

Sistemas Paralelos y Distribuidos

Máster en Ciencia y Tecnología Informática

Diseño de Sistemas Distribuidos

Máster en Ingeniería Informática

Curso 2021-2022

# Sistemas de altas prestaciones en entornos distribuidos

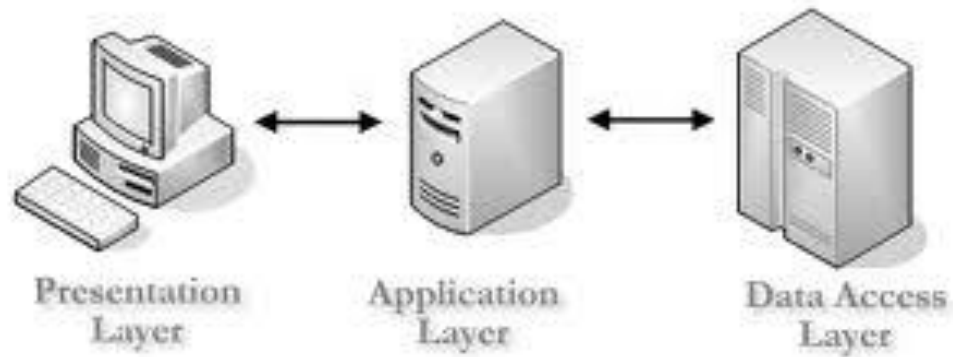
Alejandro Calderón Mateos y Félix García Carballeira

Grupo de Arquitectura de Computadores

[alejandro.calderon@uc3m.es](mailto:alejandro.calderon@uc3m.es)

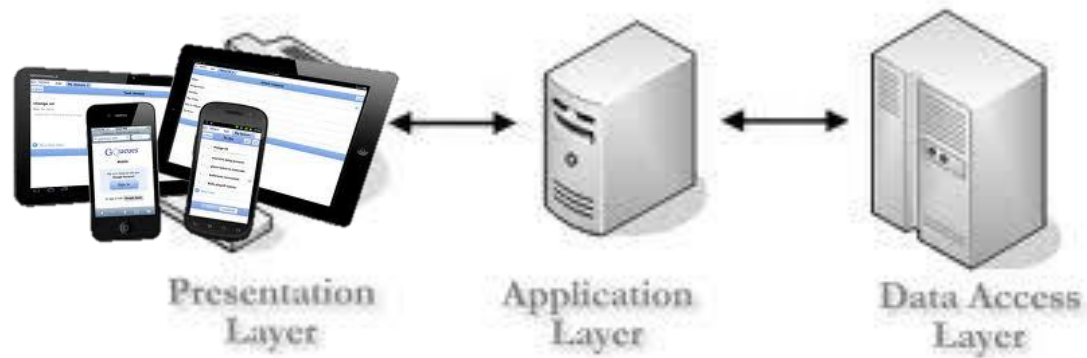
## 3-Tier Architecture

Internet / Intranet



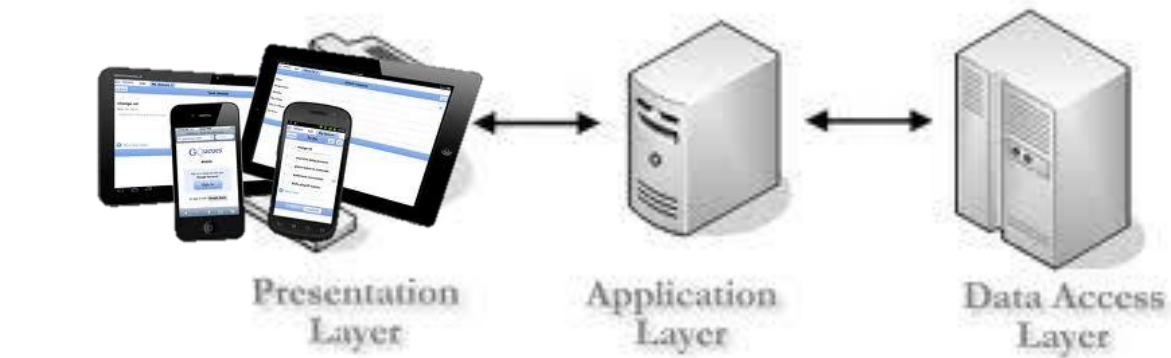
## 3-Tier Architecture

Internet / Intranet



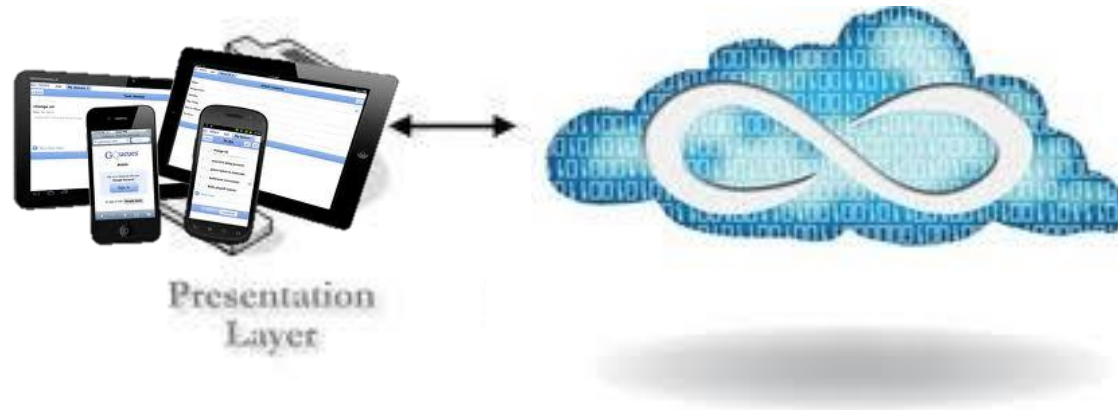
## 3-Tier Architecture

Internet / Intranet



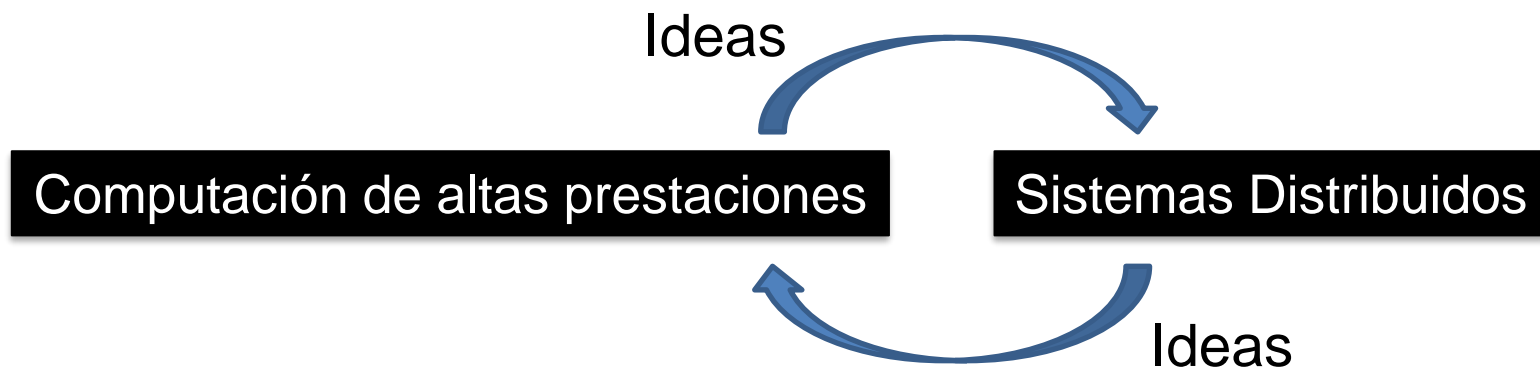
## 3-Tier Architecture

Internet / Intranet

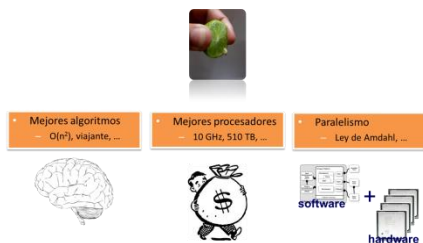


## 3-Tier Architecture

Internet / Intranet

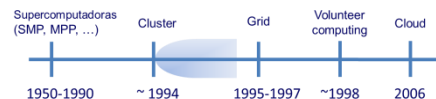


# Agenda



## Introducción a la computación de altas prestaciones

- Qué, dónde y cómo
- Hardware y software

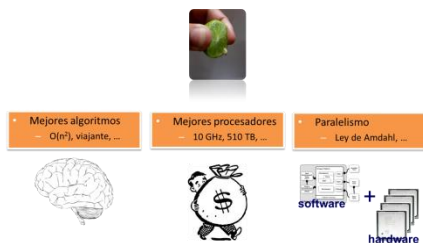


## Evolución de la computación de altas prestaciones

- Plataformas
- Tendencias

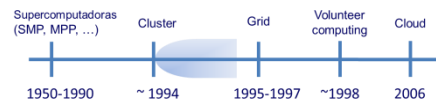


# Agenda



## Introducción a la computación de altas prestaciones

- Qué, dónde y cómo
- Hardware y software



## Evolución de la computación de altas prestaciones

- Plataformas
- Tendencias





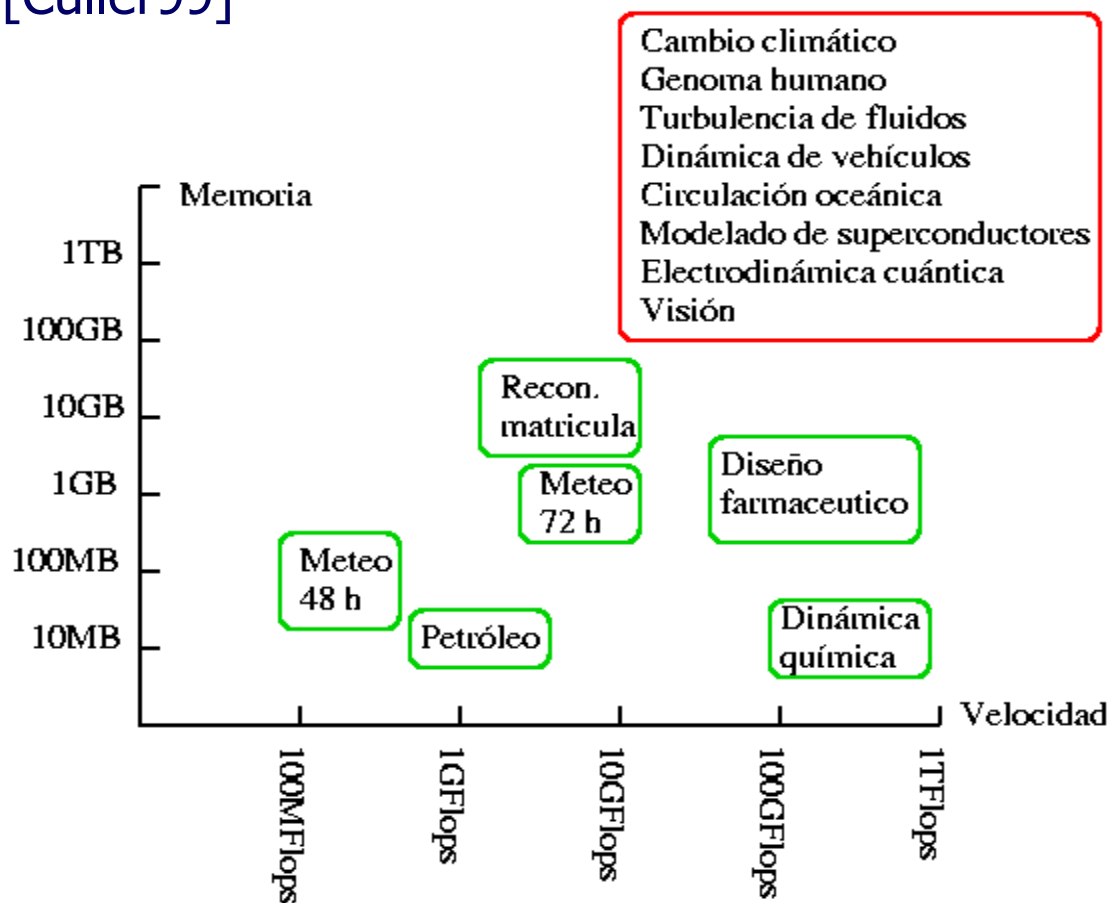
# Computación de altas prestaciones



- La computación de altas prestaciones o HPC (*High Performance Computing*) se centra principalmente en **la velocidad**.
- El objetivo es conseguir la **máxima cantidad de cómputo** posible en la **mínima cantidad de tiempo**.

# ¿Dónde se necesita?

[Culler99]



# Ejemplo 1/2: Predicción meteorológica...

(<http://www.businessinsider.com/97-million-supercomputer-in-the-uk-2014-10>)

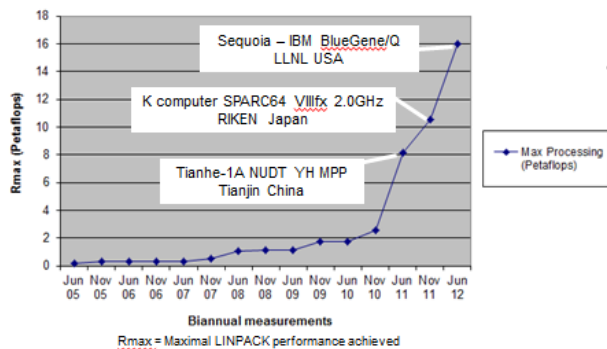


- La oficina encargada de **previsiones meteorológicas** invertirá en un **supercomputador** que le permitirá **mejores previsiones**.
  - Con **precisión de 300 metros** se podrá indicar incidencias relacionadas con niebla, rachas de viento, etc.
  - Predicciones con un **margen de 1 hora** (en lugar de 3 como ahora)

- Impacto:
  - Supondrá **97 millones de libras** (156,9 millones de dólares)
  - Estará operacional en el 2017.
  - El supercomputador pesa lo que **11 autobuses de doble planta**



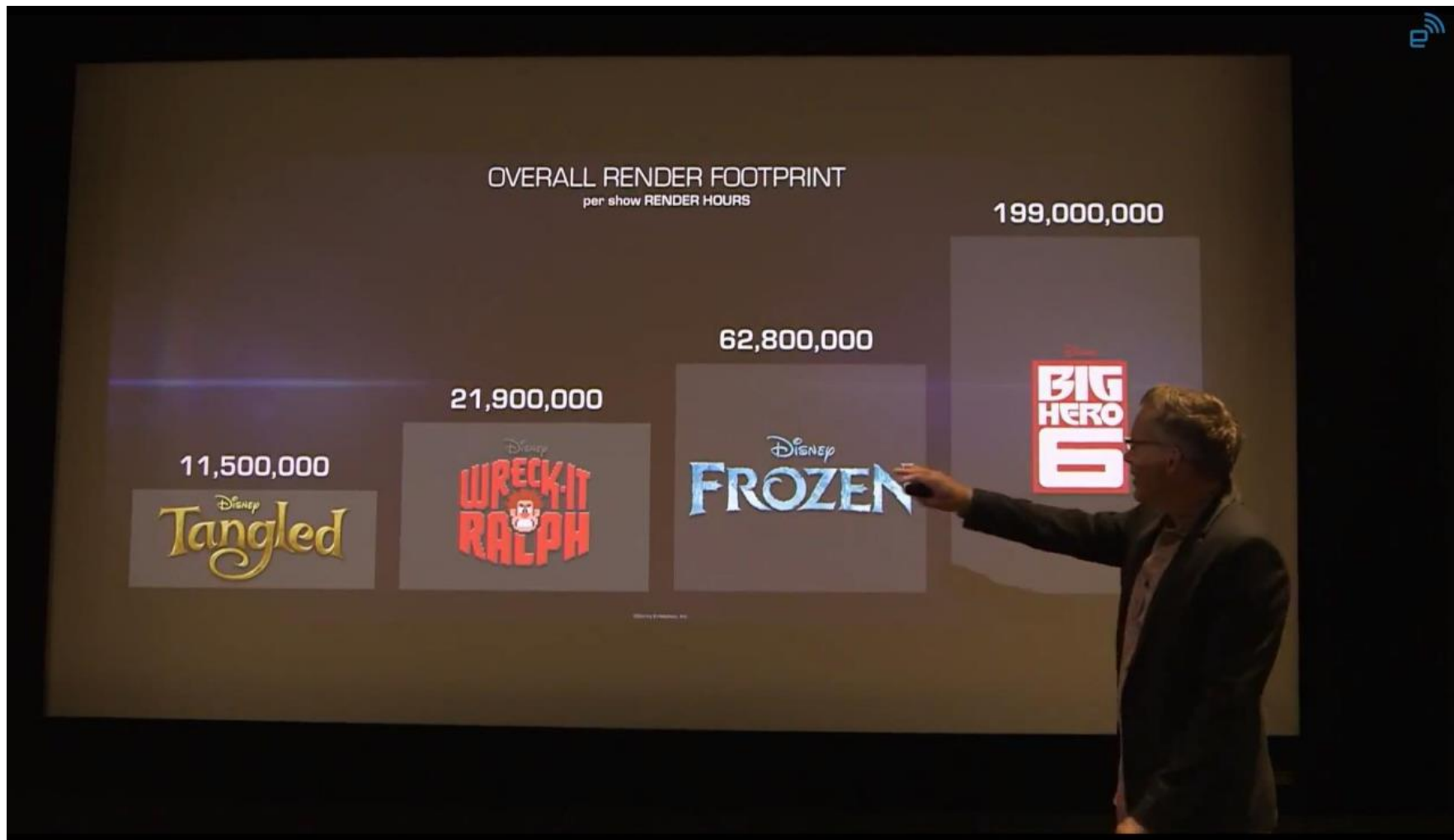
Growth in Supercomputing Capability: Jun 2005 - Jun 2012



- Capacidad computacional:
  - Será **13 veces más potente** que el que se usa ahora.
  - Tiene una capacidad aproximada de **16 petaFLOPS**.

# Ejemplo 2/2: Big Hero 6 (2014)...

(<http://www.engadget.com/2014/10/18/disney-big-hero-6/>)



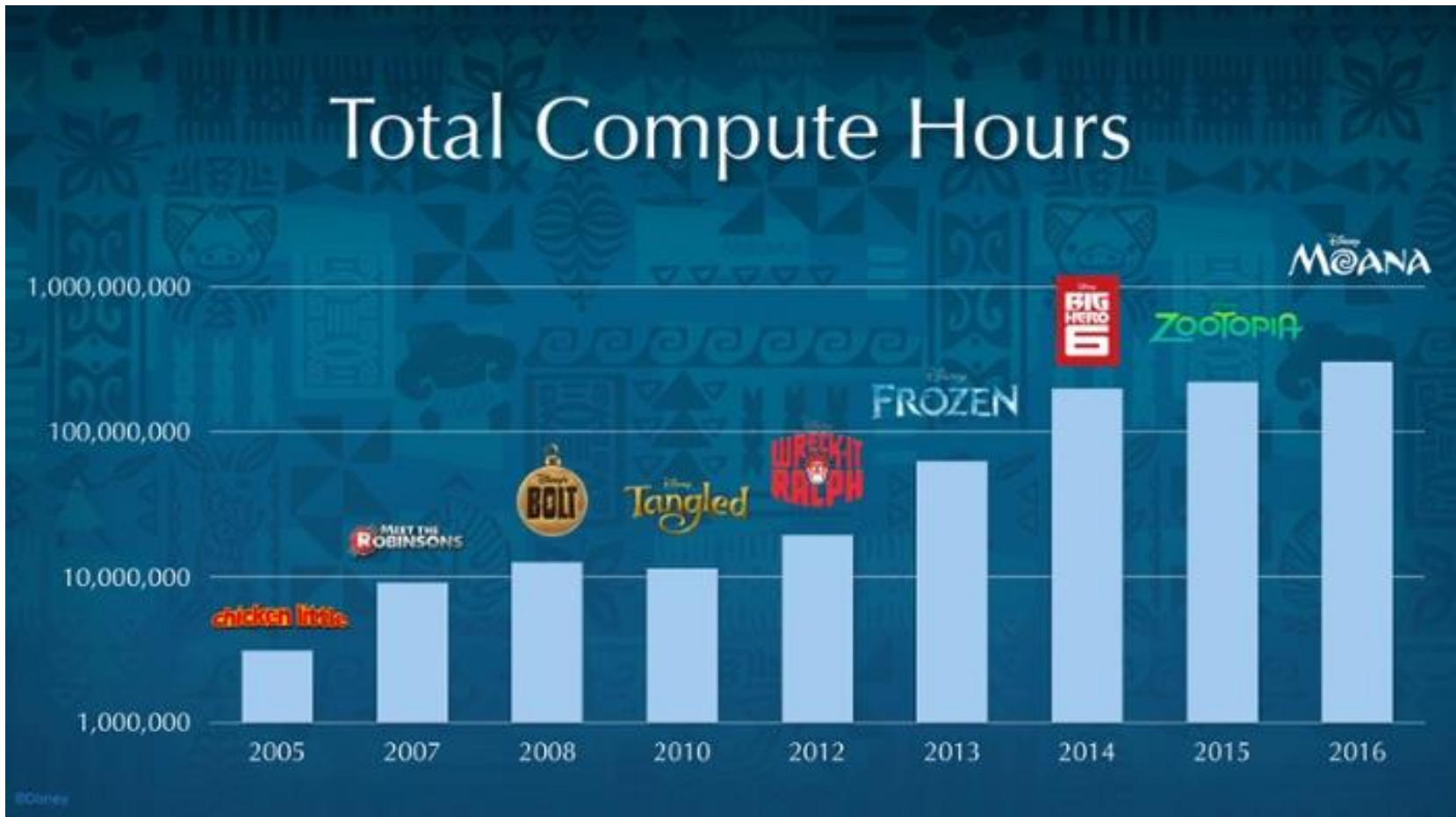
## Ejemplo 2/2: Big Hero 6 (2014)...

(<http://www.engadget.com/2014/10/18/disney-big-hero-6/>)

- To manage that cluster and the 400,000-plus computations it processes per day (roughly about 1.1 million computational hours per day), his team created software called Coda, which treats the four render farms like a single supercomputer. If one or more of those thousands of jobs fails, Coda alerts the appropriate staffers via an iPhone app.
- **The film takes 199 million core-hours (181 days) of rendering.** To put the enormity of this computational effort into perspective, Hendrickson says that **Hyperion "could render *Tangled* (2010) from scratch every 10 days."**
- If that doesn't drive the power of Disney's proprietary renderer home, then consider this: San Fransokyo contains around 83,000 buildings, 260,000 trees, 215,000 streetlights and 100,000 vehicles (plus thousands of crowd extras generated by a tool called Denizen). What's more, all of the detail you see in the city is actually based off assessor data for lots and street layouts from the real San Francisco.

# Ejemplo 2/2: and beyond (2016)...

(<https://sciencenode.org/feature/disney-hpc-animates-the-universe-within-a-grain-of-sand.php>)



# ¿Cómo se consigue más velocidad?

# ¿Cómo se consigue más velocidad?

- Mejores algoritmos
  - $O(n^2)$ , viajante, ...





# ¿Cómo se consigue más velocidad?

— Mejores algoritmos

- $O(n^2)$ , viajante, ...



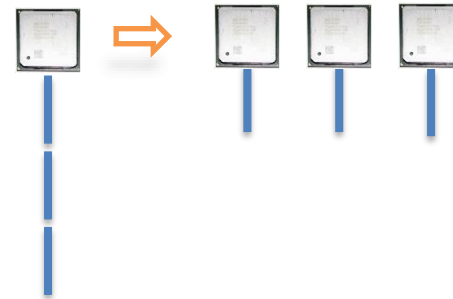
— Mejores procesadores (mejoras en la tecnología)

- CPU a 10 GHz, 510 TB de RAM, ...

# ¿Cómo se consigue más velocidad?

- Mejores algoritmos

- $O(n^2)$ , viajante, ...



- Mejores procesadores (mejoras en la tecnología)

- CPU a 10 GHz, 510 TB de RAM, ...

- Paralelismo (mejoras en el uso de la tecnología actual)

- Speedup, Ley de Amdahl, ...

# ¿Eso del paralelismo qué implica?

## – Mejores algoritmos

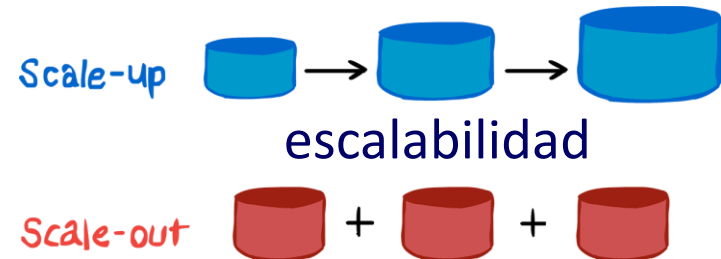
- $O(n^2)$ , viajante, ...

## – Mejores procesadores (mejoras en la tecnología)

- CPU a 10 GHz, 510 TB de RAM, ...

## Paralelismo (mejoras en el uso de la tecnología actual)

- Speedup, Ley de Amdahl, ...



# Tipos de paralelismo

- Tareas independientes:



# Tipos de paralelismo

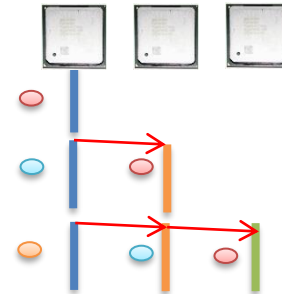
- Tareas independientes:



- Tareas cooperativas:

- *Pipeline*

- Coordinación (*mutex y conditions*)



# Tipos de paralelismo

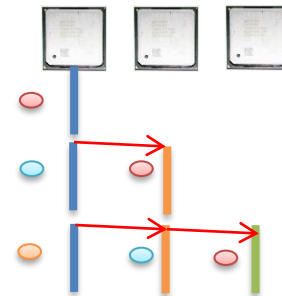
- Tareas independientes:



- Tareas cooperativas:

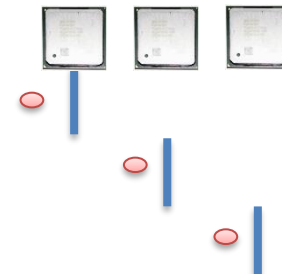
- *Pipeline*

- Coordinación (*mutex y conditions*)



- Tareas competitivas:

- Código secuencial :-S



# *Speedup*

- La mejora (o *speedup*) en la ejecución paralela con  $n$  elementos de cómputo será:

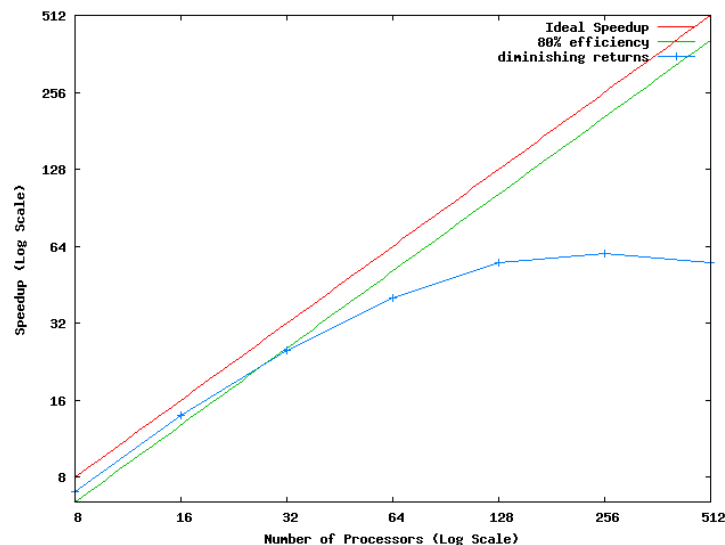
$$speedup = \text{tiempo\_de\_ejecución (1)} / \text{tiempo\_de\_ejecución (n)}$$

# Speedup

- La mejora (o *speedup*) en la ejecución paralela con  $n$  elementos de cómputo será:

$$\text{speedup} = \text{tiempo\_de\_ejecución (1)} / \text{tiempo\_de\_ejecución (n)}$$

- No siempre se obtiene un *speedup* ideal:





# Ley de Amdahl

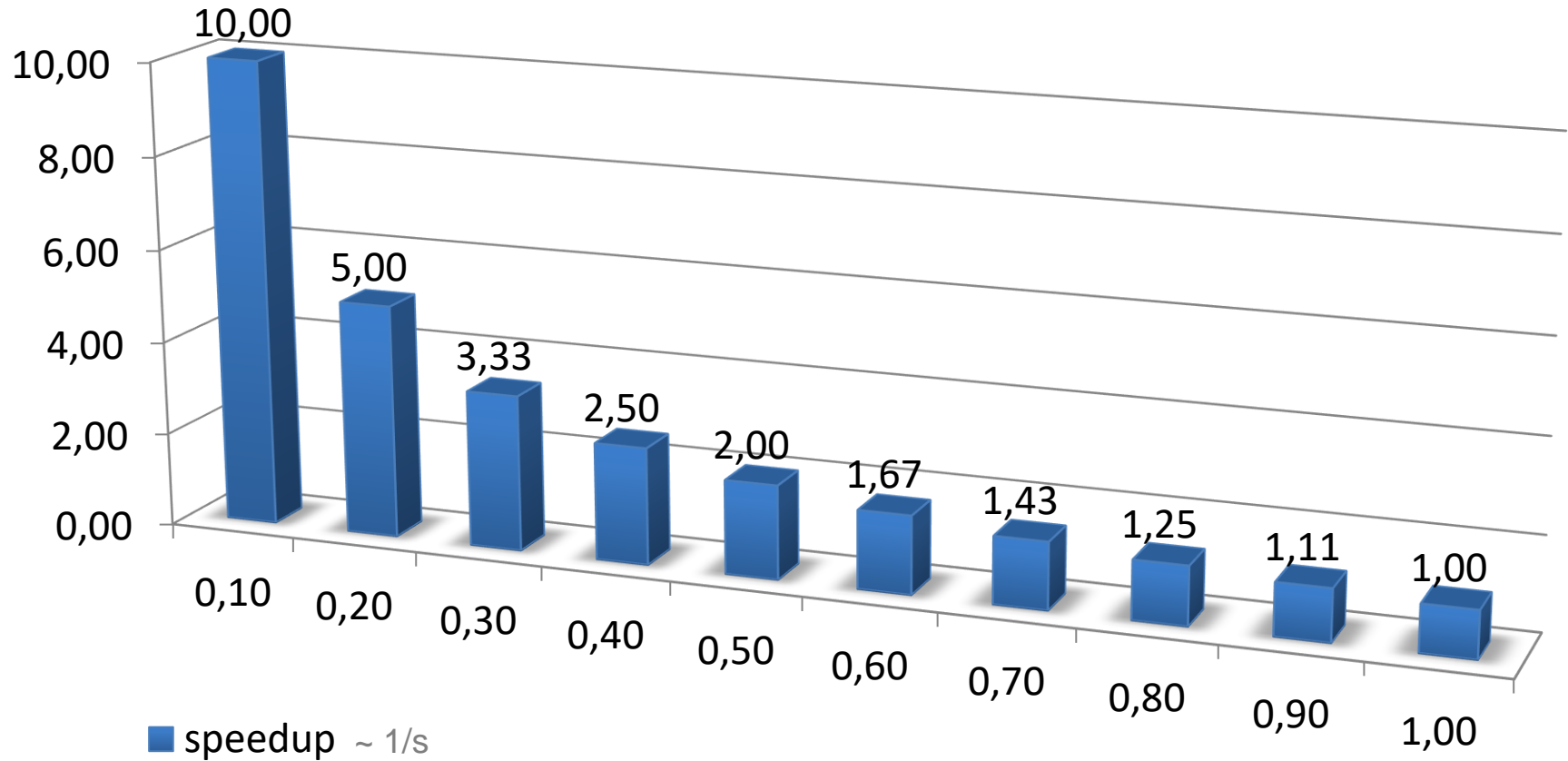
- Ley de Amdahl:

“el *speedup* teórico está limitado por la fracción secuencial s del programa”

$$speedup \leq \frac{1}{s + \frac{(1-s)}{n}}$$

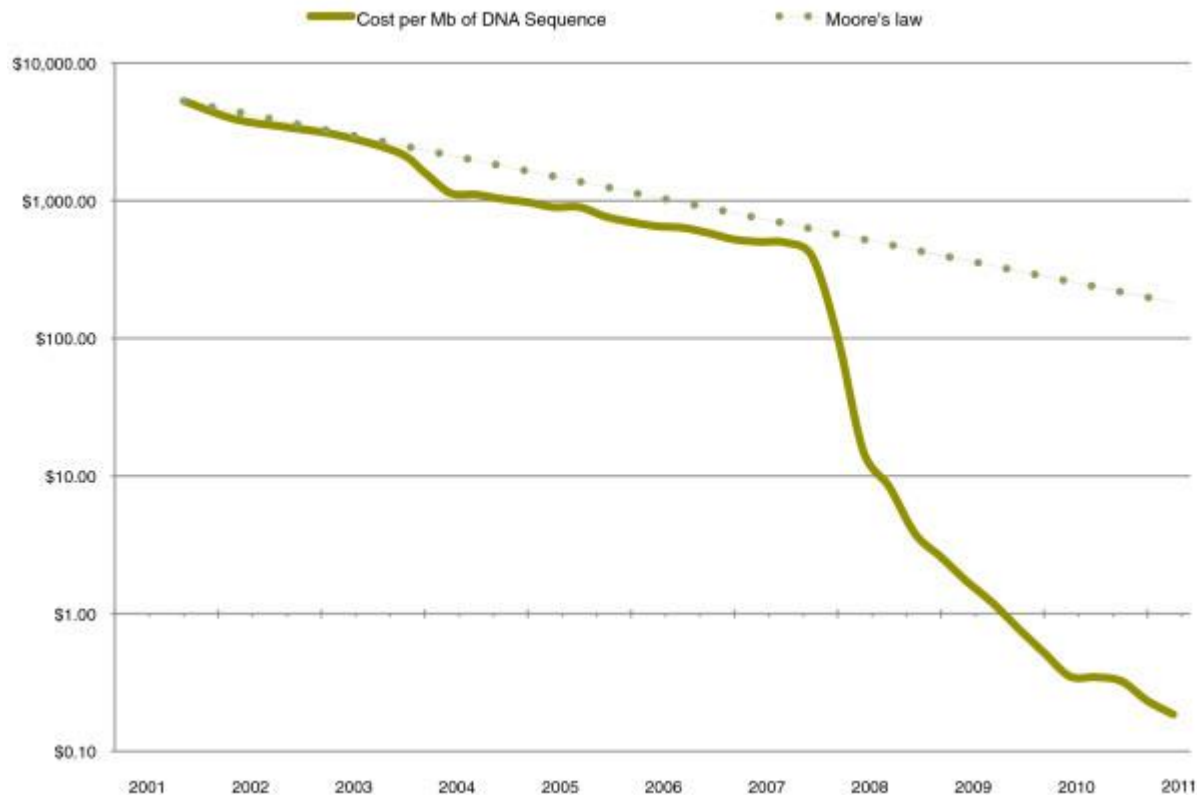
SI  $n \uparrow$  ENTONCES  $speedup \sim 1 / s$

# Ley de Amdahl



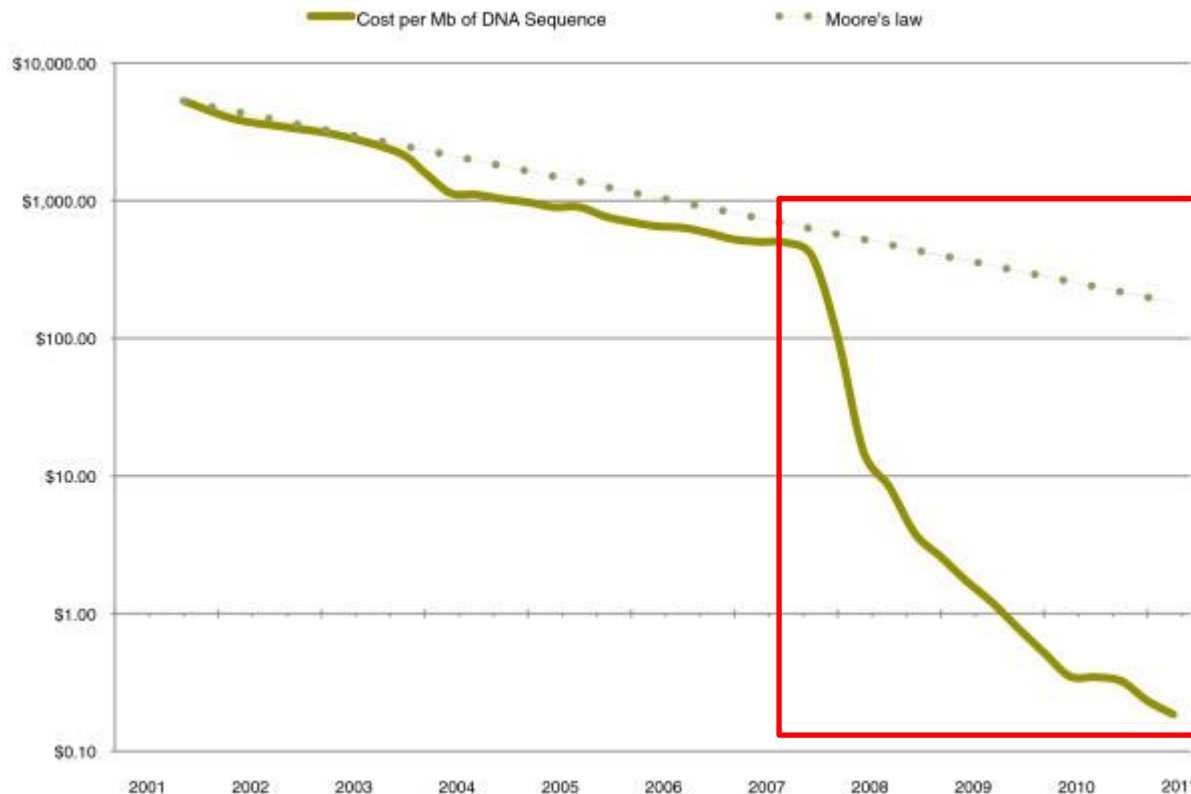
# ¿Eso del paralelismo ayuda?

## caso de estudio: genoma humano



# ¿Eso del paralelismo ayuda?

## caso de estudio: genoma humano



Yes!

# Computación de altas prestaciones



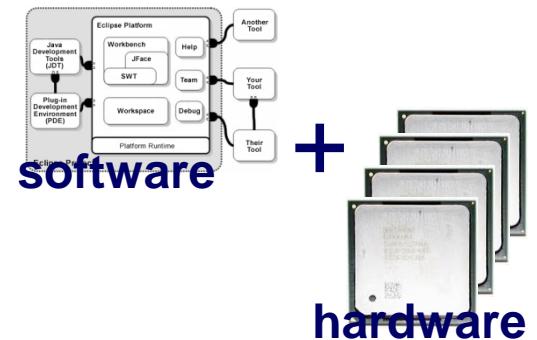
- Mejores algoritmos
  - $O(n^2)$ , viajante, ...



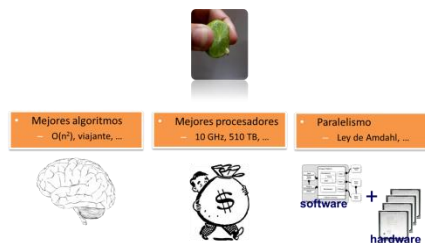
- Mejores procesadores
  - 10 GHz, 510 TB, ...



- Paralelismo
  - Ley de Amdahl, ...

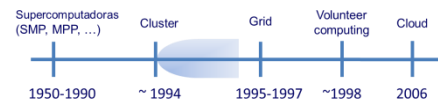


# Agenda



## Introducción a la computación de altas prestaciones

- Qué, dónde y cómo
- Hardware y software



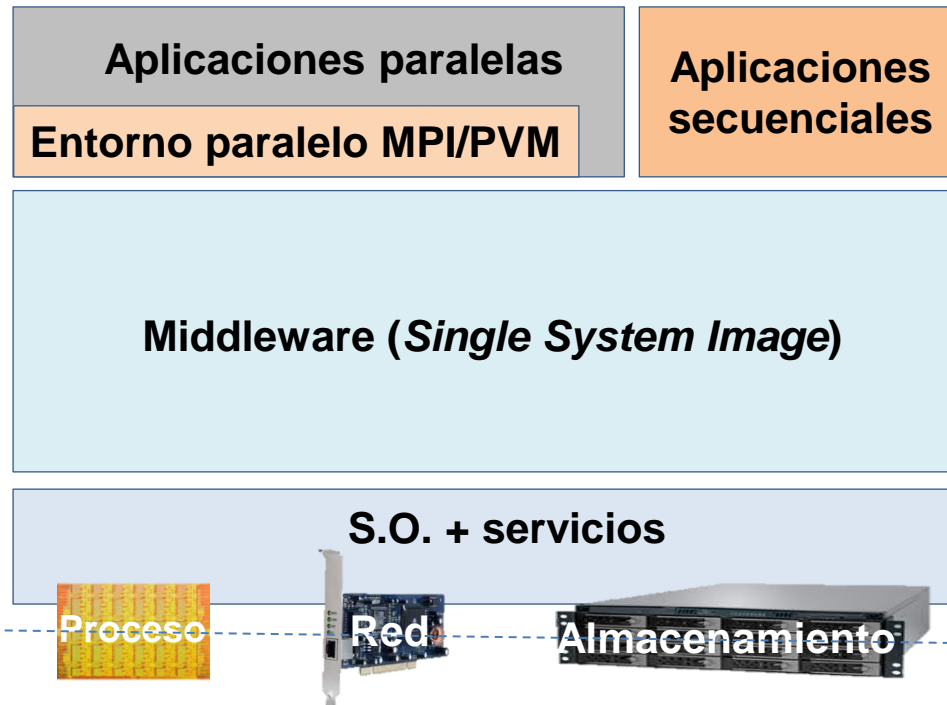
## Evolución de la computación de altas prestaciones

- Plataformas
- Tendencias



# Plataforma hardware y software

SW



HW



**Computador de altas prestaciones**

# Plataforma hardware



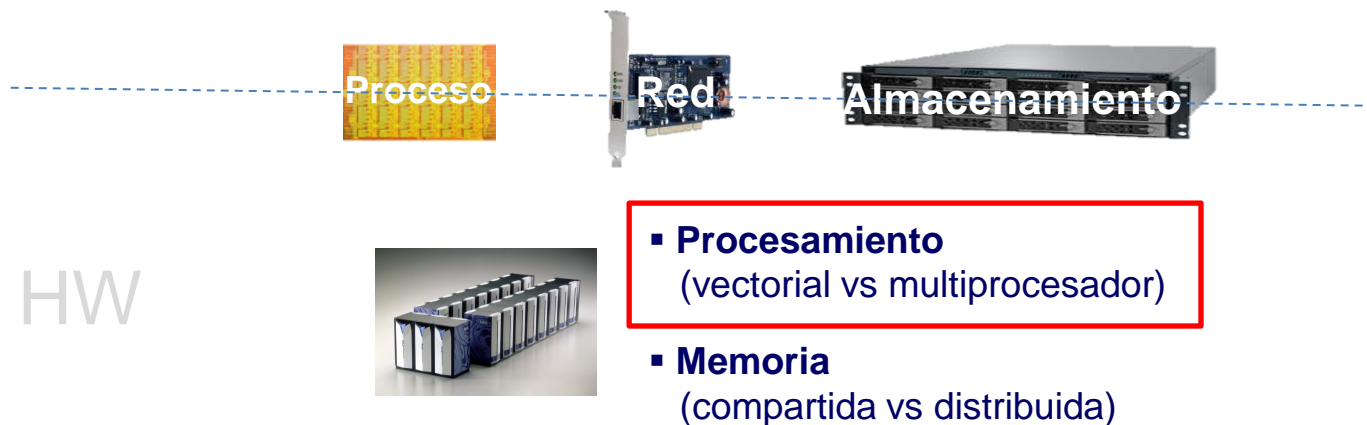
HW



- **Procesamiento**  
(vectorial vs multiprocesador)
- **Memoria**  
(compartida vs distribuida)



# Plataforma hardware



# Taxonomía de Flynn

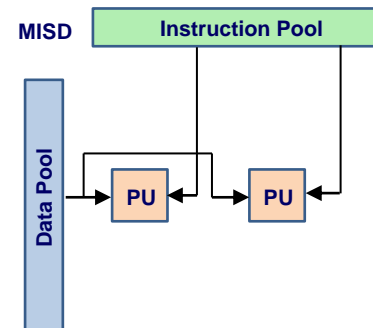
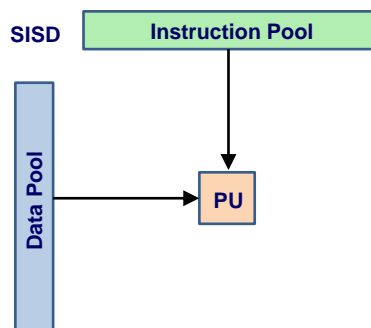
<http://www.buyya.com/microkernel/chap1.pdf>



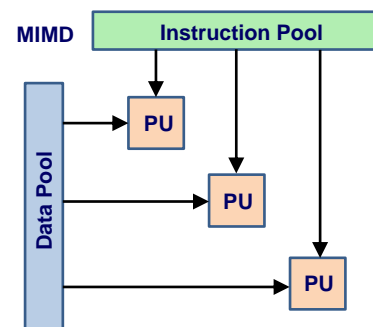
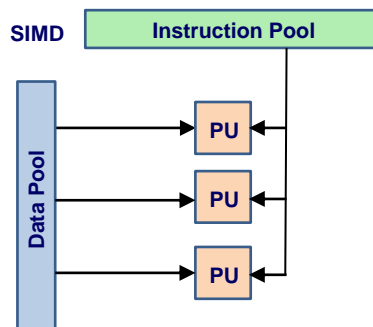
## Single Instruction

## Multiple Instruction

### Single Data



### Multiple Data



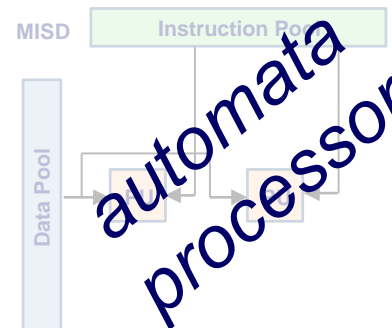
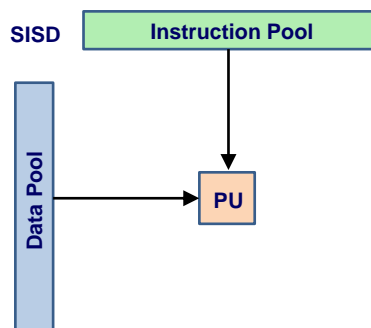
# Taxonomía de Flynn



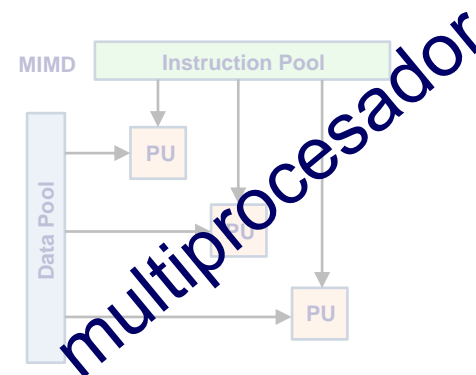
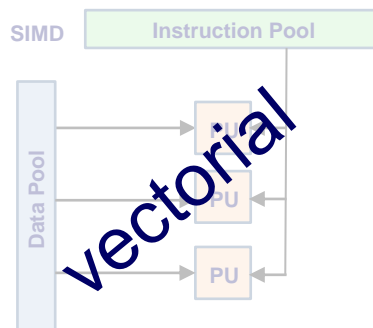
## Single Instruction

## Multiple Instruction

Single Data



Multiple Data



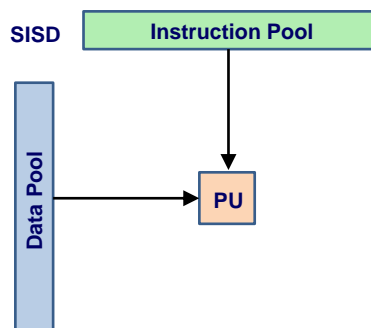
# Taxonomía de Flynn



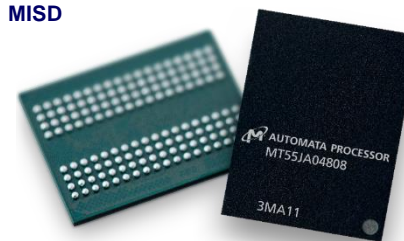
Single Instruction

Multiple Instruction

Single Data



MISD



*automata processor*

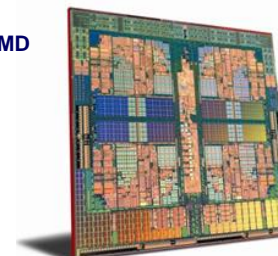
Multiple Data

SIMD



**vectorial**

MIMD



**multiprocesador**

# Plataforma hardware

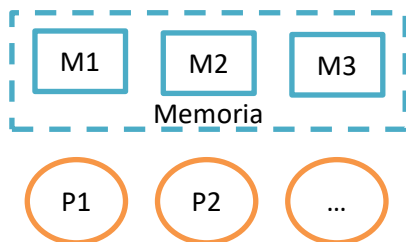
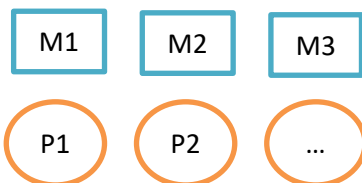
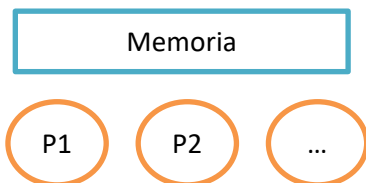


HW



- **Procesamiento**  
(vectorial vs multiprocesador)
- **Memoria**  
(compartida vs distribuida)

# Acceso a memoria



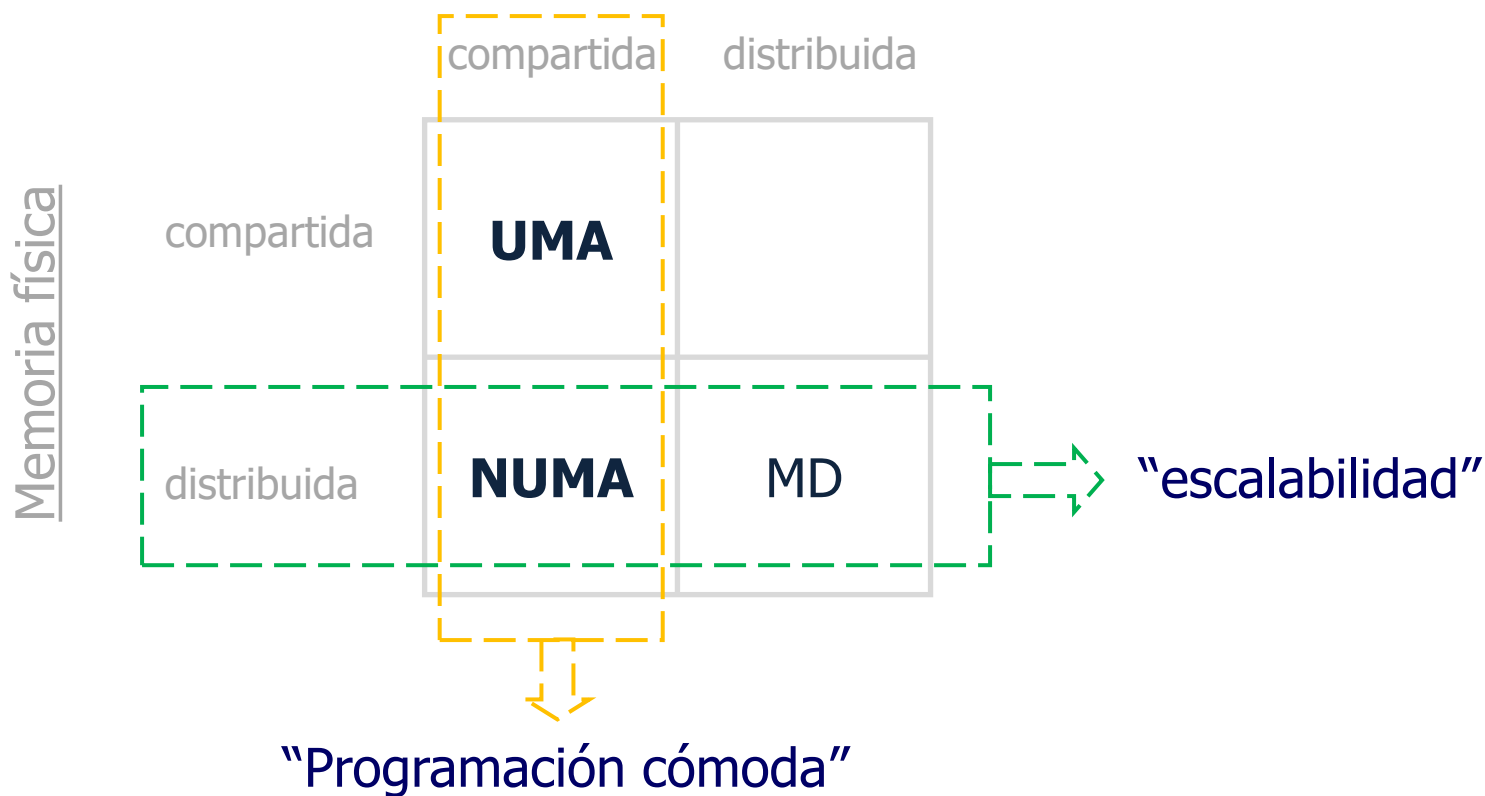
- Memoria compartida (UMA)
- Memoria distribuida (MD)
- Memoria lógicamente compartida (NUMA)

# Acceso a memoria



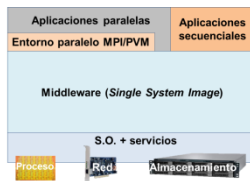
## Visión lógica de la memoria

(comunicación/sincronización)



# Plataforma software

SW

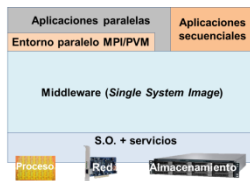


- Vectoriales
  - Uso de instrucciones especiales
- Multiprocesador
  - UMA, NUMA
    - OpenMP, ...
  - M. Distribuida
    - MPI, ...



# Plataforma software

SW



- Vectoriales
  - Uso de instrucciones especiales
- Multiprocesador
  - UMA, NUMA
    - OpenMP, ...
  - M. Distribuida
    - MPI, ...

# Cómo es OpenMP

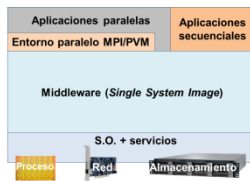
```
#include <omp.h>
#include <stdio.h>
#include <stdlib.h>

int main (int argc, char *argv[])
{
    #pragma omp parallel private(nthreads, tid)
    {
        int tid = omp_get_thread_num();
        printf("Hello World from thread = %d\n", tid);

        #pragma omp master
        {
            int nthreads = omp_get_num_threads();
            printf("Number of threads = %d\n", nthreads);
        }
    }
}
```

# Plataforma software

SW



- Vectoriales
  - Uso de instrucciones especiales
- Multiprocesador
  - UMA, NUMA
    - OpenMP, ...
  - M. Distribuida
    - MPI, ...

# Qué es MPI

- MPI es una interfaz de paso de mensaje que representa un esfuerzo prometededor de mejorar la disponibilidad de un software altamente eficiente y portable para satisfacer las necesidades actuales en la computación de alto rendimiento a través de la definición de un estándar de paso de mensajes universal.

**William D. Gropp et al.**

# Principales pilares de MPI

- **Portabilidad:**
  - Definido independiente de plataforma paralela.
  - Útil en arquitecturas paralelas heterogéneas.
- **Eficiencia:**
  - Definido para aplicaciones multihilo (*multithread*)
  - Sobre una comunicación fiable y eficiente.
  - Busca el máximo de cada plataforma.
- **Funcionalidad:**
  - Fácil de usar por cualquier programador que ya haya usado cualquier biblioteca de paso de mensajes.

# Implementaciones de MPI



## Open MPI 4.1.1 (24/4/2021)

- <http://www.open-mpi.org/>
- FT-MPI + LA-MPI + LAM/MPI + PACX-MPI

## MPICH2



## MPICH 3.4.2 (28/5/2021)

- <http://www.mpich.org/>
- Argonne National Laboratory & University of Chicago

# Cómo es MPI

```
#include <stdio.h>
#include "mpi.h"

main(int argc, char **argv)
{
    int node,size;
    int tam = 255;
    char name[255];

    MPI_Init(&argc, &argv);

    MPI_Comm_size (MPI_COMM_WORLD, &size);
    MPI_Comm_rank(MPI_COMM_WORLD, &node);
    MPI_Get_processor_name(name, &tam);
    printf("Hola Mundo2 del proceso %d de %d procesos (%s)\n", node, size, name);

    MPI_Finalize();
}
```

# Cómo es MPI: uso interactivo

```
bsc41729@login2:~/tmp> mpicc -g -o hello hello.c
```

```
bsc41729@login2:~/tmp> cat > machines
```

```
login1
```

```
login2
```

```
login3
```

```
login4
```

```
bsc41729@login2:~/tmp> mpirun -np 4 -machinefile machines hello
```

```
Hola Mundo2 del proceso 2 de 4 procesos (s41c3b03-gigabit1)
```

```
Hola Mundo2 del proceso 1 de 4 procesos (s41c3b02-gigabit1)
```

```
Hola Mundo2 del proceso 3 de 4 procesos (s41c3b04-gigabit1)
```

```
Hola Mundo2 del proceso 0 de 4 procesos (s41c3b01-gigabit1)
```



# Cómo es MPI: uso de PBS (1)

```
bsc41729@login2:~/tmp> cat hello.cmd
```

```
#!/bin/bash
```

```
# @ job_type = parallel
```

```
# @ class = q10
```

```
# @ group = bsc41
```

```
# @ initialdir = /home/bsc41/bsc41729/tmp/
```

```
# @ output = hello.out
```

```
# @ error = hello.err
```

```
# @ restart = no
```

```
# @ blocking = unlimited
```

```
# @ total_tasks = 2
```

```
# @ queue
```

```
# Program Execution.
```

```
mpirun -np 2 \
```

```
-machinefile $LL_MACHINE_LIST /home/bsc41/bsc41729/tmp/hello
```

## Cómo es MPI: uso de PBS (2)

```
bsc41729@login2:~/tmp> llsubmit hello.cmd
```

```
llsubmit: Processed command file through Submit Filter: "/etc/perf/load1/scripts/llsubmit".
llsubmit: The job "s42-gigabit1.mn.406842" has been submitted.
```

```
bsc41729@login2:~/tmp> llq
```

Id	Owner	Submitted	ST	PRI	Class	Running On
s42-gigabit1.404704.0	bsc41729	11/27 12:19	R	50	q09	s06c4b11-gigabit1
s42-gigabit1.404731.0	bsc41729	11/27 12:32	R	50	q09	s07c1b10-gigabit1
s42-gigabit1.404732.0	bsc41729	11/27 12:32	R	50	q09	s06c4b03-gigabit1
s42-gigabit1.404736.0	bsc41729	11/27 12:34	I	50	q09	
s42-gigabit1.406842.0	bsc41729	11/27 17:18	I	50	q10	

```
4 job step(s) in query, 1 waiting, 0 pending, 3 running, 0 held, 0 preempted
```

## Cómo es MPI: uso de PBS (3)

```
bsc41729@login2:~/tmp> cat hello.out
```

```
Program binary is: /home/bsc41/bsc41729/tmp/hello
```

```
Machines file is /gpfs/projects/bsc99/perf/restricted/spool/049/mlist/s42-gigabit1.mn.406849.0.machine_list
```

```
Shared memory for intra-nodes coms is enabled.
```

```
GM receive mode used: polling.
```

```
2 processes will be spawned:
```

```
Process 0 (/home/bsc41/bsc41729/tmp/hello ) on s06c1b03-gigabit1.mn
```

```
Process 1 (/home/bsc41/bsc41729/tmp/hello ) on s06c1b03-gigabit1.mn
```

```
Open a socket on s06c1b03...
```

```
Got a first socket opened on port 33735.
```

```
Shared memory file: /tmp/gmpi_shmem-811134:[0-9]*.tmp
```

```
Hola Mundo2 del proceso 1 de 2 procesos (s06c1b03-gigabit1)
```

```
Hola Mundo2 del proceso 0 de 2 procesos (s06c1b03-gigabit1)
```

```
MPI Id 1 is using GM port 2, board 0 (MAC 0060dd4846f0).
```

```
MPI Id 0 is using GM port 4, board 0 (MAC 0060dd4846f0).
```

```
Received data from all 2 MPI processes.
```

```
Sending mapping to MPI Id 0.
```

```
Sending mapping to MPI Id 1.
```

```
Data sent to all processes.
```

```
Reap remote processes:
```

```
All remote MPI processes have exited.
```

## Cómo es MPI: uso de PBS (4)

```
bsc41729@login2:~/tmp> llclass -limits
```

class name	group name	job nodes	job tasks	max jobs	max idle	max tasks	wall clock time
-----	-----	-----	-----	-----	-----	-----	-----
debug	bsc41	32	64	80	8	256	00:10:00
interactive	bsc41	1	1	14	8	14	02:00:00
papi	bsc41	320	512	80	8	640	12:00:00
q09	bsc41	256	512	80	8	640	48:00:00
q10	bsc41	256	512	80	8	640	48:00:00

# MPI 2.2 – 4.0

(<http://mpi-forum.org/docs/>)

- Estructuras de datos
  - Tipos de datos (básicos, vectores, compuestos, ...)
  - Grupo de procesos (grupos, comunicadores, ...)
- Paso de mensajes
  - Llamadas punto a punto (bloqueantes, ...)
  - Llamadas colectivas (*bcast*, *scatter*, *gather*, ...)
- Entrada y salida
  - Gestión de ficheros (apertura, cierre, ...)
  - Gestión de contenidos (vistas, punteros, ...)
- Procesos
  - Gestión de procesos (creación, ...)
  - *Profiling*

# Send-receive en MPI

```
#include <stdio.h>
#include "mpi.h"

main(int argc, char **argv) {
    int node, size;
    int tam = 255;
    char name[255];
    int num = 10;

    MPI_Init(&argc, &argv);
    MPI_Comm_size(MPI_COMM_WORLD, &size);
    MPI_Comm_rank(MPI_COMM_WORLD, &node);

    if (node == 0)
        MPI_Send(&num, 1 MPI_INT, 1, 0, MPI_COMM_WORLD);
    else
        MPI_Recv(&num, 1 MPI_INT, 0, 0, MPI_COMM_WORLD);

    MPI_Finalize();
}
```

# Send-receive en MPI

```
#include <stdio.h>
#include "mpi.h"
```

```
main(int argc, char **argv) {
    int node, size;
    int tam = 255;
    char name[255];
    int num = 10;
```

```
    MPI_Init(&argc, &argv);
```

```
    MPI_Comm_size(MPI_COMM_WORLD, &size);
```

```
    MPI_Comm_rank(MPI_COMM_WORLD, &node);
```

```
    if (node == 0)
```

```
        MPI_Send(&num, 1 MPI_INT, 1, 0, MPI_COMM_WORLD);
```

```
    else
```

```
        MPI_Recv(&num, 1 MPI_INT, 0, 0, MPI_COMM_WORLD);
```

```
    MPI_Finalize();
```

```
}
```

Tipo de datos

MPI\_CHAR  
MPI\_BYTE  
MPI\_INT  
MPI\_FLOAT  
....

Tipos de datos derivados

Dato a enviar

Proceso destinatario

Etiqueta asociada al  
mensaje

comunicador

Número de elementos

# Send-receive en MPI

```
#include <stdio.h>
#include "mpi.h"
```

```
main(int argc, char **argv) {
    int node, size;
    int tam = 255;
    char name[255];
    int num = 10;
```

```
    MPI_Init(&argc, &argv);
```

```
    MPI_Comm_size(MPI_COMM_WORLD, &size);
```

```
    MPI_Comm_rank(MPI_COMM_WORLD, &node);
```

```
    if (node == 0)
```

```
        MPI_Send(&num, 1 MPI_INT, 1, 0, MPI_COMM_WORLD);
```

```
    else
```

```
        MPI_Recv(&num, 1 MPI_INT, 0, 0, MPI_COMM_WORLD);
```

```
    MPI_Finalize();
```

```
}
```

Tipo de datos

MPI\_CHAR  
MPI\_BYTE  
MPI\_INT  
MPI\_FLOAT  
....

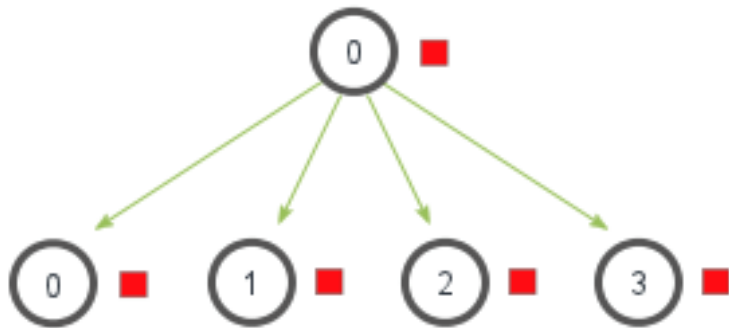
Tipos de datos derivados

Proceso emisor del mensaje

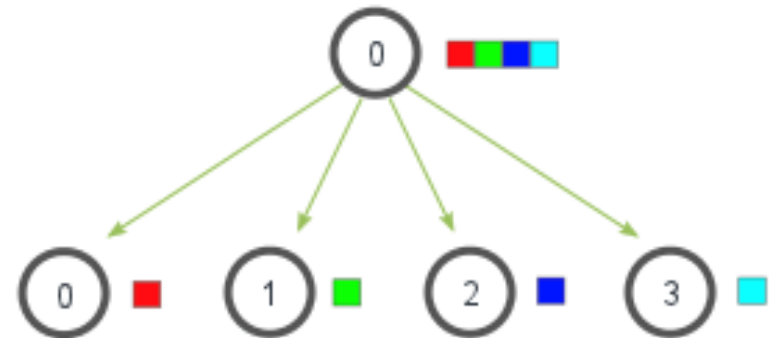


# Operaciones colectivas

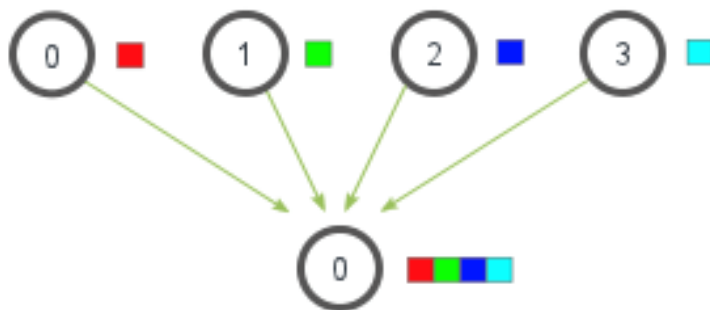
MPI\_Bcast



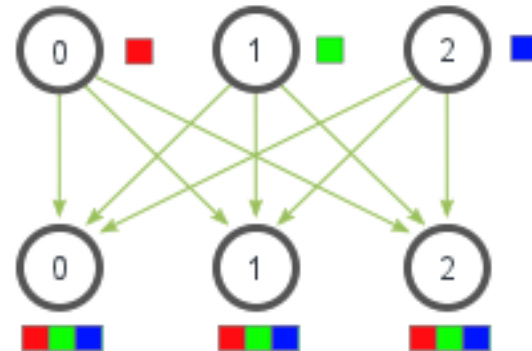
MPI\_Scatter



MPI\_Gather

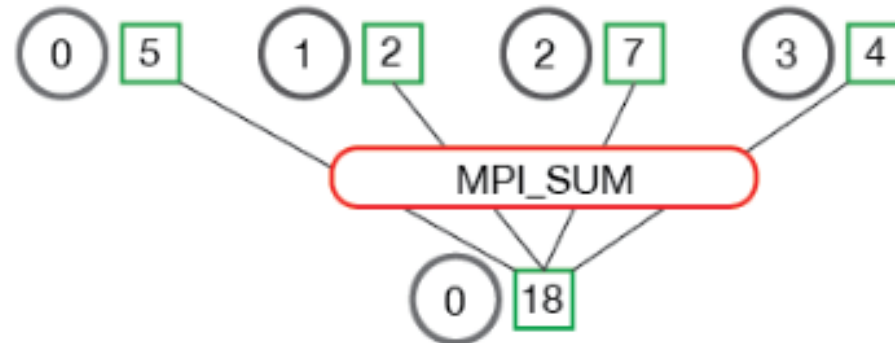


MPI\_Allgather

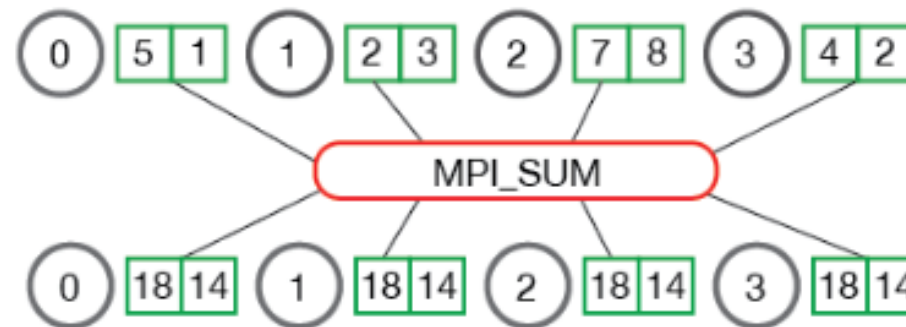


# Operaciones colectivas de reducción

MPI\_Reduce



MPI\_Allreduce



# Operaciones colectivas en MPI

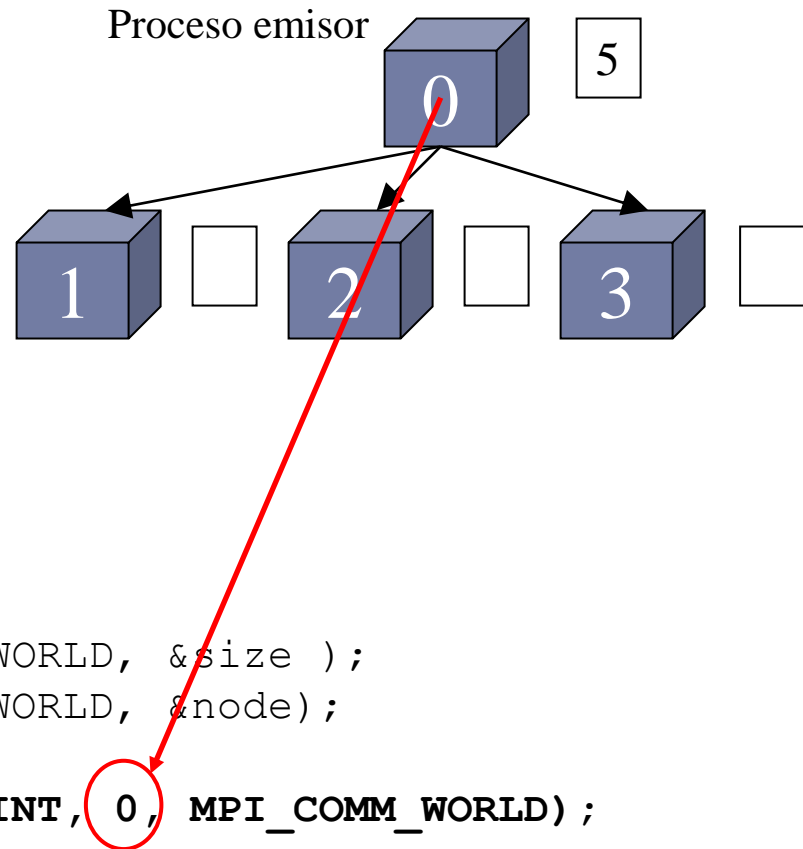
```
#include <stdio.h>
#include "mpi.h"

main (int argc, char **argv)
{
    int node, size;
    int tam = 255;
    char name[255];
    int num = 5;

    MPI_Init(&argc, &argv);
    MPI_Comm_size(MPI_COMM_WORLD, &size);
    MPI_Comm_rank(MPI_COMM_WORLD, &node);

    MPI_Bcast(&num, 1, MPI_INT, 0, MPI_COMM_WORLD);

    MPI_Barrier(MPI_COMM_WORLD);
    printf("El proceso %d recibe %d\n", node, num);
    MPI_Finalize();
}
```



# Operaciones colectivas en MPI

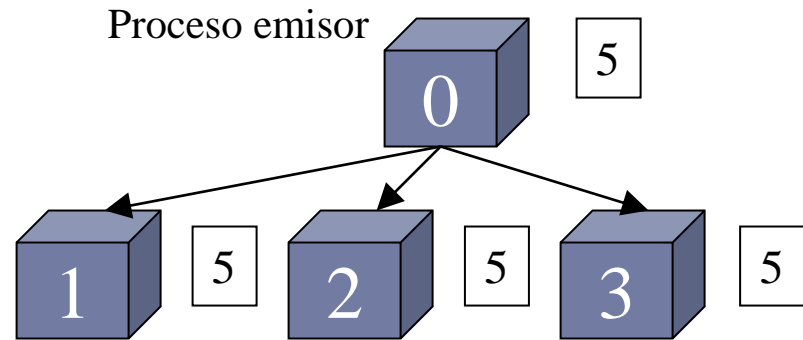
```
#include <stdio.h>
#include "mpi.h"

main (int argc, char **argv)
{
    int node, size;
    int tam = 255;
    char name[255];
    int num = 5;

    MPI_Init(&argc, &argv);
    MPI_Comm_size(MPI_COMM_WORLD, &size);
    MPI_Comm_rank(MPI_COMM_WORLD, &node);

    MPI_Bcast(&num, 1, MPI_INT, 0, MPI_COMM_WORLD);

    MPI_Barrier(MPI_COMM_WORLD);
    printf("El proceso %d recibe %d\n", node, num);
    MPI_Finalize();
}
```



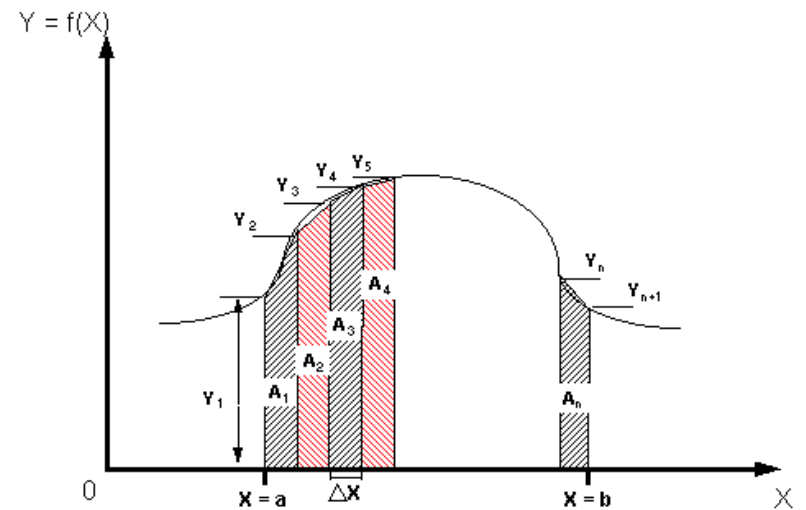
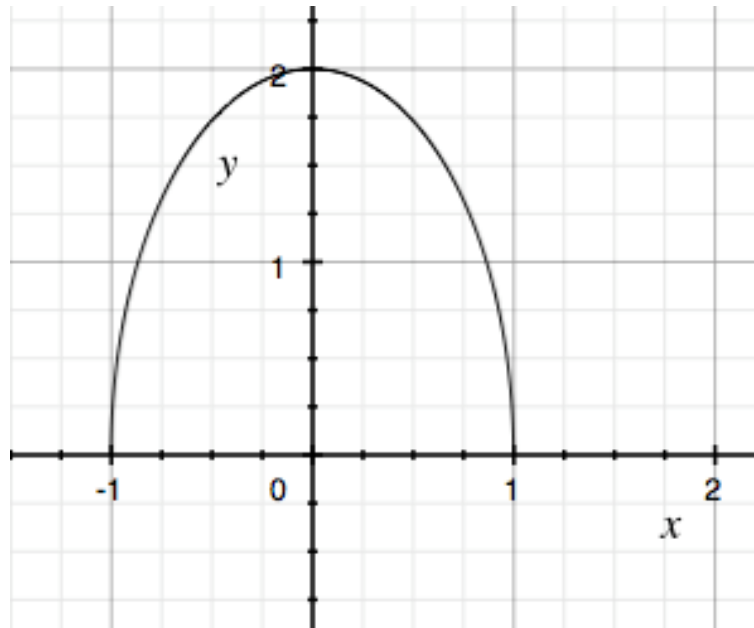
El resto de procesos recibe en num el mensaje enviado por el proceso 0

```
MPI_Bcast(&num, 1, MPI_INT, 0, MPI_COMM_WORLD);
```

```
MPI_Barrier(MPI_COMM_WORLD);
printf("El proceso %d recibe %d\n", node, num);
MPI_Finalize();
```

# Ejemplo: Cálculo de $\pi$

$$\int_0^1 \sqrt{4(1-x^2)} \, dx = \frac{\pi}{2}$$



# Cálculo secuencial

```
#include <stdio.h>
#include <math.h>
#include <stdio.h>

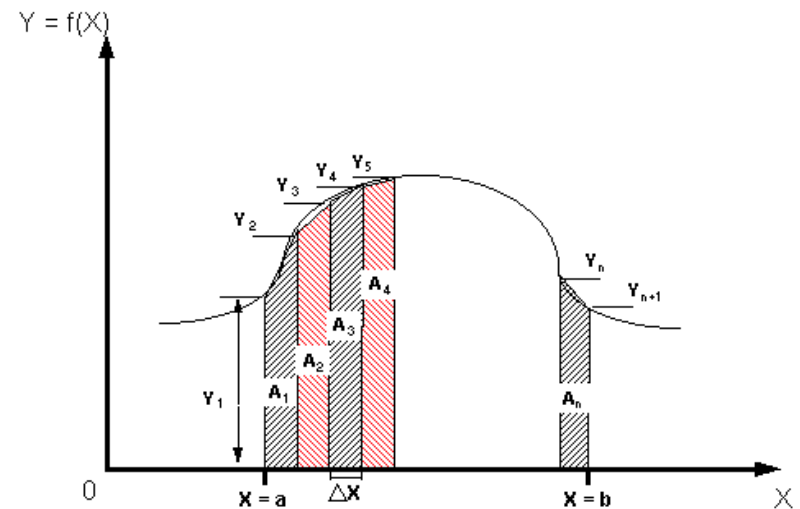
#define N 1E7
#define d 1E-7

int main (int argc, char* argv[]) {
    int i;
    double pi=0.0, result=0.0, sum=0.0, x2, x =0, s;

    result = 0;
    for (i=0; i<N; i++) {
        x2 = x * x;
        s = sqrt(1-x2) * d;
        result = result + s;
        x = x + d;
    }

    pi=4*result;
    printf("PI=%lf\n", pi);

    return 0;
}
```



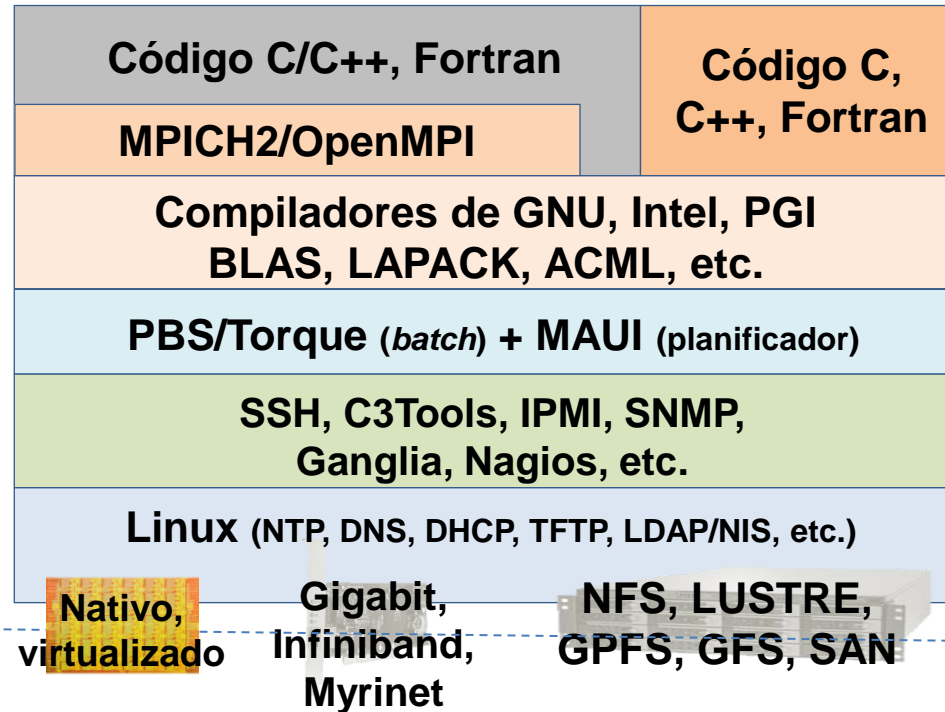
# Cálculo paralelo

```
#include "mpi.h"
#include <math.h>
int main(int argc, char *argv[])
{
    #define n 1E7
    int myid, numprocs, i;
    double mypi, pi, h, sum, x;

    MPI_Init(&argc, &argv);
    MPI_Comm_size(MPI_COMM_WORLD, &numprocs);
    MPI_Comm_rank(MPI_COMM_WORLD, &myid);
    MPI_Bcast(&n, 1, MPI_INT, 0, MPI_COMM_WORLD);
    h = 1.0 / (double) n;
    sum = 0.0;
    for (i = myid + 1; i <= n; i += numprocs) {
        x = h * ((double)i - 0.5);
        sum += 4.0 / (1.0 + x*x);
    }
    mypi = h * sum;
    MPI_Reduce(&mypi, &pi, 1, MPI_DOUBLE, MPI_SUM, 0, MPI_COMM_WORLD);
    if (myid == 0)
        printf("pi is approximately %.16f\n", pi);
    MPI_Finalize();
    return 0;
}
```

# Plataforma hardware y software

SW



HW

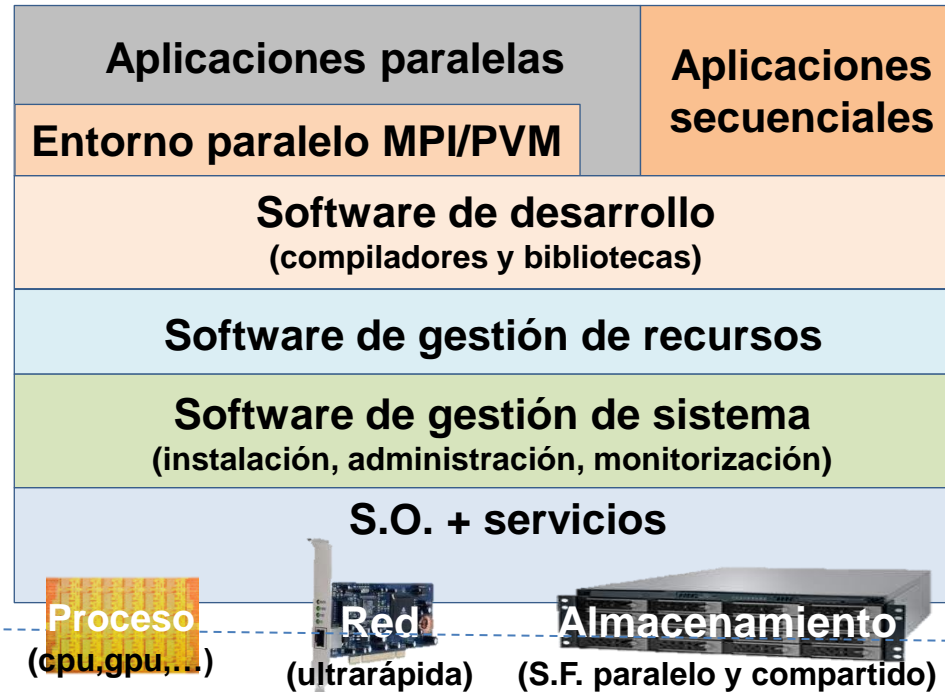


Supercomputador



# Plataforma hardware y software

SW



HW



**Supercomputador**

# Top 500 Junio 2021

(<http://www.top500.org>)

Rank	Site	Cores	R <sub>max</sub> (TFLOP/s)	R <sub>peak</sub> (TFLOP/s)	Power (kW)
1	<a href="#">Supercomputer Fugaku</a> - Supercomputer Fugaku, A64FX 48C 2.2GHz, Tofu interconnect D, Fujitsu <a href="#">RIKEN Center for Computational Science</a> Japan	7,630,848	442,010.0	537,212.0	29,899
2	<a href="#">Summit</a> - IBM Power System AC922, IBM POWER9 22C 3.07GHz, NVIDIA Volta GV100, Dual-rail Mellanox EDR <a href="#">Infiniband</a> , IBM <a href="#">DOE/SC/Oak Ridge National Laboratory</a> United States	2,414,592	148,600.0	200,794.9	10,096
3	<a href="#">Sierra</a> - IBM Power System AC922, IBM POWER9 22C 3.1GHz, NVIDIA Volta GV100, Dual-rail Mellanox EDR <a href="#">Infiniband</a> , IBM / NVIDIA / Mellanox <a href="#">DOE/NNSA/LLNL</a> United States	1,572,480	94,640.0	125,712.0	7,438
4	<a href="#">Sunway TaihuLight</a> - Sunway MPP, Sunway SW26010 260C 1.45GHz, Sunway, NRCPC <a href="#">National Supercomputing Center in Wuxi</a> China	10,649,600	93,014.6	125,435.9	15,371
5	<a href="#">Perlmutter</a> - HPE Cray EX235n, AMD EPYC 7763 64C 2.45GHz, NVIDIA A100 SXM4 40 GB, Slingshot-10, HPE <a href="#">DOE/SC/LBNL/NERSC</a> United States	706,304	64,590.0	89,794.5	2,528
6	<a href="#">Selene</a> - NVIDIA DGX A100, AMD EPYC 7742 64C 2.25GHz, NVIDIA A100, Mellanox HDR Infiniband, Nvidia <a href="#">NVIDIA Corporation</a> United States	555,520	63,460.0	79,215.0	2,646
7	<a href="#">Tianhe-2A</a> - TH-IVB-FEP Cluster, Intel Xeon E5-2692v2 12C 2.2GHz, TH Express-2, Matrix-2000, NUDT <a href="#">National Super Computer Center in Guangzhou</a> China	4,981,760	61,444.5	100,678.7	18,482
8	<a href="#">JUWELS Booster Module</a> - Bull Sequana XH2000, AMD EPYC 7402 24C 2.8GHz, NVIDIA A100, Mellanox HDR <a href="#">InfiniBand/ParTec ParaStation ClusterSuite</a> , Atos <a href="#">Forschungszentrum Juelich (FZJ)</a> Germany	449,280	44,120.0	70,980.0	1,764
9	<a href="#">HPC5</a> - PowerEdge C4140, Xeon Gold 6252 24C 2.1GHz, NVIDIA Tesla V100, Mellanox HDR Infiniband, Dell EMC <a href="#">Eni S.p.A.</a> Italy	669,760	35,450.0	51,720.8	2,252
10	<a href="#">Frontera</a> - Dell C6420, Xeon Platinum 8280 28C 2.7GHz, Mellanox InfiniBand HDR, Dell EMC <a href="#">Texas Advanced Computing Center/Univ. of Texas</a> United States	448,448	23,516.4	38,745.9	

# Top 500

(country=es)

- Junio 2014

Rank	Site	System	Cores	Rmax (TFlop/s)	Rpeak (TFlop/s)	Power (kW)
41	<a href="#">Barcelona Supercomputing Center</a> Spain	<a href="#">MareNostrum - iDataPlex DX360M4, Xeon E5-2670 8C 2.600GHz, Infiniband FDR</a> , IBM	48,896	925.1	1,017.0	1,015.6
168	<a href="#">Instituto Tecnológico y de Energías Renovables S.A.</a> Spain	<a href="#">TEIDE-HPC - Fujitsu PRIMERGY CX250 S1, Xeon E5-2670 8C 2.600GHz, Infiniband QDR</a> , Fujitsu	16,384	274.0	340.8	312

- Junio 2015

77	<a href="#">Barcelona Supercomputing Center</a> Spain	<a href="#">MareNostrum - iDataPlex DX360M4, Xeon E5-2670 8C 2.600GHz, Infiniband FDR</a> , IBM	48,896	925.1	1,017.0	1,015.6
259	<a href="#">Instituto Tecnológico y de Energías Renovables S.A.</a> Spain	<a href="#">TEIDE-HPC - Fujitsu PRIMERGY CX250 S1, Xeon E5-2670 8C 2.600GHz, Infiniband QDR</a> , Fujitsu	16,384	274.0	340.8	312

- Junio 2017

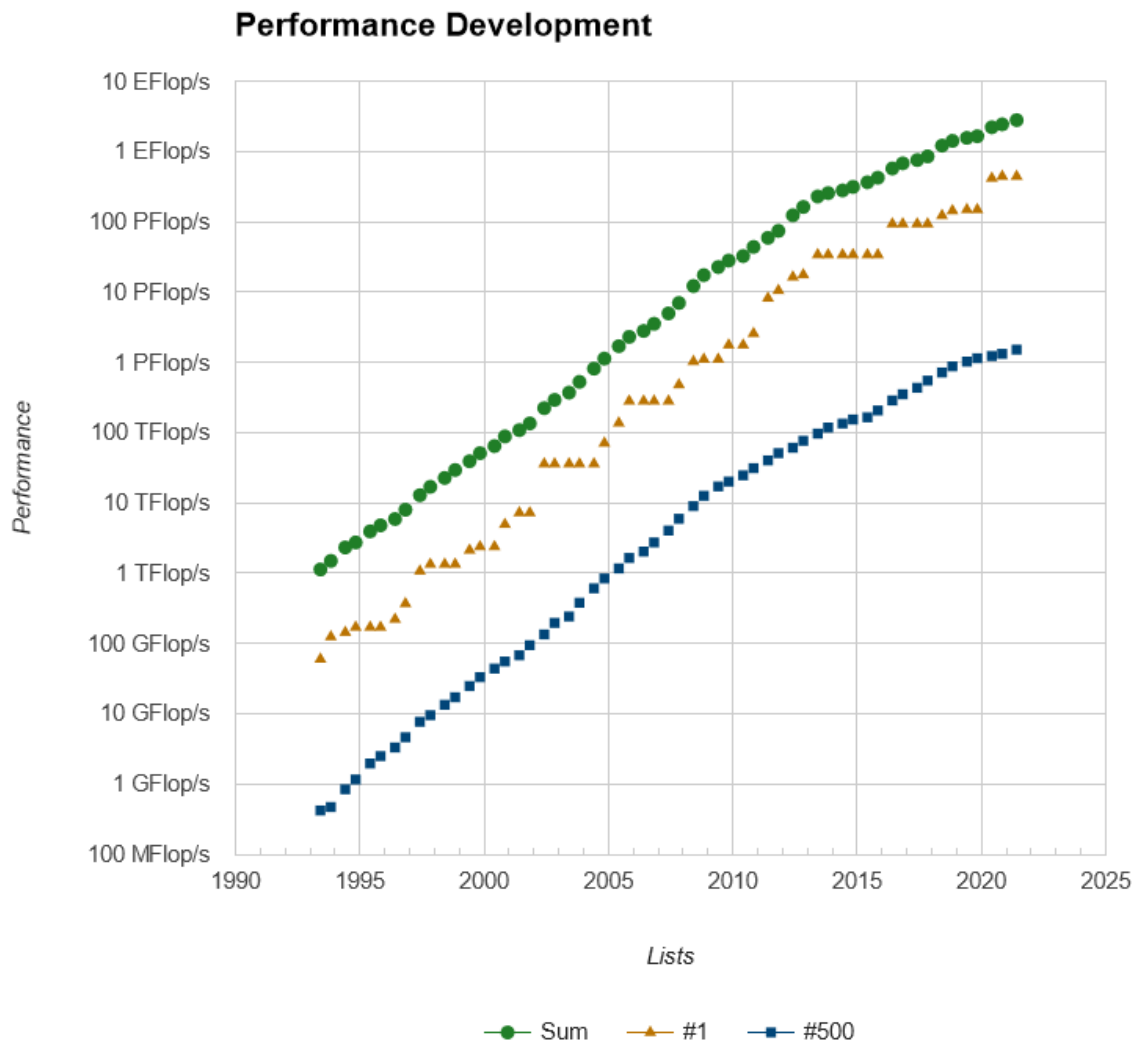
13	<a href="#">Barcelona Supercomputing Center</a> Spain	<a href="#">MareNostrum - Lenovo SD530, Xeon Platinum 8160 24C 2.1GHz, Intel Omni-Path</a> , Lenovo	148,176	6,227.2	9,957.4	1,380
----	--	---	---------	---------	---------	-------

- Junio ~~2020~~ 2021

<del>37</del> 63	<a href="#">Barcelona Supercomputing Center</a> Spain	<a href="#">MareNostrum - Lenovo SD530, Xeon Platinum 8160 24C 2.1GHz, Intel Omni-Path</a> , Lenovo	153,216	6,470.8	10,296.1	1,632
---------------------	--	---	---------	---------	----------	-------

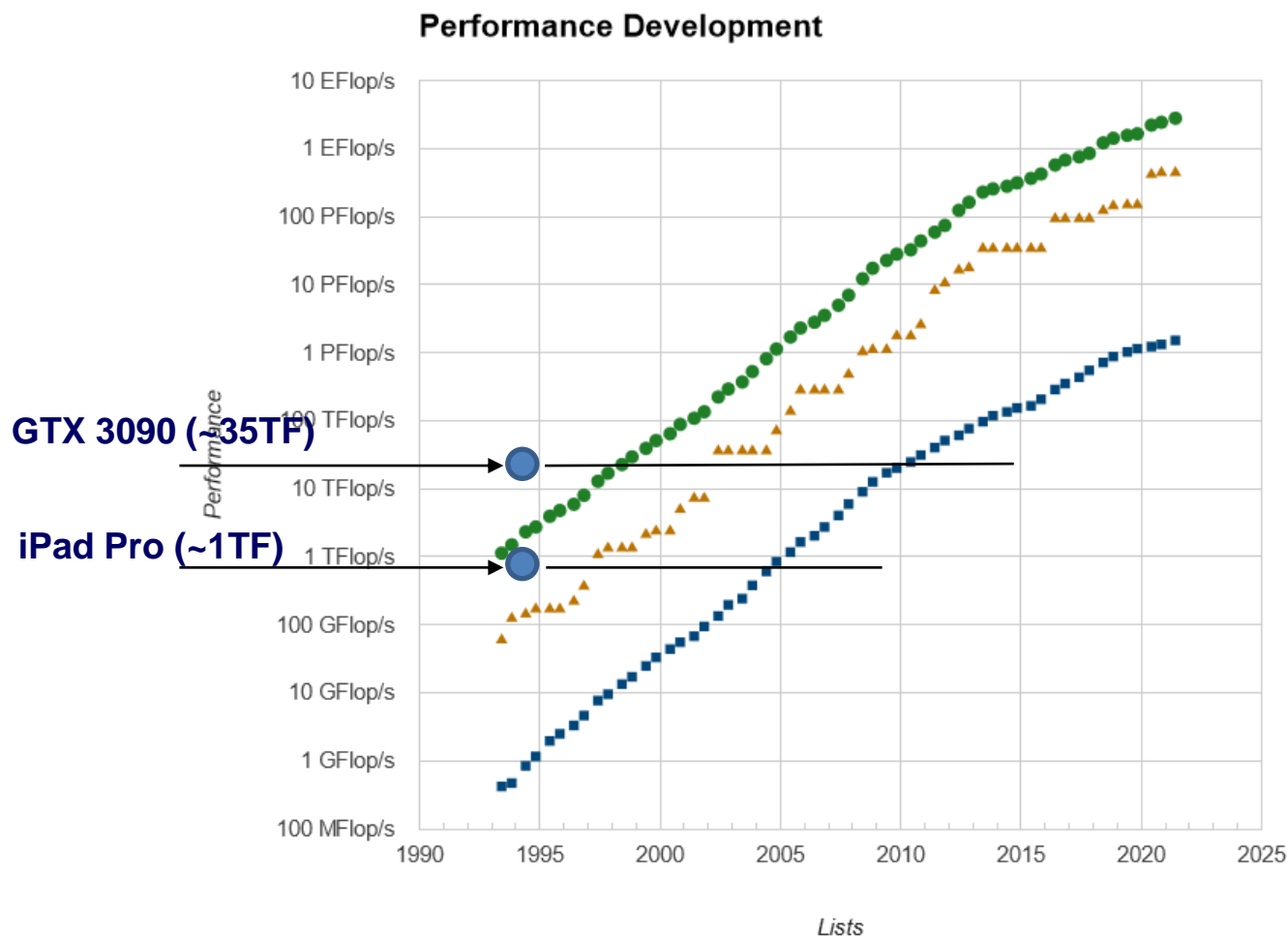
# Top 500 Junio 2021

(<http://top500.org/statistics/perfdevel/>)



# Top 500 Junio 2021

(<http://top500.org/statistics/perfdevel/>)

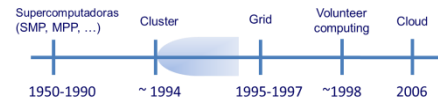


# Agenda



## Introducción a la computación de altas prestaciones

- Qué, dónde y cómo
- Hardware y software



## Evolución de la computación de altas prestaciones

- Plataformas
- Tendencias



# Evolución en las plataformas de computación de altas prestaciones

- Problemas con gran cantidad de cómputo
- Más usado en ciencia y ejército
- Uso de paralelismo masivo



Supercomputadoras  
(SMP, MPP, Sistólico, Array, ...)



# Evolución en las plataformas de computación de altas prestaciones

- Problemas con gran cantidad de datos tratados
- Más usado en administración
- Uso de paralelismo y alta frecuencia



Supercomputadoras & Mainframes  
(SMP, MPP, Sistólico, Array, ...)





# Evolución en las plataformas de computación de altas prestaciones

- Construido por Donald Becker y Thomas Sterling en 1994 (NASA)
- Formado por 16 computadores personales con procesador intel DX4 a 200 MHz interconectados por un switch Ethernet.
- Rendimiento teórico era de 3,2 Gflops
- Posibilidad de supercomputadoras "baratas"



Supercomputadoras

(SMP, MPP, Sistólico, Array, ...)

Cluster



# Evolución en las plataformas de computación de altas prestaciones

- Construido por Donald Becker y Thomas Sterling en 1994 (NASA)
- Formado por 16 computadores personales con procesador intel DX4 a 200 MHz interconectados por un switch Ethernet.
- Rendimiento teórico era de 3,2 Gflops
- Posibilidad de supercomputadoras "baratas"

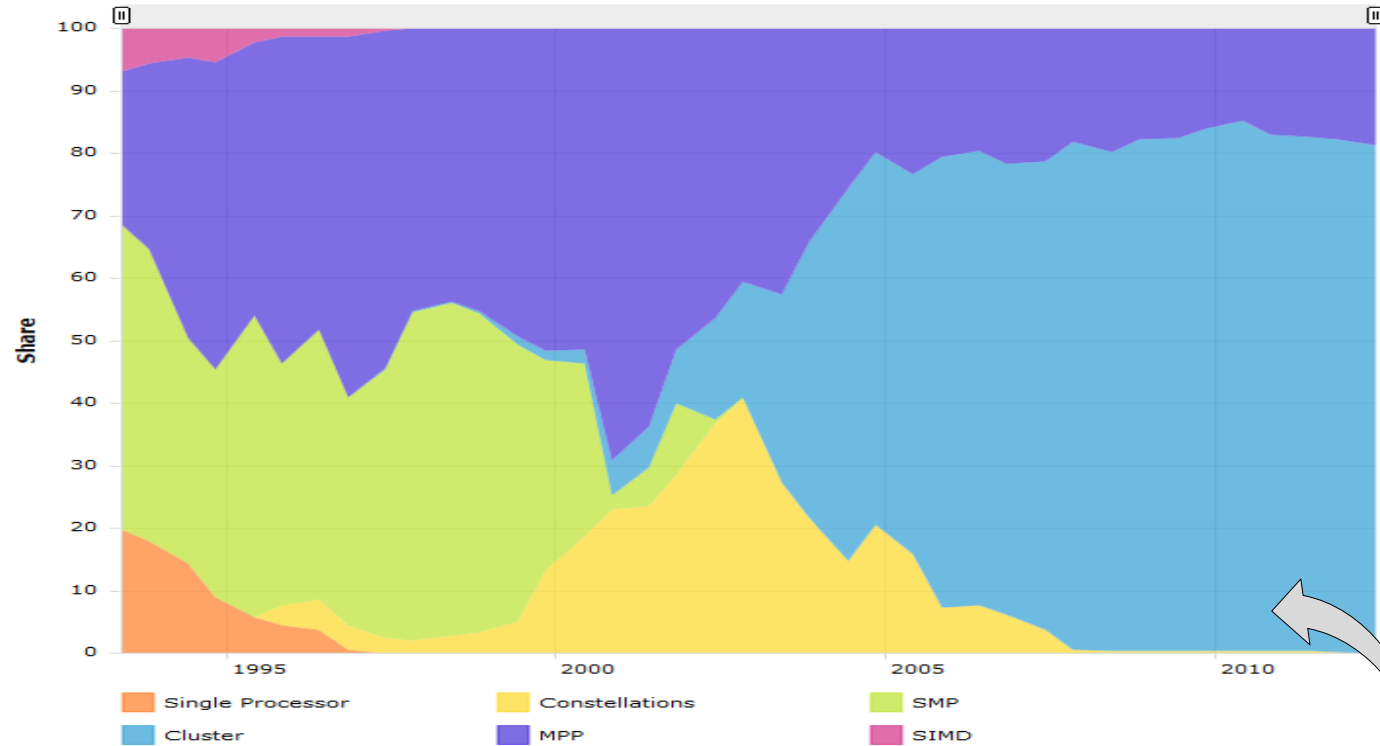


Supercomputadoras  
(SMP, MPP, Sistólico, Array, ...)

Cluster



## Architecture - Systems Share



Supercomputadoras  
(SMP, MPP, Sistólico, Array, ...)

Cluster





[Click to learn more](#)

1997: THE FIRST INTEL® TERAFLAP COMPUTER  
consisted of:

**9,298** INTEL  
PROCESSORS

and occupied:

**72** SERVER  
CABINETS

THE INTEL® XEON® PHI™ COPROCESSOR  
will provide:

**1** TERAFLAP OF  
PERFORMANCE

and occupy:

**1** PCIe  
SLOT



Supercomputadoras  
(SMP, MPP, Sistólico, Array, ...)

Cluster



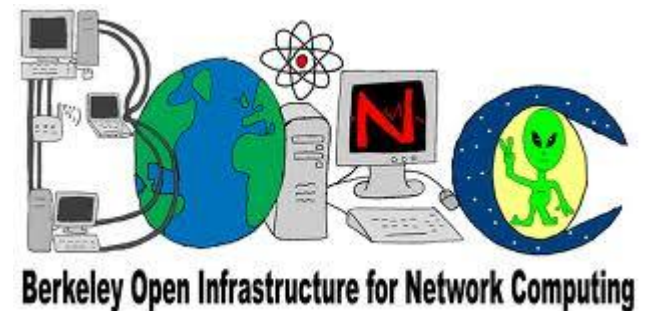
# Evolución en las plataformas de computación de altas prestaciones

- Antecesor: metacomputing por Larry Smarr (NCSA) al inicio de los 80
  - Centros de supercomputación interconectados: más recursos disponibles
  - I-WAY demostrado en 1995
- Grid aparece en un seminario dado en 1997 en ANL por Ian Foster y Carl Kesselman



# Evolución en las plataformas de computación de altas prestaciones

- Término acuñado por Luis F. G. Sarmenta (Bayanihan)
- En 1999 se lanza los proyectos SETI@home y Folding@home
- A día 26/09/2021 todos los proyectos BOINC suponen ~[22275 TeraFLOPS](#)

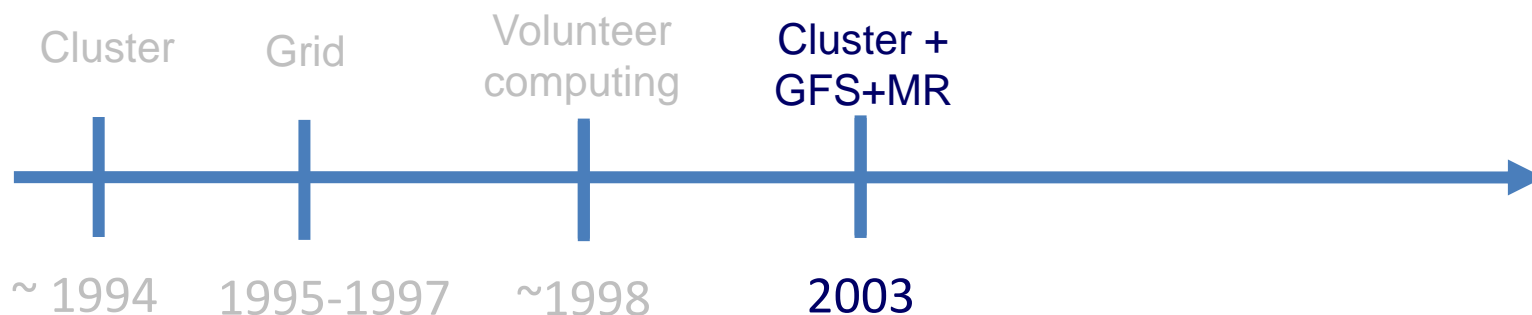


# Evolución en las plataformas de computación de altas prestaciones

- Google presenta:
  - MapReduce como *framework* para trabajar con grandes conjuntos de datos: la misma función se aplica a diferentes particiones de datos (map) y después estos resultados se combinan (reduce)
  - GFS como forma de almacenar petabytes de datos (ordenadores normales, distribución escalable y tolerancia a fallos)
- GFS+MR permite a los usuarios construir “mainframes baratos” (GFS+MR vs mainframe similar a cluster vs supercomputador)

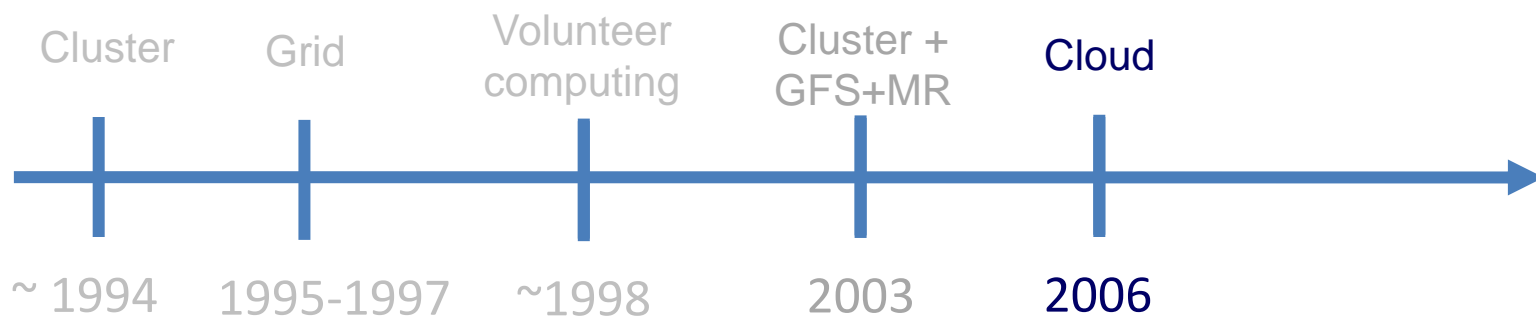
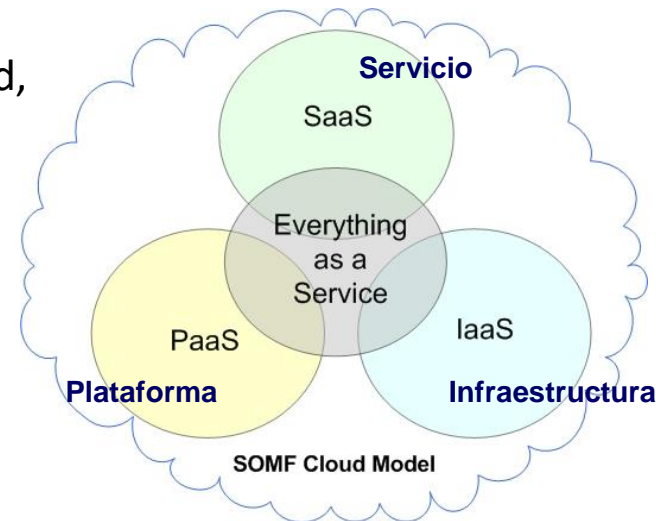


Doug Cutting  
y Hadoop



# Evolución en las plataformas de computación de altas prestaciones

- Amazon inspira el *Cloud computing* actual:
  - *data centers* pensando en las compras de Navidad, el resto del tiempo se usaban ~10%
  - Dos pilares fundamentales: *utility computing* y virtualización
- Principales mejoras: agilidad, coste, escalabilidad, mantenimiento, ...
- Openstack: construir un *cloud* con un cluster



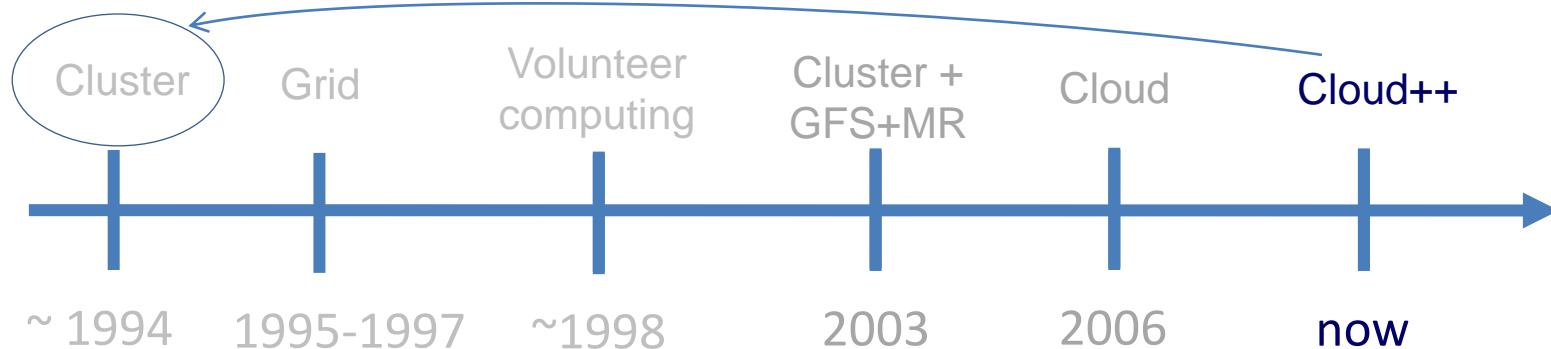
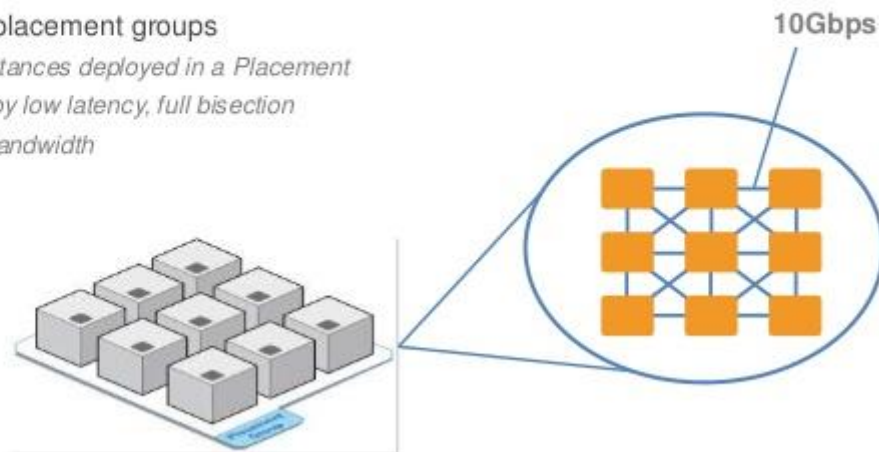


# Amazon Cluster Compute Instance

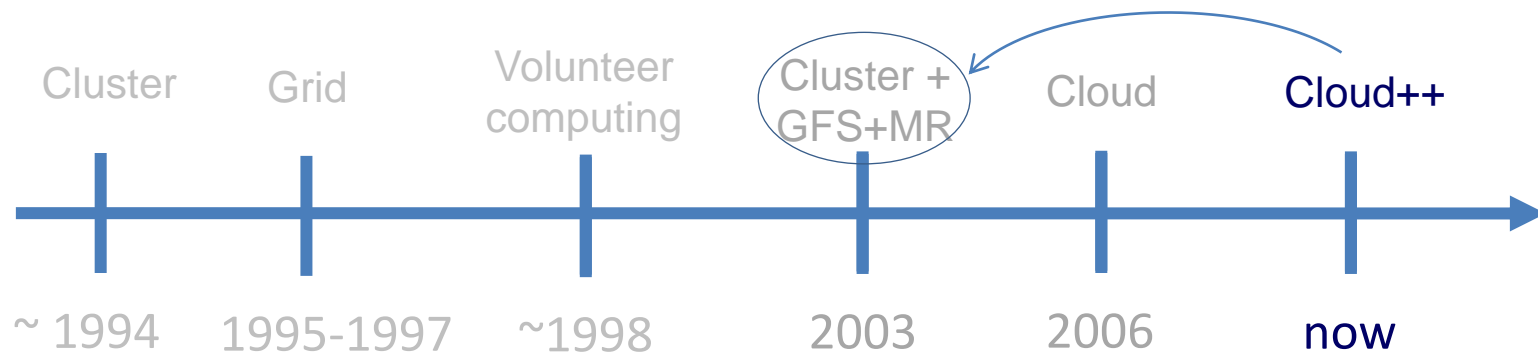
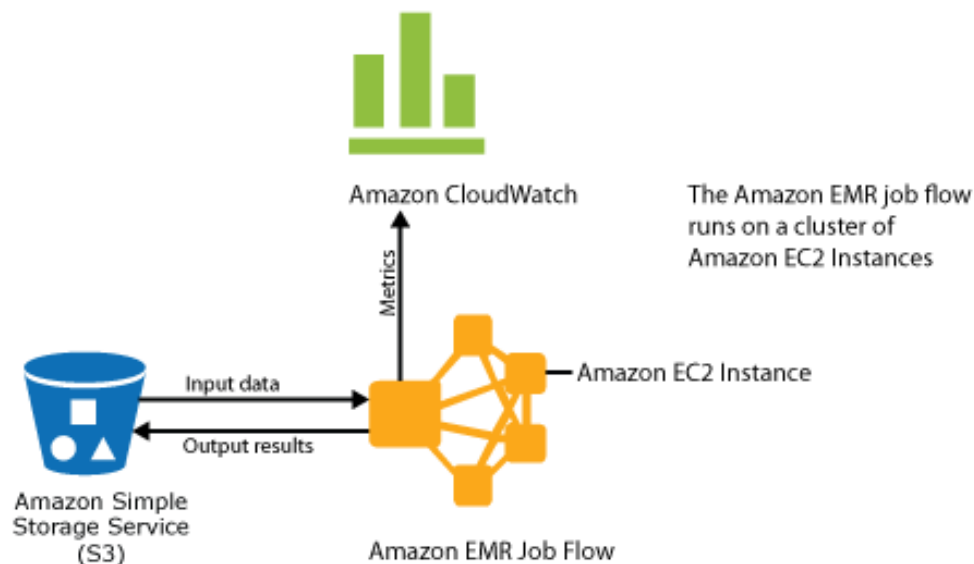
## Tightly coupled

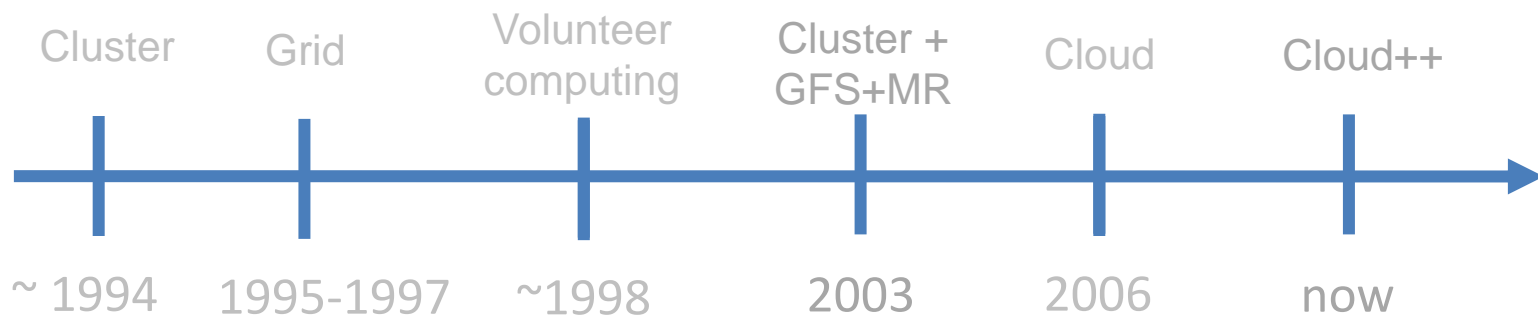
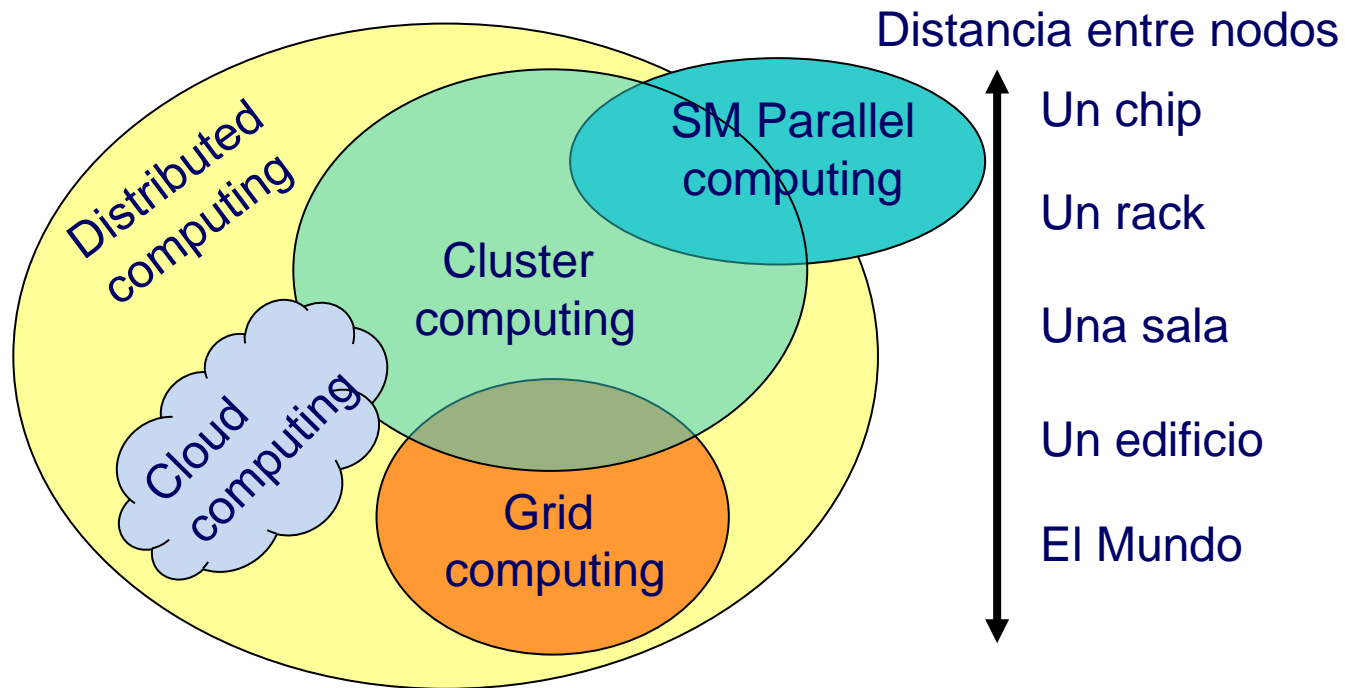
### Network placement groups

*Cluster instances deployed in a Placement Group enjoy low latency, full bisection 10 Gbps bandwidth*



# Amazon Elastic MapReduce





# Nvidia DGX-320G

(<https://www.nvidia.com/es-es/data-center/dgx-2/>)

## ANNOUNCING NVIDIA DGX STATION 320G

Workgroup AI Supercomputer-in-a-Box

Plug-into-the-Wall Instant AI Infrastructure

2.5 petaFLOPS

320 GB at 8 TB/sec

7.68 TB NVMe

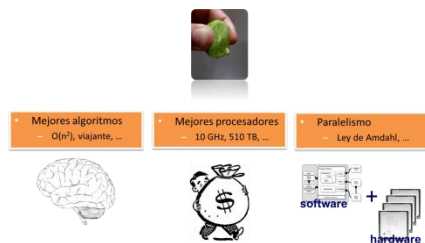
28 MIGs

1500W and < 37db

\$149,000 or \$9,000/Month Subscription

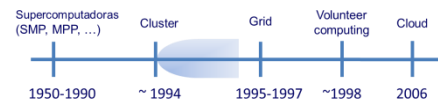


# Agenda



## Introducción a la computación de altas prestaciones

- Qué, dónde y cómo
- Hardware y software



## Evolución de la computación de altas prestaciones

- Plataformas
- Tendencias



# Principales tendencias



Supercomputadoras  
(SMP, MPP, ...)

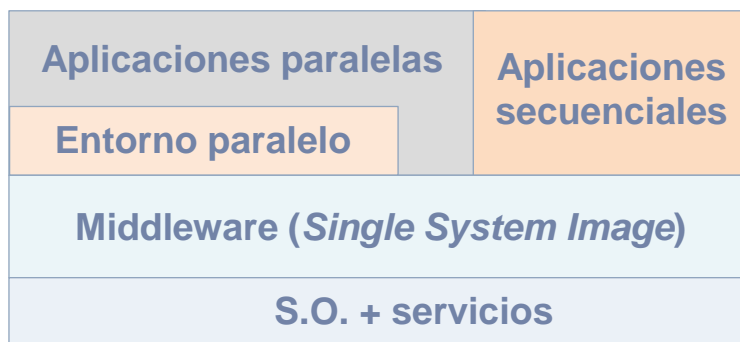
Cluster

Grid

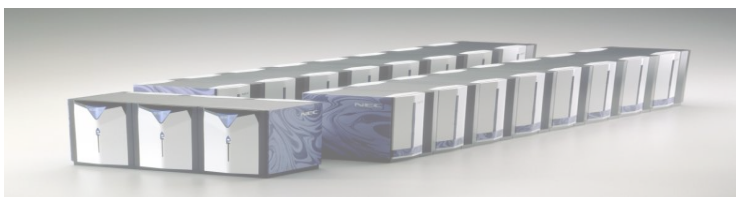
Volunteer  
computing

Cloud

Plataforma



Software



Computador de altas prestaciones

Hardware



# Principales tendencias



Supercomputadoras  
(SMP, MPP, ...)

Cluster

Grid

Volunteer  
computing

Cloud

Plataforma



Aplicaciones paralelas

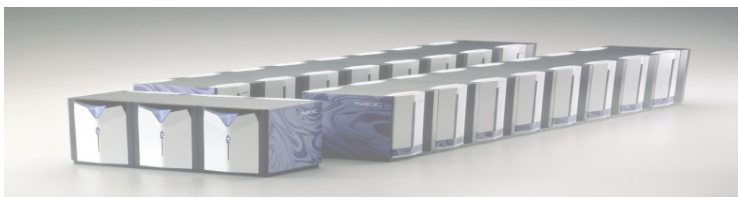
Entorno paralelo

Aplicaciones  
secuenciales

Middleware (*Single System Image*)

S.O. + servicios

Software



Computador de altas prestaciones

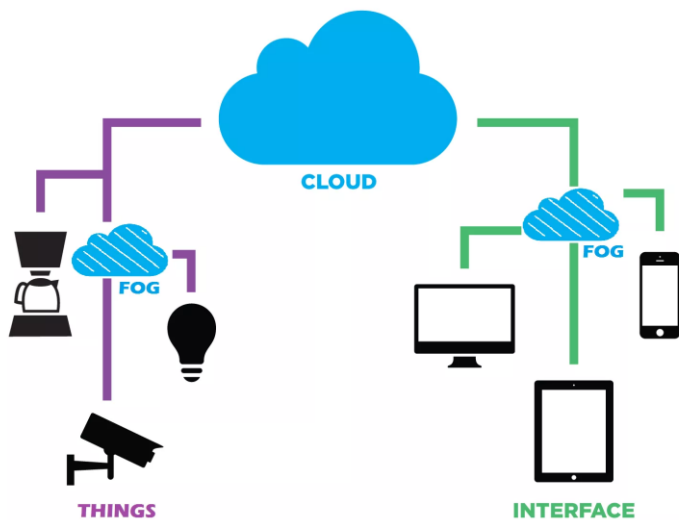
Hardware





# Plataforma:

## uso de recursos distribuidos



- **Clouds:** empleo de recursos distribuidos alquilados bajo demanda
- **Fog/Edge:** acercar el cloud a los dispositivos que lo usan

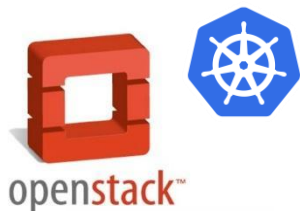
<https://iot.do/ngd-openfog-fog-computing-2016-10>





# Plataforma:

uso eficiente de recursos



- **Clouds privados y públicos:** ajuste de infraestructura para minimizar gasto
- **Green computing:** uso de recursos distribuidos de distintas organizaciones
- **Internet computing:** uso de ordenadores personales a escala global (SETI@home)

# Principales tendencias



Supercomputadoras  
(SMP, MPP, ...)

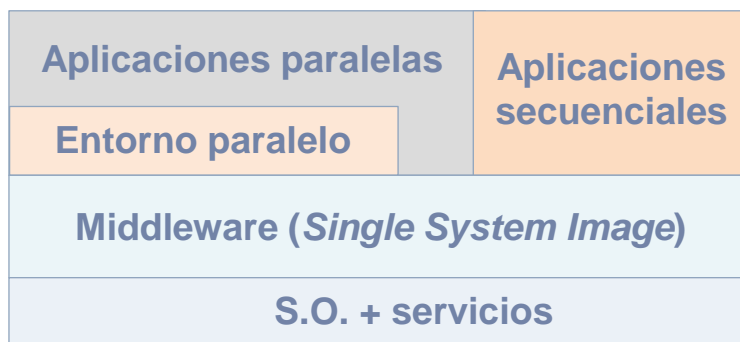
Cluster

Grid

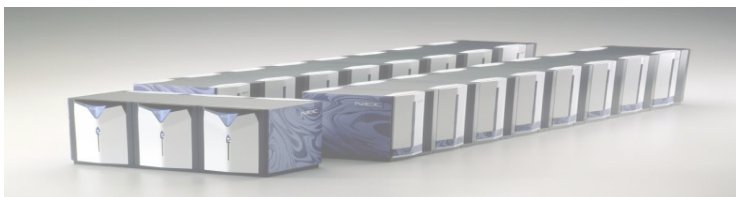
Volunteer  
computing

Cloud

Plataforma



Software

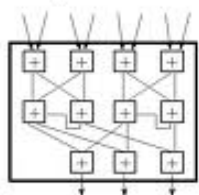


Computador de altas prestaciones

Hardware



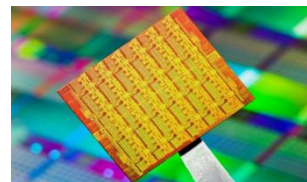
# Hardware



**A nivel de bit**



**A nivel de  
instrucción**

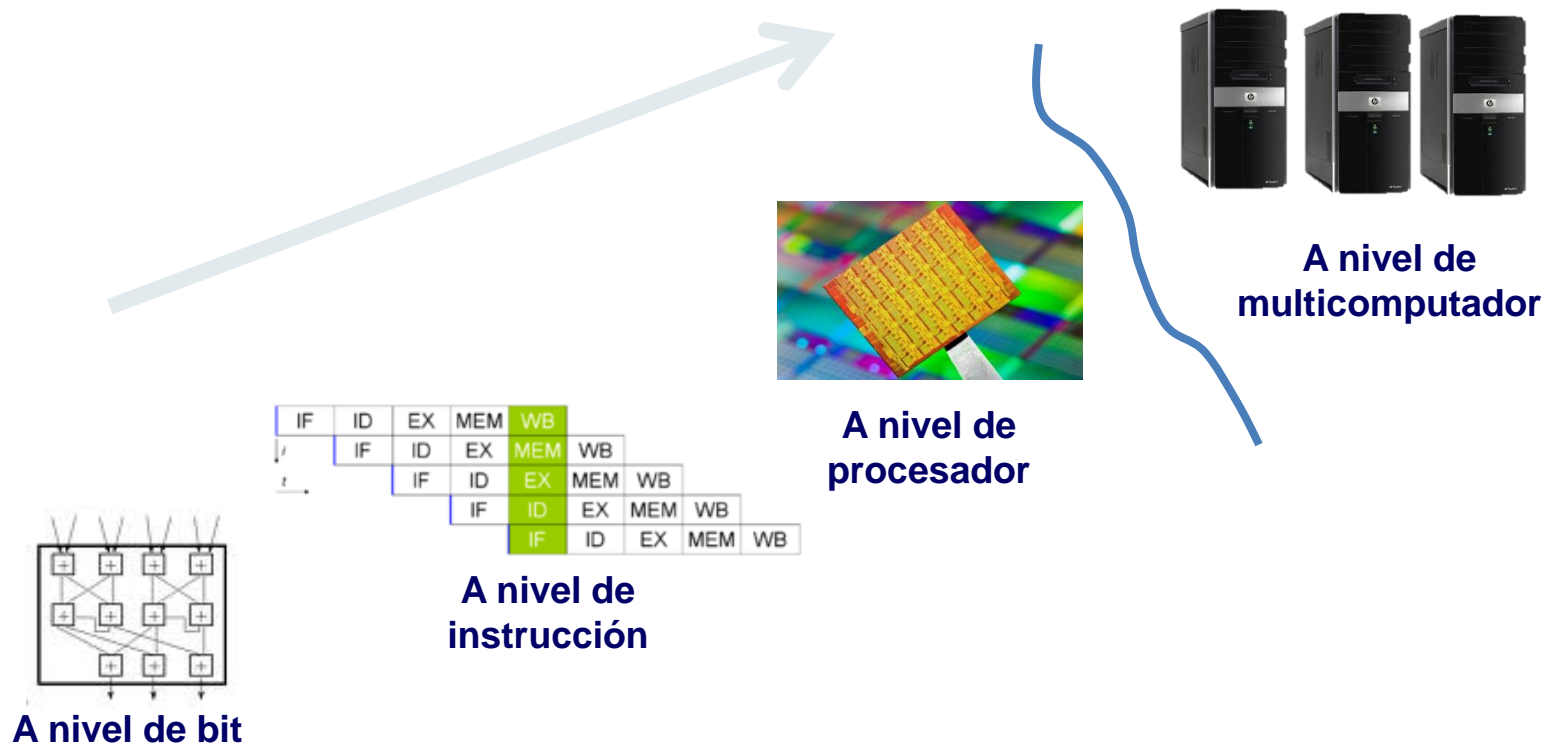


**A nivel de  
procesador**



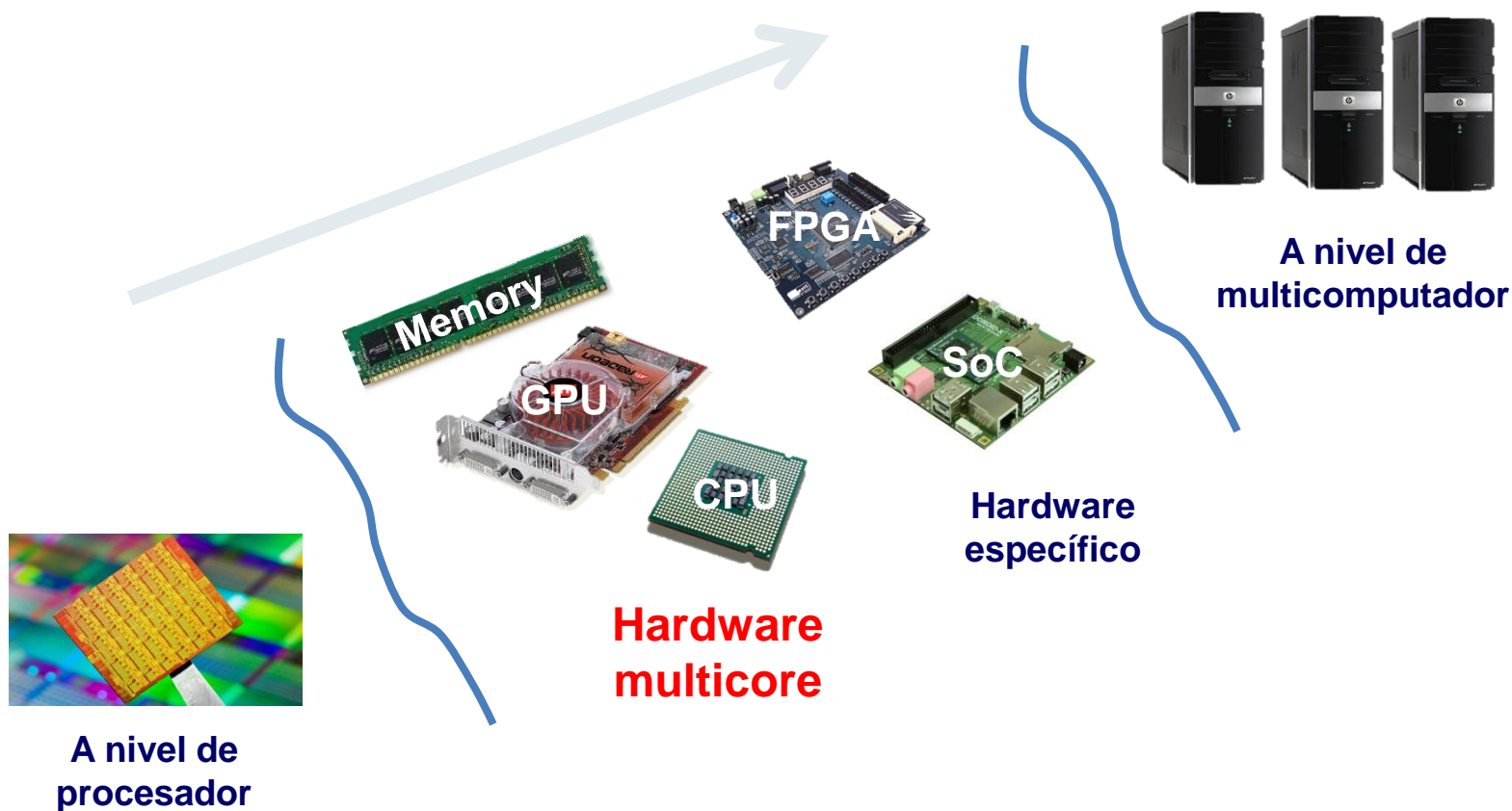
**A nivel de  
multicomputador**

# Hardware



# Hardware:

más procesadores y cores heterogéneos





# Hardware:

más procesadores y cores heterogéneos



- **Tarjetas gráficas:** uso de la capacidad de procesamiento de las potentes tarjetas gráficas actuales



# Hardware:

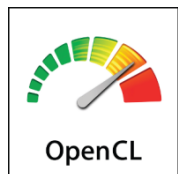
más procesadores y cores heterogéneos

- **Tarjetas gráficas:** uso de la capacidad de procesamiento de las potentes tarjetas gráficas actuales



- **CUDA:**

Entorno de programación para poder usar la potencia de las tarjetas gráficas de NVidia



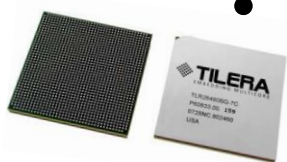
- **OpenCL:**

lenguaje basado en C99 extendido para operaciones vectoriales y eliminando ciertas funcionalidades



# Hardware:

más procesadores y cores heterogéneos



- **Procesadores many-core:** gran cantidad de procesadores en un mismo chip

- <http://gizmodo.com/5846060/this-crazy-64+core-processor-wants-to-be-in-your-smartphone>
- <http://www.tgdaily.com/hardware-features/33451-tilera-announces-64-core-processor>

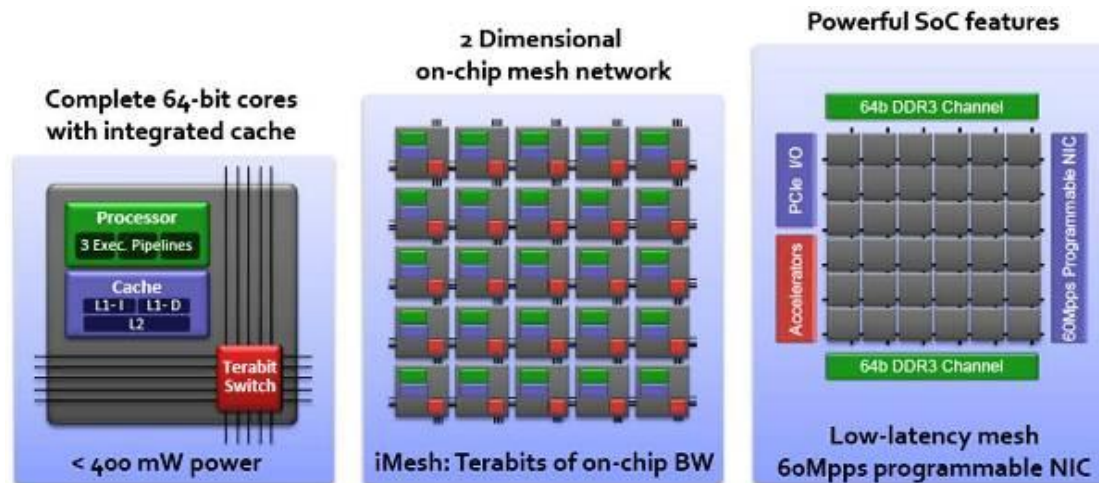




# Hardware:

## más procesadores y cores heterogéneos

- **Procesadores many-core:** gran cantidad de procesadores en un mismo chip





# Hardware:

más procesadores y cores heterogéneos

- **Procesadores many-core:** gran cantidad de procesadores en un mismo chip
  - <memoria compartida>:  
*SMP Linux 2.6*
  - <paso de mensaje>:  
*Hypervisor (VMs)*



# Hardware:

más procesadores y cores heterogéneos

- **Procesadores heterogéneos:**  
gran cantidad de procesadores con coprocesadores especializados

1997: THE FIRST INTEL® TERAFL0P COMPUTER consisted of: **9,298 INTEL PROCESSORS** and occupied: **72 SERVER CABINETS**

THE INTEL® XEON® PHI™ COPROCESSOR will provide: **1 TERAFL0P OF PERFORMANCE** and occupy: **1 PCIe SLOT**

[Click to learn more](#)

- [http://es.wikipedia.org/wiki/Intel\\_MIC](http://es.wikipedia.org/wiki/Intel_MIC)
- <http://hothardware.com/News/Intel-Demos-Knights-Ferry-Development-Platform-Tesla-Scores-With-Amazon/>



# Hardware:

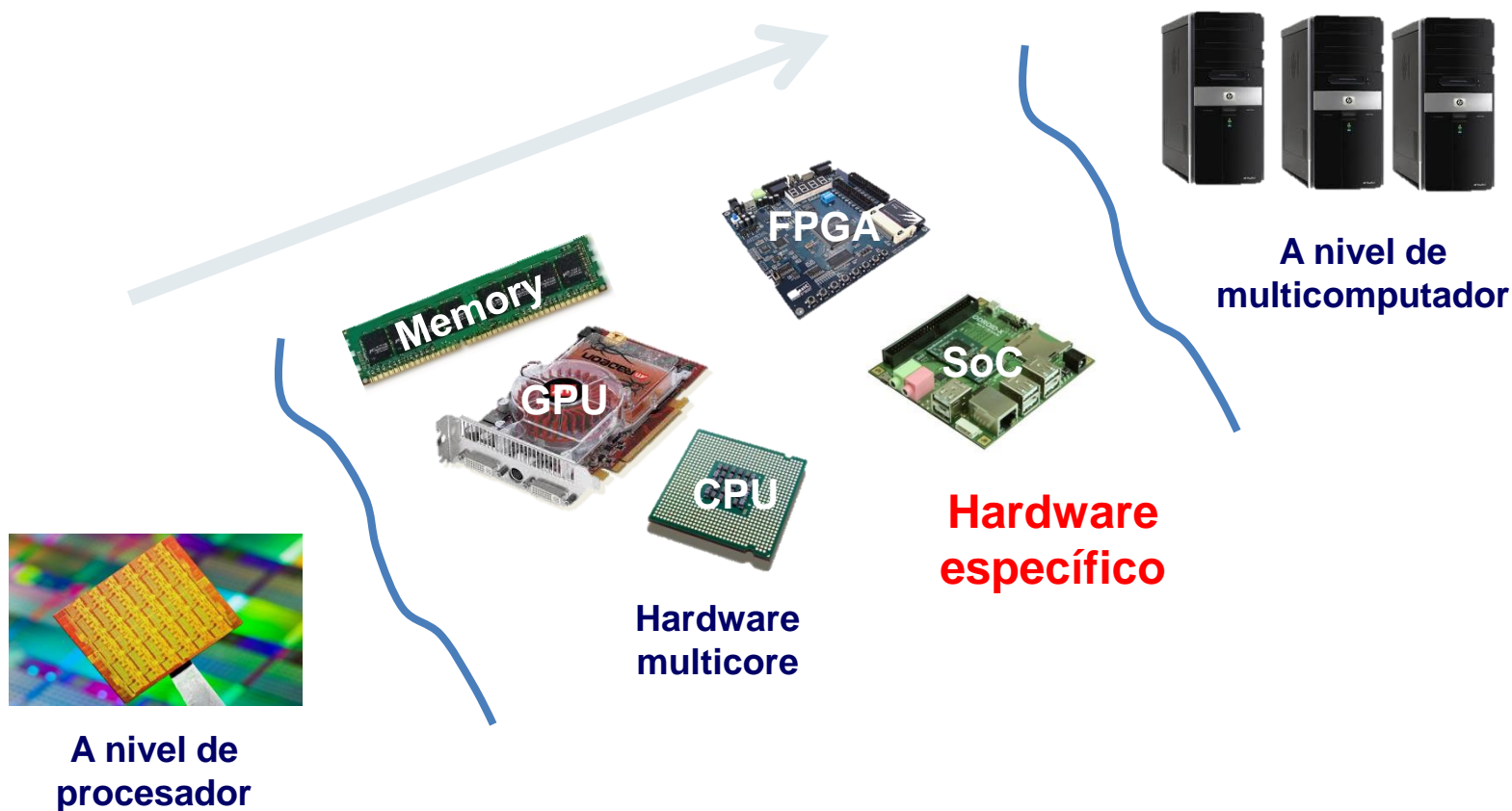
## más procesadores y cores heterogéneos

- **Procesadores heterogéneos:**  
gran cantidad de procesadores con coprocesadores especializados
  - <memoria compartida>:  
Intel Cilk (plus), Intel Threading Building Blocks, OpenMP, ¿OpenACC?, OpenCL
  - <paso de mensaje>:  
Intel MPI

- <http://goparallel.sourceforge.net/parallel-programming-intel-mic-early-experiences-tacc/>
- <http://www.drdobbs.com/parallel/intels-50-core-mic-architecture-hpc-on-a/232800139>

# Hardware:

más procesadores y cores heterogéneos

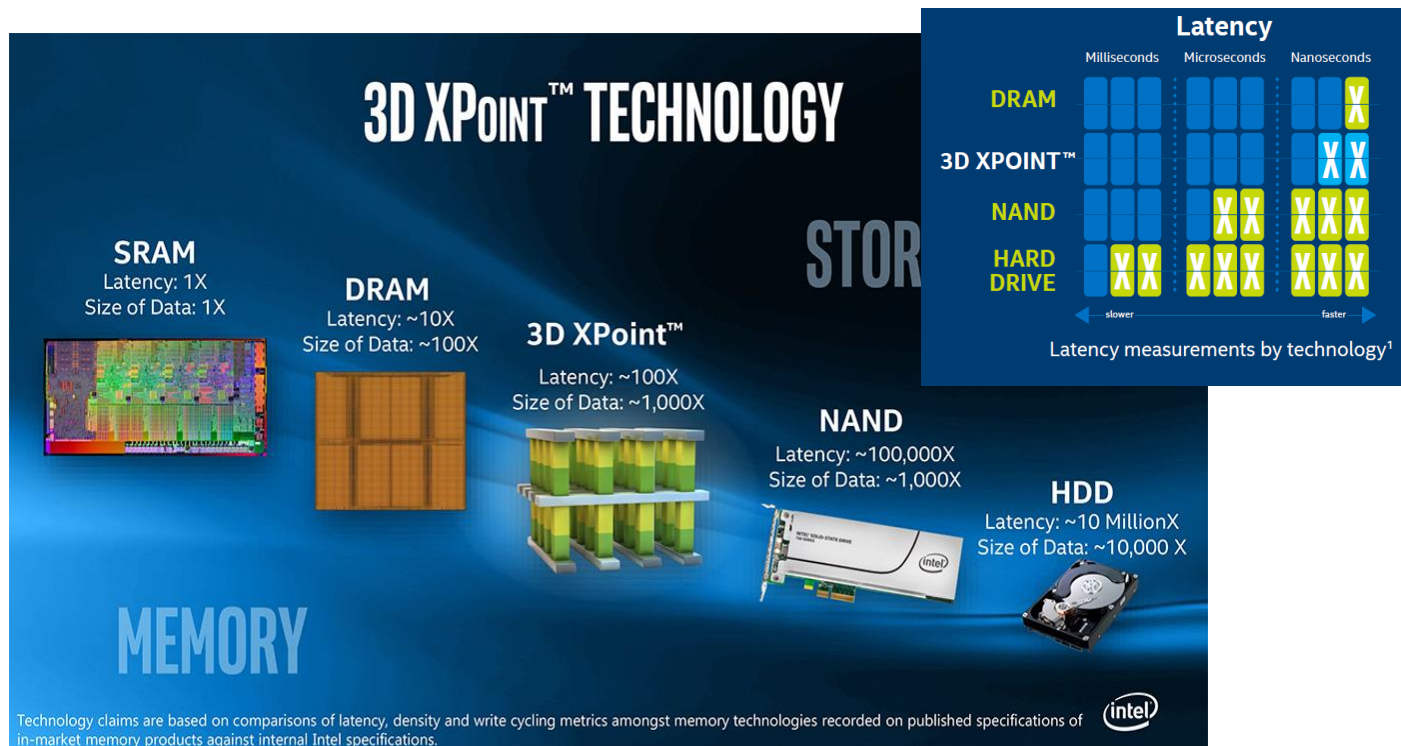




# Hardware:

memoria persistente, de gran capacidad y baja latencia

- **Memoria 3D-XPoint:**





# Hardware:

“memoria” con capacidad de cómputo

- **Memoria “activa”:**  
computo simple en la propia memoria

Planted Motif Search Problem	Automata Processor	UCONN - BECAT Hornet Cluster
Processors	48 (PCIe Board)+CPU	48 CPU (Cluster/OpenMPI)
Power	245W-315W <sup>1</sup>	>2,000W <sup>1</sup>
Cost	TBD	~\$20,000 <sup>1</sup>
Performance (25,10)	12.26 minutes <sup>2</sup>	20.5 minutes
Performance (26,11)	13.96 minutes <sup>2</sup>	46.9 hours
Performance (36,16)	36.22 minutes <sup>2</sup>	Unsolved

- Planted Motif Search problem is a leading problem in bioinformatics and is NP Hard. Attempts to find common genomic sequences in noisy data.
- Solutions involving high match lengths and substitution counts are often presented to HPC clusters for processing.
- Independent research predicts the Micron Automata Processor significantly outperforms a multi-core HPC cluster in speed, power and estimated cost.

<sup>1</sup> Micron Technology Estimates, Not including Memory of 4GB DRAM /Core

<sup>2</sup> Research conducted by Georgia Tech (Roy/Aluru)





# Hardware:

## aceleradores específicos por USB



- Conector USB Type A.
- VPU (Vision Processing Unit) Myriad 2.
- 4 GB de memoria LPDDR3.
- Soporte del framework “Caffe”.
- Compatible con FP16 (precisión media).
- Consumo de 1 vatio.
- Precio: **79 dólares** (2017)

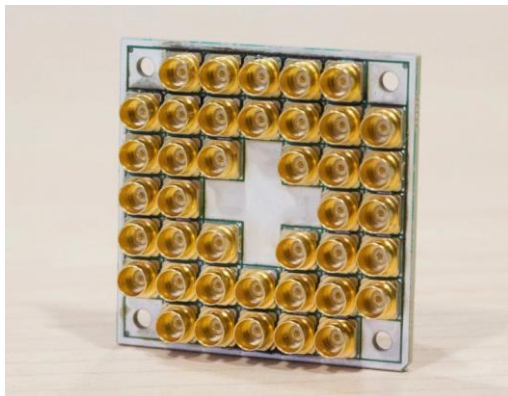
- <https://www.muycomputer.com/2017/07/20/movidius-neural-compute-stick/>
- <https://www.movidius.com/MyriadX>





# Hardware:

## qubit-chip



- “...While quantum computers promise greater efficiency and performance to handle certain problems, they won’t replace the need for conventional computing or other emerging technologies like [neuromorphic computing](#). We’ll need the technical advances that Moore’s law delivers in order to invent and scale these emerging technologies...”

# Principales tendencias



Supercomputadoras  
(SMP, MPP, ...)

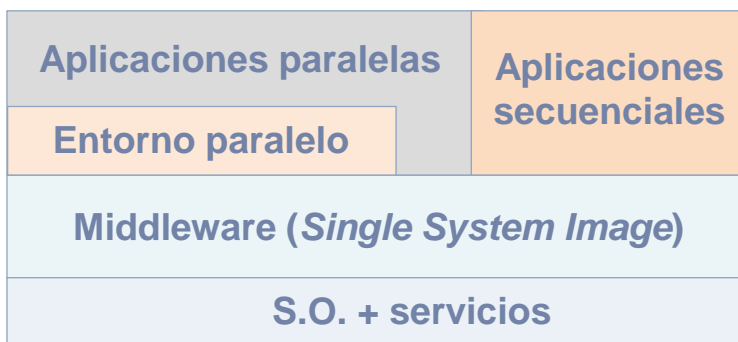
Cluster

Grid

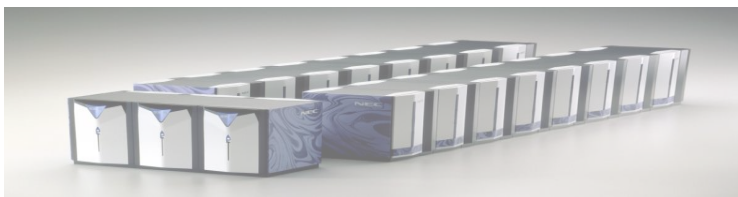
Volunteer  
computing

Cloud

Plataforma



Software



Computador de altas prestaciones

Hardware





# Software

## Vectoriales

SSE, AVX, AVX2, ...

## Multiprocesador

UMA, NUMA

OpenMP,

iTBB, ...

## M. Distribuida

MPI,...

Map-reduce

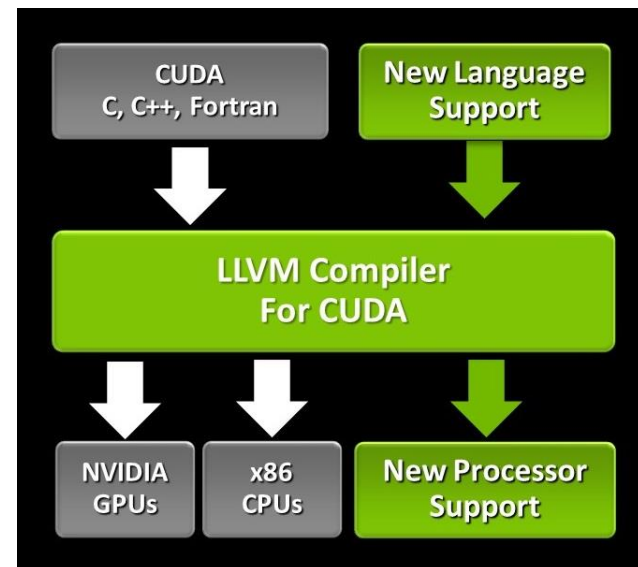
- **Integrar soluciones vectoriales y multiprocesador (dentro de las herramientas de desarrollo)**



# Ejemplo:

## CUDA/LLVM adaptado a nuevos entornos

- CUDA Compiler SDK
- Versión de Clang/LLVM con:
  - Generación de código para GPU
  - Compilación con CUDA
- Soporte para:
  - MacOS
  - Windows
  - Linux (algunos)



▪ <http://developer.nvidia.com/cuda/cuda-llvm-compiler>



# Software

## Vectoriales

SSE, AVX, AVX2, ...

## Multiprocesador

UMA, NUMA

OpenMP,

iTBB, ...

## M. Distribuida

MPI,...

Map-reduce

- Integrar soluciones vectoriales y multiprocesador (dentro de las herramientas de desarrollo)
- **Integrar soluciones de memoria compartida y paso de mensaje con ayuda del sistema operativo.**



# Ejemplo:

## MPI 3.x: adaptación a requisitos actuales

- Programación híbrida
- Tolerancia a fallos
- Acceso remoto a memoria
- Comunicación colectiva y topología
- Soporte de herramientas
- Persistencia
- Compatibilidad hacia atrás

▪ [http://meetings.mpi-forum.org/MPI\\_3.0\\_main\\_page.php](http://meetings.mpi-forum.org/MPI_3.0_main_page.php)



# Software

Vectoriales

SSE, AVX, AVX2, ...

Multiprocesador

UMA, NUMA

OpenMP,

iTBB, ...

M. Distribuida

MPI,...

Map-reduce

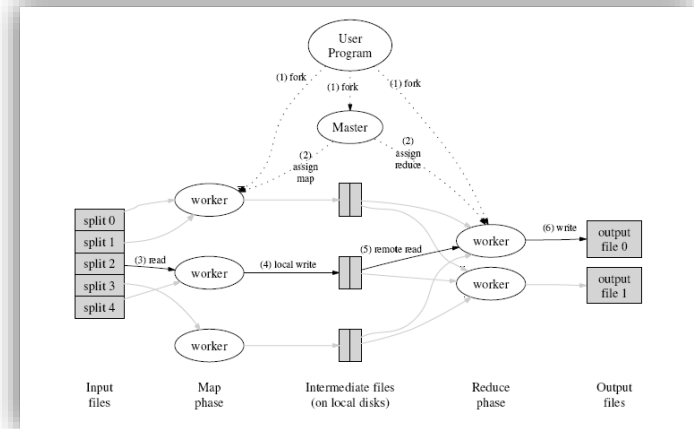
- Integrar soluciones vectoriales y multiprocesador (dentro de las herramientas de desarrollo)
- Integrar soluciones de memoria compartida y paso de mensaje con ayuda del sistema operativo.
- **Buscar perfiles simplificados que permitan la mayor escalabilidad posible.**



# Sistemas distribuidos:

Computación de altas prestaciones

- Google:
  - Modelo MapReduce



- Sistemas de ficheros de Google
- Algoritmos de clasificación (K-Means + Canopy)

- <http://code.google.com/edu/parallel/mapreduce-tutorial.html>
- <http://code.google.com/edu/submissions/mapreduce-minilecture/listing.html>
- <http://en.wikipedia.org/wiki/MapReduce>





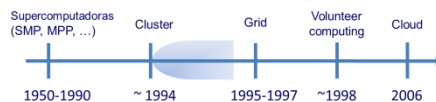
# Aplicaciones:

## Adaptación a computación de altas prestaciones

- Ejemplos:
  - Primal and dual-based algorithms for sensing range adjustment in WSNs
  - The unified accelerator architecture for RNA secondary structure prediction on FPGA
  - Protein simulation data in the relational model
  - Dynamic learning model update of hybrid-classifiers for intrusion detection

▪ <http://www.springer.com/computer/swe/journal/11227>

# Agenda



## Introducción a la computación de altas prestaciones

- Qué, dónde y cómo
- Hardware y software

## Evolución de la computación de altas prestaciones

- Plataformas
- Tendencias



# Bibliografía

- **Parallel Computer Architectures: a Hardware/Software Approach.**  
D.E. Culler, J.P. Singh, with A. Gupta
  - Capítulo 1
- **Organización y Arquitectura de Computadores (5ta. ed.)**  
William Stallings
  - Capítulo 16: Procesamiento Paralelo.
- **Organización de Computadoras (4ta. ed.)**  
Andrew S. Tanenbaum
  - Capítulo 8: Arquitecturas de computadoras paralelas.

# Bibliografía

- **GPU + CPU**
  - <http://www.hardwarezone.com.ph/articles/view.php?cid=3&id=2786>
- **Cluster**
  - <http://www.democritos.it/~baro/slides/LAT-HPC-GRID-2009/Part1.pdf>
- **TOP500 Supercomputer Sites**
  - <http://www.top500.org/>
- **Beowulf**
  - <http://www.beowulf.org/overview/index.html>

Sistemas Paralelos y Distribuidos

Máster en Ciencia y Tecnología Informática

Diseño de Sistemas Distribuidos

Máster en Ingeniería Informática

Curso 2021-2022

# Sistemas de altas prestaciones en entornos distribuidos

Alejandro Calderón Mateos y Félix García Carballeira

Grupo de Arquitectura de Computadores

[alejandro.calderon@uc3m.es](mailto:alejandro.calderon@uc3m.es)