Cloud-based software engineering

Antonio Brogi

Department of Computer Science
University of Pisa


Cloud computing 101
- motivations

Planning a BeepBeep start-up

Define **business model** and **business plan**

CapEx

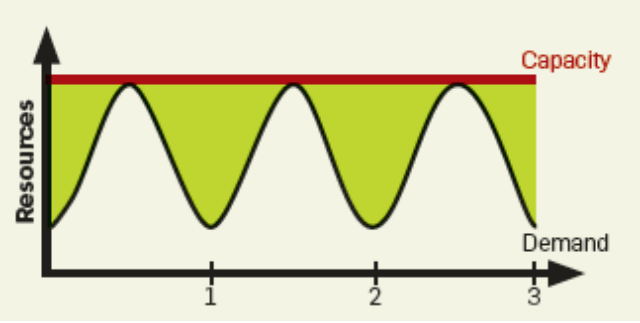
- Get hardware
- Get software licences
- Get IT staff
- Get premise
- ..

OpEx

- Maintain & update hardware
- Pay software licences
- Pay IT staff
- Manage app (deploy, scale, mantain, evolve)
- Pay premise
- …

Estimating service demand

Service demand changes with time, nobody knows how



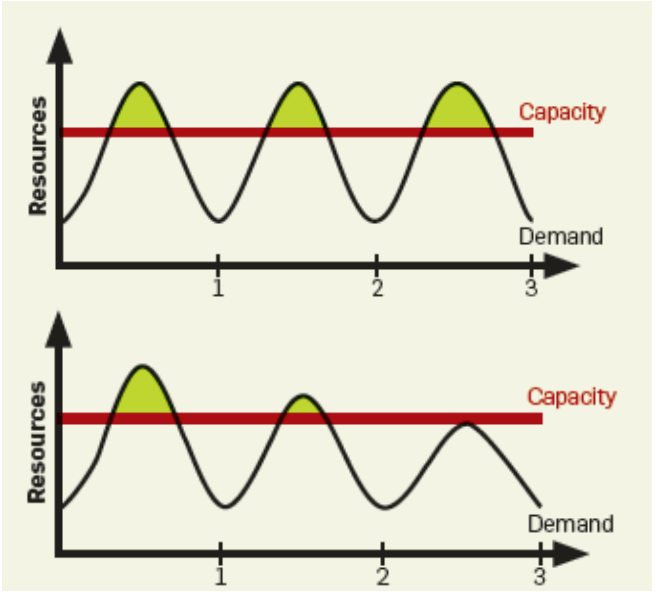
Overprovisioning

Ensuring in advance provisioning for expected demand peaks (due to diurnal or seasonal patterns or unexpected demand bursts) leads to wasting resources (if prediction is correct - even worse if spike is overestimated)

Underprovisioning

If spike is underestimated then underprovisioning may accidentally turn away excess users

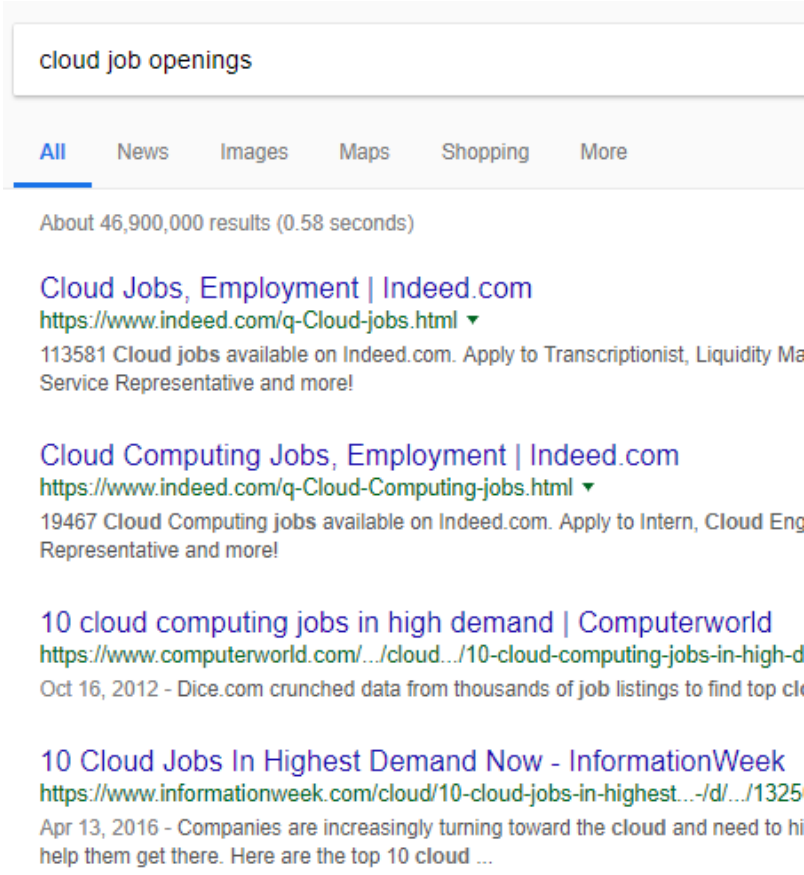
Cost of underprovisioning more difficult to measure, but as serious as cost of overprovisioning - not only do rejected users generate zero revenue, they may never come back




Seemingly infinite resources,
available on demand

	2016	2017	2018	2019	2020
SaaS	48.2	58.6	71.2	84.8	99.7
PaaS	9.0	11.4	14.2	17.3	20.8
IaaS	25.4	34.7	45.8	58.4	72.4

Public cloud services revenue forecast (BUSD) [Gartner, Oct2017]




Cloud computing 101
- motivations
- definition

NIST
National Institute of Standards and Technology
U.S. Department of Commerce

2. The NIST Definition of Cloud Computing

Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.

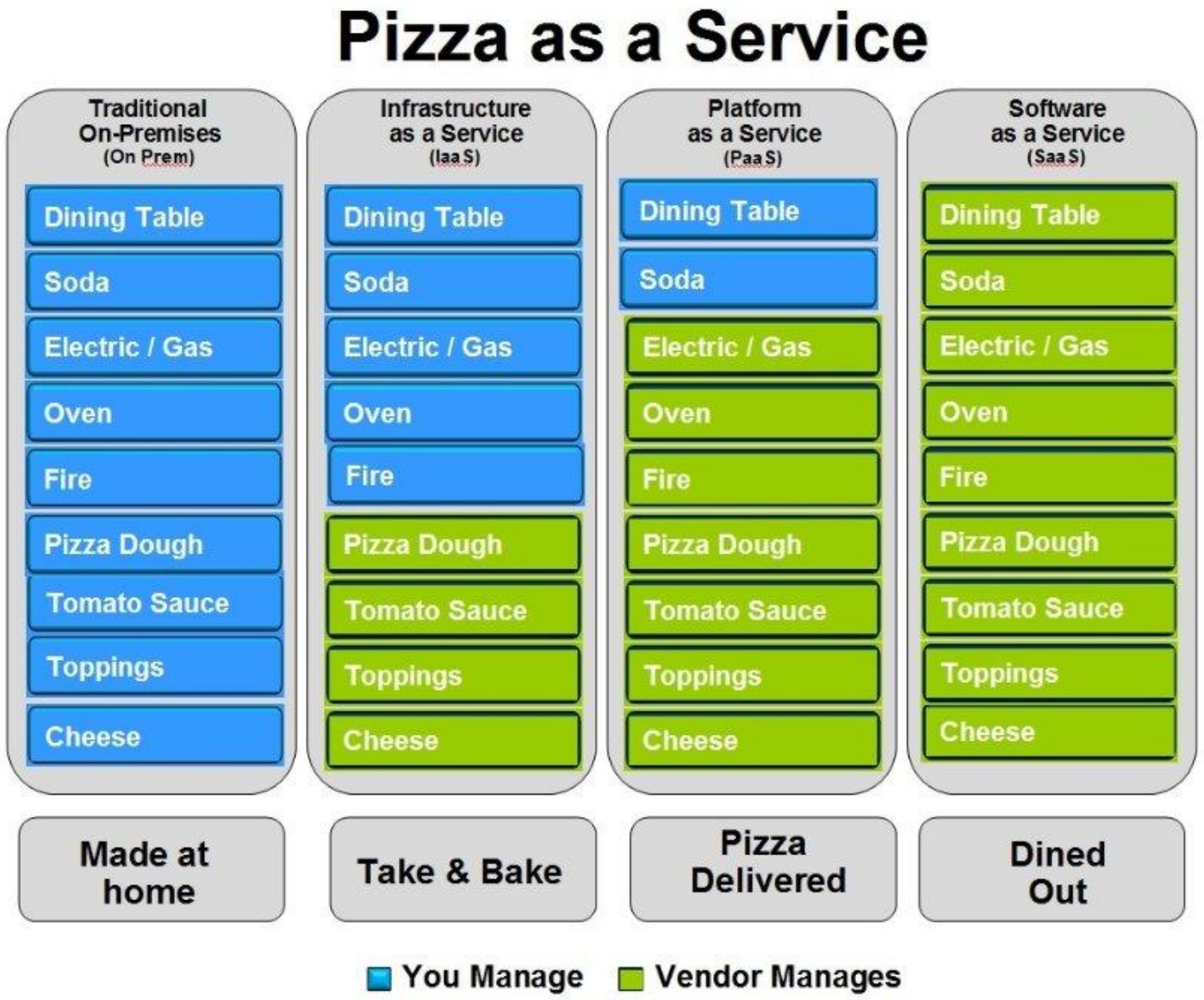
Key ideas

- Efficient pooling of on-demand, self-managed virtual infrastructures, consumed as services
- Delivery of dynamically scalable, virtualised resources over the Internet to multiple clients
- Decoupling delivery of computing services from underlying technology

Economics

- Elimination of up-front commitment by cloud users (converting CapEx to OpEx)
- Pay-per-use
  - customers just love this! ❤️
  - even if pay-per-use more expensive, cost is outweighed by economical benefits of ***elasticity*** and ***transference of risk***

Service models



[Albert Barron - IBM Software]

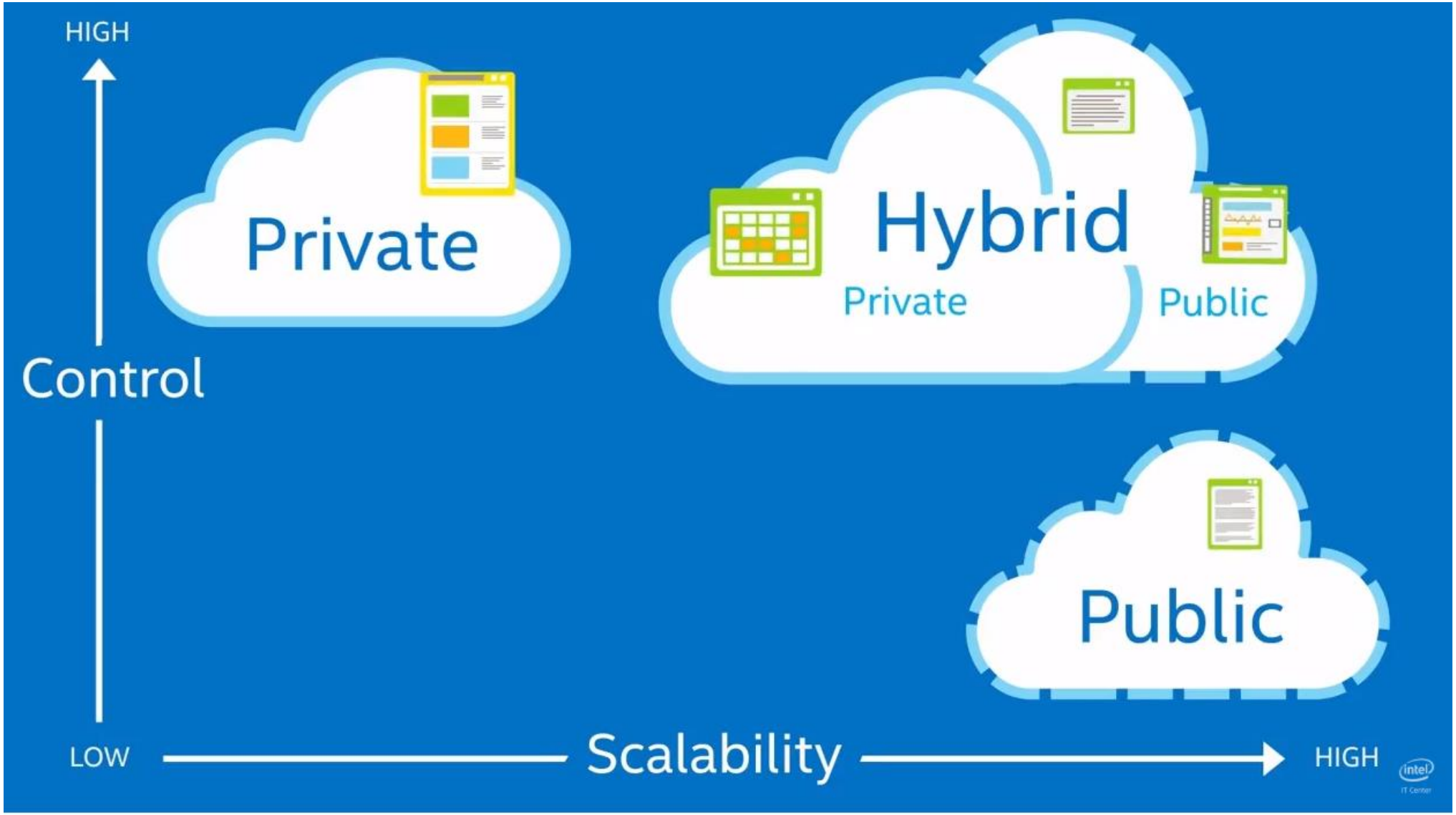
Service models

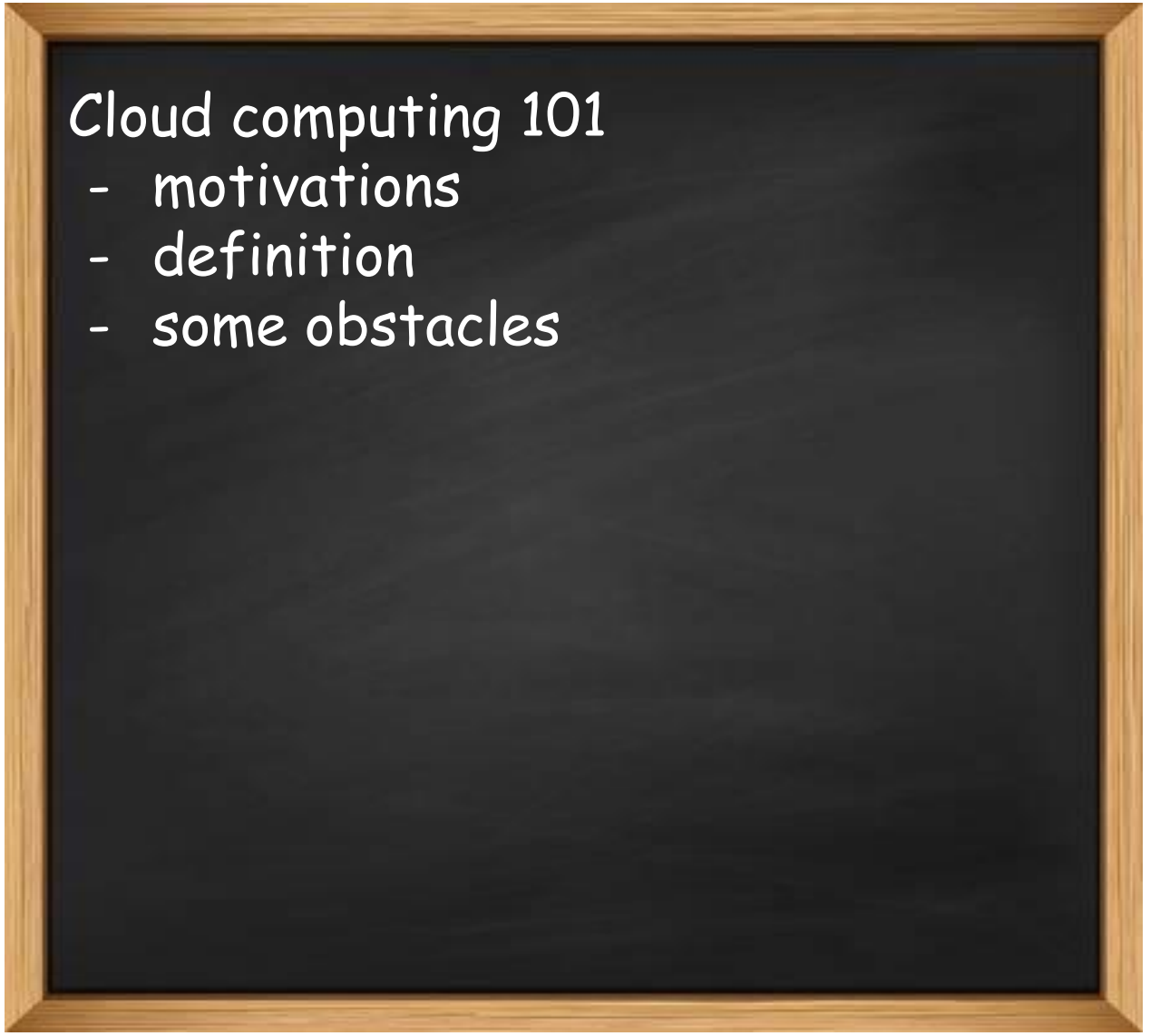
- SaaS provides software on-demand for use, accessible via thin clients or APIs
- SaaS provider manages infrastructure + OS + app
- Client responsible for nothing
- Example: salesforce, …

- PaaS provides whole platform as a service (VMs, OS, apps, services,SDKs,…)
- PaaS provider manages infrastructure + OS + enabling SW
- Client responsible for installing and managing app
- Examples: Heroku, Azure, GAE ...

- IaaS provides (virtualized) servers, storage, networking
- IaaS service provider manages all infrastructure
- Client responsible for all other aspects of the deployment (e.g., OS, app)
- Example: EC2, S3, …

Deployment models




Cloud computing 101
- motivations
- definition
- some obstacles

Some obstacles to cloud adoption

Data confidentiality

- Where will our data be stored, concretely?
- Will data privacy and integrity be guaranteed? How?
- How can we know whether a problem occurred?

Dropbox Security Bug Made Passwords Optional

Business continuity / service availability

- What if cloud provider goes out of business?
- CS mantra: “no single point of failure”...

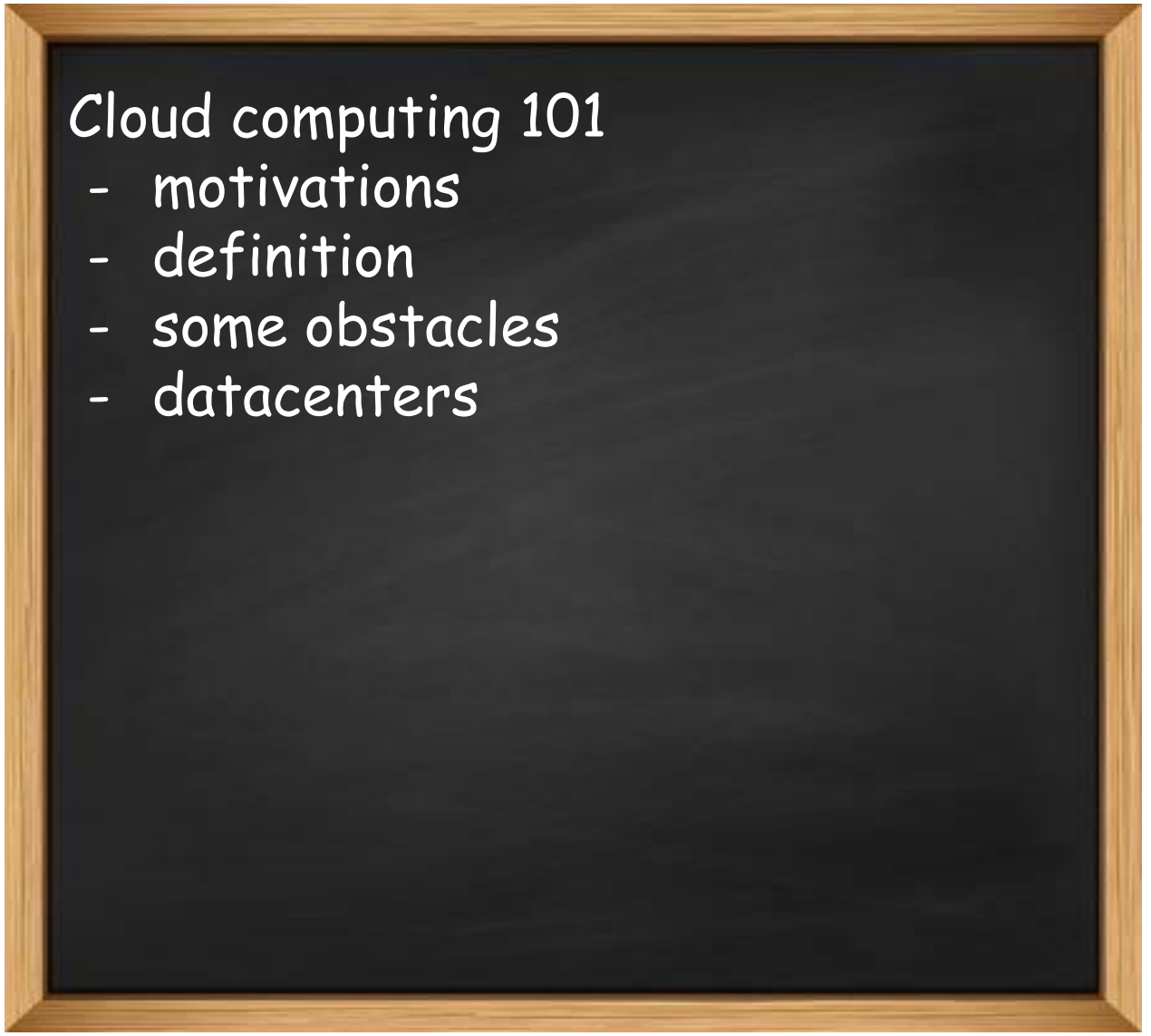
Forbes / Tech

Amazon AWS Takes Down Netflix On Christmas Eve

2017 outages (first half)

January 26 – IBM Bluemix cloud
January 31 – GitLab
February 9 – Instapaper
February 24 – Facebook
February 28 – AWS
March 16 – Azure
March 21 – Microsoft Office 365
May 22- Lululemon
June 19 – Microsoft Skype
June 28 – Apple iCloud

Vendor lock-in ?

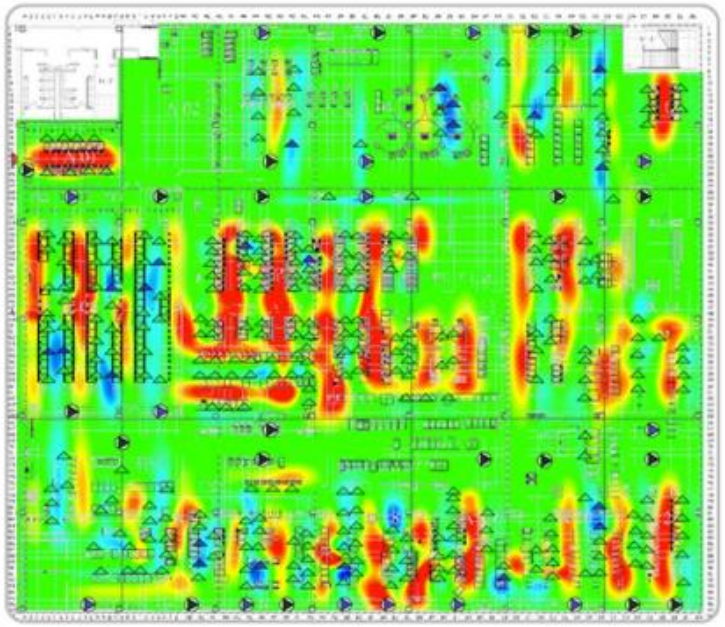

Cloud computing 101
- motivations
- definition
- some obstacles
- datacenters

Datacenters

Amazon

Google

Facebook

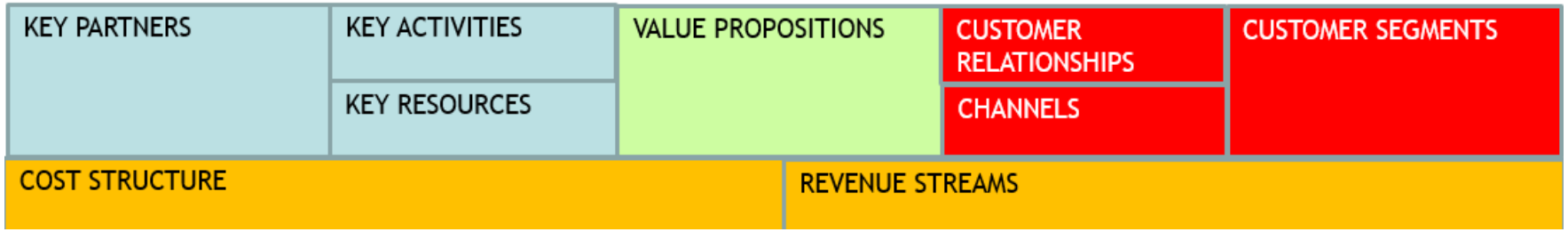


Cooling

DCIM

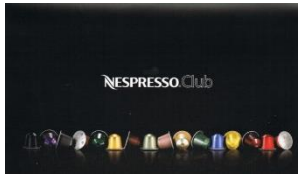

Cloud computing 101
- motivations
- definition
- some obstacles
- datacenters
- business models

Business model canvas



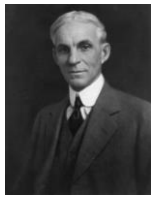
KEY PARTNERS	KEY ACTIVITIES	VALUE PROPOSITIONS	CUSTOMER RELATIONSHIPS	CUSTOMER SEGMENTS
-Machine manufacturers -Raw material suppliers	-Coffee procurement -Marketing -Selling -Post purchase	-High quality coffee -Post purchase service -Innovative product -Make customer special -Coffee maker design -Recognition	-Nespresso club -Personal assistance	-Elite (high class) -Niche market -Social status -People who want one coffee at a time
	KEY RESOURCES -Coffee beans -Coffee boutiques -Workers in shops		CHANNELS -Online shops -Boutiques	

COST STRUCTURE	REVENUE STREAMS
-Manufacturing -Distributing -Selling	-Big revenue on capsules


NESPRESSO Club

New business models

If I had asked people what they wanted, they would have said faster horses



Henry Ford

What if … we provide free music to everybody?

Spotify

Freemium

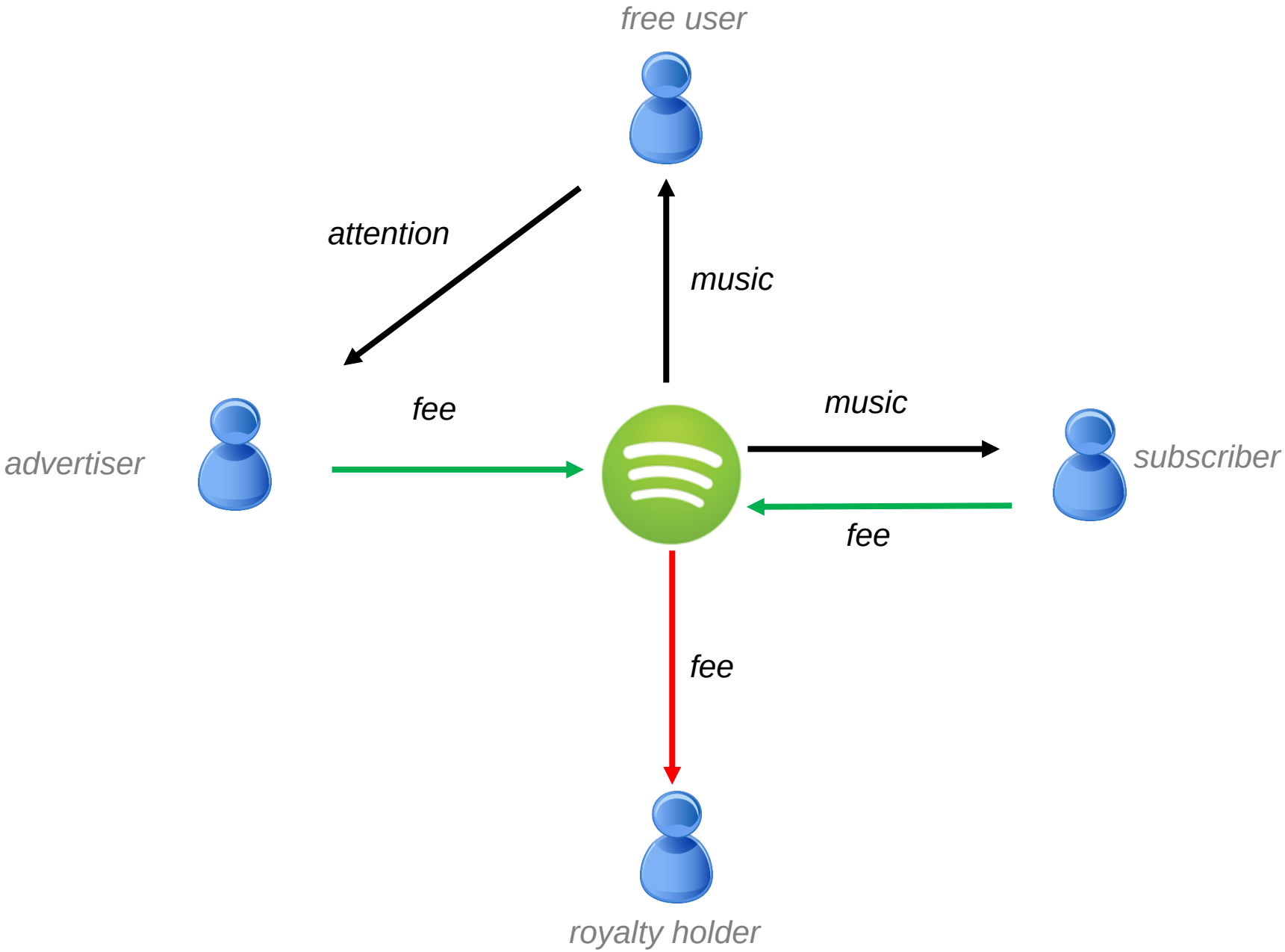
What if … we provide free storage to everybody?

Freemium

What if … we provide a free search engine to everybody?

Google

Customised advertising


free user
attention
music
advertiser
fee
music
subscriber
fee
fee
royalty holder

Advertisement fees for F free users

Fees from S subscribers

Costs for F+S users

F=97M, **S=83M** for Spotify (Q2 2018)

		free users	subscribers
if	number	100	50
	monthly fee	0	$9,99
	avg. daily plays	30	60
	fee per play	$0,004	

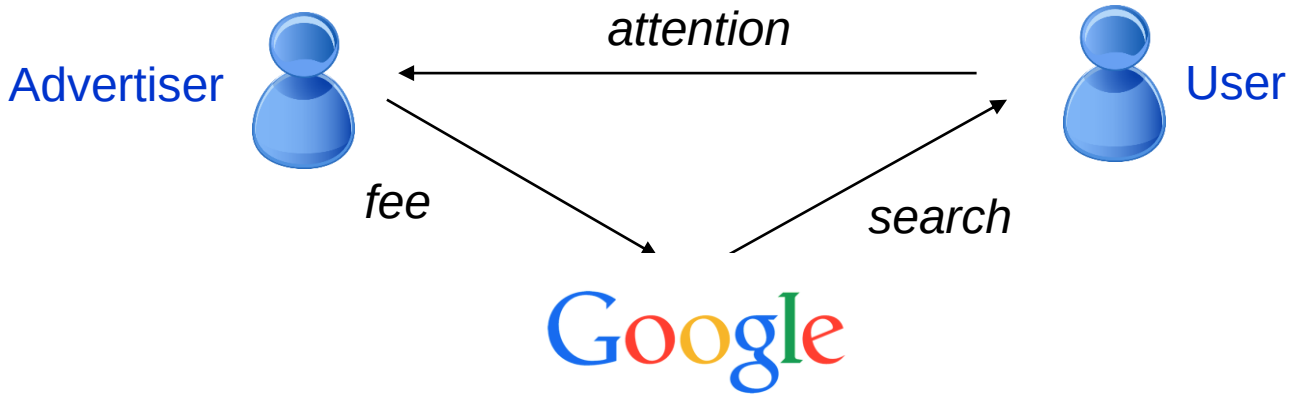
then	*monthly*	
	get from subscribers	$499,50
	pay for subscribers	$240,00
	pay for free users	$240,00
		$19,50

		free users	subscribers
	number	100	50
	monthly fee	0	$9,99
	avg. daily plays	60	120
	fee per play	$0,004	

	monthly	
	get from subscribers	$499,50
	pay for subscribers	$480,00
	pay for free users	$480,00
		-$460,50

		free users	subscribers
	number	200	50
	monthly fee	0	$9,99
	avg. daily plays	60	120
	fee per play	$0,004	

	monthly	
	get from subscribers	$499,50
	pay for subscribers	$480,00
	pay for free users	$960,00
		-$940,50

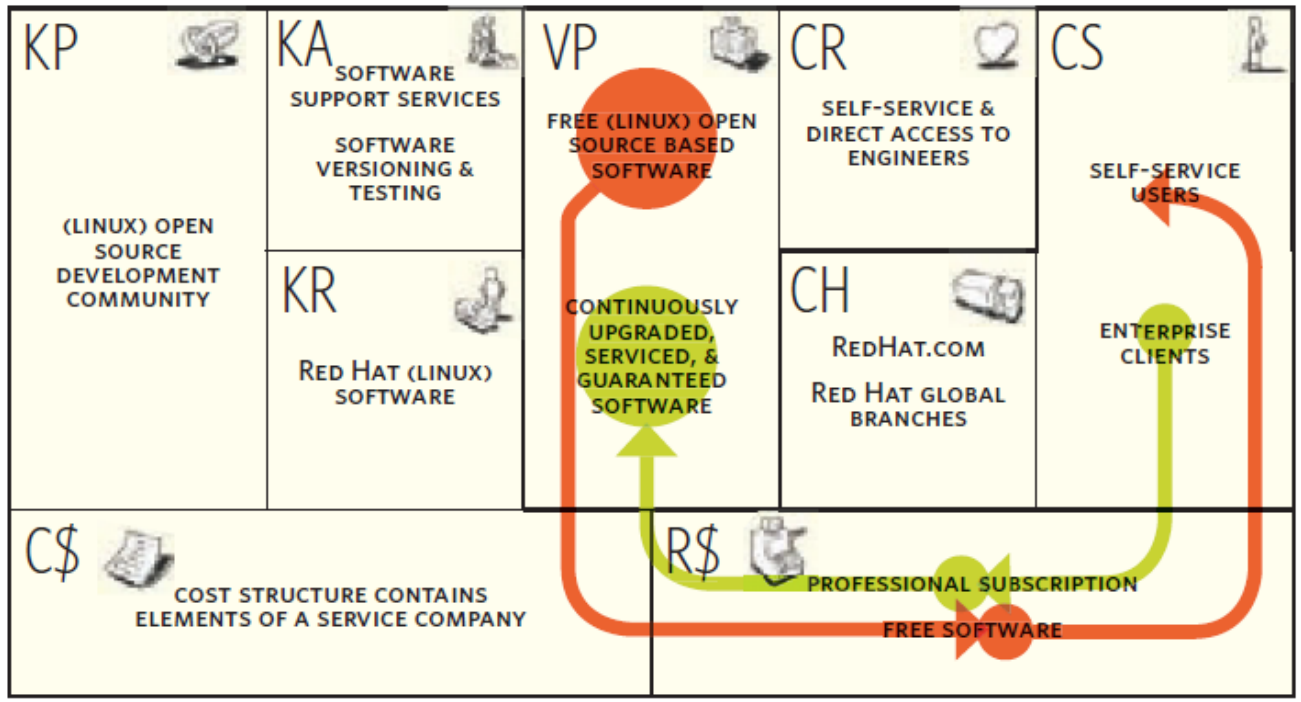


Google's business: **Customised advertising**


NETFLIX


amazon
web services

redhat



BUSINESS NEWS OCTOBER 28, 2018 / 7:09 PM / 17 DAYS AGO

IBM to acquire software company Red Hat for $34 billion

Some startup statistics

50% of owners of small businesses are over 50

50% of small businesses fail in the first 4 years

- 4% makes it to second year
- leading causes of failure
  - cash flow problems 82%
  - incompetence 46%
  - lack of managerial experience 30%

[source: smallbiztrends.com, May 2018]

[Job seeking]

The partial order

Company

Salary

What

Where

«Impact» [Millennials]

Career perspective

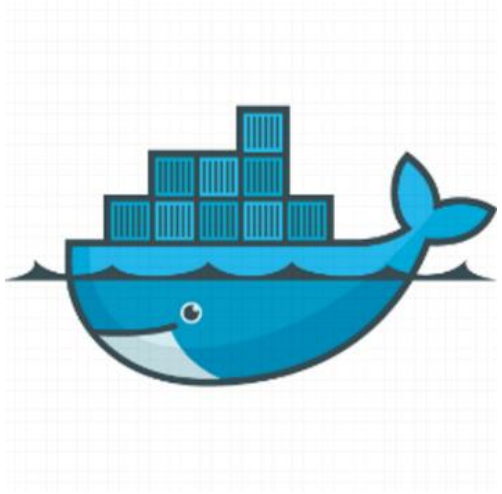
…

Where would you like to be in 2023?

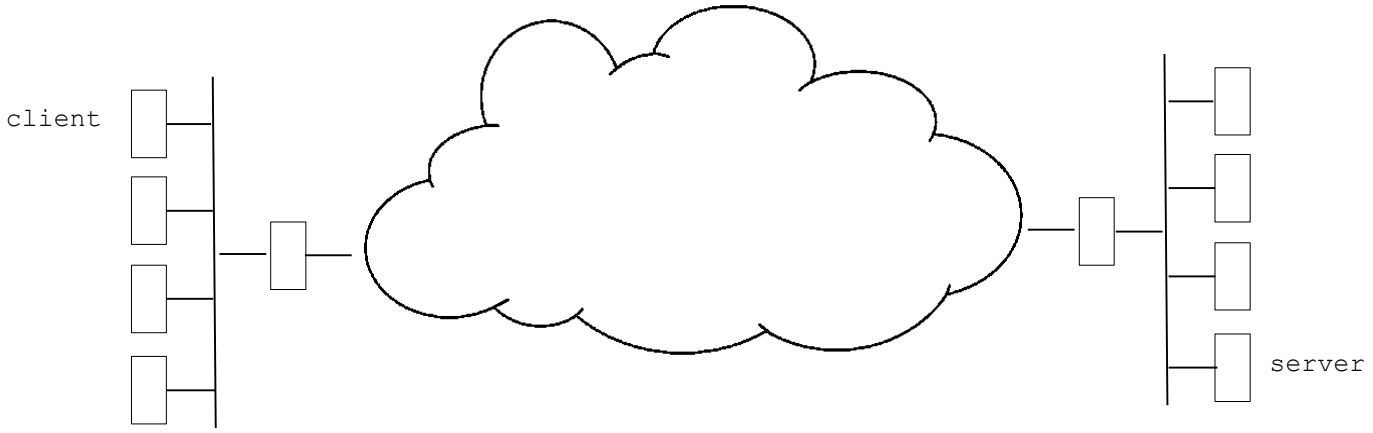

Cloud computing 101
- motivations
- definition
- some obstacles
- datacenters
- business models
- conclusions

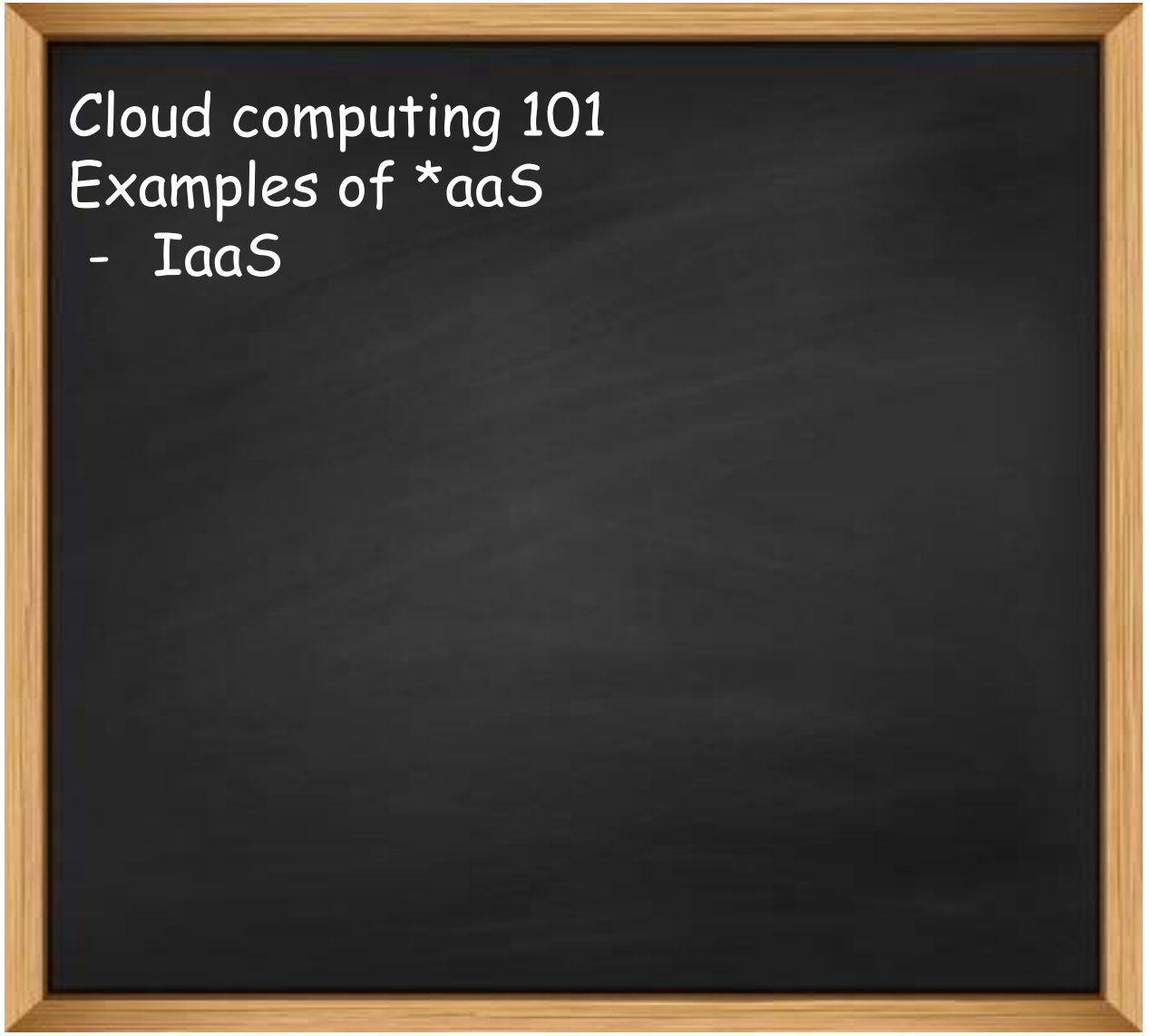
Conclusions

Cloud computing is here to stay

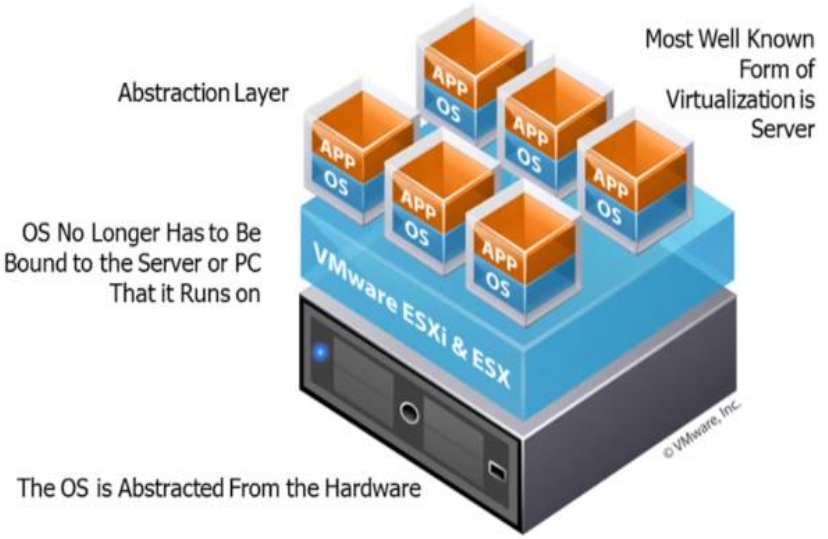


(PS: Why the name «cloud»?)




Cloud computing 101
Examples of *aaS
- IaaS

Server virtualization & hypervisors



Definition of virtualization

- Virtualization is an "abstraction layer"
- OS no longer has to be bound to the server/pc it runs on
- OS abstracted from the HW, OS isn't installed directly on the HW

Server virtualization

- Virtualization layer between physical server and the OS you would normally install
- Virtual machines: where you actually install the OSs you are used to install, and the Apps

Definition of Hypervisor

- Creates the virtualization layer
- Contains the Virtual Machine Manager (VMM)

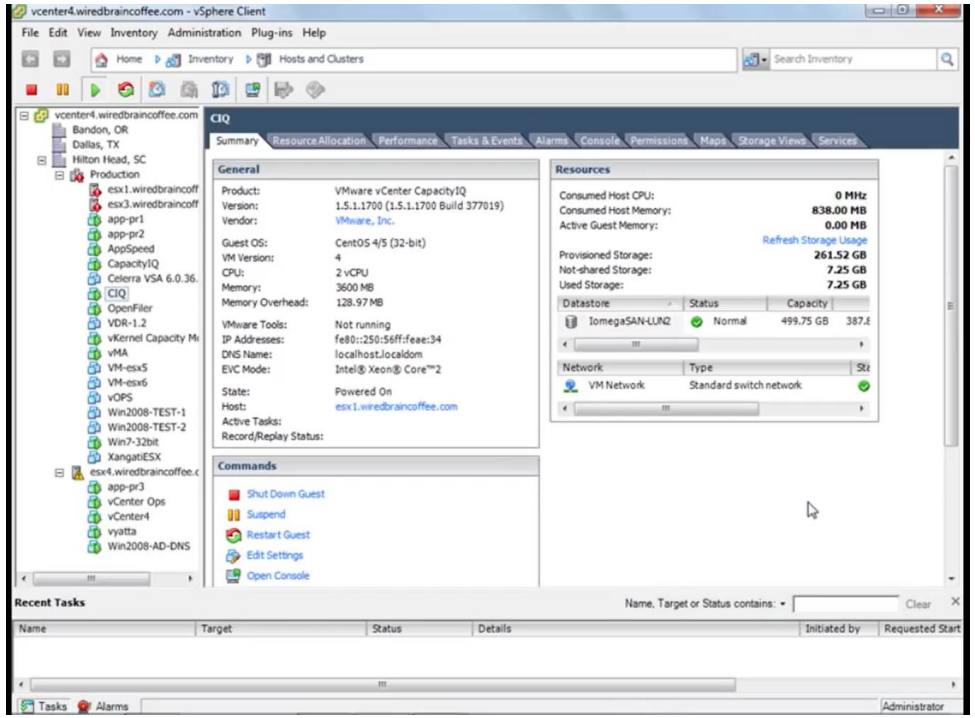
Type 1 vs. Type 2 Hypervisors

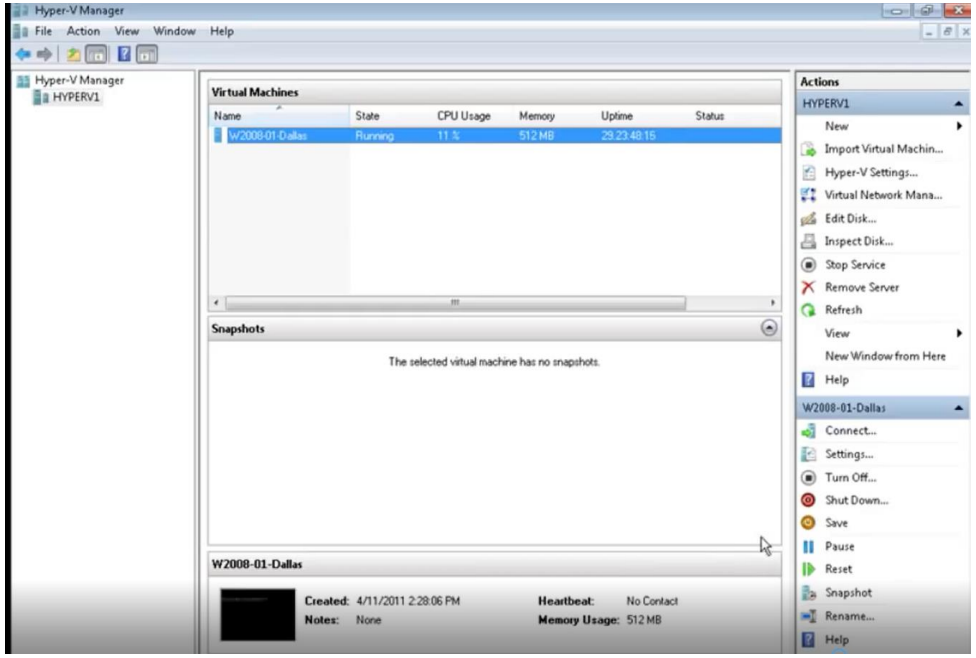
- Type 1 loaded directly on the HW, Type 2 loaded in an OS running on the HW
- Type 2 have greater overhead / lower consolidation ratio than Type 1
- Type 1 for data center, Type 2 for desktop/laptop

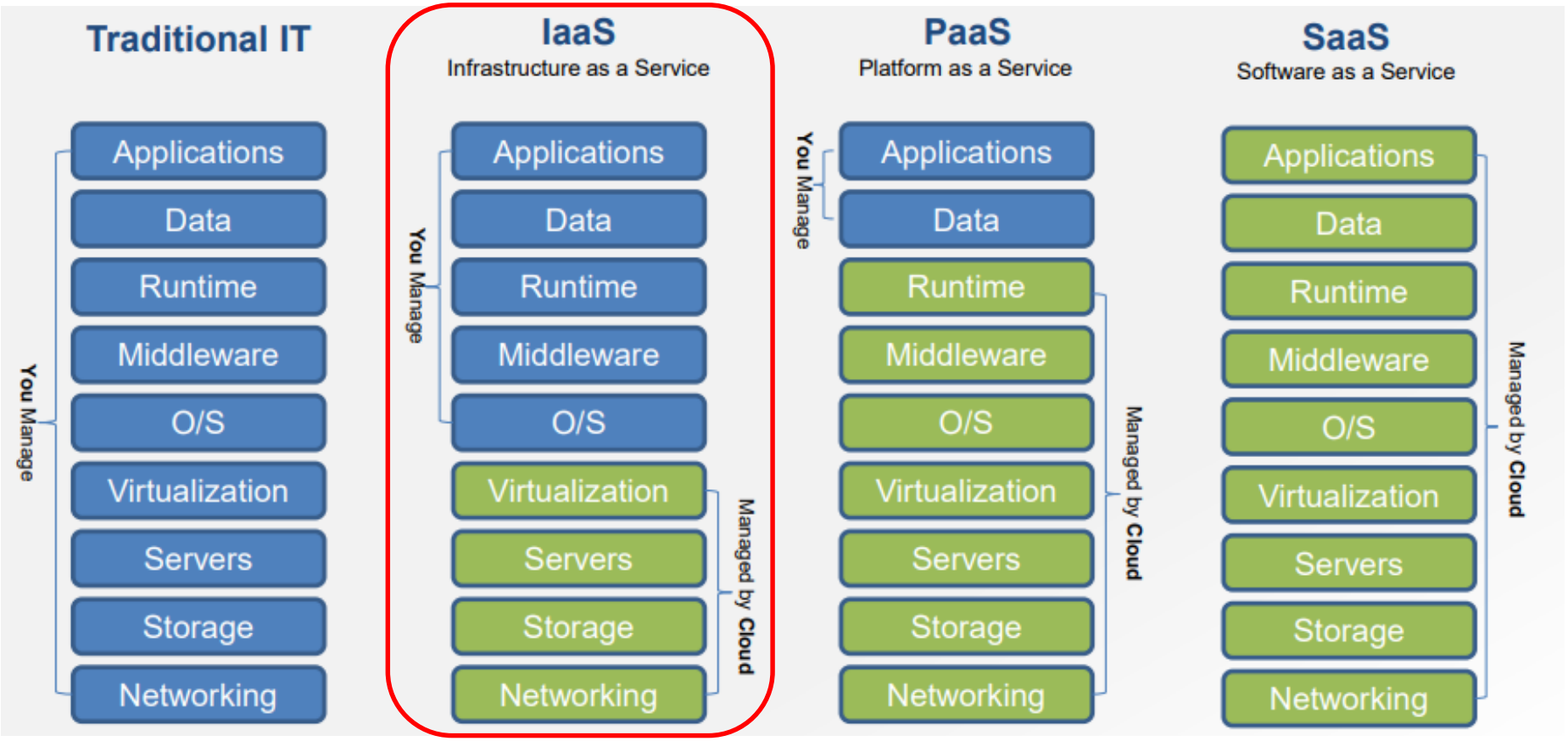
Server virtualization & hypervisors

Administering enterprise virtualization

- Client apps used by administrators (e.g., VMware vSphere, MS Hyper-V) to create/run/turn off/reboot/take snapshots of/clone VMs



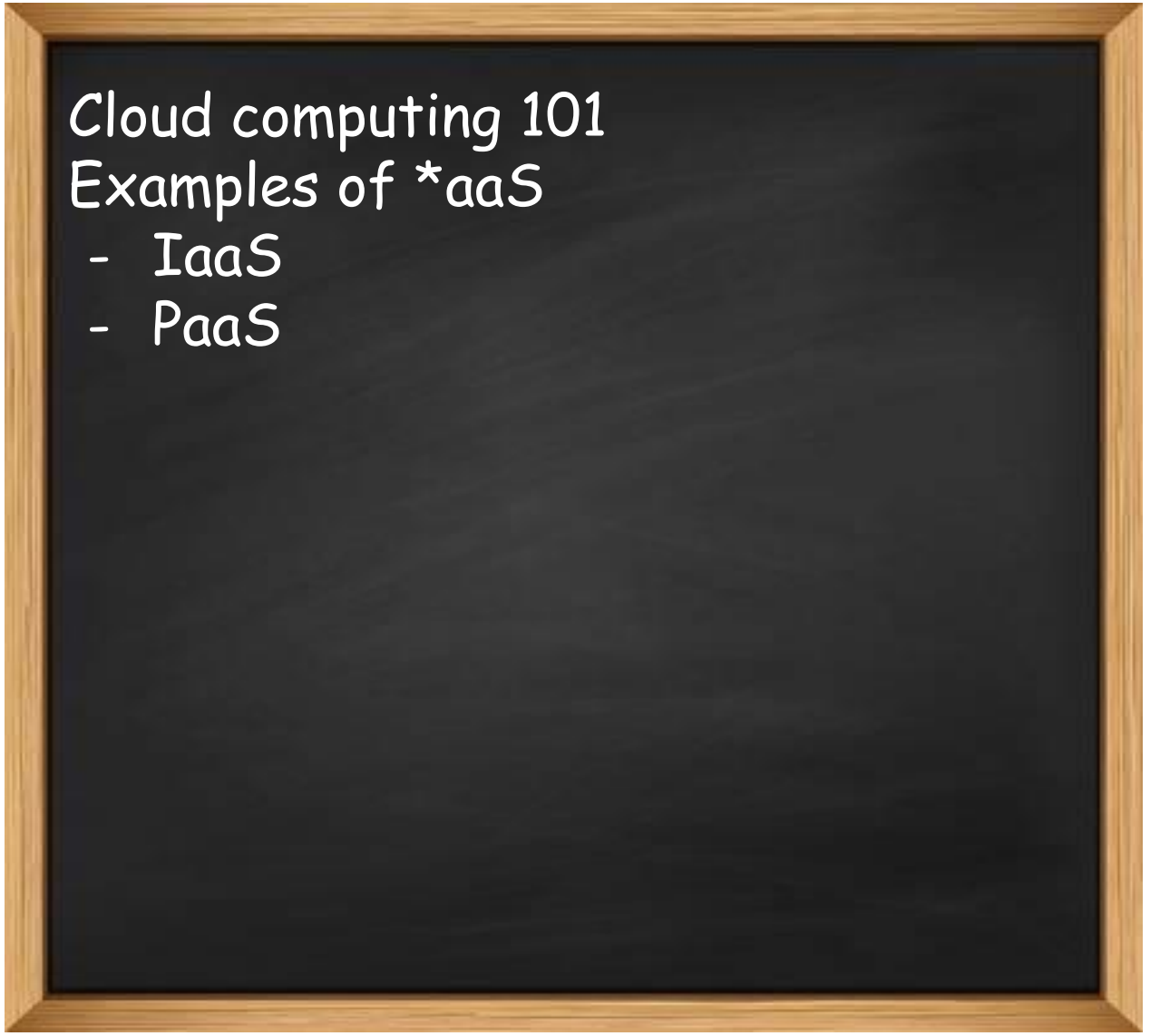


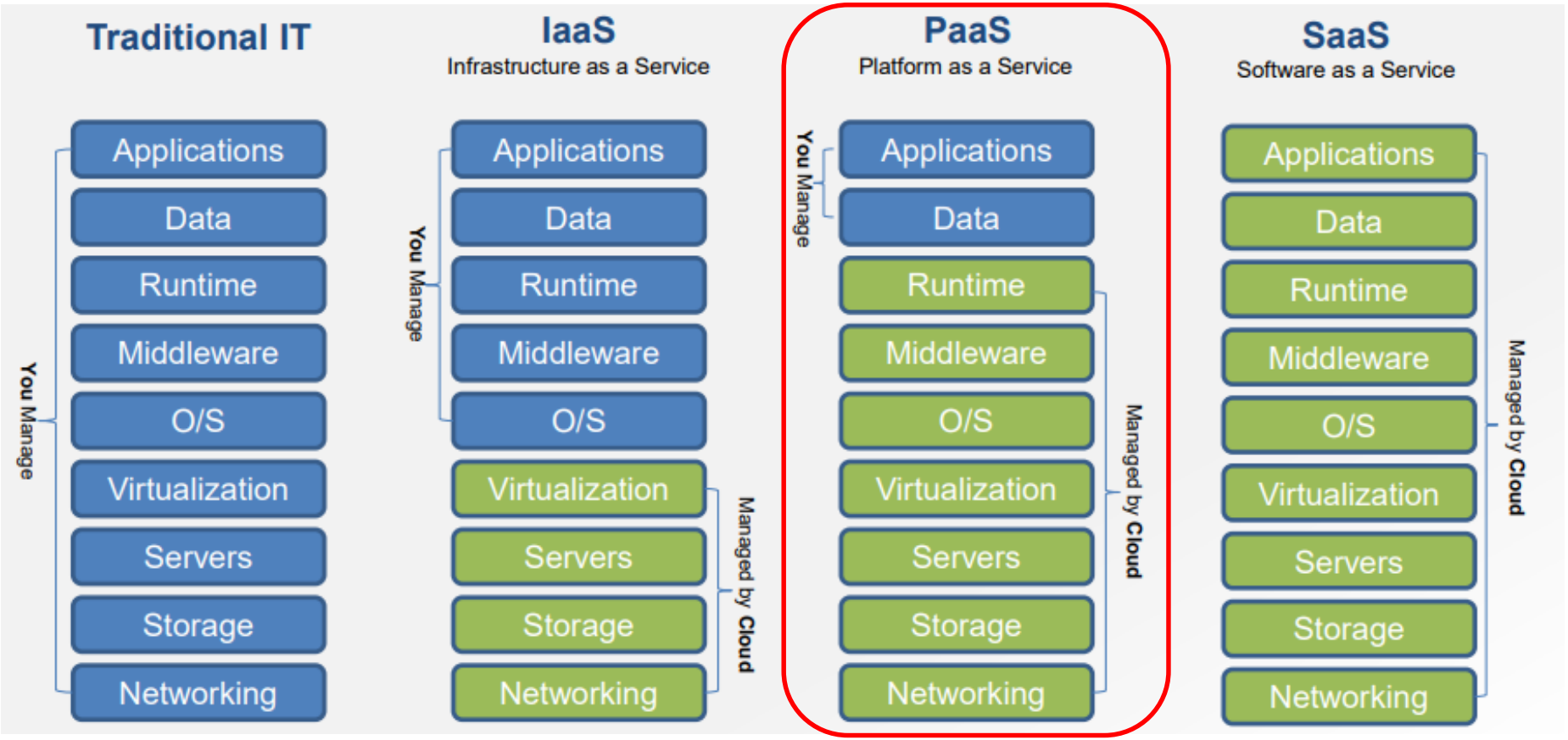

Traditional IT
Applications
Data
Runtime
Middleware
O/S
Virtualization
Servers
Storage
Networking
You Manage
IaaS
Infrastructure as a Service
Applications
Data
Runtime
Middleware
O/S
Virtualization
Servers
Storage
Networking
You Manage
Managed by Cloud
PaaS
Platform as a Service
Applications
Data
Runtime
Middleware
O/S
Virtualization
Servers
Storage
Networking
You Manage
Managed by Cloud
SaaS
Software as a Service
Applications
Data
Runtime
Middleware
O/S
Virtualization
Servers
Storage
Networking
Managed by Cloud

Company	2017 Revenue	2017 Market Share (%)	2016 Revenue	2016 Market Share (%)	2017-2016 Growth (%)
Amazon	12,221	51.8	9,775	53.7	25.0
Microsoft	3,130	13.3	1,579	8.7	98.2
Alibaba	1,091	4.6	670	3.7	62.7
Google	780	3.3	500	2.7	56.0
IBM	457	1.9	297	1.6	53.9
Others	5,902	25.0	5,392	29.6	9.5
Total	**23,580**	**100.0**	**18,213**	**100.0**	**29.5**

Source: Gartner (August 2018)

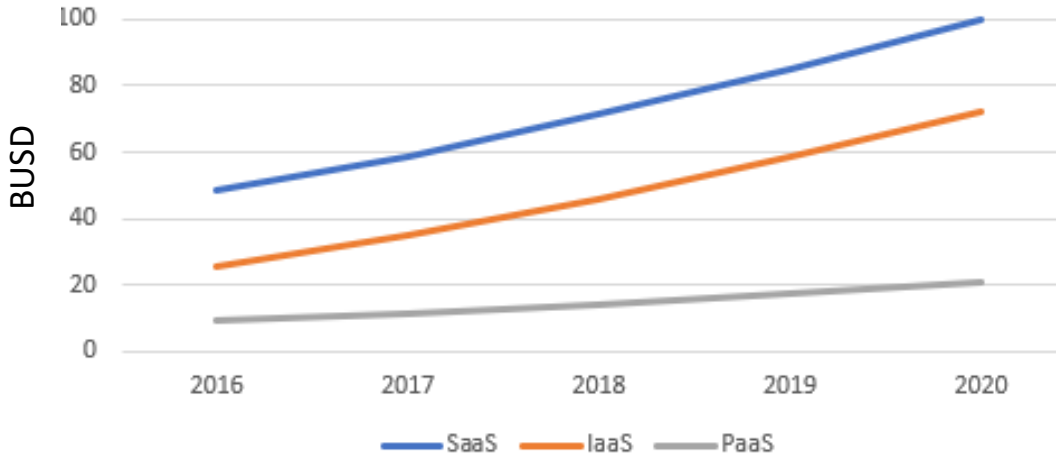
IaaS examples


Cloud computing 101
Examples of *aaS
- IaaS
- PaaS


Traditional IT
Applications
Data
Runtime
Middleware
O/S
Virtualization
Servers
Storage
Networking
You Manage
IaaS
Infrastructure as a Service
Applications
Data
Runtime
Middleware
O/S
Virtualization
Servers
Storage
Networking
You Manage
Managed by Cloud
PaaS
Platform as a Service
Applications
Data
Runtime
Middleware
O/S
Virtualization
Servers
Storage
Networking
You Manage
Managed by Cloud
SaaS
Software as a Service
Applications
Data
Runtime
Middleware
O/S
Virtualization
Servers
Storage
Networking
Managed by Cloud

PaaS ?

Gartner (October 2017)



"Strategic adoption of PaaS offerings is also outperforming previous expectations, as enterprise-scale organizations are increasingly confident that PaaS will be their primary form of application development platform in the future"

Sid Nag, Gartner (October 2017)

Gartner’s PaaS Magic Quadrant

Heroku

Heroku is a cloud platform based on a managed container system, with integrated data services and a powerful ecosystem, for deploying and running modern apps

Born in 2007 (only Ruby supported)

Now polyglot (Node, Ruby, Java, PHP, Pythin, Go, Scala, Clojure)

Developer can build, run and scale apps in a similar manner across all languages

Acquired by Salesforce in 2010 (212 MUSD)

What is Heroku? (by analogy)

Heroku Dynos

Containerization abstracts away the burden of managing hardware or virtual machines

You deploy an app to Heroku, which packages the app's code and dependencies into ***containers***

- lightweight, isolated environments that provide compute, memory, an OS, and an ephemeral filesystem
- typically run on a shared host, yet completely isolated one another

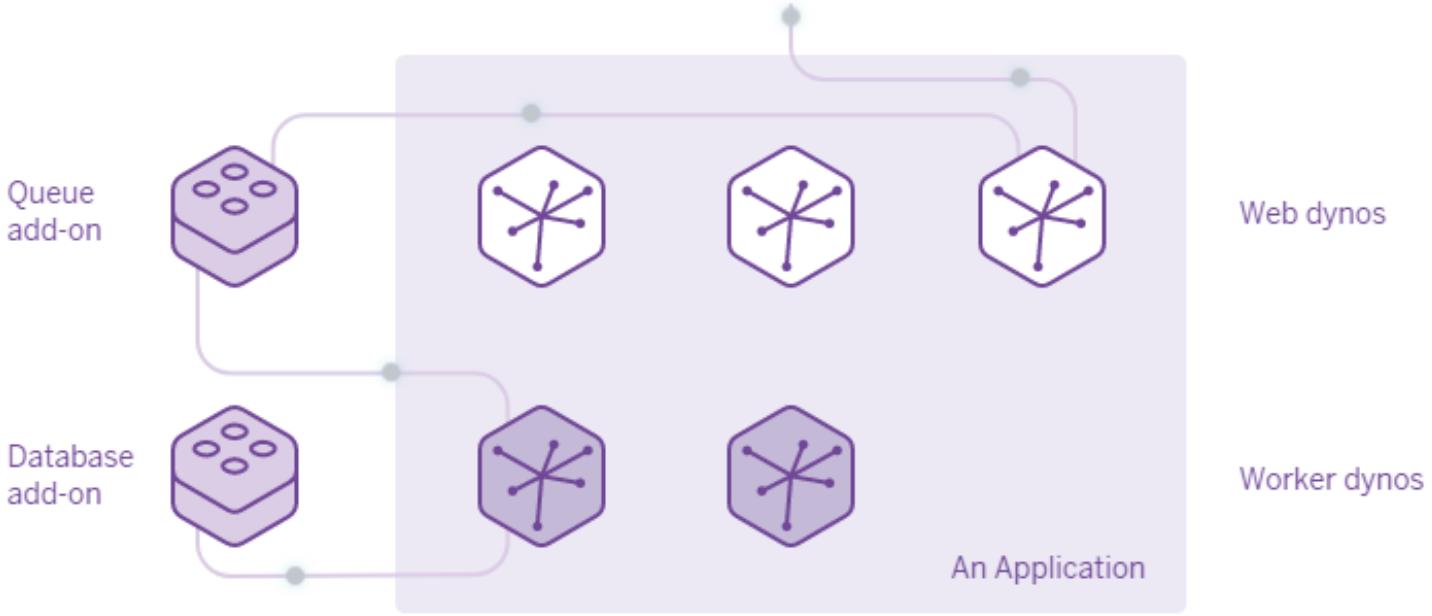
Heroku Dynos

Heroku Platform uses the container model to run & scale all Heroku apps

- Containers at Heroku are called “dynos”
- Dynos are isolated, virtualized Linux containers designed to execute code based on a user-specified command
  - app can scale to any specified number of dynos based on its resource demands
  - easy for user to scale and manage number, size, and type of dynos for app

Deploying to dynos (and relying on Heroku's dyno management) makes it easy to build and run flexible, scalable apps - freeing user from managing infrastructure

Heroku Dynos

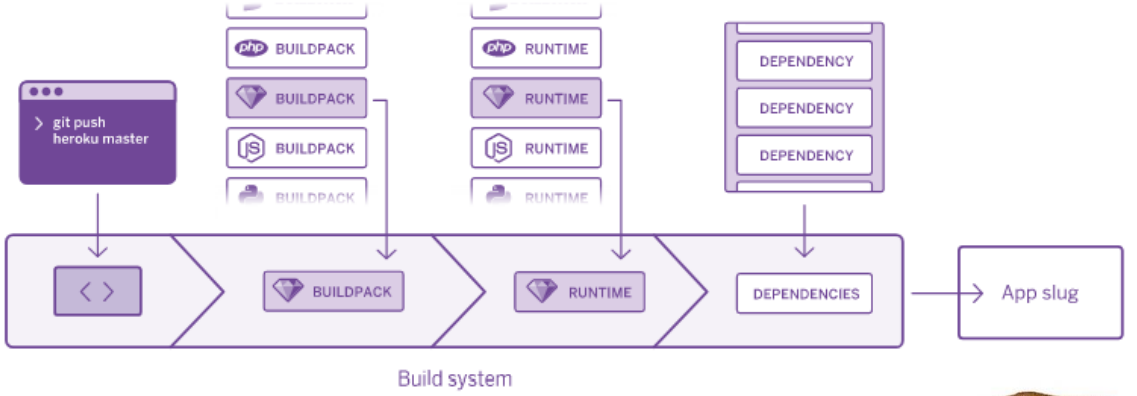


- Application receives request
- Request delivered to random Web dyno
- Request placed in queue (success message returned to user)
- Worker dyno picks up request and does the work, can persists result in database

Buildtime

To deploy an app, Heroku needs only three things from the developer:

- source code
- a list of dependencies
- a “Procfile” (text file indicating which command to use to start the code running)



The automated build system

- receives your code
- fetches a buildpack, language runtime, and code dependencies
- produces a slug
  - a bundle of source, dependencies, runtime, output, that is injected into a dyno to run your app

The final component needed to run app is the OS – on Heroku called the "stack", an Ubuntu operating system image maintained by Heroku

Runtime

When you deploy or scale your app, Heroku will automatically create one or more dynos, each loaded with the same stack and slug representing your app

Heroku's Dyno Manager then executes the command you provided in your configuration file to start your application running on Heroku

Heroku enables developers to fine-tune their app’s runtime resources by choosing from a broad range of dyno

- types
  - free, hobby
  - standard (h-scalability)
  - performance (h-scalability, autoscaling)
- and configurations (web/worker/one-off)

Heroku Add-ons

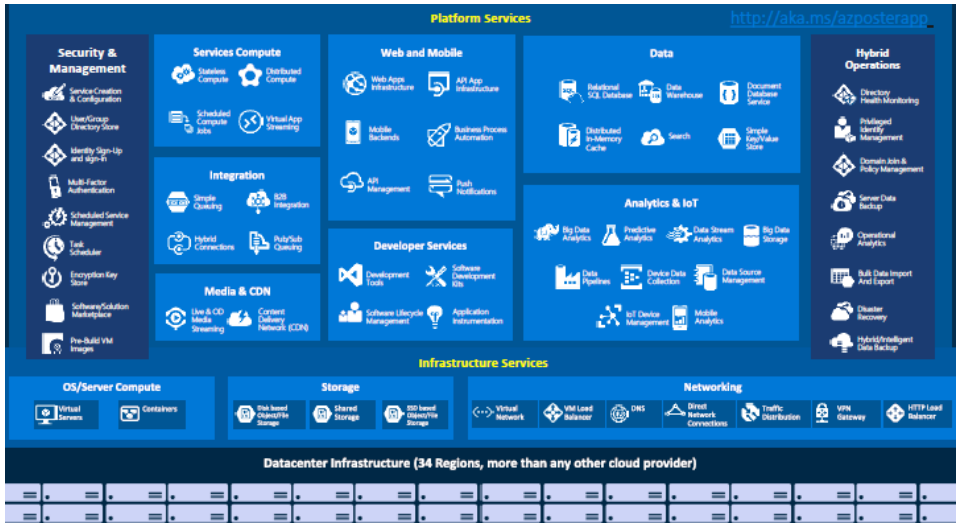
150+ 3rd party cloud services that developers can use to immediately extend their apps with a range of functionality such as data stores, logging, monitoring and more

Heroku provides three fully-managed data service Add-ons: Heroku Postgres, Heroku Redis, and Apache Kafka on Heroku

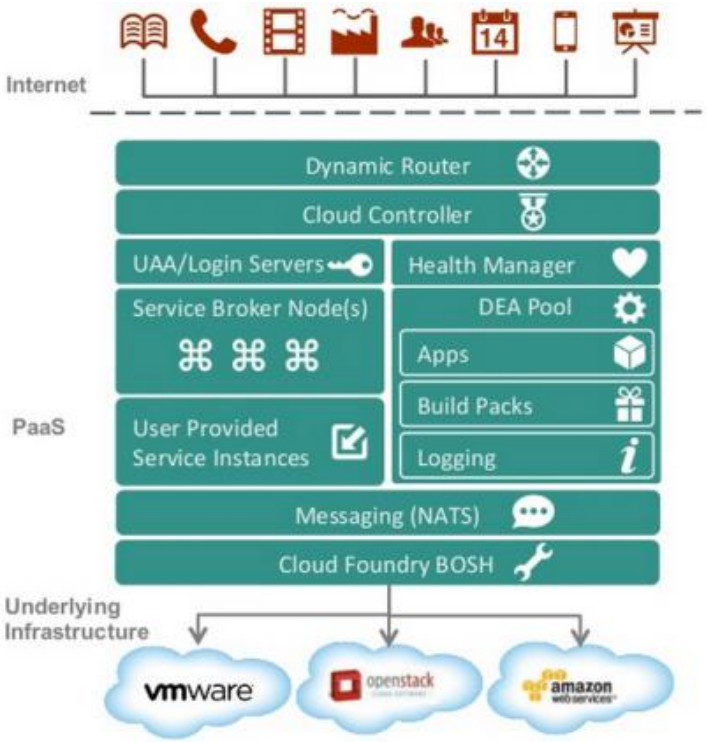
ADD-ON CATEGORIES

- Data Stores
- Data Store Utilities
- Monitoring
- Logging
- Email/SMS
- Caching
- Errors and Exceptions
- Content Management
- Search
- Metrics and Analytics
- Testing
- Messaging and Queueing
- Network Services
- Alerts and Notifications
- User Management
- Development Tools
- Security
- Dynos
- Content
- Document Processing
- Image Processing
- Video Processing
- Deployment
- Utilities

Others ...

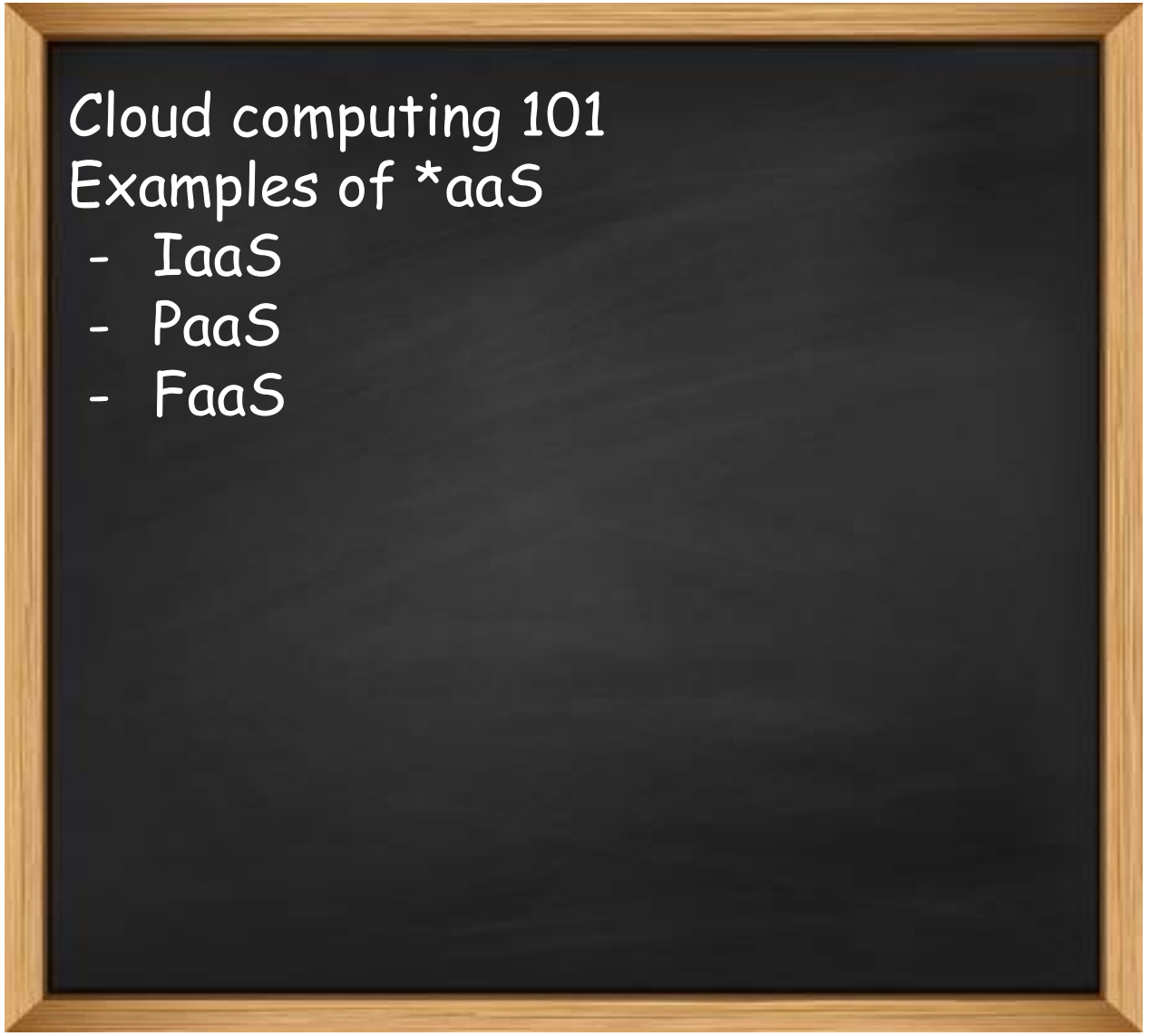


Microsoft Azure



Cloud Foundry

OPENSHIFT


Cloud computing 101
Examples of *aaS
- IaaS
- PaaS
- FaaS

AWS Lambda

AWS Lambda lets you run code without provisioning or managing servers. You pay only for the compute time you consume - there is no charge when your code is not running. With Lambda, you can run code for virtually any type of application or backend service - all with zero administration. **Just upload your code and Lambda takes care of everything required to run and scale your code with high availability.** You can set up your code to automatically trigger from other AWS services or call it directly from any web or mobile app.

AWS Lambda

AWS Lambda

Easy to use

- Upload your lambda function / design it in AWS IDE / select from list of pre-built samples
  → AWS SDK makes it is easy to call other AWS services!
- Select the event source to monitor (e.g., S3 bucket)

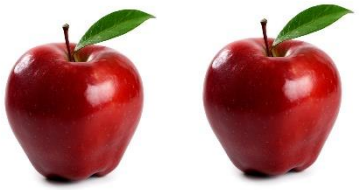
The service

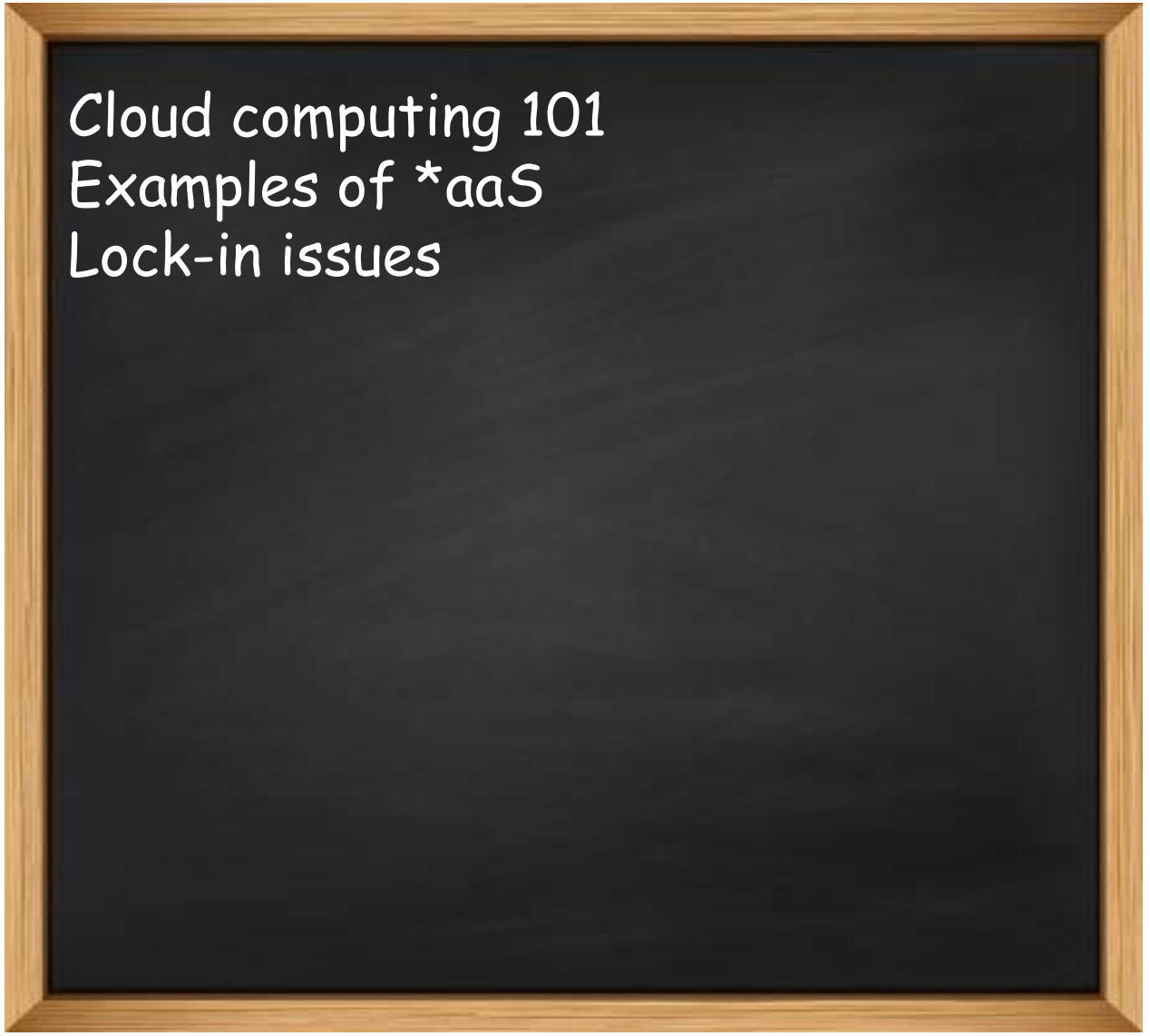
- triggers your function automatically when an event occurs
- handles all capacities, scaling, patching and admin of the infrastructure to run your code – and publishes real time metrics and logs
- with a low cost service (low fee per request)

Google cloud functions

Microsoft Azure functions

IBM Cloud Functions




Cloud computing 101
Examples of *aaS
Lock-in issues

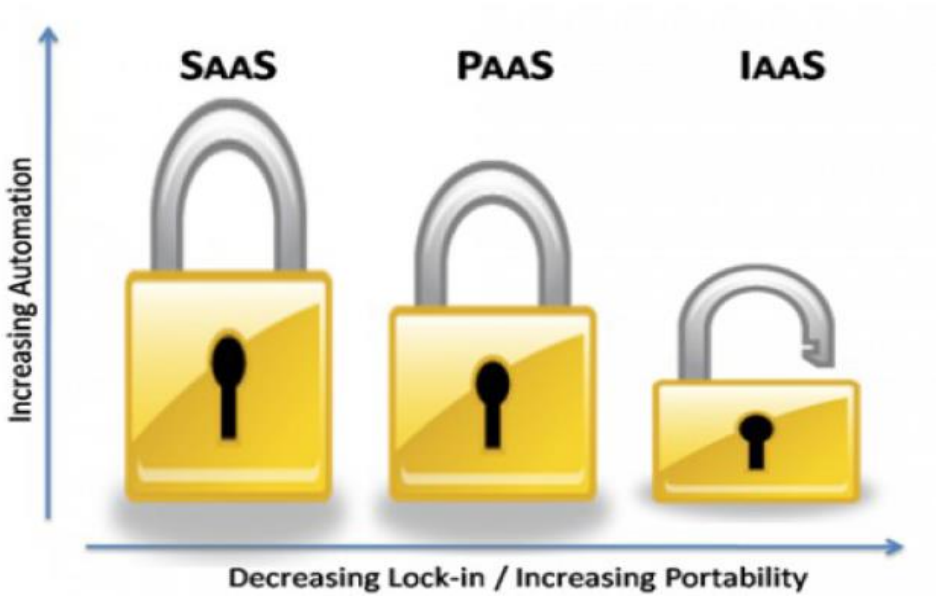
The lock-in problem

Definition - Vendor lock-in makes a customer dependent on a vendor for products or services, unable to use another vendor without substantial switching costs.

Thorsten's Lock-in Hypothesis

The higher the cloud layer you operate in,

the greater the lock-in.



Vendor lock-in?

We may wish to change provider for various reasons (e.g., price, QoS)

… but changing provider may get expensive due to ecosystem dependencies (e.g., «data gravity», portability issues -especially with add-ons)

Tips to avoid/reduce lock-in

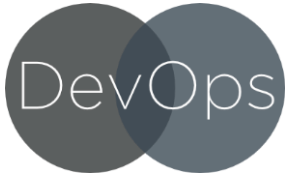
"Amen"

"Carefully choose provider"
"Date providers don't marry them"
"Think carefully before using proprietary cloud-vendor services"

"Reconsider the way you think about purchasing services"
"Plan your exit strategy"

"Ensure portability of data"
"Use unified interfaces"
"Choose open standards and open-source technologies"

"Use multiple clouds"



"Use loosely coupled architecture, API/REST integration"
"Use microservices architectures"
"Use containers and devops tools"

Consortium standards

CDMI (Cloud Data Management Interface) ISO //cloud storage

OCCI (Open Cloud Computing Interface) //IaaS management

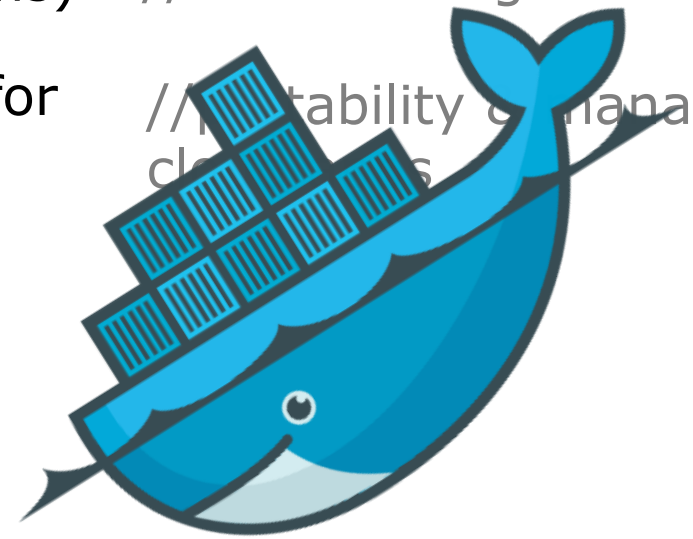
CIMI (Cloud Infrastructure Management Interface) //IaaS management

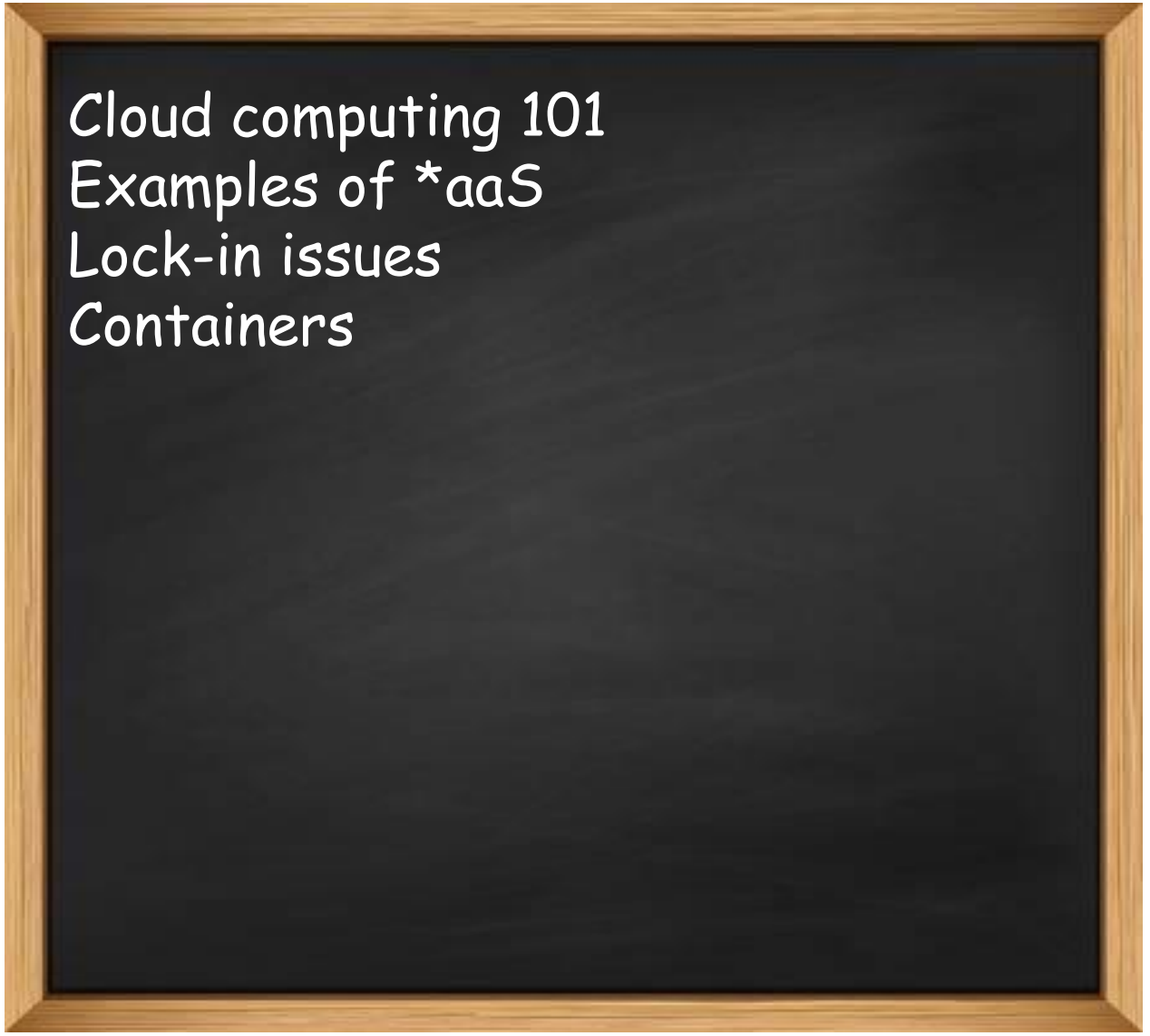
OVF (Open Virtualization Format) ANSI //sw packaging & distribution

CAMP (Cloud Application Management for Platforms) //PaaS management

TOSCA (Toplogy and Orchestration Specification for Cloud Applications) //portability & management of cloud apps

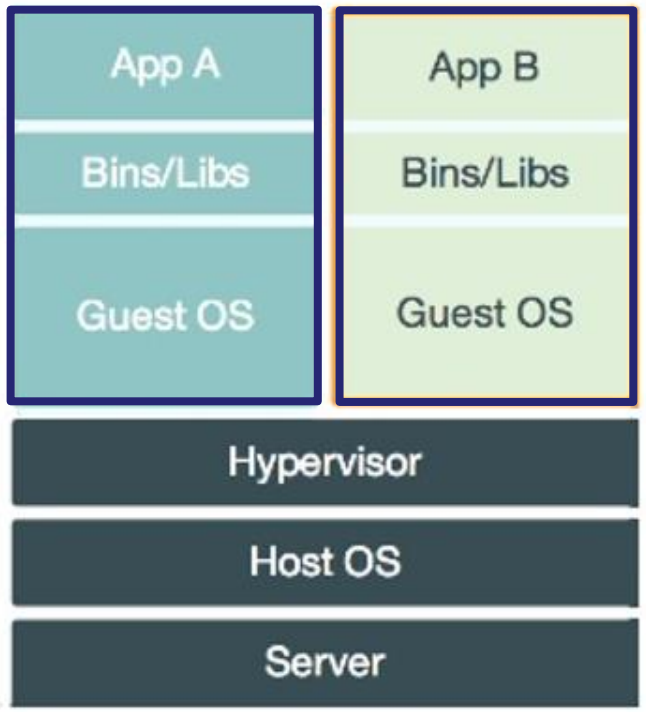
…

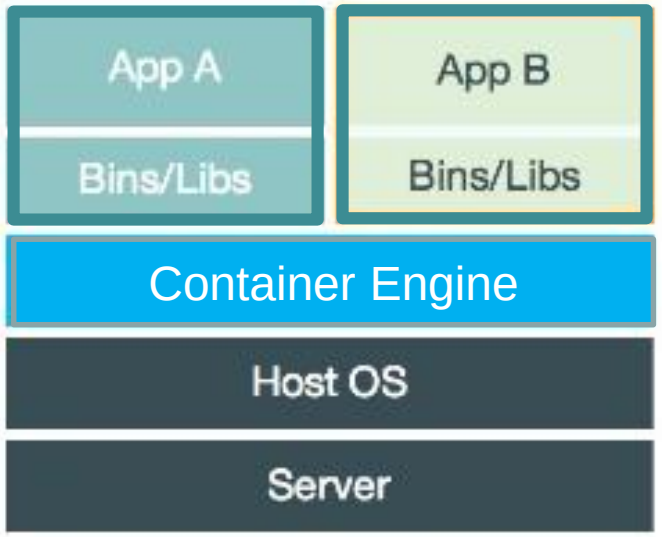



Cloud computing 101
Examples of *aaS
Lock-in issues
Containers

OS kernel permits to have multiple isolated user-space instances (a.k.a. **containers**)

VMs *vs.* Containers





+ lighter (less resources)
+ faster to start
+ simpler to build
- less secure (less isolation)

How old are containers?

For decades, UNIX *chroot* command provided a simple form of filesystem isolation

1998 – FreeBSD *jail* utility extended *chroot* sandboxing to processes

2005 – Google started developing *CGroups* for Linux kernel and began moving its infrastructure to containers

2008 – Linux Containers (LXC) provided a complete containerization solution

2013 - Docker added the missing pieces - *portable images* and *friendly UI* – to the containerization puzzle, and containers entered the mainstream

The Docker platform consisted of:

- **Docker Engine** (for creating and running containers)
- **Docker Hub** **(**for distributing containers)

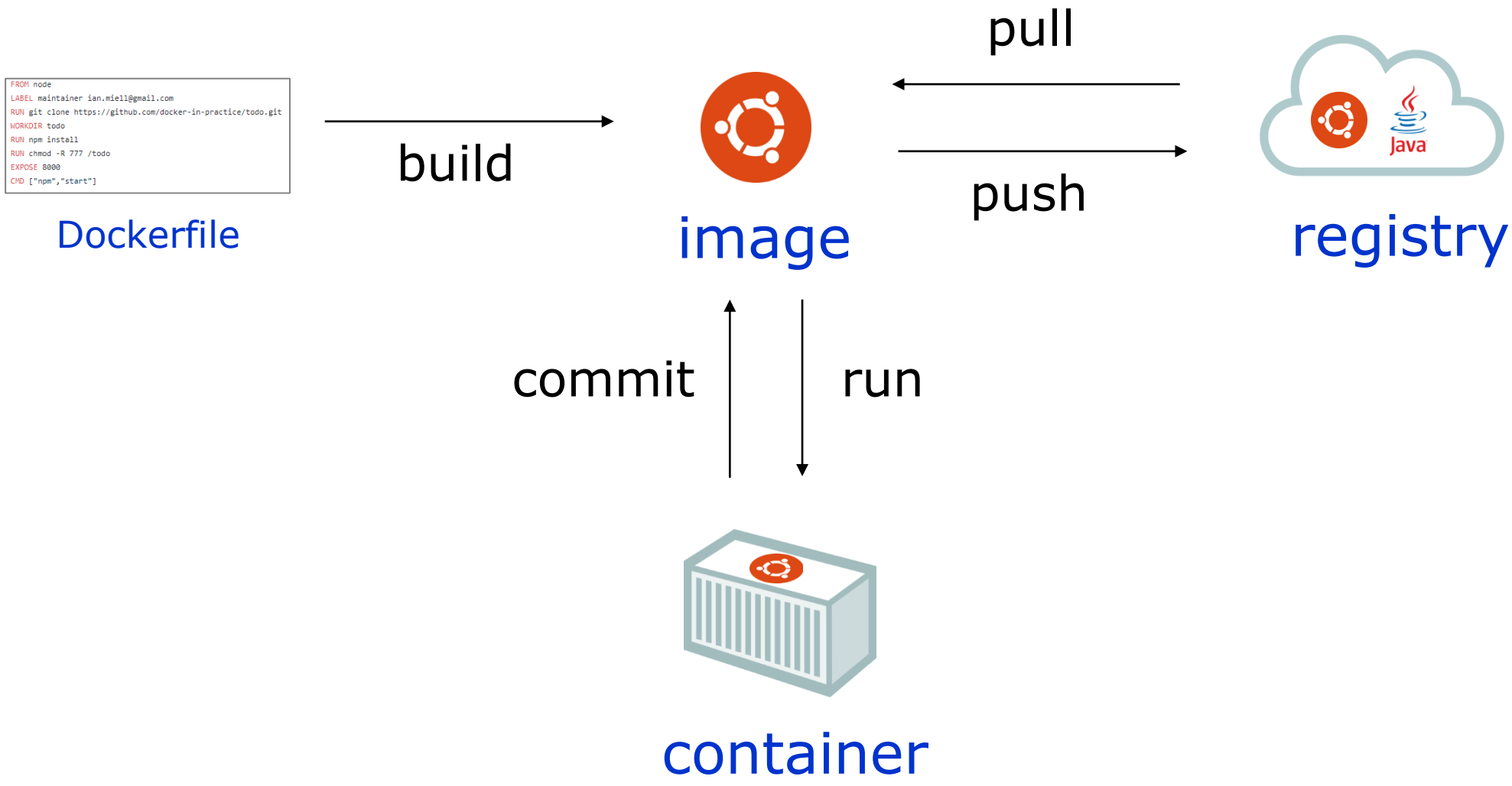
Docker

Docker exploits container-based virtualization to run multiple isolated guest instances on the (same) OS

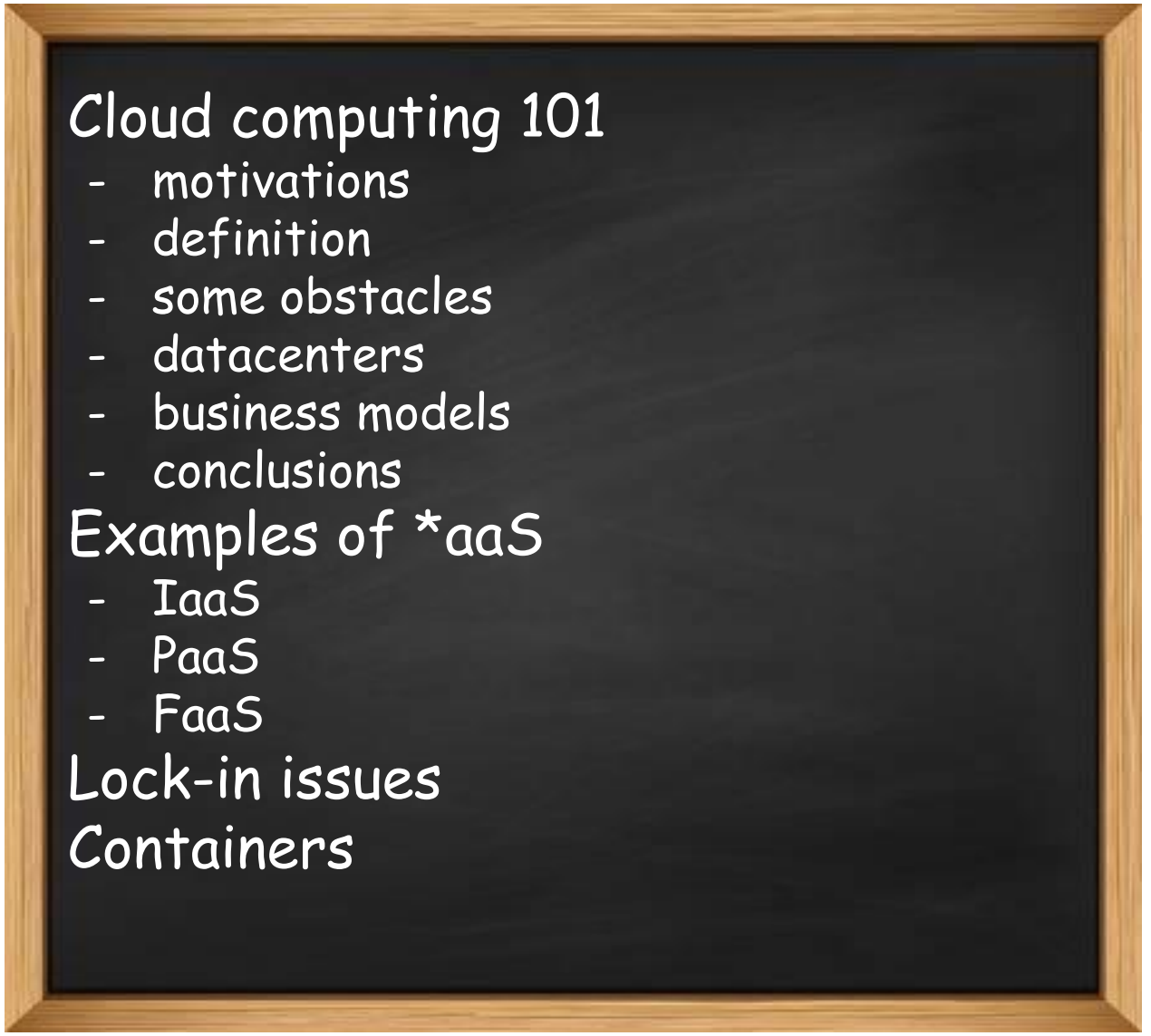
Software components are packaged into ***images***, which are exploited as read-only templates to create and run ***containers***

External ***volumes*** can be mounted to ensure data persistence

“Build, ship, and run any app, anywhere”


FROM node
LABEL maintainer ian.miell@gmail.com
RUN git clone https://github.com/docker-in-practice/todo.git
WORKDIR todo
RUN npm install
RUN chmod -R 777 /todo
EXPOSE 8000
CMD ["npm","start"]
Dockerfile
build
image
pull
push
registry
Java
commit
run
container

Learn Docker in 12 min


Cloud computing 101
- motivations
- definition
- some obstacles
- datacenters
- business models
- conclusions
Examples of *aaS
- IaaS
- PaaS
- FaaS
Lock-in issues
Containers