Cheat Sheet (1)

***QoS**:* la qualità del servizio è fondamentale, non basta che sia economicamente conveniente ma deve essere anche affidabile.

***SLA**:* parte del contratto dove viene indicata l'affidabilità del servizio. Viene scritta da tre figure: avvocato, business expert e sviluppatore. Esempi:

- S3: da nessuna parte è riportato che abbiamo un qualche tipo di garanzia o indice di affidabilità, c'è soltanto una spiegazione su quanto "credito" l'utente può ricevere se si dovessero verificare dei disservizi. Leggendo attentamente le clausole si arriva alla conclusione che un eventuale rimborso è quasi impossibile da ricevere.
- Google Compute Engine: per singola istanza garantisce un Monthly Uptime Percentage ≥ 99.5%. Se non è in grado di fornire servizi, il cliente ha diritto a un financial credit (rimborso) come segue:
  - 95.00% < Monthly Uptime Percentage < 99.50% → 10%
  - 90.00% < Monthly Uptime Percentage < 95.00% → 25%
  - Monthly Uptime Percentage < 90.00% → 100%
- Microsoft: non garantisce neanche che il Cloud sia sicuro, anzi ci invita a fare regolarmente un backup dei nostri dati del Cloud su un'altra piattaforma.
- Facebook: avverte che qualsiasi contenuto condiviso pubblicamente puo essere utilizzato da Facebook o essere ceduto a terzi in tutto il mondo che possono utilizzarlo a loro volta per qualsiasi scopo.

***Monthly Uptime Percentage**:* minuti in un mese, meno i minuti di Downtime di tutti i periodi di Downtime in un mese, divisi per i minuti in un mese.

Q: How much credit can I get if my virtual machine instance was not reachable from 8am to 5:30pm on April 1st and 2nd?
(I tried to connect to it every 30', each time for 5' - no success)

$$\frac{(30 \times 24 \times 60) - (2 \times 20 \times 5)}{30 \times 24 \times 60} = \frac{43000}{43200} = 0{,}995370370$$

***Downtime Period**:* periodo di uno più minuti consecutivi di Downtime.

***Customer Must Request Financial Credit**:* al fine di ricevere uno qualsiasi dei financial credit sopra descritti, il cliente deve informare l'assistenza tecnica di Google entro 60 giorni dal momento in cui diventa idoneo a ricevere il financial credit. Il cliente deve inoltre fornire a Google i file di log del server che mostrano errori di perdita di connettività esterna e la data e l'ora in cui si sono verificati gli errori. Se il cliente non rispetta questi requisiti, perderà il diritto a ricevere un financial credit.

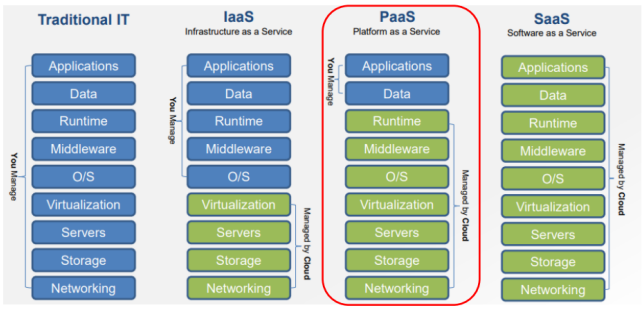
***Overprovisioning**:* porta a sprecare risorse poichè si prevede un flusso maggiore di richieste di quelle effettive, ma anche nel caso di una previsione corretta avviene comunque una perdita economica.

***Underprovisioning**:* porta a errori e crash del sistema con conseguente perdita di utenti.

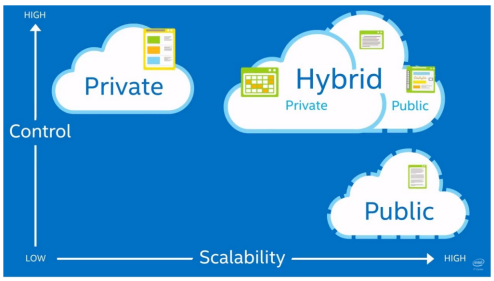
***IaaS**:* fornisce servers e memoria virtualizzati, gestendo tutta l'infrastruttura, il cliente è responsabile di OS, applicazione, dati ed esecuzione (EC2, S3). +31.3%

***PaaS**:* fornisce un'intera piattaforma come un servizio, gestendo infrastruttura, OS ed enabling software, il cliente è responsabile di installare e gestire l'applicazione e i dati (Heroku, Azure, Google Application Engine, Openshift, Firebase). Il mercato globale del PaaS ammonta a $60.1B ed è principalmente relativo allo sviluppo di applicazioni.

***SaaS**:* fornisce software on-demand accessibile mediante client thin o API, gestendo infrastruttura, OS, applicazione e dati (tutto), il cliente non è responsabile di nulla (Salesforce, Google Drive).



Modelli di servizio



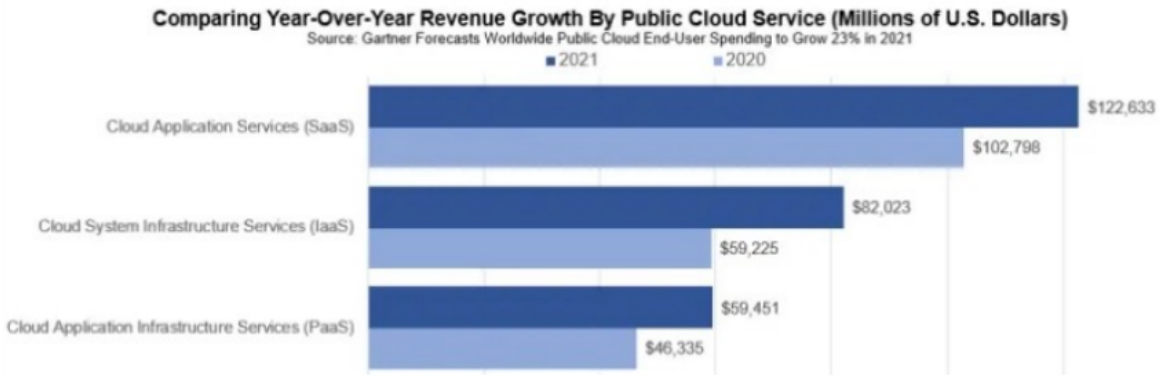
Modelli di deployment

Table 1. Worldwide IaaS Public Cloud Services Market Share, 2020-2021 (Millions of U.S. Dollars)

Company	2021 Revenue	2021 Market Share (%)	2020 Revenue	2020 Market Share (%)	2020-2021 Growth (%)
Amazon	35,380	38.9	26,201	40.8	35.0
Microsoft	19,153	21.1	12,659	19.7	51.3
Alibaba	8,679	9.5	6,117	9.5	41.9
Google	6,436	7.1	3,932	6.1	63.7
Huawei	4,190	4.6	2,681	4.2	56.3
Others	17,056	18.8	12,697	19.8	34.3
Total	**90,894**	**100.0**	**64,286**	**100.0**	**41.4**

Source: Gartner (June 2022)

IaaS Market share 2020 - 2021



SaaS > IaaS > PaaS (Revenue) - Gartner Aprile 2020- 2021

***Virtualizzazione**:* è un livello di astrazione nell'architettura di una macchina: il sistema operativo non è più legato al server su cui è in esecuzione, ma è astratto dall'hardware, cioè il sistema operativo non viene installato direttamente sull'hardware.

***Server virtualization**:* è il livello di virtualizzazione tra il server fisico e il sistema operativo. Sopra questo livello si possono installare le macchine virtuali che contengono all'interno sistema operativo e applicazione.

***Hypervisor**:* è una funzione che crea il livello di virtualizzazione e si occupa della gestione delle macchine virtuali. Ne esistono due tipi:

- Type 1: l'hypervisor viene caricato direttamente sull'hardware della macchina ed è impiegato nei data center.
- Type 2: l'hypervisor viene caricato sul sistema operativo dell'hardware come una normale applicazione, ha un overhead maggiore e dunque un consolidation ratio inferiore (il numero di server virtuali che possono girare contemporaneamente su una macchina fisica). Usato tipicamente nei laptop.

***AWS EC2** (IaaS):* mette a disposizione dei server virtuali chiamati istanze in modo semplice, veloce ed economico. Offre anche servizi di sicurezza come VPC e persistenza dei dati con EBS (ogni volume viene automaticamente replicato all'interno della sua availability zone). Esistono diversi tipi di istanze:

- On demand: paghi solo per quello che utilizzi. Da $0.0048 (0.5GB) a $0.3072 (32GB).
- Spot istances: sono istanze non utilizzate che Amazon mette all'asta ogni 5 minuti e possono costare tra il 50% e il 90% in meno rispetto alle precedenti.
- Reserved istances: pagamento mensile, risparmiando rispetto alle on demand, si usa quando sappiamo già quali e quante macchine ci serviranno. Da $1.90 ($0.0042 on demand) al mese a $187.61 al mese ($0.4080 on demand).
- Dedicated hosts: affittare un intero server fisico con istanze EC2 dedicate.
- Free tier: livello freemium con limitazione.

Costi → Dedicated hosts > On demand > Reserved istances > Spot istances

---

***AWS S3** (IaaS):* fornisce uno storage sicuro e facile da usare (drag and drop di file in un bucket). S3 per assicurarci di non perdere i nostri dati esegue 3 backup su 3 unita di memoria diverse e consente di mantenere tutte le vecchie versioni dei file in modo da poterli recuperare se cancellati erroneamente. Si paga per GB e in base alla classe di memorizzazione e si ha un free usage tier di 5GB. Esistono tre classi di memorizzazione:

- Standard: per i dati a cui si accede con maggiore frequenza, garantisce bassa latenza e throughput elevato. $0.023 (50TB) al mese, $0.022 (450TB) al mese, $0.021 (500TB) al mese.
- Standard Infrequent Access: per i dati a cui si accede con minore frequenza ma richiedono un accesso rapido in caso di necessità. Offre durabilità e velocità

effettiva elevate e bassa latenza della classe Standard, a un minor costo di archiviazione per GB e una minor tariffa di recupero per GB. $0.0125 al mese.

- Glacier: per i dati a lungo termine ad accesso raro che non richiedono accesso immediato a basso costo. Glacier Instant Retreval permette un risparmio fino al 68% sui costi di archiviazione rispetto a Standard Infrequent Access mentre Glacier fino al 10% in meno rispetto a S3 Glacier Instant Retrieval. $0.004 o $0.0036 o $0.00099 al mese.

Costi → Standard > Standard Infrequent Access > Glacier

Dropbox *(IaaS):* servizio di file hosting che offre memorizzazione Cloud, sincronizzazione dei file e strumenti di collaborazione. Nasce con l'obiettivo di fornire gratuitamente spazio di archiviazione con limitazioni, e chi vuole puo acquistare l'account premium per avere più capacita di memorizzazione. A marzo 2018 (o 2016) Dropbox contava più di 500 milioni di utenti. Per i primi 8 anni Dropbox ha sempre usato i server di Amazon S3. Per passare al 100% del controllo progettò hardware apposito chiamato "Diskotech" e sviluppò un nuovo software "Magic Pocket" che non performava sulle nuove macchine. Dovevano trasferire 4 petabyte prima del rinnovo del contratto con Amazon. Secondo Forbes, le entrare di Dropbox nel 2021 e 2022 sono destinate a diminuire.

Container*:* sono pacchetti di software che contengono tutti gli elementi necessari per l'esecuzione in qualsiasi ambiente (virtualizzano il sistema operativo e sono eseguibili ovunque). Vantaggi e svantaggi:

- + Più leggeri (chiede meno risorse rispetto ad una macchina virtuale)
- + Tempi di avvio più brevi (rispetto ad una macchina virtuale)
- + Più semplici da implementare (rispetto ad una macchina virtuale)
- - Meno sicuri, hanno meno isolamento (rispetto ad una macchina virtuale)

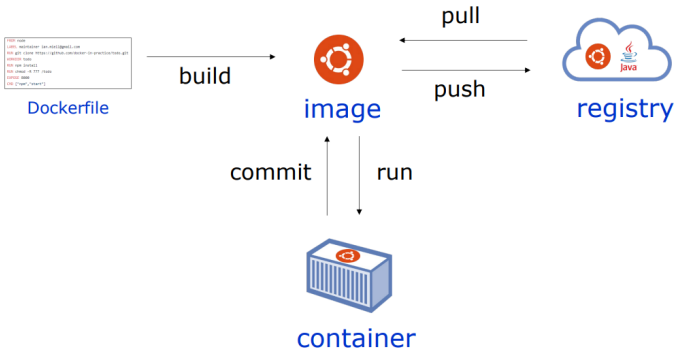
Docker*:* piattaforma che consente di runnare applicazioni in un ambiente isolato sfruttando i container, al posto dell'hypervisor viene montato il cosiddetto Docker engine che permette di creare i container. I container sono volatili ovvero non tengono traccia dei dati una volta terminata l'esecuzione. Per questo motivo vi è un altro concetto dei volumi per memorizzare i dati in modo persistente. L'applicazione viene impacchettata in una immagine, ovvero un file che rappresenta l'intera applicazione, questo diventa un template che poi verra usato per creare e mandare in esecuzione un Container. I template vengono memorizzati in un registry (per esempio Docker registry) che è strutturato in repositories. Ogni repository contiene

un insieme di immagini per versioni diverse del software. Le immagini vengono identificate dalle coppie repository : tag, e sono strutturate in livelli: a livello più basso vi è il base image. Si compone di due componenti:

- Docker Engine: per creare e runnare i containers
- Docker Hub: per la distribuzione dei containers

I comandi principali di Docker sono:

- pull: scarica un'immagine da un Docker registry
- run: manda in esecuzione il container definito dall'immagine
- commit: per creare ed effettuare delle modifiche che possono essere committate in una nuova immagine



- build: permette di specificare la costruzione dell'immagine in un file chiamato dockerfile (permette di specificare come si vuole che sia formata l'applicazione multiservizio da mandare in esecuzione)

Docker compose è un tool che permette di definire ed eseguire applicazioni Docker multi-container. Il comando docker compose up aggrega gli output di tutti i container.

Il comando docker ps mostra tutti container in esecuzione e stoppati, mentre il comando docker images mostra tutte le immagini al livello superiore, insieme alle loro repository, i loro tag e la loro dimensione. Il comando docker container ls mostra la lista dei container mentre il comando docker container prune rimuove tutti i container stoppati.

Il comando RUN all'interno del dockerfile crea un nuovo image layer al di sopra del precedente, quindi è sempre raccomandato di concatenare tutti i comandi RUN insieme.

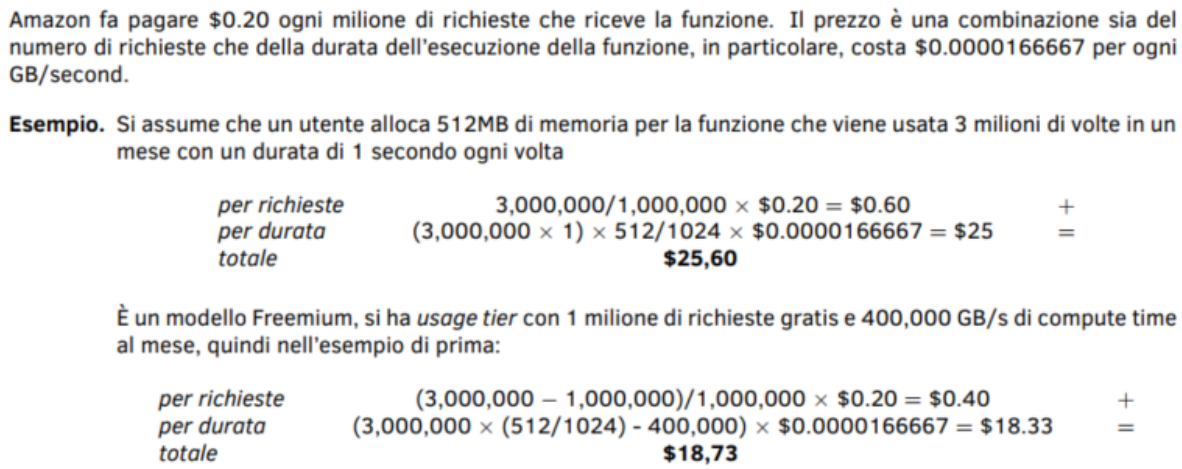
Il comando CMD permette di impostare un comando di default che verrà eseguito ogni volta che eseguiamo un container Docker. Se nel comando RUN forniamo un comando aggiuntivo, l'argomento del comando CMD verrà ignorato. Nel caso in cui il comando CMD abbia più argomenti, solo l'ultimo verrà eseguito.

Un’altra funzione di Docker è lo swarm che permette di gestire un insieme (cluster) di host Docker. Quindi i vari task vengono ripartiti tra host diversi. I nodi del cluster possono avere due ruoli:

- Manager, che delegano e distribuiscono i task
- Workers, che eseguono i task

***AWS Lambda** (FaaS):* permette agli utenti di mandare in esecuzione del codice senza gestire in alcun modo i server, Lambda si prende l’incarico di mandare in esecuzione il codice e scalarlo opportunamente. Un’altra funzionalità è quella di associare automaticamente al codice dei trigger, ovvero, degli eventi che scatenano l’esecuzione della funzione. In AWS Lambda quando creiamo una funzione dobbiamo specificare un Runtime da utilizzare, questo però contiene solo funzionalità e librerie di base. Se si vogliono utilizzare ulteriori librerie è necessario creare dei livelli Lambda che verranno utilizzati dalle nostre funzioni. Un livello Lambda è un archivio di file con estensione .zip che può contenere codice o dati aggiuntivi. I livelli Lambda offrono un metodo pratico per impacchettare librerie e altre dipendenze che è possibile utilizzare insieme alle funzioni Lambda. In AWS Lambda paghi soltanto il tempo di calcolo necessario per eseguire la funzione caricata dall’utente. Vi sono tre modi per mandare in esecuzione la funzione:

- Caricare una funzione definita dall’utente
- Usare l’IDE di AWS Lambda per scrivere la funzione
- Lambda fornisce un insieme di template



Amazon fa pagare $0.20 ogni milione di richieste che riceve la funzione. Il prezzo è una combinazione sia del numero di richieste che della durata dell'esecuzione della funzione, in particolare, costa $0.0000166667 per ogni GB/second.

Esempio. Si assume che un utente alloca 512MB di memoria per la funzione che viene usata 3 milioni di volte in un mese con un durata di 1 secondo ogni volta

per richieste	3,000,000/1,000,000 × $0.20 = $0.60	+
per durata	(3,000,000 × 1) × 512/1024 × $0.0000166667 = $25	=
totale	**$25,60**	

È un modello Freemium, si ha *usage tier* con 1 milione di richieste gratis e 400,000 GB/s di compute time al mese, quindi nell'esempio di prima:

per richieste	(3,000,000 − 1,000,000)/1,000,000 × $0.20 = $0.40	+
per durata	(3,000,000 × (512/1024) - 400,000) × $0.0000166667 = $18.33	=
totale	**$18,73**	

Prezzi

Commerciali	Open source
AWS Lambda	Apache Openwhisk
Google Cloud Functions	Fission
MS Azure Functions	Fn
	Knative
	Kubeless
	Nuclio
	OpenFaaS

Top 10 FaaS

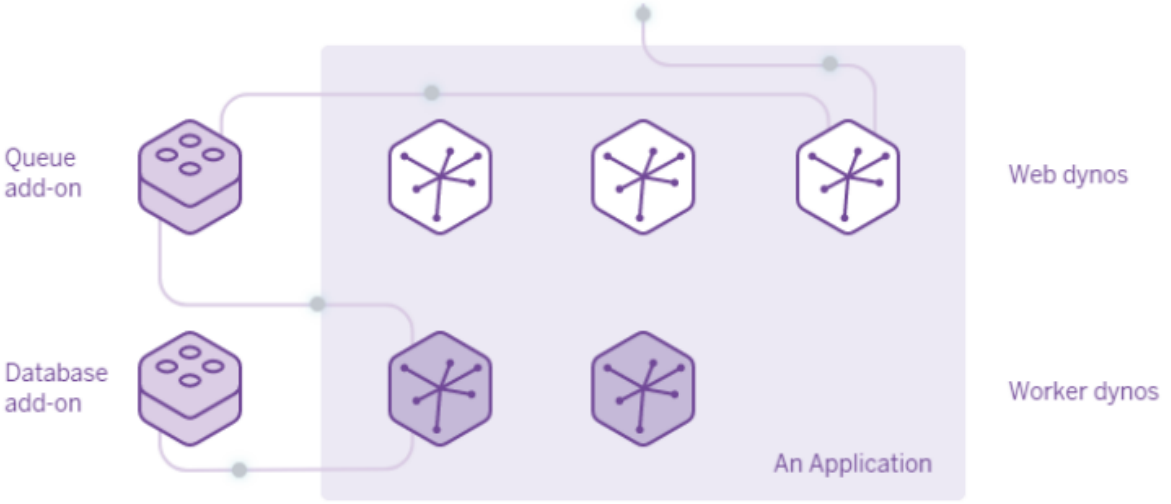
- Business View:
  - Licensing
    - Commerciali: tutti licenze proprietarie (Azure alcune open source)
    - Open source: tutti licenze permissive (Apache 2.0)
  - Installation
    - Commerciali: tutti as-a-service (Azure può essere installato)
    - Open source: quasi tutti on-premise (installabili)
  - Source code
    - Commerciali: no codice sorgente
    - Open source: codice sorgente spesso implementato in Go ospitato su GitHub
  - Release
    - Commerciali: sempre aggiornate e in produzione
    - Open source: leggermente indietro
  - Interface
    - Commerciali: CLI (API e GUI non sempre fornite)
    - Open source: CLI (API e GUI non sempre fornite)

    - Community
        - Commerciali: Stack Overflow per AWS Lambda
        - Open source: GitHub
    - Documentation
        - Commerciali: si, per sviluppo dell’app, si, per sviluppo della piattaforma
        - Open source: si, per sviluppo dell’app, no, per sviluppo della piattaforma
- Technical View:
    - Development
        - Commerciali: Node.js, Java, Python, Docker + IDE o editor
        - Open source: Node.js, Java, Python, Docker + IDE o editor
    - Versioning
        - Commerciali: Dedicate Versioning
        - Open source: implicit Versioning
    - Event sources
        - Commerciali: invocazione sincrona basata su HTTP, poche piattaforme supportano invocazione asincrona
        - Open source: invocazione sincrona basata su HTTP, poche piattaforme supportano invocazione asincrona
    - Function orchestration
        - Commerciali: la maggioranza delle piattaforme offre un “orchestratore” dedicato
        - Open source: la maggioranza delle piattaforme offre un “orchestratore” dedicato
    - Testing & Debugging
        - Commerciali: testing e debugging integrati all’interno dell’ambiente
        - Open source: solo test call o debugging basato su log
    - Observability
        - Commerciali: strumenti di monitoraggio forniti dal provider
        - Open source: strumenti di monitoraggio forniti da terze parti

- Application delivery
  - Commerciali: quasi tutte le piattaforme usano un approccio dichiarativo al deployment, permettendo all’utente di dichiarare qual è lo stato da raggiungere senza dichiarare i passi
  - Open source: quasi tutte le piattaforme usano un approccio dichiarativo al deployment, permettendo all’utente di dichiarare qual è lo stato da raggiungere senza dichiarare i passi
- Code reuse
  - Commerciali: Lambda e Azure offrono tante funzioni gia pronte con un Function Marketplace
  - Open source: nessuna funzione pronta
- Access managment
  - Commerciali: processi di autenticazione e controllo degli accessi in maniera nativa
  - Open source: rimandano queste funzionalità all’ambiente di hosting

FaaStener Prototype permette di scegliere il FaaS più adatto.

Heroku *(PaaS):* piattaforma Cloud che fornisce una serie di servizi integrati e un intero sistema che permette, non solo di fare il deployment delle applicazioni, ma anche di completarle, mandarle in esecuzione e gestirle. Utilizza un sistema di container che si chiamano Dynos (container Docker ≠ container Dynos) che eseguono codice dato dall’utente. I Dynos permettono una scalabilita molto flessibile, quindi a seconda delle richieste e delle risorse, l’applicazione può scalare arbitrariamente il numero, il tipo e la dimensione dei Dynos per applicazione. Funziona nel seguente modo: l’app riceve la richiesta e la invia a uno dei Web Dynos che la analizza e la inserisce in una coda asincrona (ottima per la scalabilità orizzontale), il Worker Dynos prende la richiesta e la soddisfa, se necessario può inserire il risultato in un database per ottenere persistenza.



Funzionamento

Dyno Type	Price per dyno/month
free	$0
hobby	$7
standard-1x	$25
standard-2x	$50
performance-m	$250
performance-l	$500

Prezzi Dynos

Si divide in tre fasi:

1. Buildtime: ha bisogno di codice sorgente, lista di dipendenze (librerie) e procfile (file di testo che indica qual è il comando per far partire l'app). Una volta date in input queste tre cose parte il sistema automatico di built: dopo aver ricevuto il codice, scarica il Build Packet (linguaggio, dipendenze, librerie, ...), produce uno slug (copia compressa e preconfezionata dell'app ottimizzata per la distribuzione) e lo mette in esecuzione in un Dynos. Il componente finale per eseguire l'applicazione è il sistema operativo (Ubuntu) che è aggiunto da Heroku e si chiama stack.

2. Runtime: quando si fa il deployment o si scala manualmente l’applicazione, Heroku crea uno o più Dynos dove ognuno avra lo stesso stack e slug che rappresenta l’app. A questo punto Heroku esegue il comando che l’utente ha specificato e fa partire l’applicazione. Heroku permette di configurare le risorse che a tempo di esecuzione si vogliono utilizzare, per esempio esistono quattro tipi principali di Dynos:
    a. Free, hobby, in cui vi sono le funzionalità di base
    b. Standard, supportano la scalabilità orizzontale
    c. Performance, oltre alla scalabilità orizzontale supportano l’autoscaling
3. Add-ons: Heroku ha pi`u di 150 add-ons che gli sviluppatori possono utilizzare per estendere l’applicazione (autenticazione, log, monitoring, data stores). L’aspetto negativo è che questo ci lega all’utilizzo della piattaforma e instaura una forma di vendor lock-in rendendo la nostra applicazione più difficilmente portabile.

Microsoft Azure *(PaaS):* offre funzionalità simili a Heroku come scalabilità verticale, supporta diversi linguaggi, SQL database, machine learning, servizi media per live on-demand, meccanismi di sicurezza e data storage. Offre anche servizi IaaS.

OpenShift *(PaaS):* fornisce molte funzionalità semplici per il build e il deployment automatico. OpenShift puo monitorare lo stato delle applicazioni e farle ripartire in caso di fallimenti. OpenShift è basato su kubernetes. Piattaforma DevOps che permette la costruzione e l’erogazione dei servizi.

Amazon Web Services (AWS) Elastic Beanstalk
Oracle Cloud Platform (OCP)
Google App Engine
Microsoft Azure
Salesforce aPaaS
Red Hat OpenShift PaaS
Mendix aPaaS
IBM Cloud Platform
SAP Cloud Platform
Engine Yard

Top 10 PaaS

Only 10% of organizations are "very concerned" about IaaS/PaaS public cloud vendor lock-in. Another 32% are "somewhat concerned”.
451 Research, 2020

Generally, the calculus in the enterprise market is shifting more toward speed, and we'll worry about platform later.
Dave Bartoletti, Forrester Research, 2020

Considerazioni sul vendor lock-in

PaaSfinder permette di trovare il PaaS che più fa al caso nostro.
