20.000 regs
de Viaje submarino

Bienvenido - Welcome - Witam

Juanmi Taboada
@juanmitaboada
www.juanmitaboada.com

OpenSouthCode'19 - Málaga, 25 may 2019

1996

1997

1999

UNIVERSIDAD DE MÁLAGA | uma.es

1999

grupo de estudios en biomimética

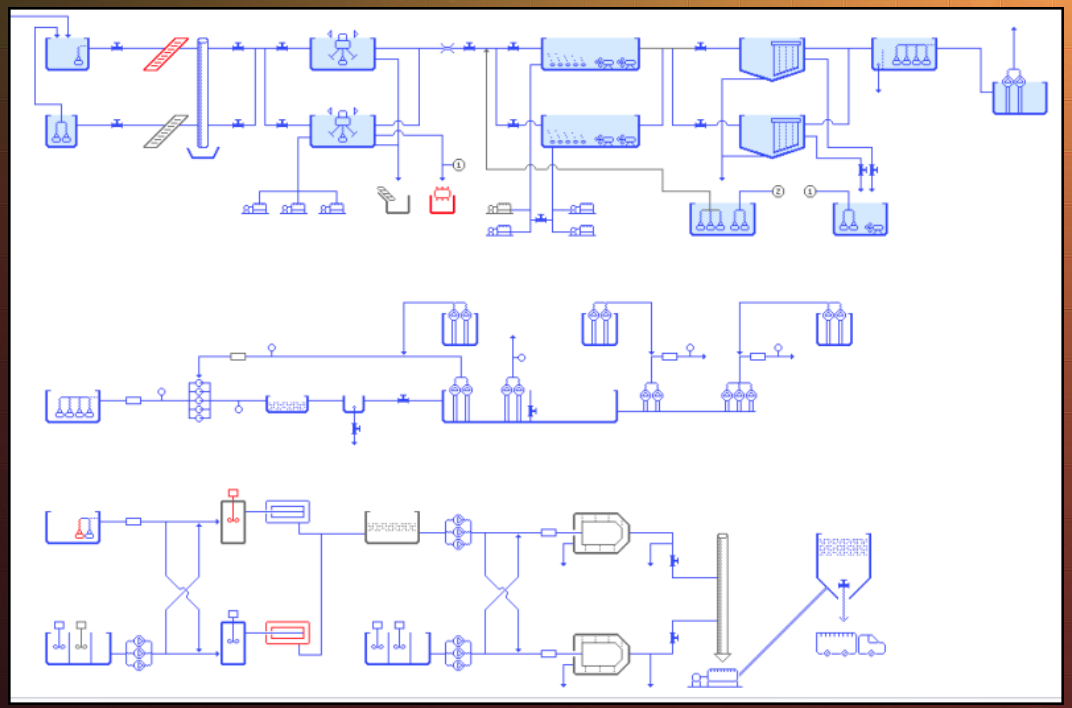
2001

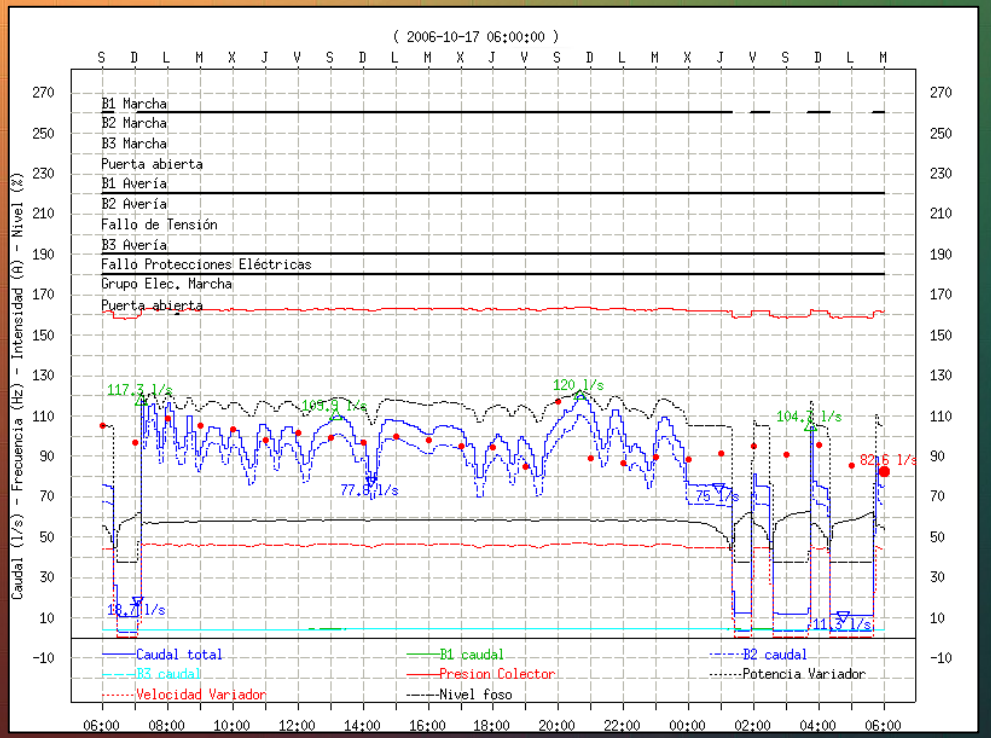
axaragua

2005

python

Likindoy
The monitoring company





juanmitaboada.com

python SOFTWARE FOUNDATION

2008

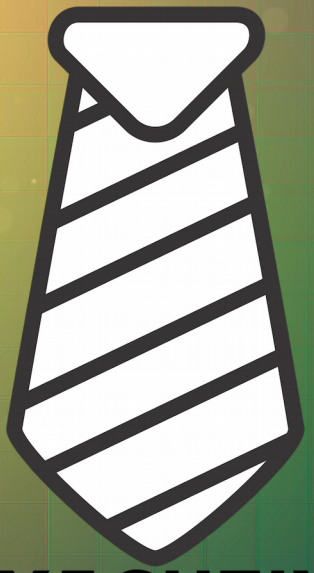
django

Seguros

Webservices

Fotovoltáica

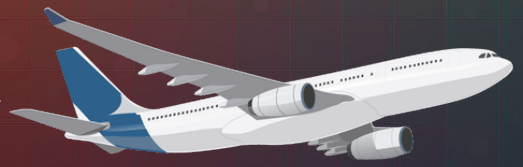
Adquisición Datos



EXECUTIVE

MBA

Aeronáutica



- Hoy -

20.000 regs/seg

o más … porque no los contamos

¿Información?

Información

es un conjunto organizado de datos procesados,

que constituyen un mensaje

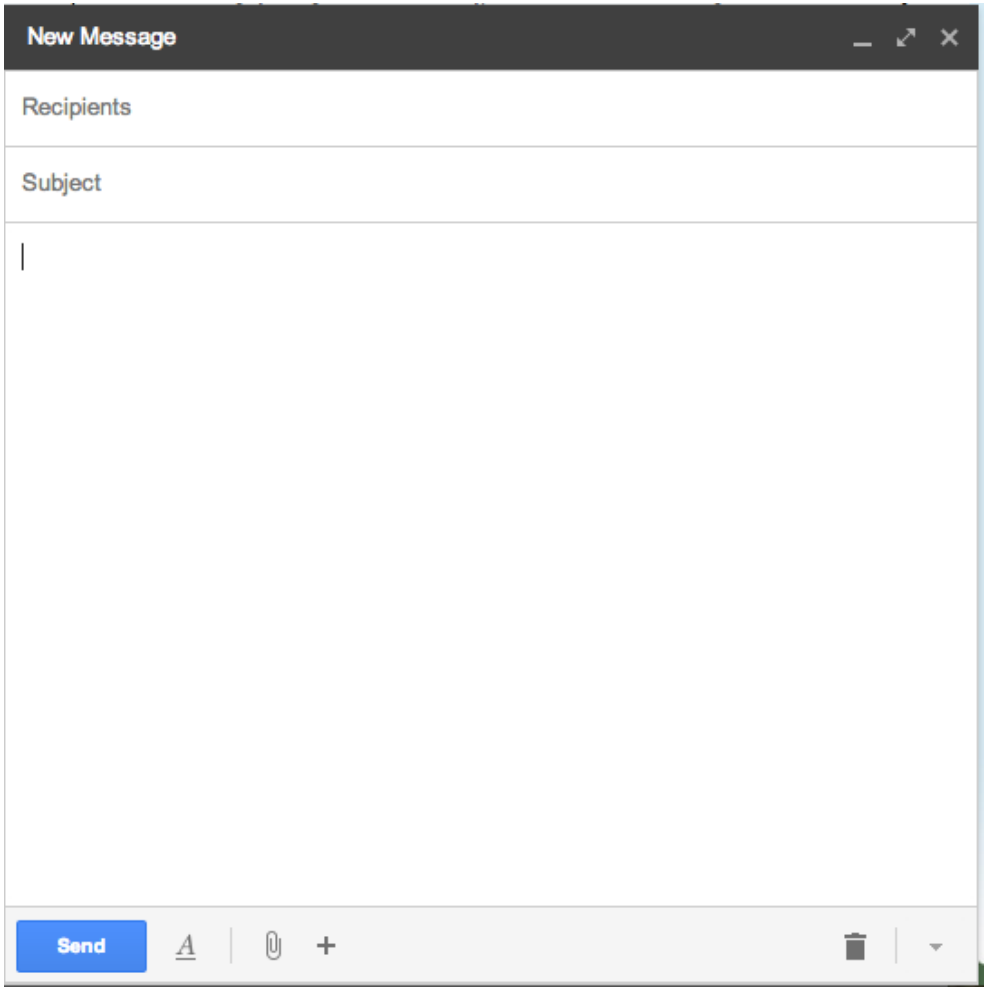
que cambia el estado de conocimiento

del sujeto o sistema que recibe dicho mensaje.

(fuente Wikipedia)

Información digital

Información digital → Email



Información digital → Email

New Message

Recipients

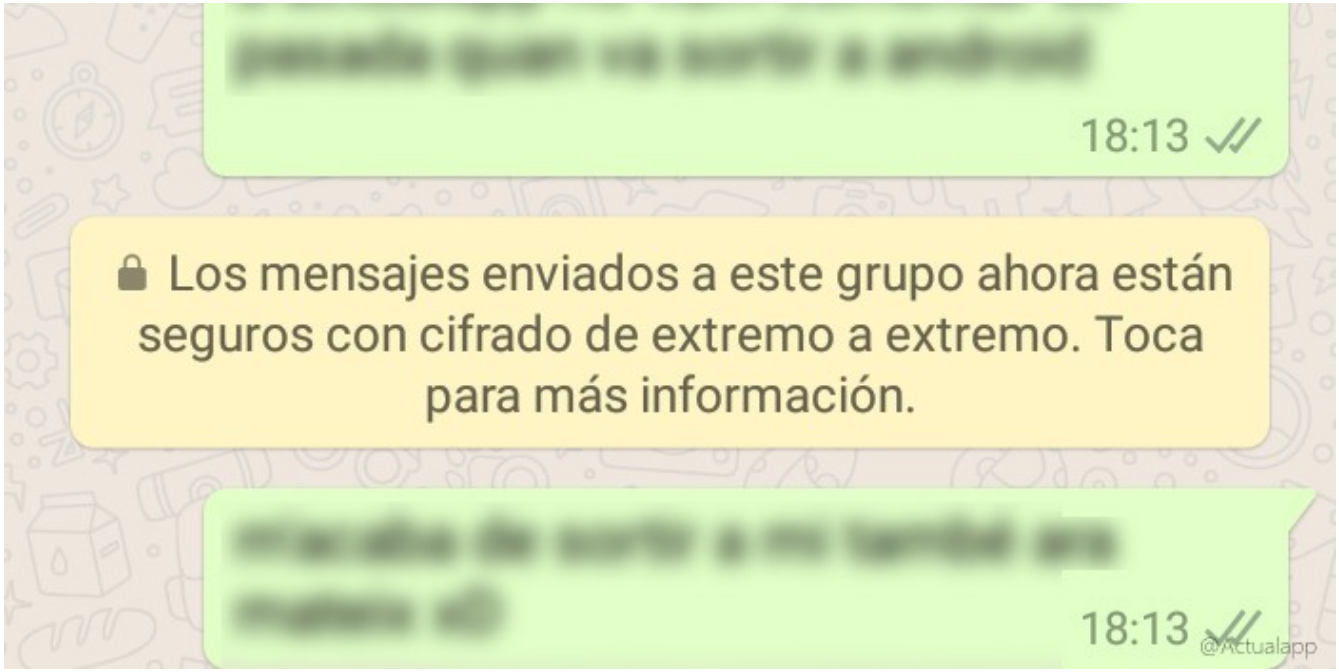
Subject

-----BEGIN PGP MESSAGE-----
Version: OpenPGP.js v.1.20130712
Comment: http://openpgpjs.org

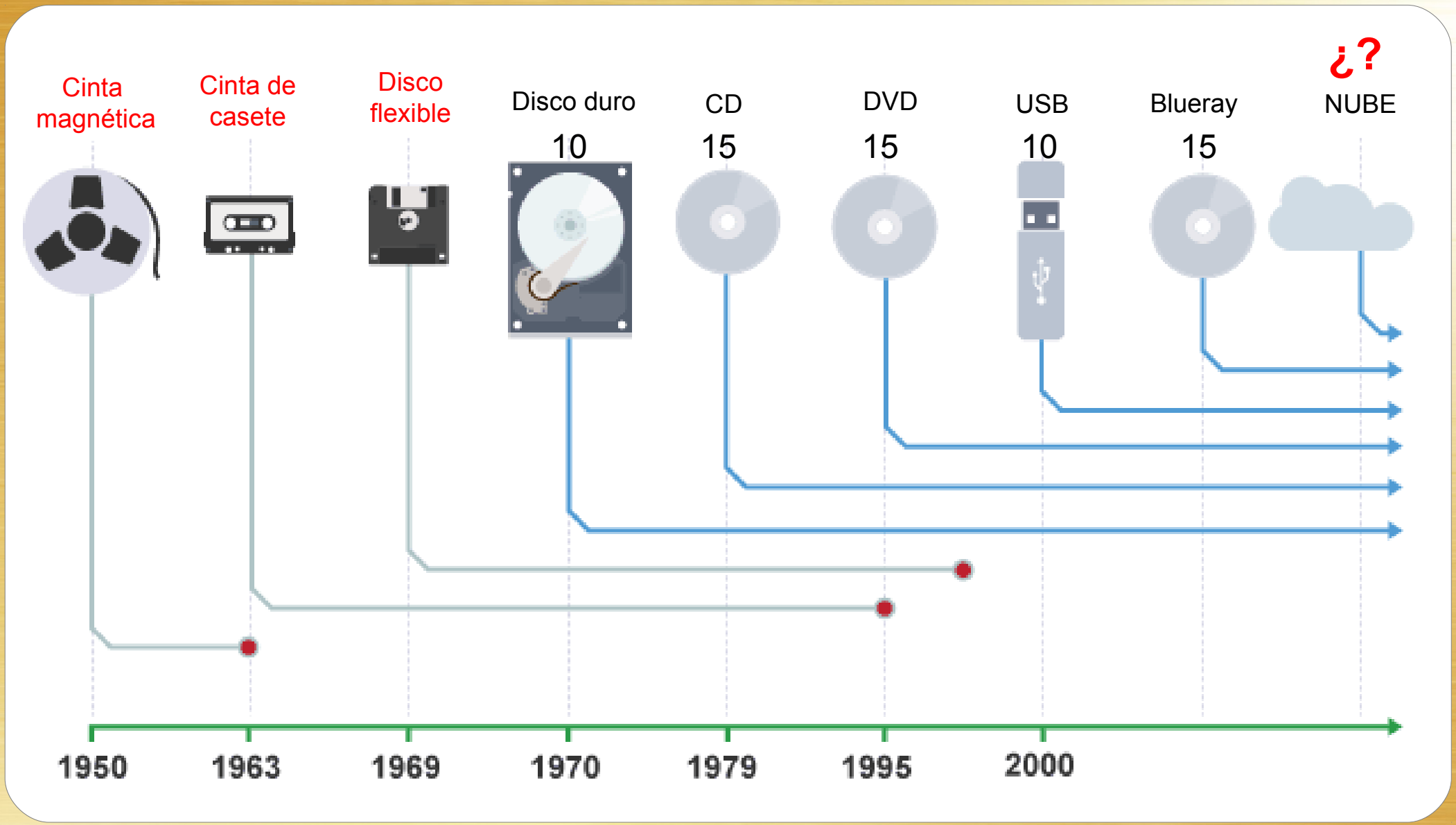
wYwDzoryw22LIbsBBACgY0920+TBs9wr3imXPonZ/sul3z1nTwMVfqZI7BHb
49KIbqKwePlQnLZeu2ckfybJsC4enHQTzn8ZTJlvlETHf/o0plEF87jf82Ml
mRDekE1fcdj5consy2VFAsJahTHOOfBxoLk72m7iuPK0mf0SgN8hX+R53MzJ
1RC+tBoafcHATAOp8CsqJMp8SgElALrtDdwuq0eCqjw7Jv3cAf7OBVxC/74F
rCDWl1FC6Tl3lBVXlrZ2182/GGND+yvWXyZsP5vBAdVIkv+mbApsvvCWYc5p
oybkxpuCcT1IWBQRAkZtdePNXTkNH6yHiagkOdYLoGsTl9ThZl4jHtY8DJ2u
+iFO3tsAY3UD0zeLpIoLmmvwMOFfEb6DUVdU1Uglko4uMxG6qAvc6dTLGmPQ
Y/VKjNw5TwGOFeKFAAgSgmkr7eAcmnD7cn2p9fKJpGq50BWgi6cOoMxZth0Z
8ClP7n2CbHQFQUvK1F/alrLhDQ7NClnCosOYMwF2VmPYP/AU1DH/8khdGGCe
jusKjBS/nnvSrQFdViFOGgcEFvTTzrCb1mKR1lmCwB2DlUreclmXpGzs9aWr
uHlqjuq2BjagW9qOD3+V5KIVt86CaUjlAAISnqW08nRZVWe/dktDhE7fvfUx
VBZfX+CFLoK17rNxfKdnnrB/PbW4zqL5w1erO9A6QYMXOqY3BLbYjA3eyzs9
JPMvUsAeSjK8CECt7uV8qv348CisGZT0nfQIXuE4H+0qnMS1sUKHPF5m6X7b
Yflg
=D7x6
-----END PGP MESSAGE-----

Información digital → Audio y Video

Información digital → Email

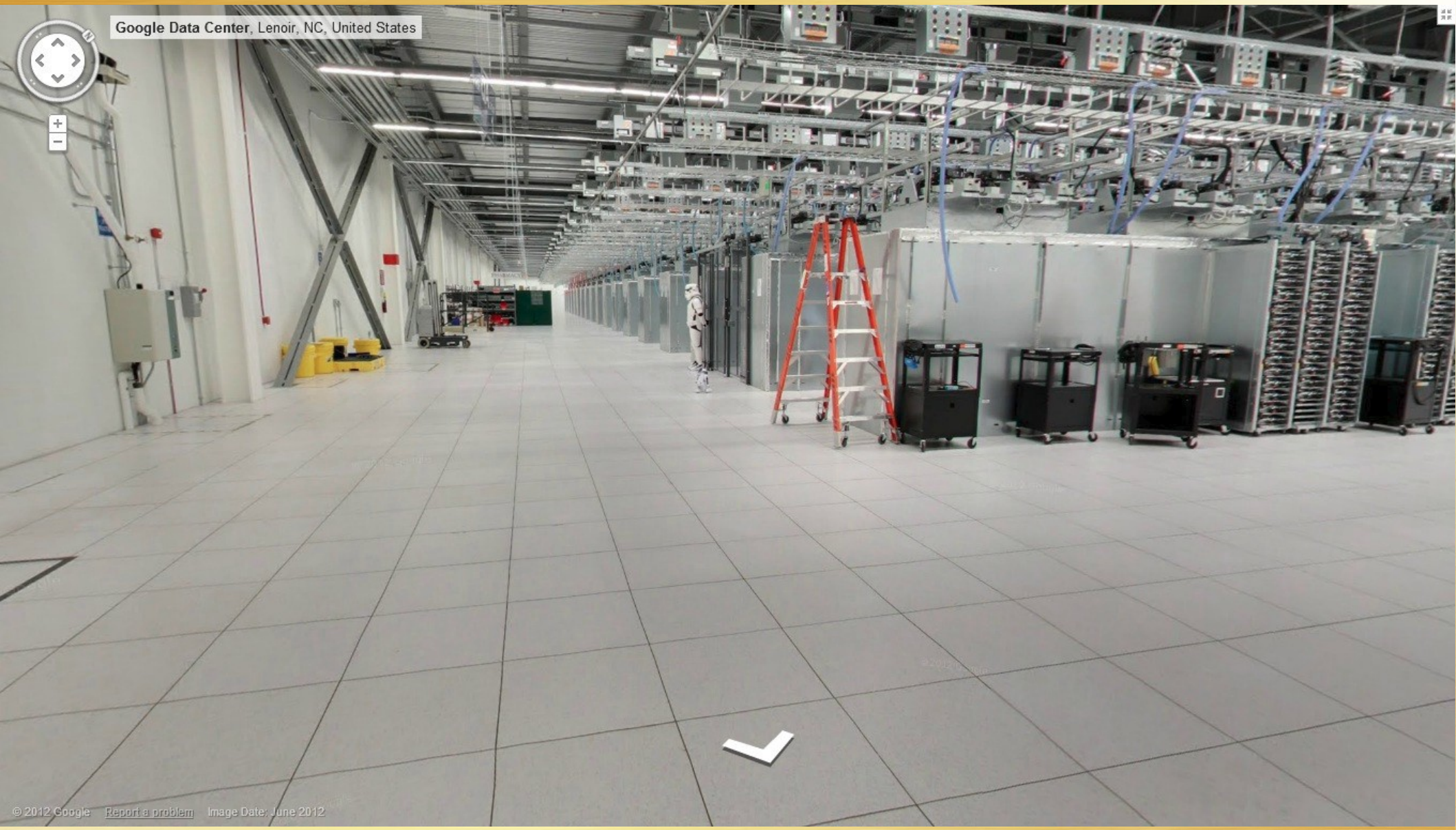


¿Información?

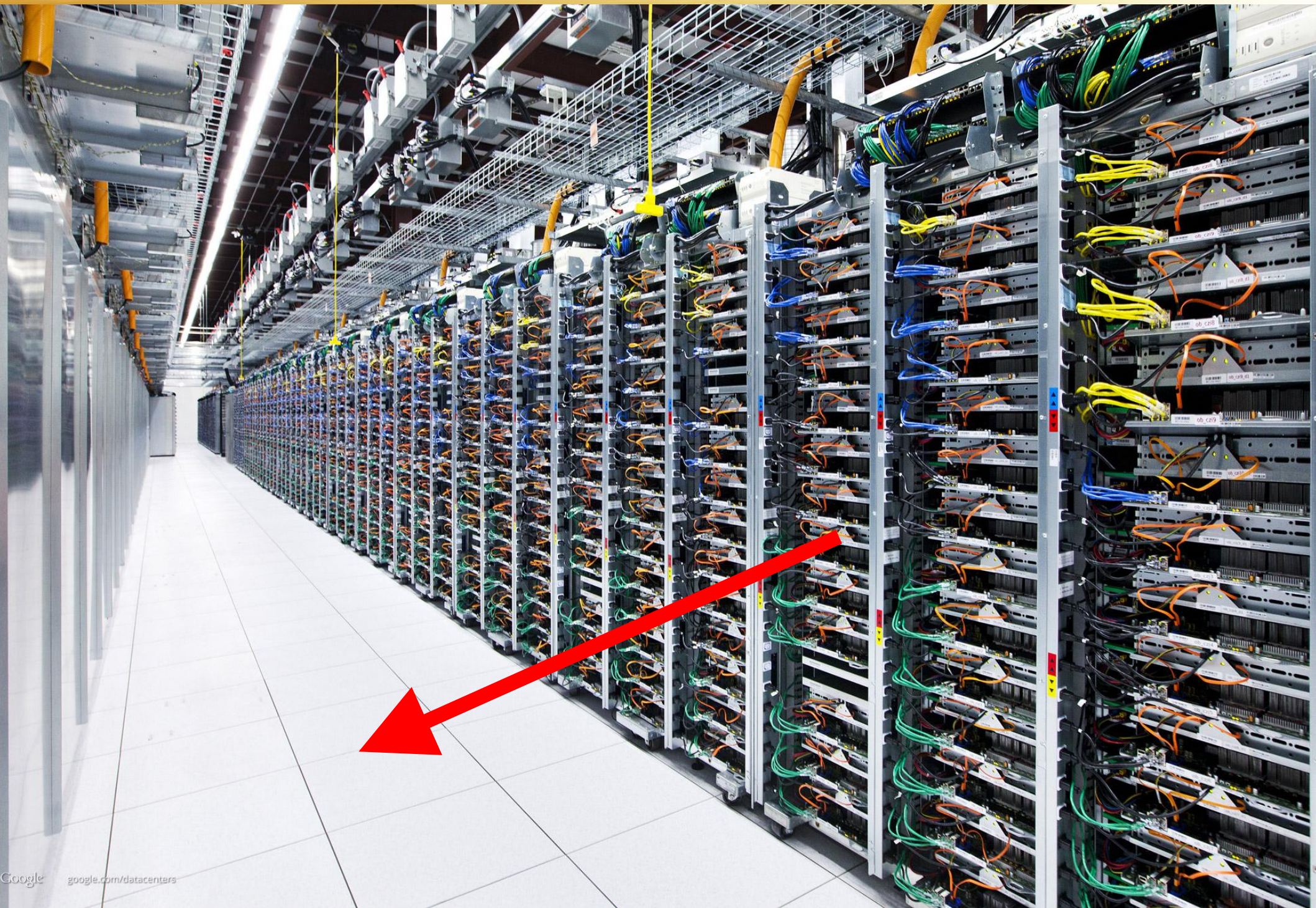



La NUBE

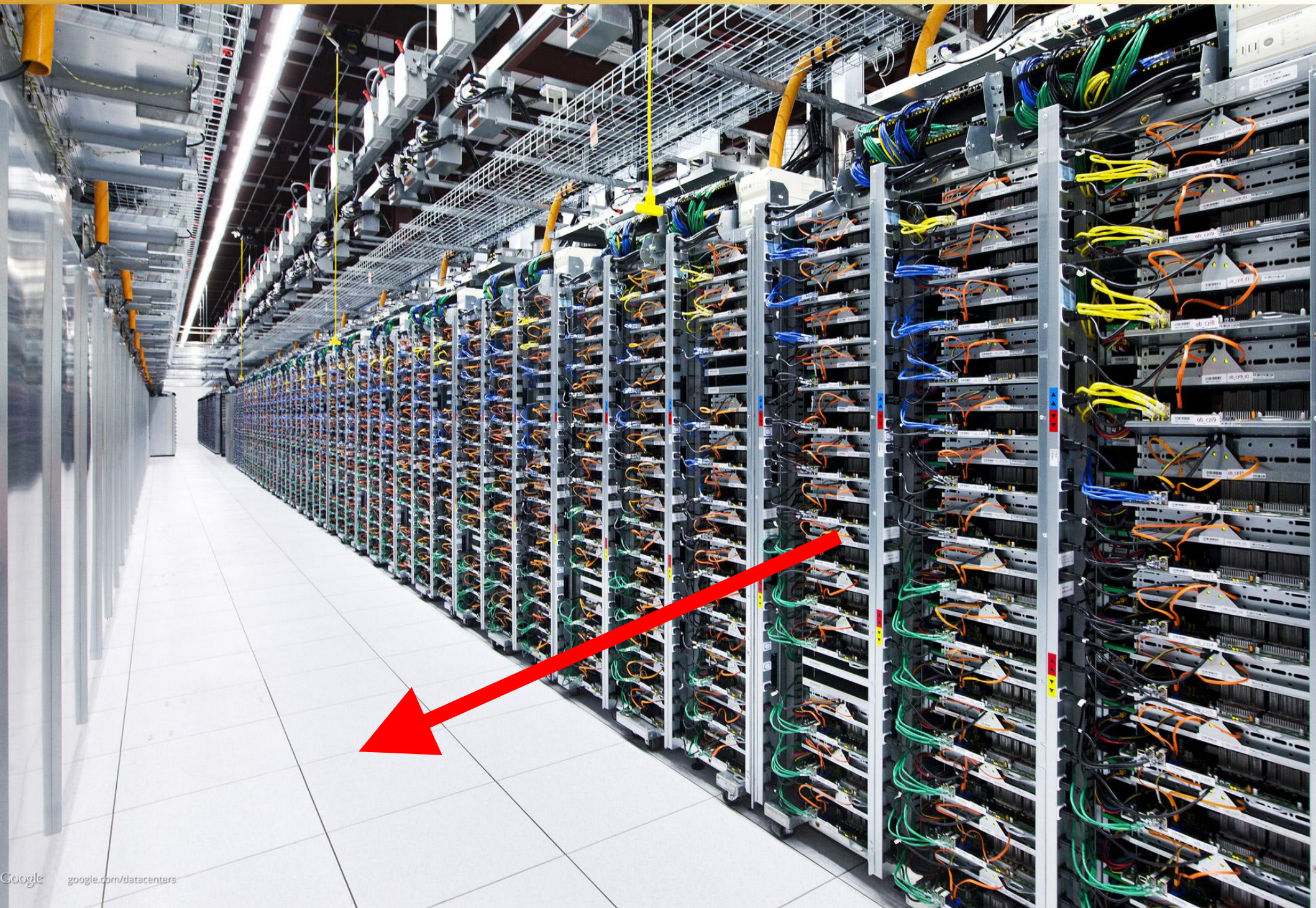
La NUBE: Datacenter



La NUBE: saquemos un equipo



La NUBE: saquemos un equipo



La NUBE: abramos el equipo

La NUBE: interior del ordenador

1 disco duro = 10 años = 3 TBytes

10 años

¡Sincronicemos todo el tiempo!

¿Cómo pinta todo esto?

¿Por qué?

Monitorización

Juanmi Taboada

Big Data

La cruda realidad

Recibimos :

- 1 registro tiene 100 datos útiles (18 campos)
- 20.000 registros por segundo

1.200.000 registros por minuto

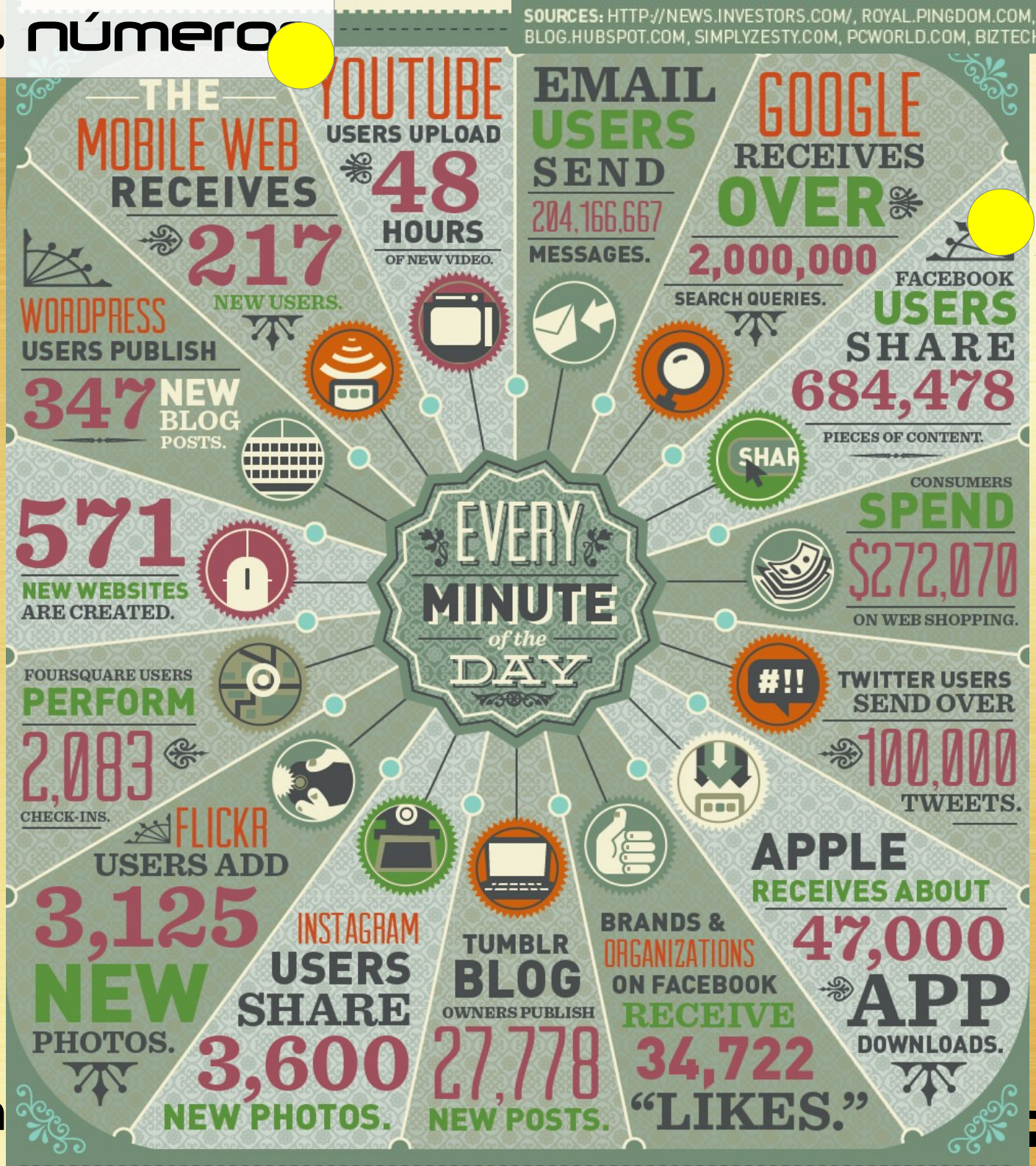
(120.000.000 datos útiles por minuto)

- 72M por hora
- 1.728M por día

0,6 Billones por año

Grandes números

2012
2.1 Billones de usuarios en Internet



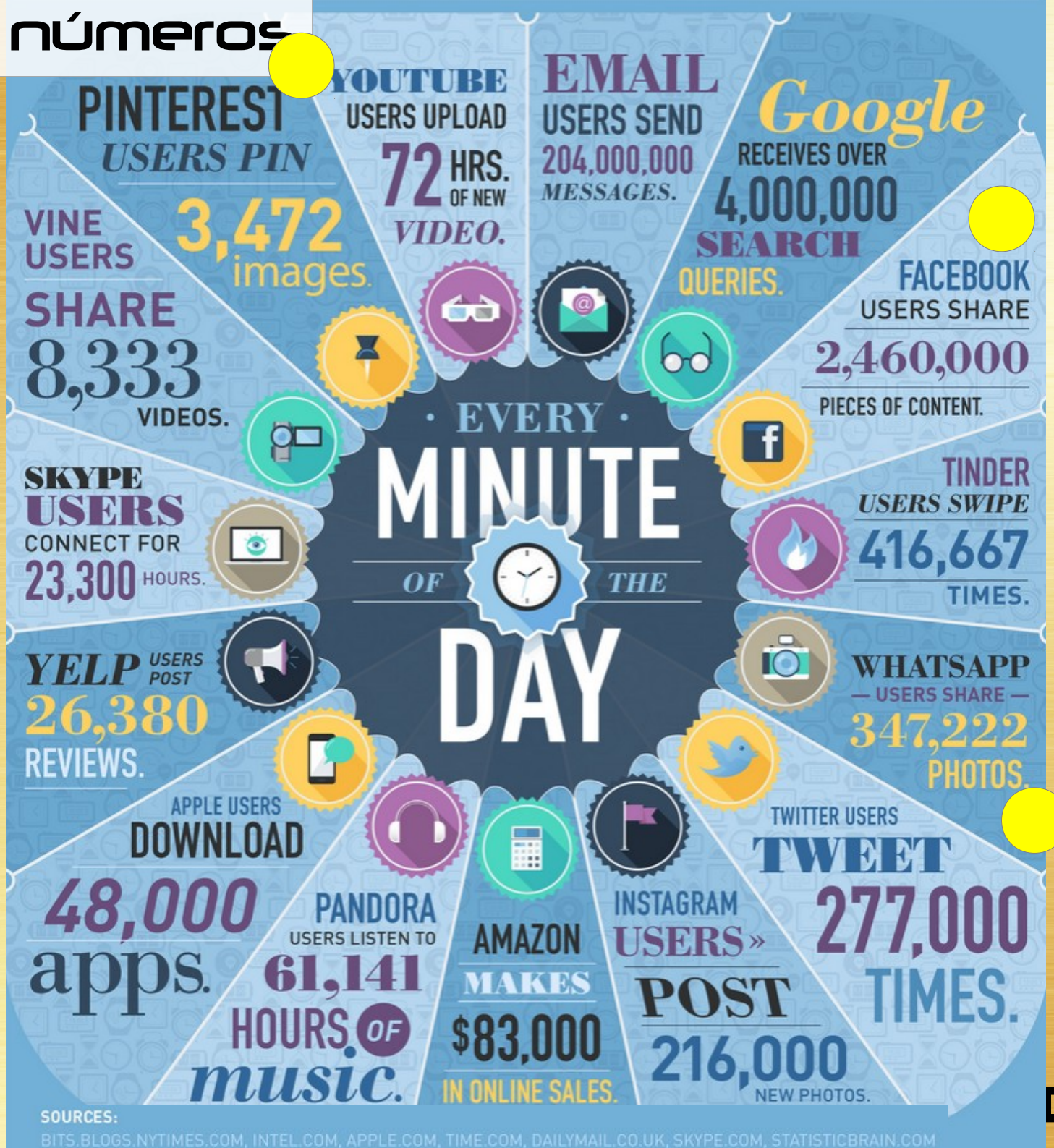
Juanm

Data

Grandes números

2014
De 2.1
a 2.4
billones

2012 - 2.1
2014 - 2.4



Juanmi

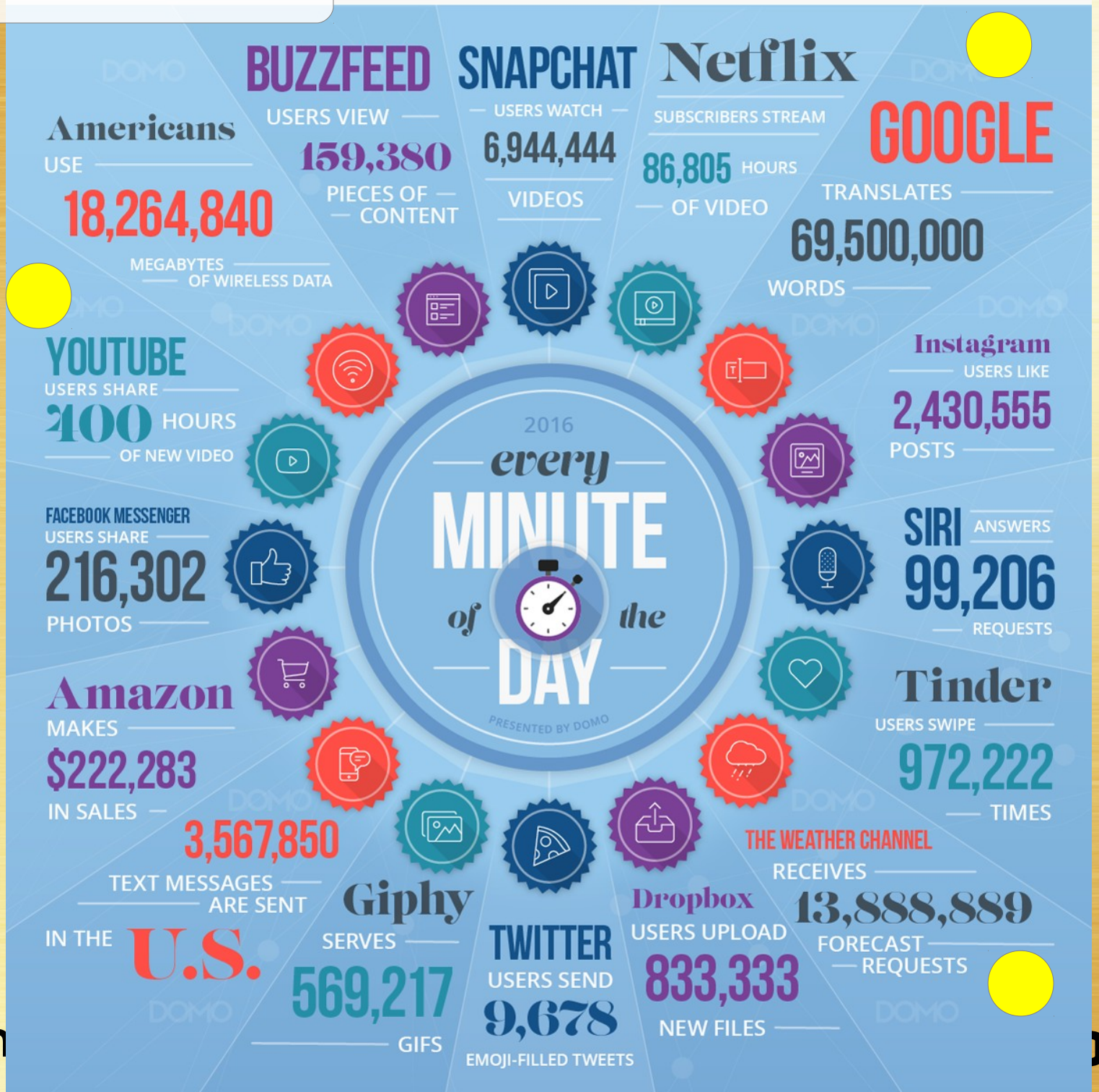
Data

Grandes números

2016

De 2.4
a 3.4
billones

2012 - 2.1
2014 - 2.4
2016 - 3.4



Juanm

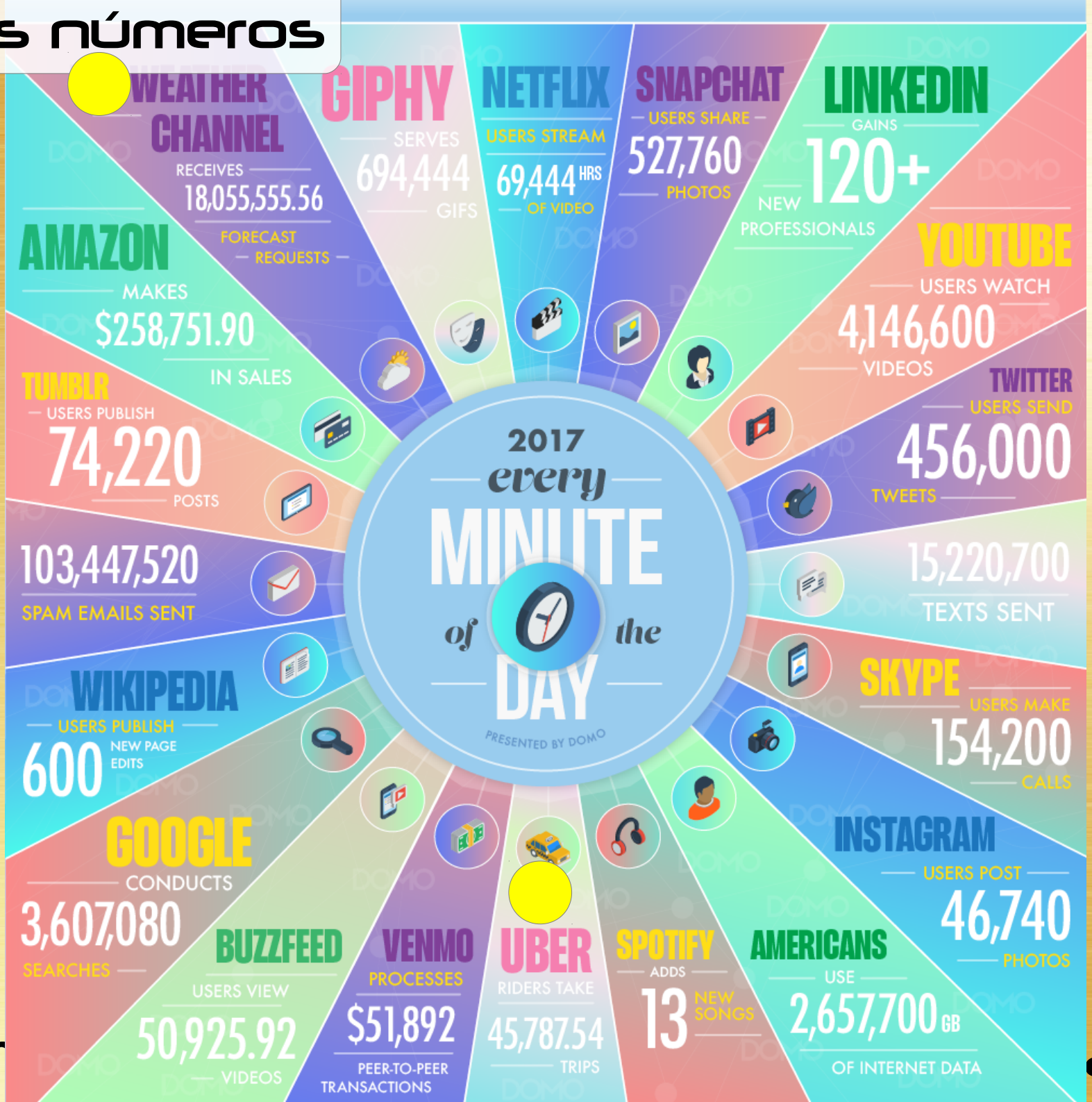
Data

Grandes números

2017

De 3.4
a 3.7
billones

2012 - 2.1
2014 - 2.4
2016 - 3.4
2017 - 3.7



Juanr

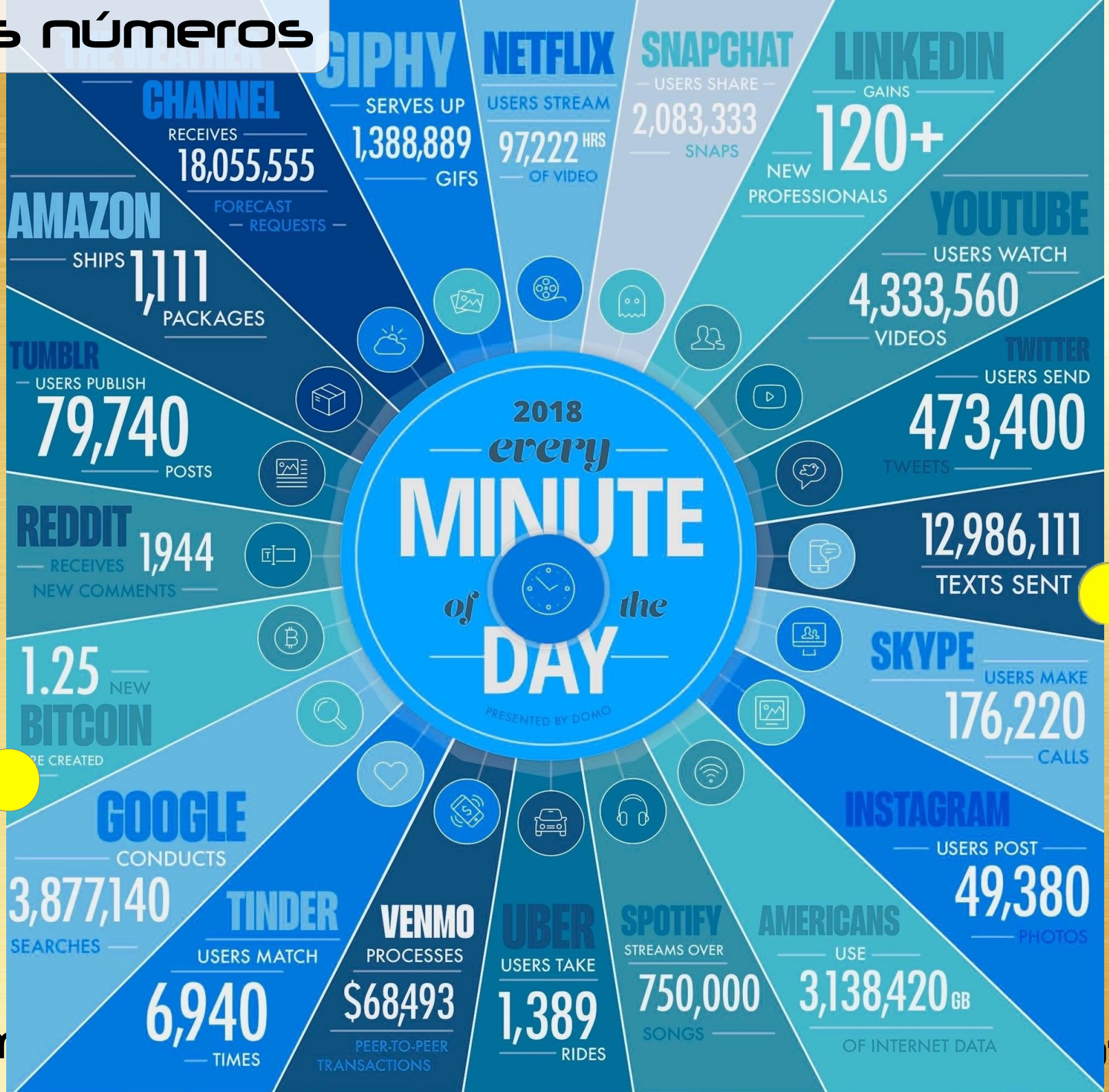
ata

Grandes números

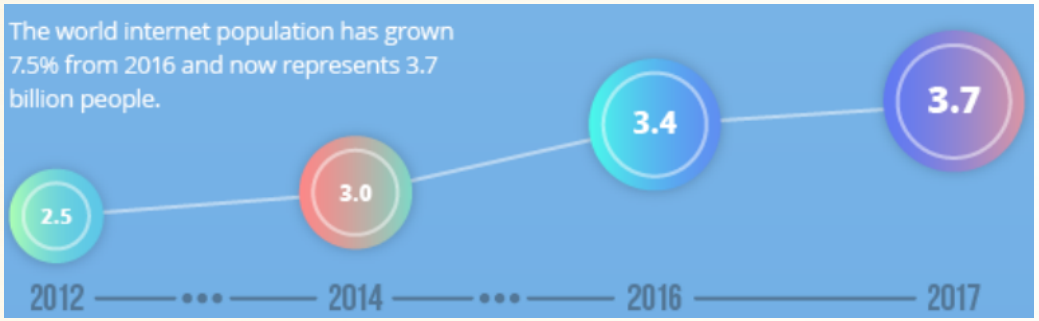
2018

De 3.4
a 3.7
billones

2012 - 2.1
2014 - 2.4
2016 - 3.4
2017 – 3.7
2017 – 4.4



Usuarios de Internet:
2012 – 2.1 Billones
2014 – 2.4 Billones
2016 – 3.4 Billones
2017 – 3.7 Billones
2018 – 4.4 Billones



Hoy: 86% del tráfico son videos

Toda la historia de películas juntas cruzan Internet **cada 3 minutos**

A lo largo de 1 año hablamos de 1 trillón de Gigabytes o …

1 Zetabbyte

IBM 2018: 2.5 Quintillones de bytes/día

¿Cuanto es un quintillón?

- Una única **gota de agua** contiene:
  1.7 quintillones de moléculas de agua.

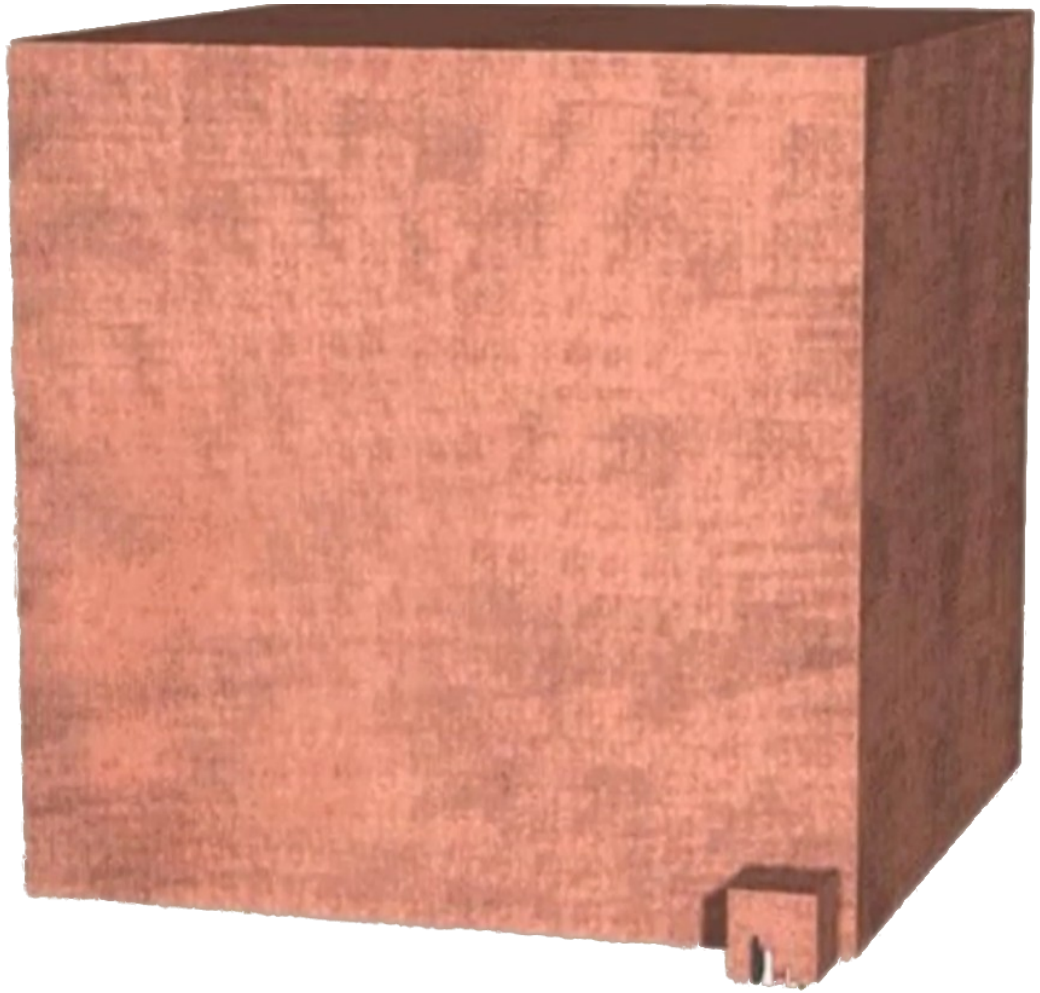
- La distancia de la **Via Láctea hasta Andrómeda** es de:
  2 millones de años luz
  18,87 quintillones de kilómetros
  11,73 quintillon miles

- **La tierra** completa contiene unos:
  1.234 quintillones de litros de agua
  326 quintillon gallons of water

- Si cortamos **la tierra por la mitad**, la sección tendría un área aproximada de:
  1.275 quintillones de centímetros cuadrados

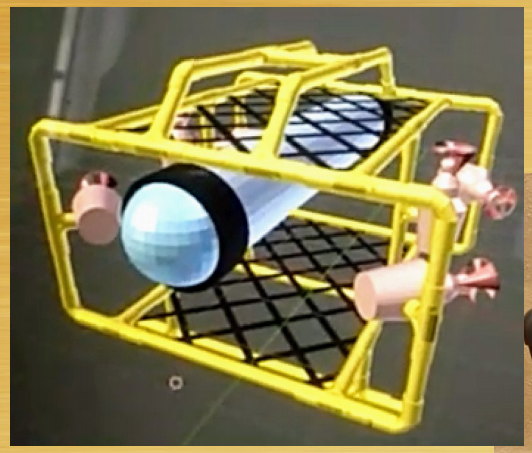
- ¿Cuanto es un quintillón de céntimos o peniques de dólar?

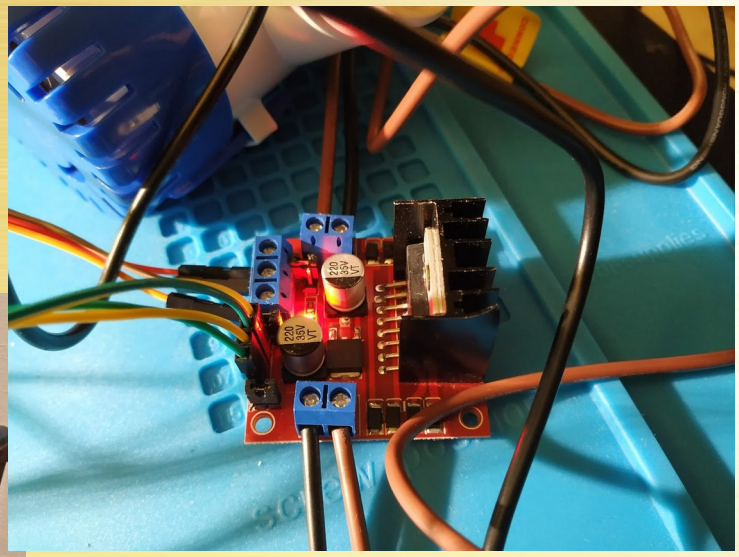
¿Cuanto es un quintillón... de peniques?

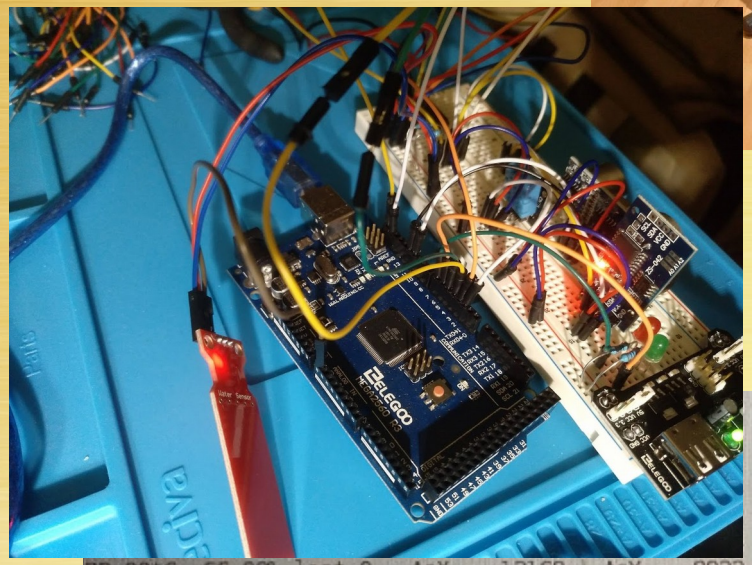


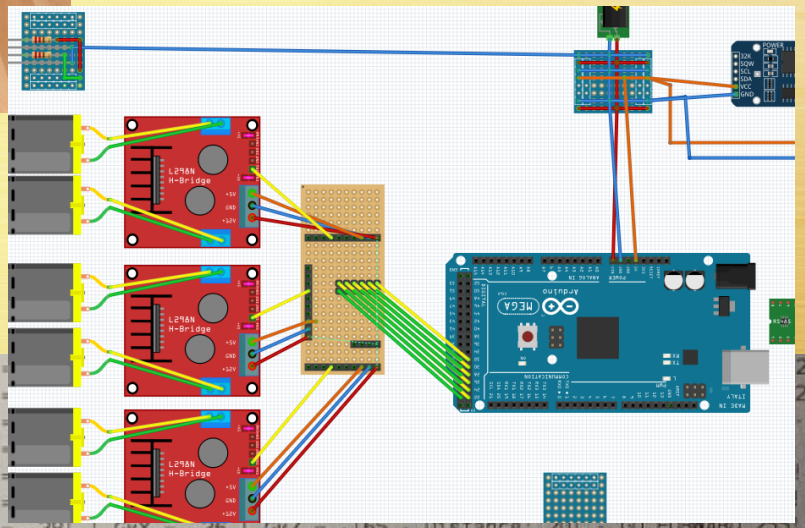
1.000.067.088.384.000.000 peniques
1 quintillón, 67 trillones, 88 billones, 384 millones de peniques
Un cubo de **8,32 kilómetros** de lado

¿Buceamos en los datos?










```
22.00*C, 66.00% last:0 - AcX = -13168 | AcY = -8932 | AcZ = -4228 | Tmp = 27.73 | GyX = -381 | GyY = -36 | GyZ = -165 - Distance: 201 - GO FORWARD 255!
22.00*C, 66.00% last:0 - AcX = -13224 | AcY = -9044 | AcZ = -4428 | Tmp = 27.73 | GyX = -307 | GyY = -19 | GyZ = -108 - Distance: 202 - GO FORWARD 255!
22.00*C, 66.00% last:0 - AcX = -13308 | AcY = -8956 | AcZ = -4288 | Tmp = 27.78 | GyX = -377 | GyY = -45 | GyZ = -239 - Distance: 202 - GO FORWARD 255!
22.00*C, 66.00% last:0 - AcX = -13224 | AcY = -8956 | AcZ = -4200 | Tmp = 27.73 | GyX = -340 | GyY = -34 | GyZ = 13 - Distance: 202 - GO FORWARD 255!
```

Juanmi Taboada

Big Data

Eric Brewer (2000)

C: Consistency → Consistencia

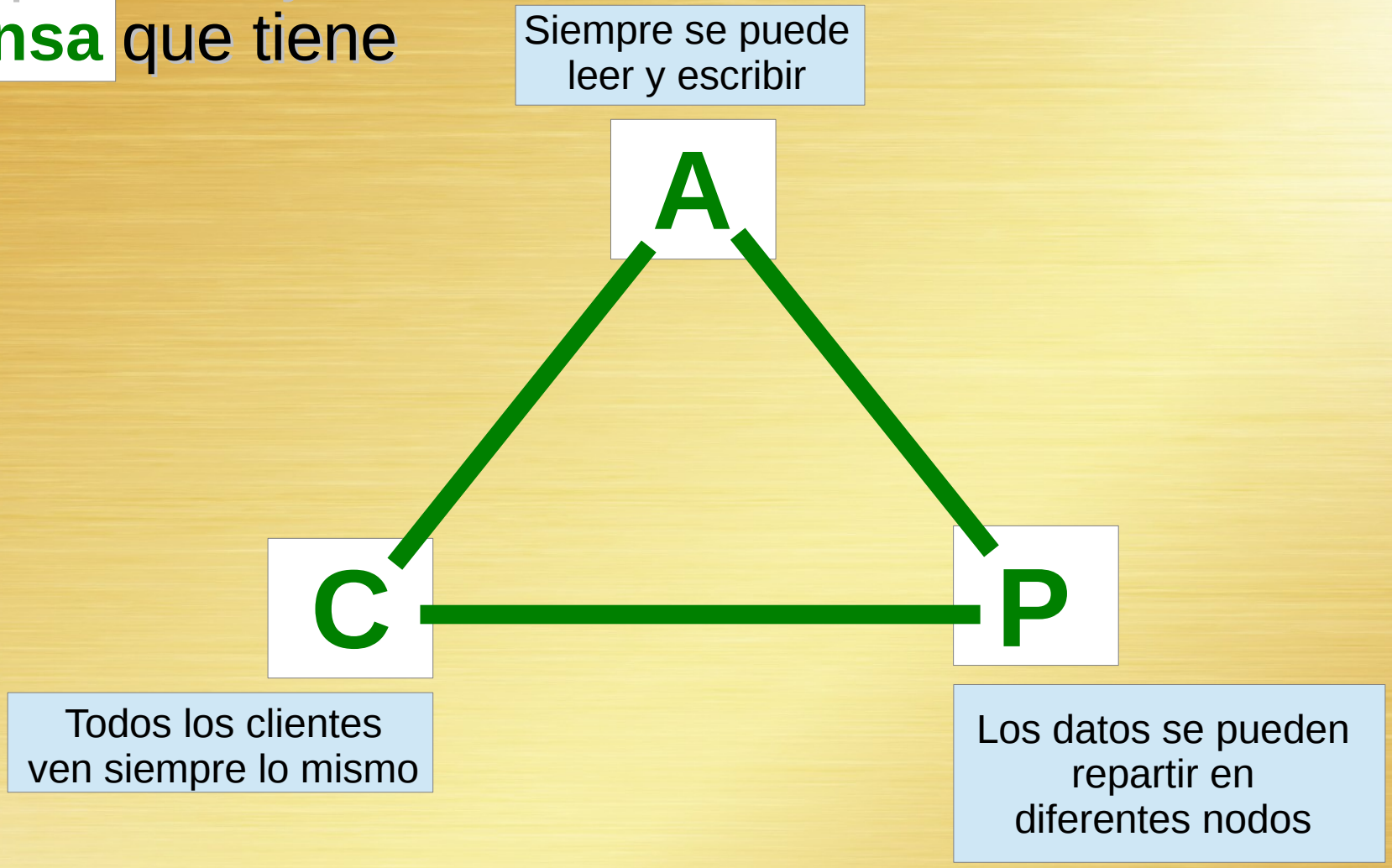
A: Availability → Disponibilidad

P: Partition tolerance → Tolerancia al particionado

Sólo puedes llegar a 2 de las 3

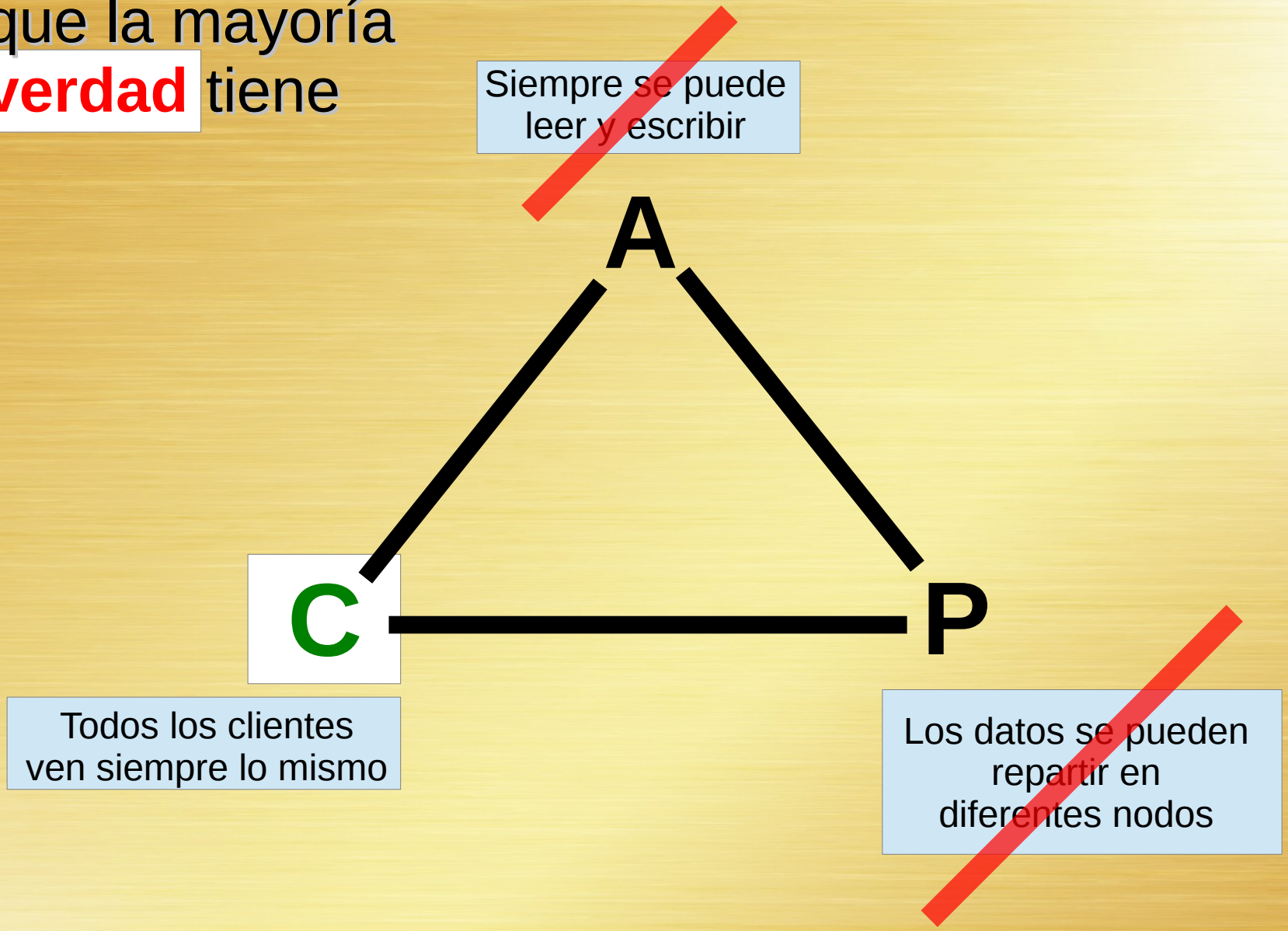
Es imposible para un sistema de cómputo distribuido garantizar simultáneamente la **consistencia**, la **disponibilidad** y ser tolerante al **particionado** de los datos (separación y distribución).

Lo que la mayoría **piensa** que tiene



Teorema de CAP

Lo que la mayoría de **verdad** tiene



Lo cierto es que todos buscamos la **disponibilidad (A)**

Pero ... ¡¡¡ tenemos que elegir entre... !!!

Escalabilidad (P)

y

Consistencia (C)

ACID

A: Atomicidad

C: Consistencia

I: Aislamiento (Isolation)

D: Durabilidad

En grandes sistema ocurre que:
Disponibilidad y Rendimiento

BASE

BA: Básicamente disponible

S: Estado flexible/maleable (Soft state)

E: Eventualidad del estado final

Da **menos** importancia a la **consistencia**
en **pro** de la tolerancia al **particionado**
aparece la **consistencia eventual**

¿Qué es la consistencia eventual?

Que ... eventualmente será consistente

Podemos introducir un dato y que
no esté disponible inmediatamente después

Convergencia natural a la consistencia

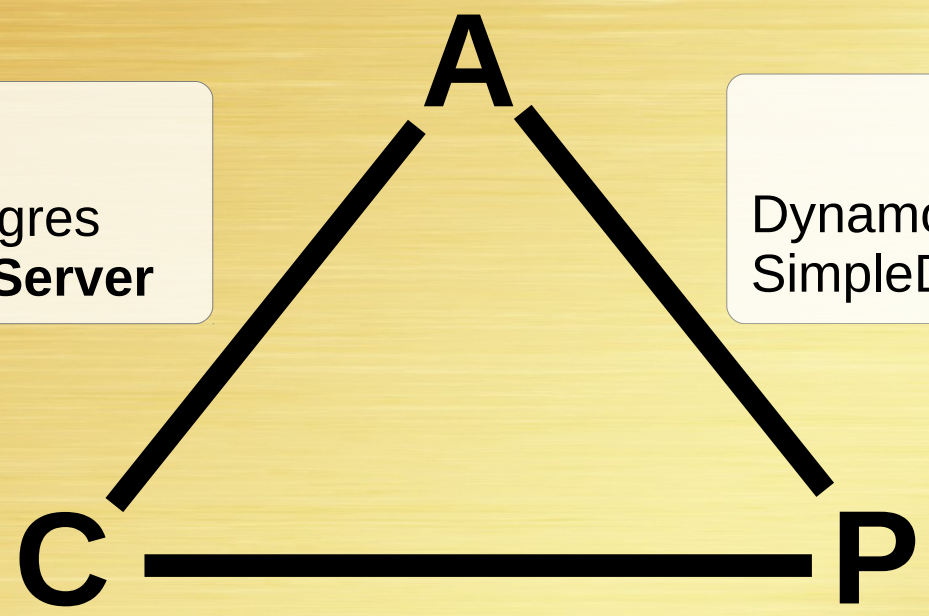
Métodos de resolución de conflictos

Evitan la entropía (control de versiones)

Reconciliación (elección de estado final)
[generalmente: “last write wins”]

Garantizar la seguridad de una operación

Strong eventual consistency (SEC)



Siempre se puede leer y escribir

CA

MySQL Postgres
Oracle **SQLServer**

AP

Dynamo **Cassandra**
SimpleDB CouchDB

Todos los clientes ven siempre lo mismo

CP

MongoDB Redis
MemcacheDB Hbase
Berkely DB

Los datos se pueden repartir en diferentes nodos

CA

MySQL
Postgres
Oracle
SQLServer
Aster Data
Greenplum
Vertica
Clickhouse

AP

Dynamo
Voldemort
Tokyo Cabinet
KAI
Cassandra
SimpleDB
CouchDB
Riak

CP

BigTable
Hypertable
HBase
MongoDB
Terrastore
Scalaris
Berkely DB
MemcacheDB
Redis


You
Tube

- Planificar antes de actuar, no te dejes llevar por las emociones. ¿Cuales son los objetivos?

- Diseño orientado a consulta: ¿Cuales son las preguntas a responder?

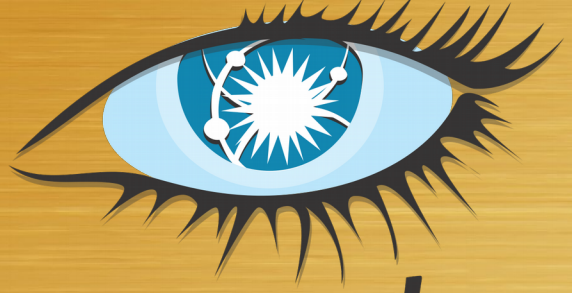
- ~~DRY~~: Vale repetir la información

- El almacenamiento no debe ser un coste significativo (HA, resilencia, velocidad, esparcimiento,…)

- El motor de BD BigData va en f(proyecto, dato)

¿Quién usa Big Data?

Fuente: http://planetcassandra.org/companies/



CALL OF DUTY

CERN

COMCAST

Alcatel·Lucent

agoda

CREDIT SUISSE

The New York Times

accenture
High performance. Delivered.

KASPERSKY lab

OPERA software

Aol.

NBCUniversal

reddit

Adobe

Spotify

THOMSON REUTERS

vmware

Disney

AVIRA

Bla Bla Car

FedEx

SanDisk

hp

avast!

Centrologic

NASA

rackspace

UBISOFT

CISCO

PayPal

Expedia

Sony Entertainment

softonic

Symantec

ebay

DELL

ERICSSON

GitHub

NETFLIX

BlackBerry

IBM

despegar.com

ING

SHAZAM

Telefónica

Instagram
Fast beautiful photo sharing

MuleSoft
connecting the new enterprise

Microsoft

unity

Juanmi Taboada

Big Data

¿Quién usa Big Data?

CERN

Al finalizar la última iteración del LHC en el que se descubrió el "**Bosón de Higgs**", el CERN almacenaba más de **100Pbytes**

El LHC del CERN arrancó en **Abril de 2015** en busca de la **Supersimetría** (capaz de producir 1Pbytes/segundo)

1Petabyte = 1.000 Terabytes = 1 Millón de Gigabytes

NETFLIX

Ejecuta 235 clusters separados, con un total de 7.000 nodos.
1 Millón de escrituras por segundo (factor 3)

unity

Ecosistema para desarrolladores de juegos que teniendo problemas con MongoDB migraron a Cassandra.

Disponibilidad contínua

Simplicidad en la gestión entre servidores
Sin un único punto de fallo

Escalabilidad linear
- Si 2 nodos procesan 100 transacciones/seg
- 4 nodos procesan 200 transacciones/seg
- 8 nodos procesan 400 transacciones/seg

Sistema **descentralizado** (sin master)

Relaciones por grupo: Nodo → Datacenter → Cluster
Replicación personalizada

Gossip: mantiene la red informada

Partitioner: deciden como se distribuyen los datos

Replication factor: número de réplicas en el cluster

Replica placement strategy: donde poner las réplicas

Snitch: define grupo de máquinas destinadas a réplicas

Gossip:

- Protocolo punto a punto

- Comunicación entre nodos

- Detección de fallos y recuperación

- Autodetección de topología e información

Particionador: encargado de "**esparcir**" los datos.

Desaconsejados:
Random: Hashes con MD5
ByteOrdered: orden lexicográfico sobre las claves
OrderPreserving: se asumen claves en formato UTF8

Murmur3: Funcionalmente idéntico a Random pero es más rápido sin efectos colaterales. Se centra en la distribucción espacial.

Planificar el deploy de un cluster

Hardware:

- RAM: 16GB-64Gb (mínimo 8Gb)
- CPU: 8-cores dedicados o 4-8 cores en virtuales
- Disco: mejor entre 500Gb y 1Tb por nodo (según I/O)
- 2 discos (commit log + data)
- Sistema de ficheros XFS
- Red mínimo Gigabit

Planificar el deploy de un cluster

Espacio útil disco: 45%-70% de espacio total RAW

Antipatrones:
- Usar un NAS
- Sistemas de ficheros compartidos
- SELECT … IN
- Leer antes de escribir (multiples hits)
- Balanceadores de carga
- Falta de testing
- Bajo conocimiento de Linux

CQL
(Cassandra Query Language)

SQL

```
USE myDatabase;

/* Creating Tables */
CREATE TABLE IF NOT EXISTS myTable (id INT PRIMARY KEY);

/* Altering Tables /*
ALTER TABLE myTable ADD myField INT;

/* Creating Indexes */
CREATE INDEX myIndex ON myTable (myField);

/* Inserting Data */
INSERT INTO myTable (id, myField) VALUES (1, 7);

/* Selecting Data */
SELECT * FROM myTable WHERE myField = 7;

/* Counting Data */
SELECT COUNT(*) FROM myTable;

/* Deleting Data */
DELETE FROM myTable WHERE myField = 7;
```

CQL

```
USE myDatabase;

/* Creating Tables */
CREATE TABLE IF NOT EXISTS myTable (id INT PRIMARY KEY);

/* Altering Tables /*
ALTER TABLE myTable ADD myField INT;

/* Creating Indexes */
CREATE INDEX myIndex ON myTable (myField);

/* Inserting Data */
INSERT INTO myTable (id, myField) VALUES (1, 7);

/* Selecting Data */
SELECT * FROM myTable WHERE myField = 7;

/* Counting Data */
SELECT COUNT(*) FROM myTable;

/* Deleting Data */
DELETE FROM myTable WHERE myField = 7;
```


```
USE miBaseDatos;

/* Creando Tablas */
CREATE TABLE IF NOT EXISTS miTabla (id INT PRIMARY KEY);

/* Cambiando Tablas /*
ALTER TABLE miTabla ADD miCampo INT;

/* Creating Indexes */
CREATE INDEX miIndice ON miTabla (miCampo);
```

/* Insertando Datos */
INSERT INTO miTabla (id, miCampo) VALUES (1, 7);

/* Seleccionando Datos */
SELECT * FROM miTabla WHERE miCampo = 7;

/* Contando Datos */
SELECT COUNT(*) FROM miTabla;

/* Borrando Datos */
DELETE FROM miTabla WHERE miCampo = 7;


```
/* Crear un nuevo keyspace en CQL */
CREATE KEYSPACE miBaseDatos WITH replication =
   {'class': 'SimpleStrategy', 'replication_factor': 1};

/* Crear una nueva base de datos en SQL */
CREATE DATABASE miBaseDatos;
```

No existen:
JOIN, **GROUP BY**, y **FOREIGN KEY**

1.- Las escrituras son baratas. Escribe todo del modo en que vas a leerlo.

2.- UPSERT: INSERT (inserta ó actualiza), UPDATE (inserta ó actualiza), ya que Cassandra no hace un read durante estos procesos.

3.- Filas con TTL, pasado X tiempo la fila caduca.

4.- DELETE ... ¡no borra!

/* Seleccionar datos de un rango */
SELECT * FROM myTable
WHERE miCampo > 5000 AND miCampo < 100000;

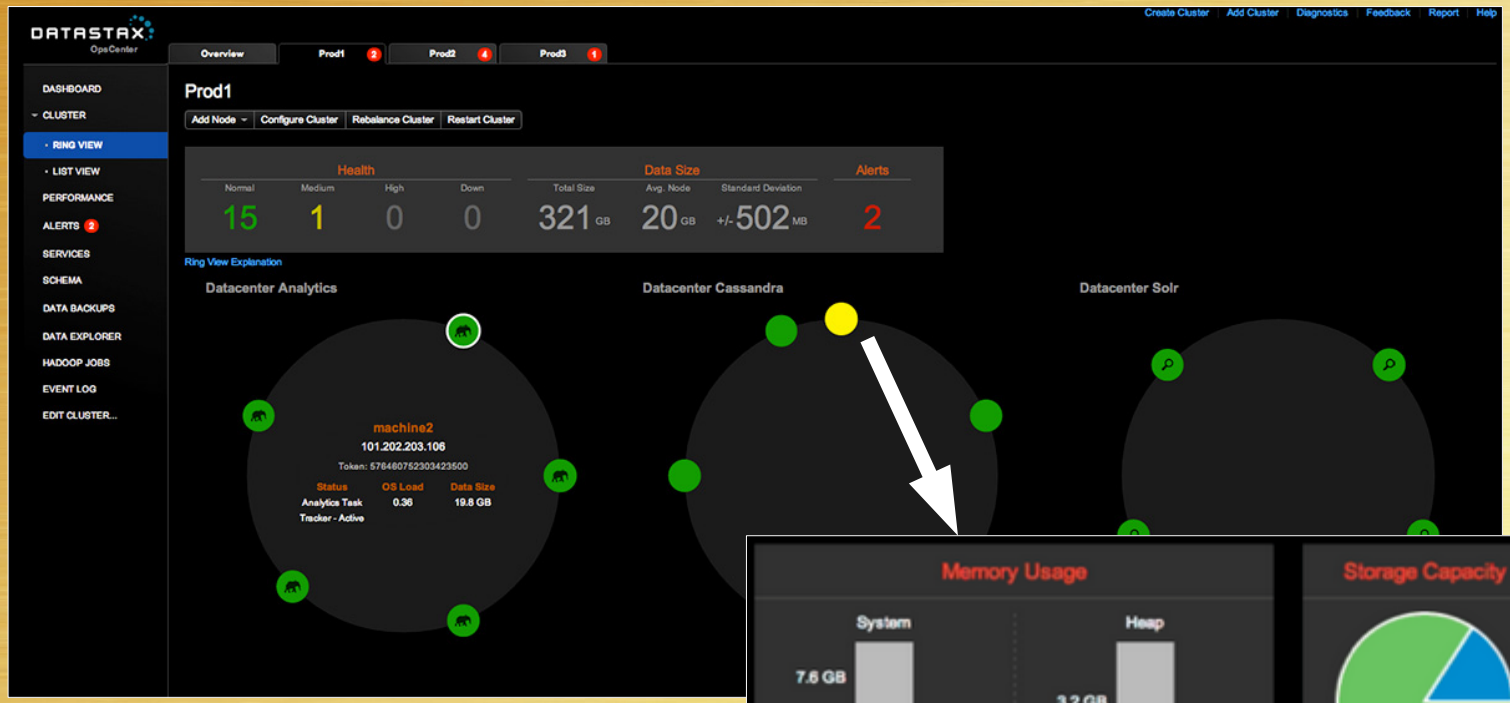
Bad Request: Cannot execute this query as it might involve data filtering and thus may have unpredictable performance. If you want to execute this query despite the performance unpredictability, use ALLOW FILTERING.

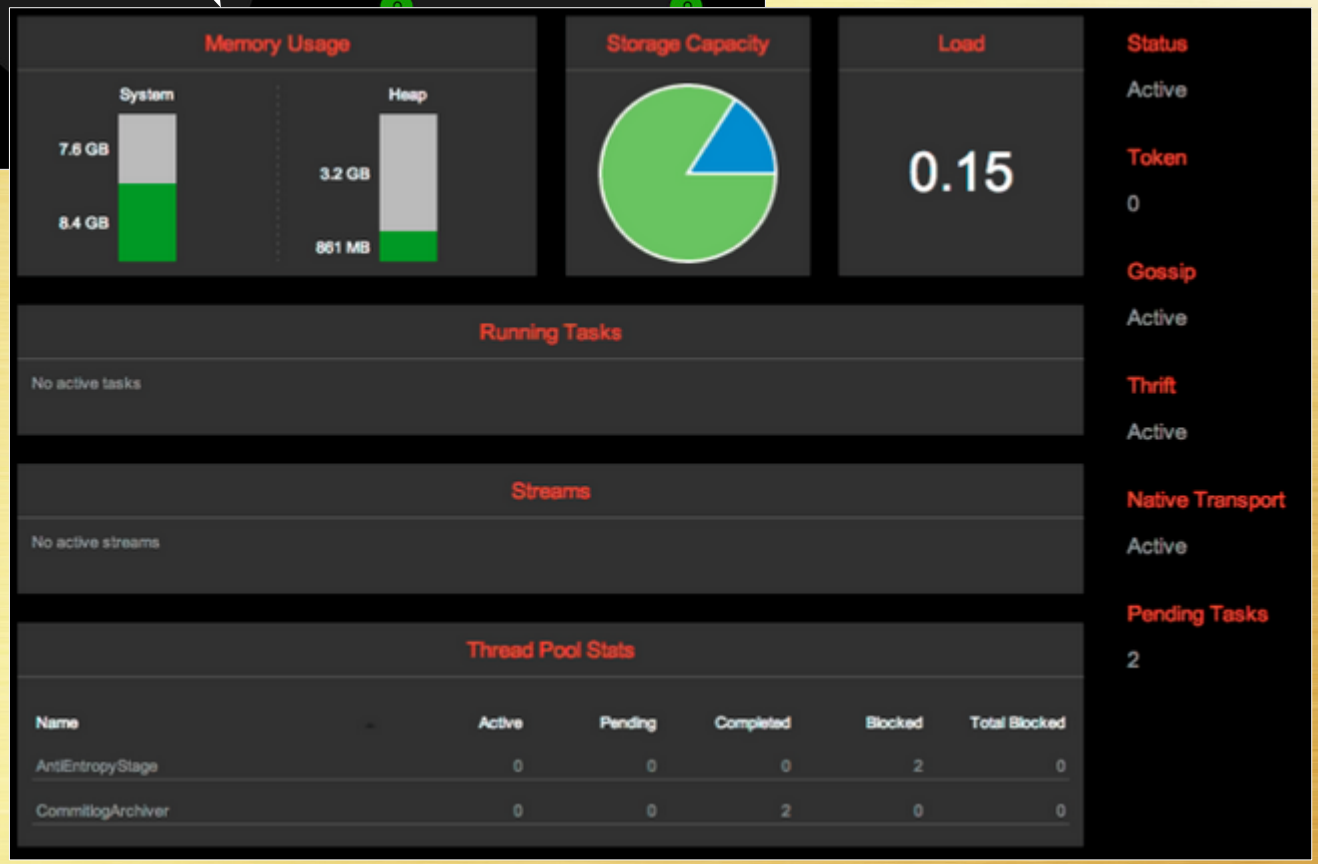
Opscenter

Juanmi Taboada

Big Data

Opscenter

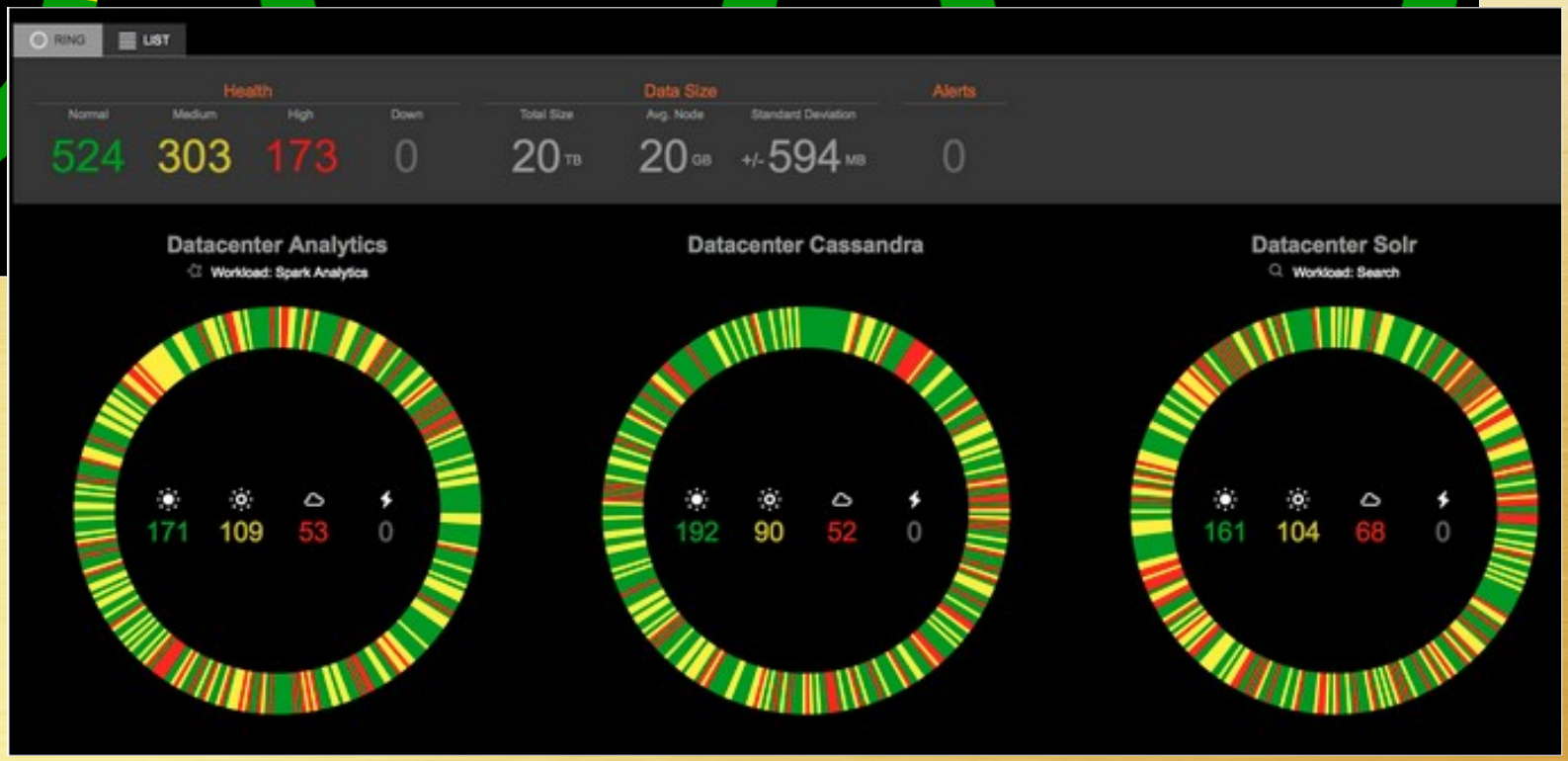




Juanmi Taboada

Big Data

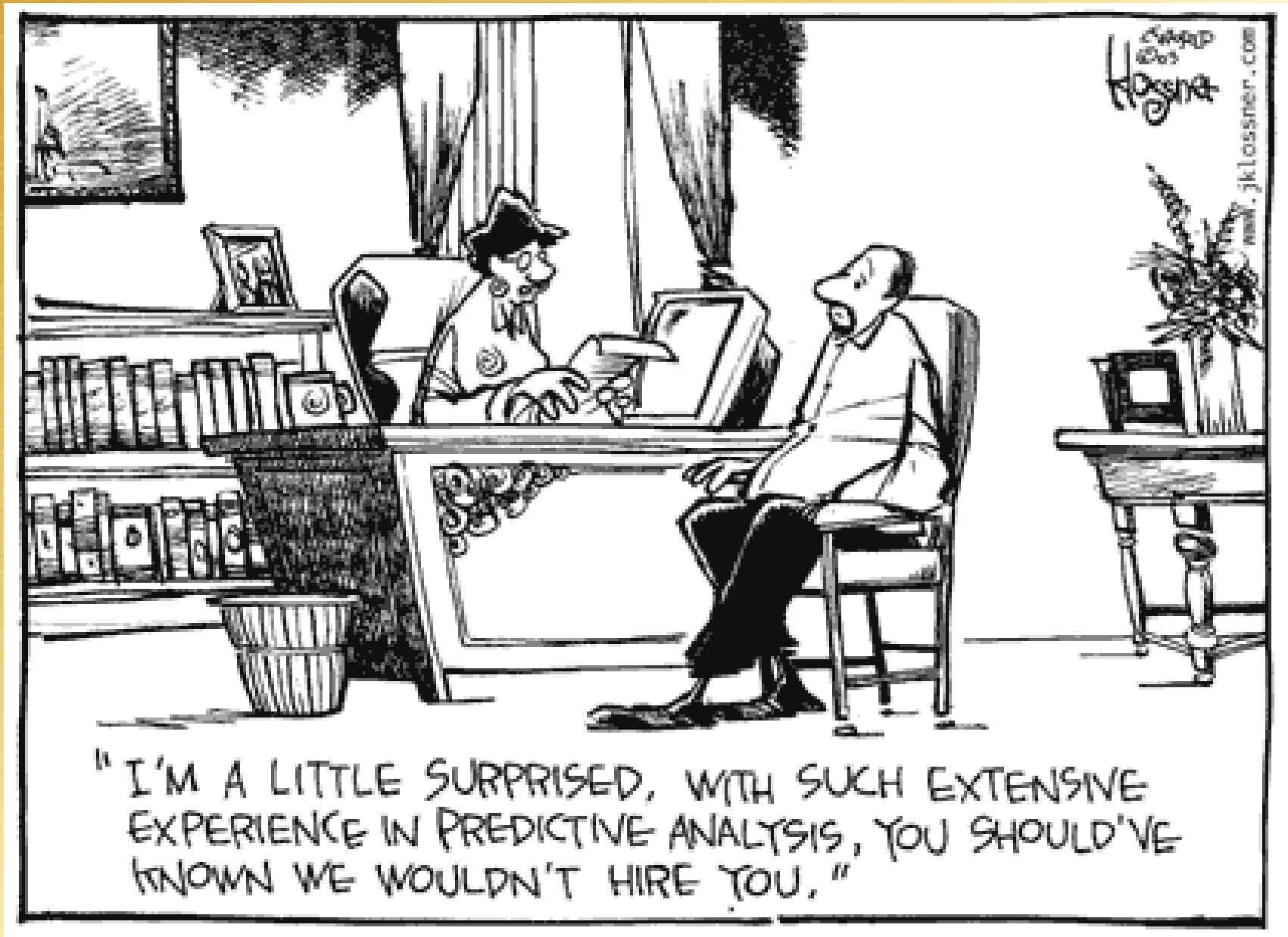
Opscenter



Juanmi Taboada

Big Data

Una dosis de realidad



Una dosis de realidad

- Predicción de datos

- La tareas más ardua es filtrar y preparar los datos

- Desechar información para evitar males mayores

- PHP vs Python vs C, todo vale

- El rendimiento homogéneo de la BD es clave (red, Discos, CPU, RAM,…, tipo de partición)

- Hay bases de datos para diferentes tipos de datos (también existen bases de datos en memoria)

- Parar unos segundos son ciento de ventas perdidas

¿Dudas?

Muchas Gracias

Thank you - Dziękuję

Juanmi Taboada
@juanmitaboada
www.juanmitaboada.com