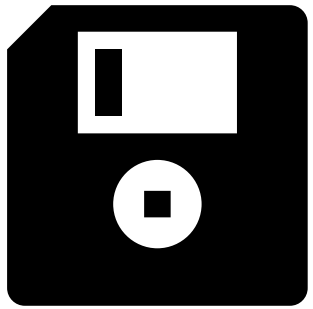
SISTEMAS DE INFORMACIÓN

DEFINICIÓN, ELEMENTOS Y COMPONENTES

Realizado por: Edwin Willer Narváez Burbano

DATO



- Puede ser un número, una palabra, una imagen
- Un dato es un documento, una información o un testimonio que permite llegar al conocimiento de algo

INFORMACIÓN

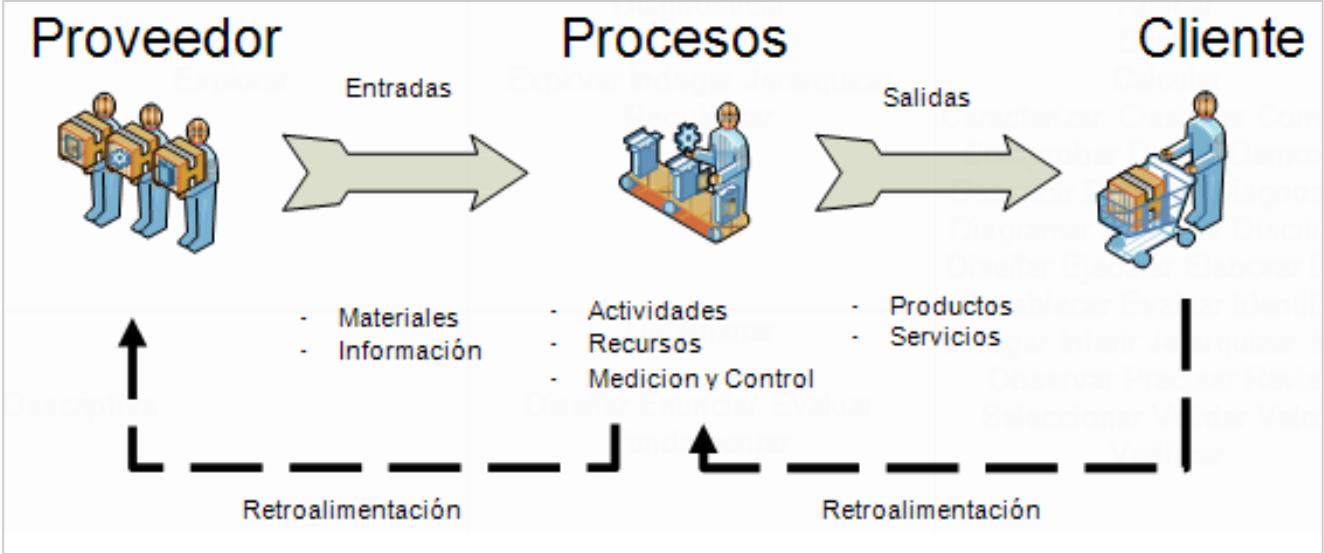


- Son datos que dentro de un contexto dado tienen un significado para alguien.



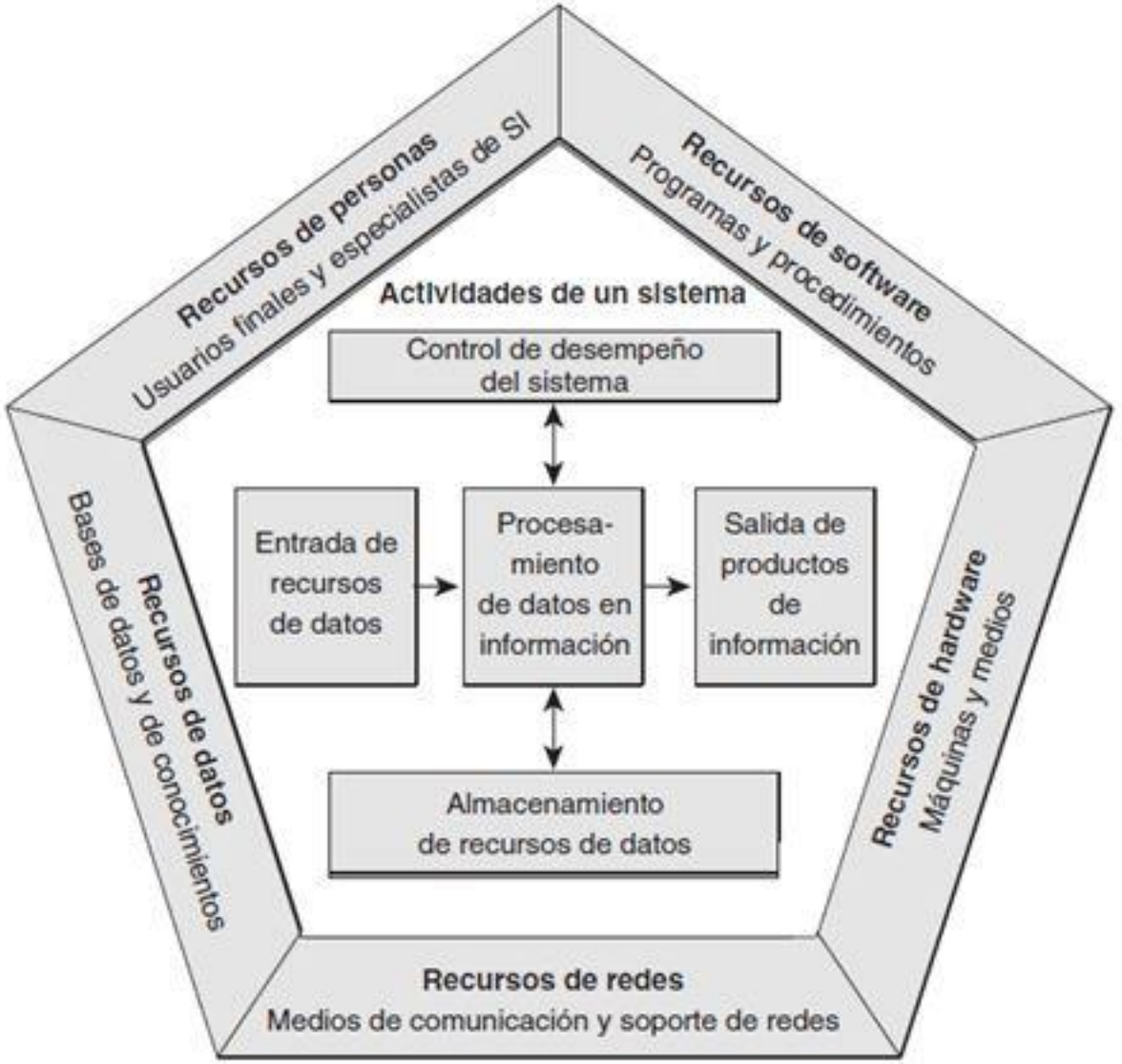
SISTEMA DE INFORMACIÓN

SISTEMA



- Es un grupo de componentes interrelacionados, con un límite definido con claridad, que trabajan juntos hacia un objetivo común, al recibir entradas y producir salidas en un proceso organizado de transformación.

SISTEMA DE INFORMACIÓN



PERSONAS

- Usuarios finales: personas que utilizan un sistema de información o la información que éste produce. Pueden ser clientes, vendedores, ingenieros, oficinistas, contadores o gerentes.

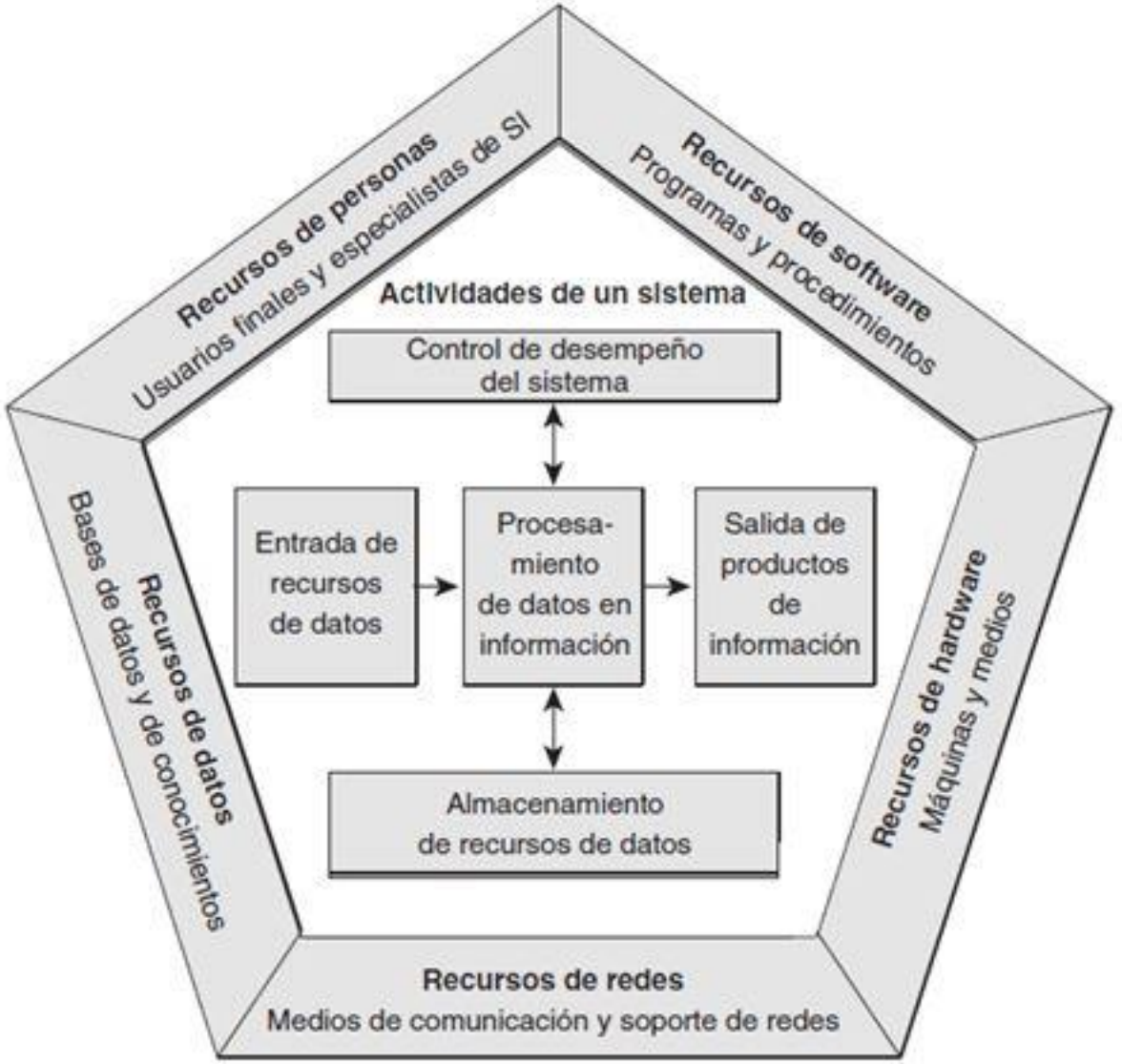


PERSONAS

- Especialistas de SI: personas que desarrollan y operan los sistemas de información. Incluyen analistas de sistemas, desarrolladores de software, operadores de sistemas y demás personal gerencial, técnico y de oficina.



SISTEMA DE INFORMACIÓN



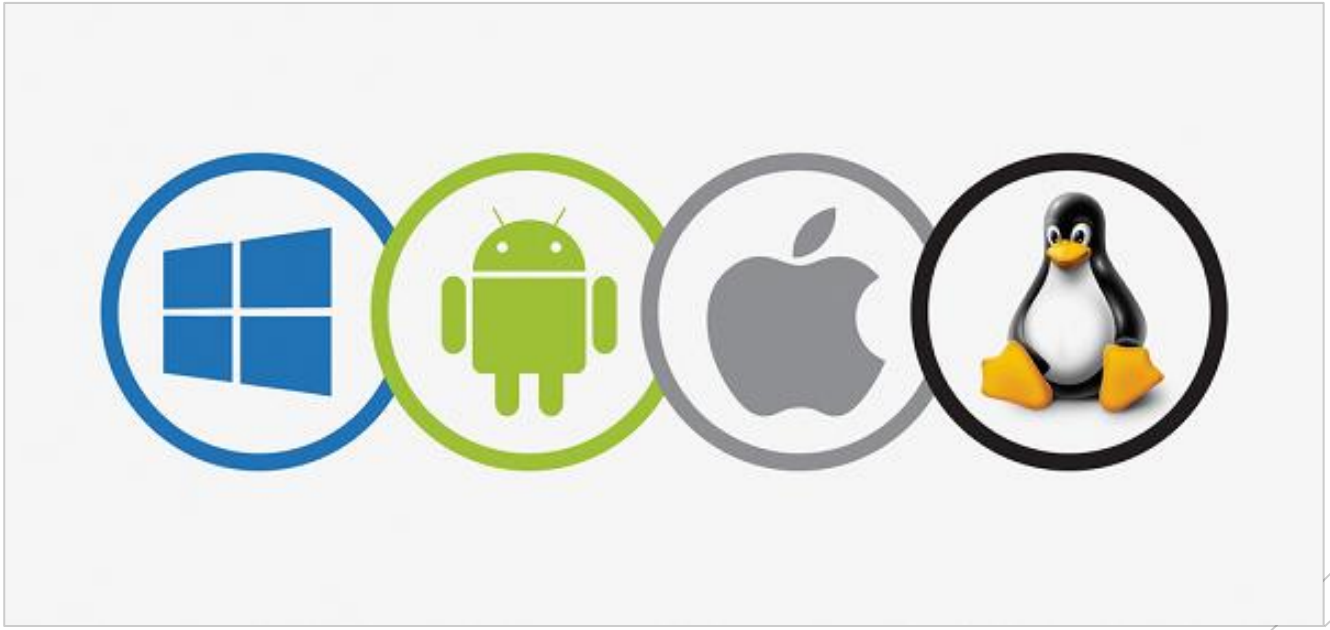
SOFTWARE

- **Incluye no sólo los conjuntos de instrucciones operativas llamados programas, los cuales dirigen y controlan el hardware informático, sino también los grupos de instrucciones para el procesamiento de información, llamados procedimientos, que las personas necesitan.**



SOFTWARE

- **Software de sistema:** sistema operativo, el cual controla y apoya las operaciones de un sistema informático.



SOFTWARE

- **Software de aplicación:** son programas que dirigen el procesamiento hacia un uso particular de las computadoras por parte de los usuarios finales.

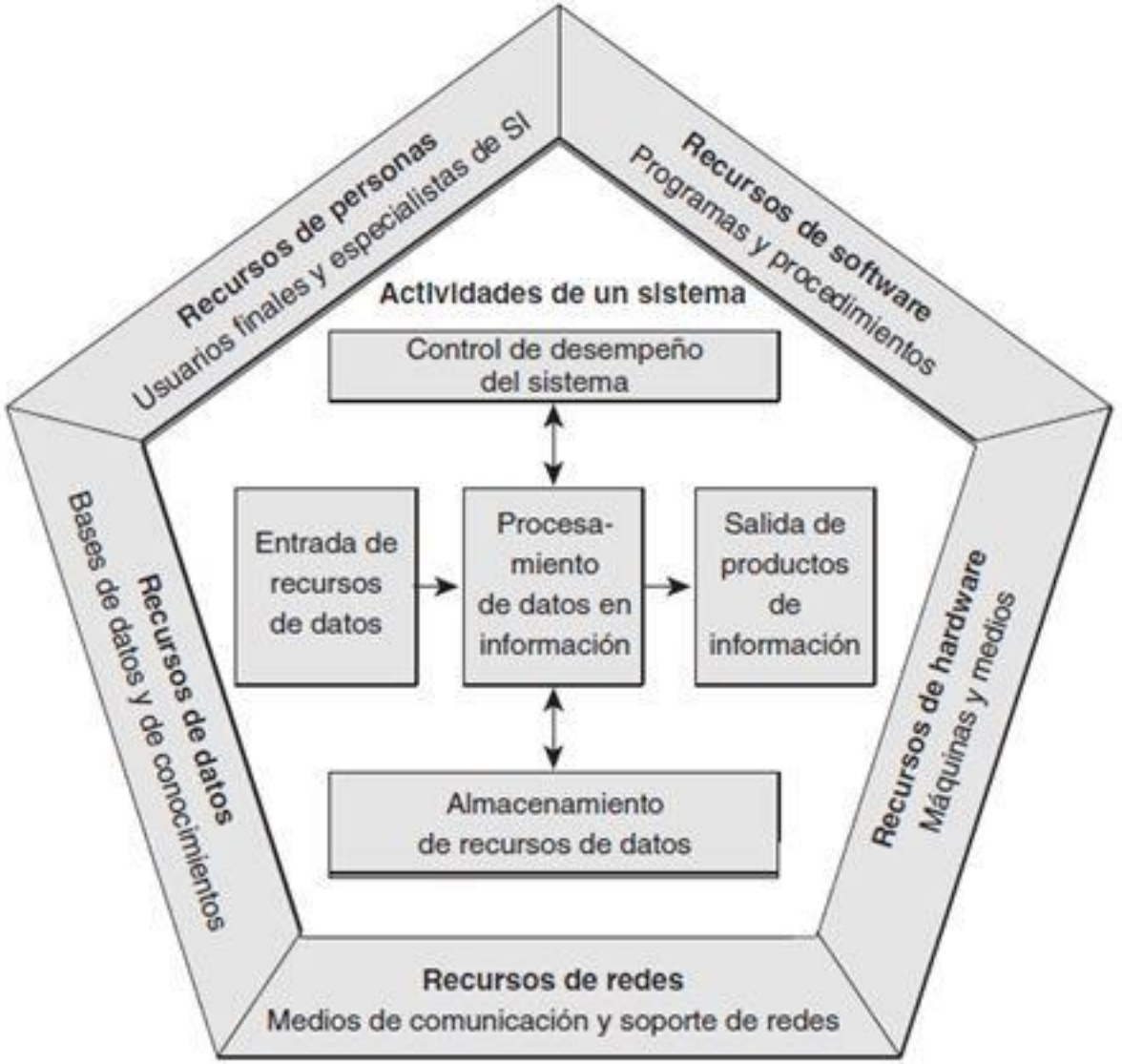


SOFTWARE

- **Procedimientos**: son instrucciones de operación para las personas que utilizarán un sistema de información.



SISTEMA DE INFORMACIÓN

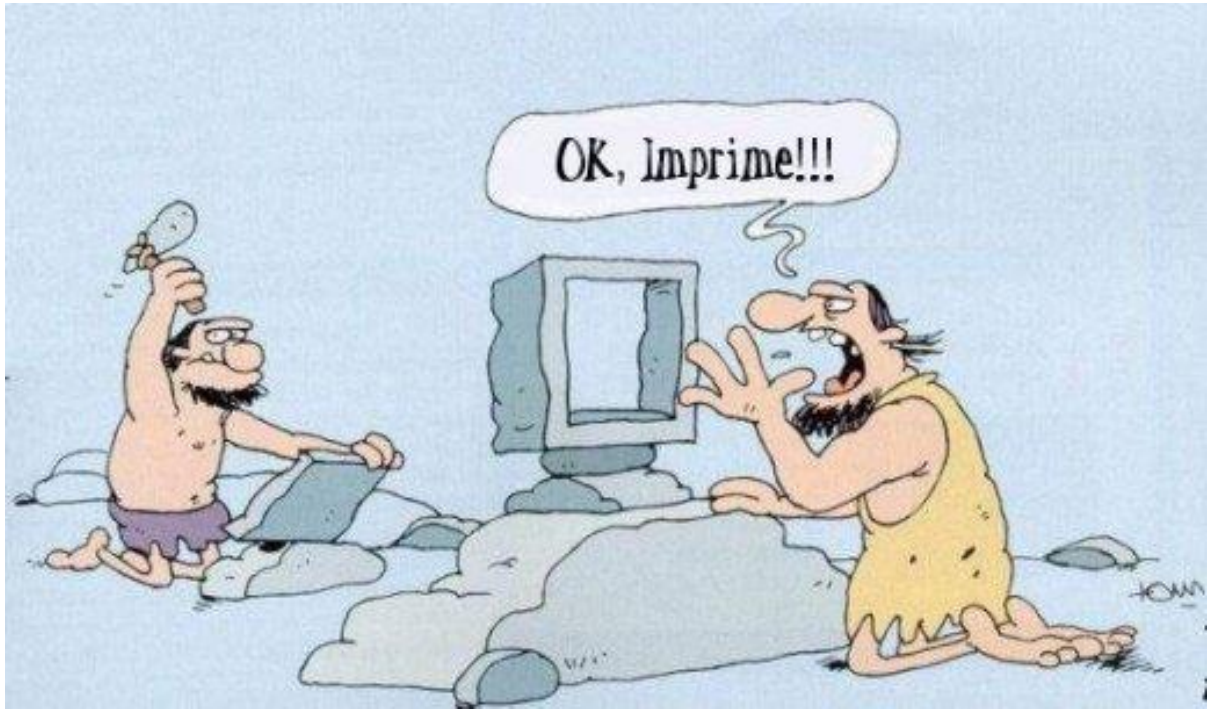


HARDWARE

- Incluye todos los dispositivos y materiales físicos utilizados en el procesamiento de la información, esto incluye a todos los objetos tangibles en los que se registran los datos.



HARDWARE



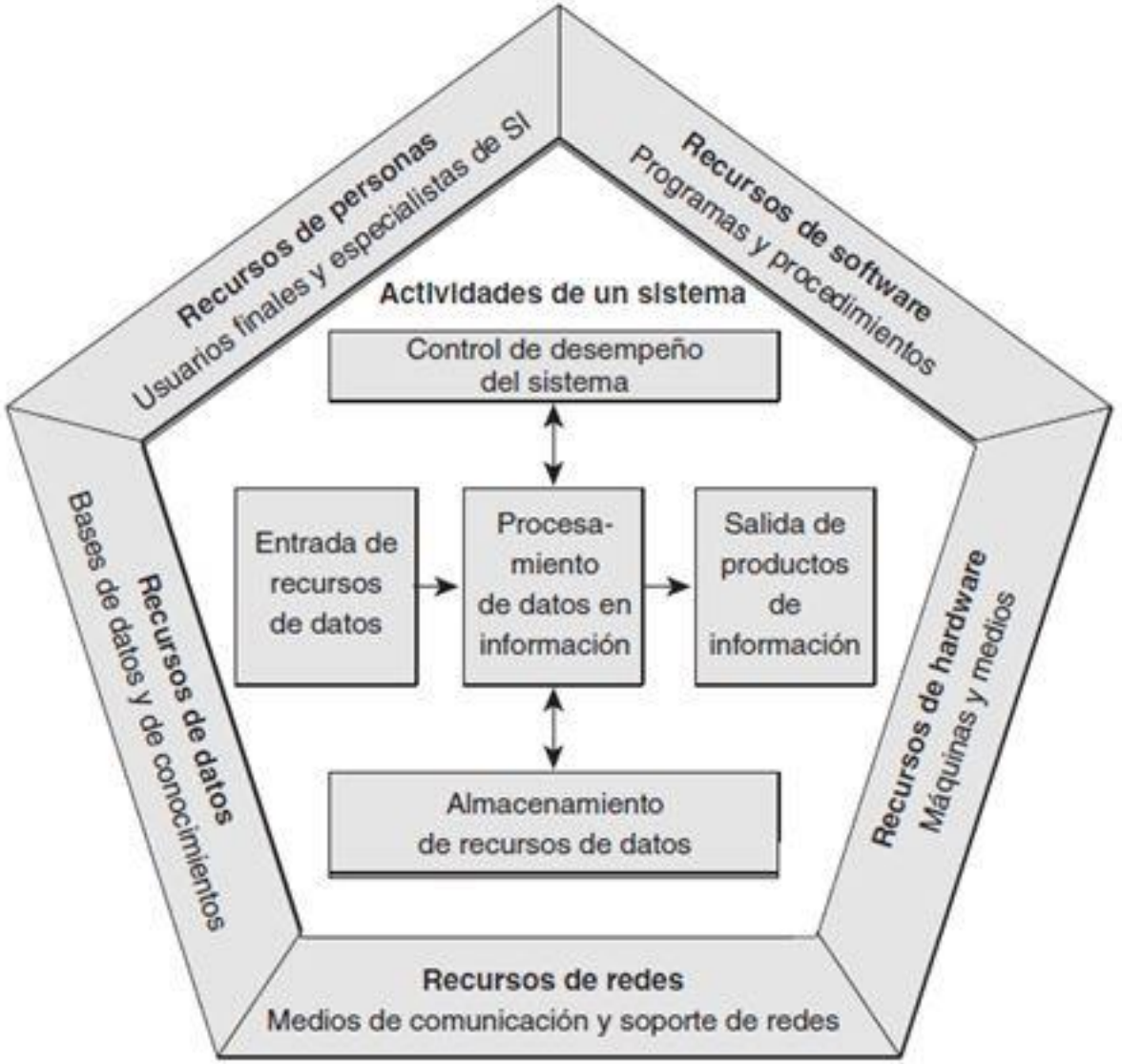
- **Sistemas de cómputo:** consisten en unidades centrales de procesamiento que contienen microprocesadores y una variedad de dispositivos periféricos interconectados. Pueden ser computadoras de mano, laptops o computadoras de escritorio y los grandes sistemas de cómputo central (mainframe).

HARDWARE



- **Periféricos de cómputo:** son dispositivos que sirven para la entrada de datos y comandos, así como para la salida de la información, y para el almacenamiento de los recursos de datos.

SISTEMA DE INFORMACIÓN



REDES

- Las tecnologías y redes de telecomunicaciones, son esenciales para las operaciones exitosas de negocio y comercio electrónicos de todo tipo de organizaciones y de sus sistemas de información basados en computadoras. Los recursos de redes enfatiza que las tecnologías y redes de comunicaciones son un componente fundamental de recursos de todos los sistemas de información.

REDES

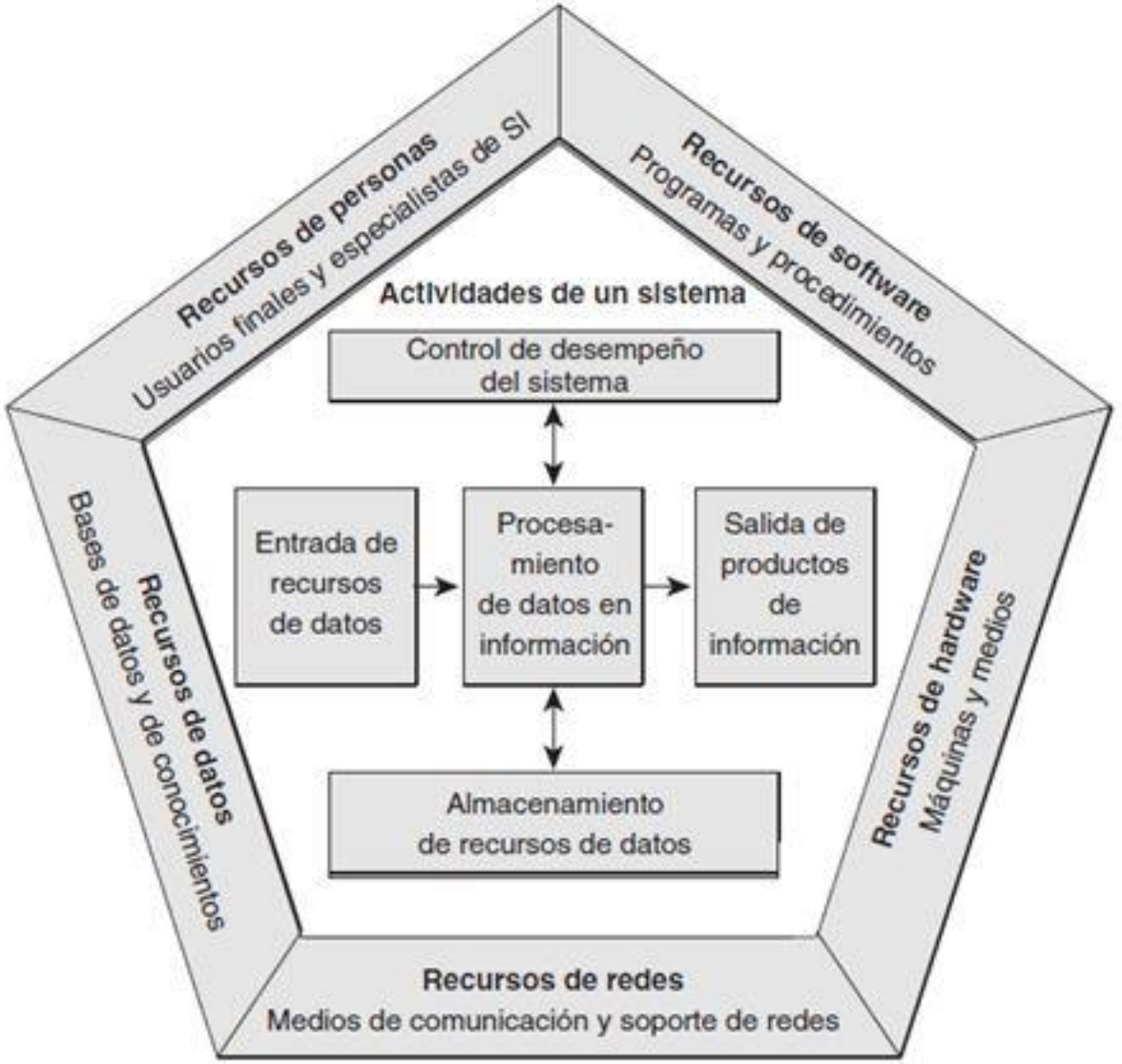
MEDIOS DE COMUNICACIÓN:

- Comprenden cables de par trenzado, cables coaxiales y de fibra óptica; y las tecnologías inalámbricas de microondas, celular y satelital.

INFRAESTRUCTURA DE REDES:

- Se necesitan muchas tecnologías de hardware, software y de datos para apoyar la operación y el uso de una red de comunicaciones. Pueden ser módems y procesadores de grupos de redes, y software de control de comunicaciones.

SISTEMA DE INFORMACIÓN



DATOS

- Constituyen recursos **valiosos** en la organización. Por eso deben ser administrados de manera **eficaz** para **beneficiar a todos los usuarios finales de una organización**.
- Los datos capturados se **almacenan, procesan y analizan**, mediante el uso de **aplicaciones muy sofisticadas de software que pueden revelar relaciones complejas acerca de ventas, clientes, competidores y mercados.**

DATOS

BASES DE DATOS

- Mantienen los datos procesados y organizados.



DATOS

BASES DE CONOCIMIENTO:

- Mantienen el conocimiento en diversas formas, tales como hechos, reglas y ejemplos de casos acerca de las prácticas de negocio exitosas.



CLASIFICACIÓN DE LOS SISTEMAS DE INFORMACIÓN

6 TIPOS PRINCIPALES

- Sistemas de procesamiento de transacciones
- Sistemas de control de procesos de negocio
- Sistemas de colaboración empresarial
- Sistemas de Información de Gestión
- Sistemas de apoyo a la toma de decisiones
- Sistemas de Información Ejecutiva

TEMA1. El ordenador y sus componentes



1. Introducción
2. Evolución de los ordenadores
3. Generaciones de ordenadores
4. Representación de la información en el ordenador
5. Conceptos básicos

1. INTRODUCCIÓN

¿PARA QUÉ SIRVEN LOS ORDENADORES?

- Hace no mucho tiempo, imaginar que en cada hogar podría existir un ordenador de uso doméstico era casi ciencia-ficción. Las computadoras eran enormes aparatos .
- Los precios se abarataron y se fueron creando aplicaciones para unos ordenadores que iban siendo más pequeños y más asequibles.
- Escribir un trabajo, dibujar, editar fotografías, montar una película de vídeo, crear una pagina web, navegar en Internet...











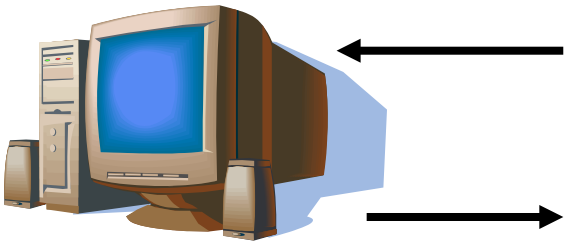






FUNCIONAMIENTO DE UN ORDENADOR

En el ordenador siempre se realiza algún tipo de tratamiento de la información. Puede ser tan sencillo como calcular la suma de dos números, o tan complejo como obtener datos de planetas remotos.

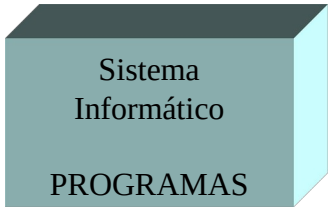


INFORMACIÓN DE ENTRADA

INFORMACIÓN DE SALIDA

¿Cómo procesa la información? → PROGRAMA

- El ordenador necesita conocer cómo debe procesar la información. Esta característica la obtiene a través de un **programa** que tiene almacenado y que contiene todas las instrucciones para la elaboración de los datos.



Ejemplo de un programa

```
using System;
using System.Collections.Generic;
using System.Text;

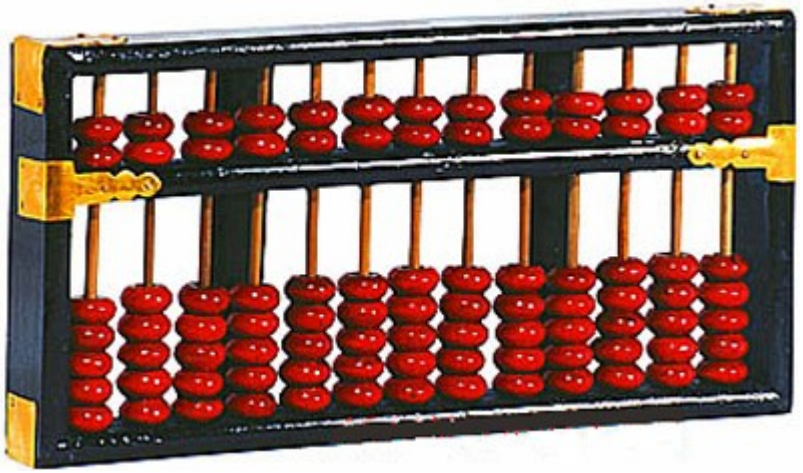
/* ============================================================
   Ejemplo 4.1
   Programa que lee un numero por teclado y calcula su cuadrado
  *
  * Ejemplo de sentencias simples
    ============================================================*/
namespace Ejemplo01
{
    class Program
    {
        static void Main(string[] args)
        {
            int n;          // Guardara el nº leido de teclado
            int cuadrado_n; // Cuadrado de n
            char car;       // caracter para finalizar el programa
            string cadena;

            Console.Write("Introduzca un número :");    // mostramos mensaje
            cadena = Console.ReadLine();                // leemos el número
            n = int.Parse(cadena);
            cuadrado_n = n * n;                             // calculamos su cuadrado
            Console.Write("\nEl cuadrado de {0} es {1}\n",    // mostramos resultados
                          n,cuadrado_n);
            Console.WriteLine("Pulse INTRO para finalizar ...");
            Console.Read();
        }
    }
}
```

2. EVOLUCIÓN DE LOS ORDENADORES

- (2000 y 1000 a.C.) Primer instrumento para facilitar el cálculo: ábaco.
- Puede ser considerado como el origen de las máquinas de calcular.
- A pesar de su antigüedad se sigue usando en algunos países orientales





- La historia de la computadora empieza propiamente con la primera computadora electrónica digital, construida prácticamente a finales de la *Segunda Guerra Mundial.* Pero esta historia sólo ha podido darse gracias a la aportación de hombres que idearon máquinas empleadas para ejecutar cálculos rápidos y precisos que dieran solución a algún problema matemático.

- La computadora nace en 1834, con *Charles Babbage (1791-1871)* y su máquina analítica. Este diseño contenía todos los elementos que constituyen a una computadora moderna.

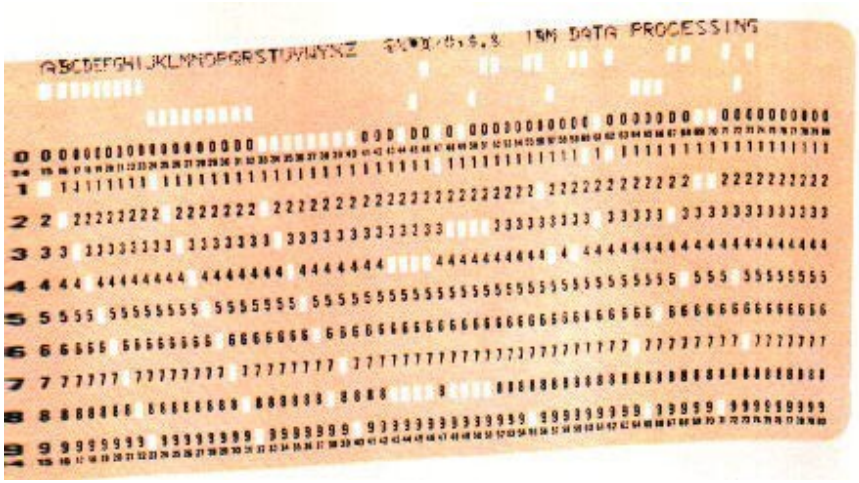
Fig. Charles Babbage



- La máquina analítica recibía los datos con los que se iba a trabajar por medio de tarjetas perforadas. La máquina ejecutaba las instrucciones que indicaban que hacer con los datos proporcionados, para obtener los resultados deseados.



MAQUINA BABBAGE



- Después de casi cien años de *Babbage*, en 1944, se construyó en la *Universidad de Harvard* la computadora ***Mark I***, cuyo diseño fue dirigido por *Howard H. Aiken.* Medía 15 m de largo y 2,5 de alto, pesaba 5 toneladas y tenía 800 Km de cables

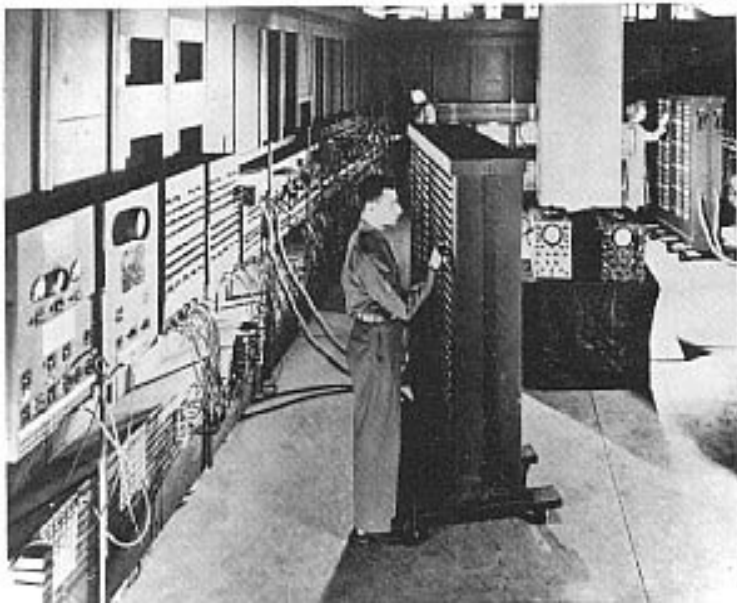


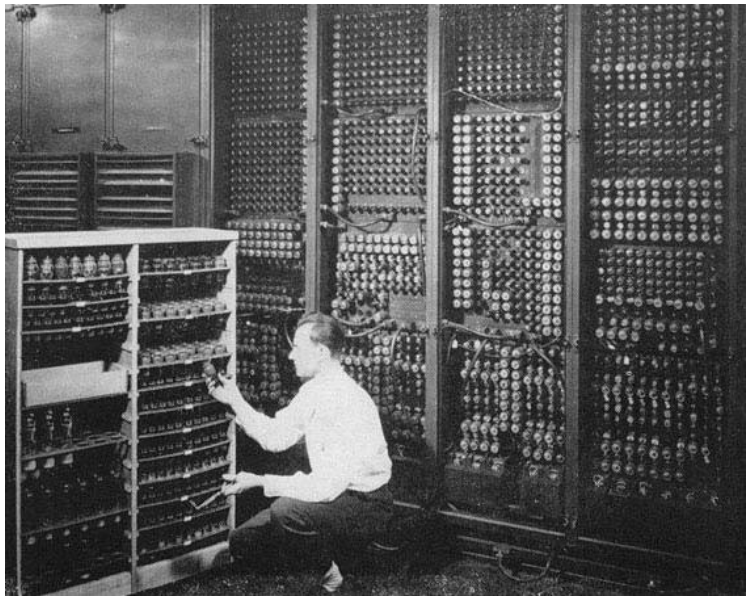


MARK I

- En 1947, un equipo encabezado por *John Mauchly* y *John Eckert*, de la *Universidad de Pennsylvania* construye una enorme computadora llamada ***ENIAC*** (Electronic Numerical Integrator And Calculator) que es la primera computadora digital electrónica de la historia.

- Esta máquina ocupaba 140m^2 y pesaba 30 toneladas, poseía mas de 18,000 tubos de vacío, consumía 200KW y requería de un sistema industrial de aire acondicionado. Pero era capaz de efectuar aproximadamente 5,000 operaciones aritméticas en un segundo.





3. GENERACIONES DE ORDENADORES

3.1 Primera generación (1946-1955)

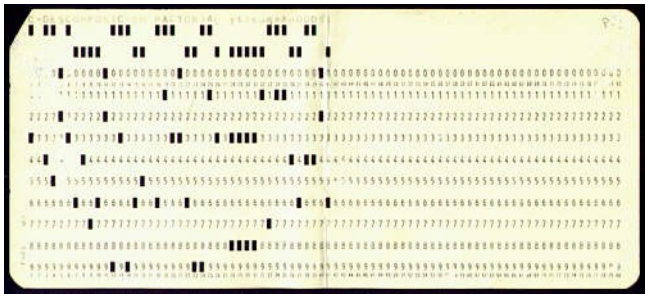
- Esta primera etapa se situó en la década de 1950. Las máquinas de esta generación estaban construidas por medio de ***válvulas de vacío.***

- Así, en 1951 se construyó la *UNIVAC I,* el primer ordenador basado en válvulas de vacío. Pesaba más de 30 Toneladas y su programación era totalmente manual. Se utilizó para procesar el censo de Estados Unidos.

- La forma de procesar la información era secuencial (un proceso detrás de otro) y los datos se introducían mediante **tarjetas perforadas**.



VALVULA DE VACIO



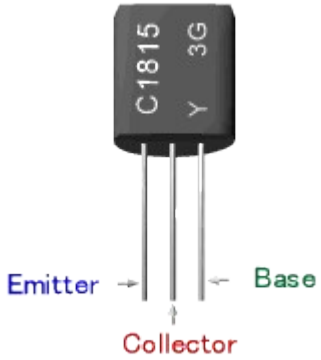
TARJETA PERFORADA

3.2 Segunda generación (1955-1964)

- Las máquinas de la segunda generación estaban fabricadas con **transistores** .
- Los transistores son mucho más pequeños, desprenden menos calor y se averían menos que las antiguas válvulas de vacío.
- Entre los primeros modelos de la segunda etapa se puede mencionar la *Philco 212*, la *UNIVAC M460*, la *CDC 1604*, la *IBM 7090*….



Philco 212



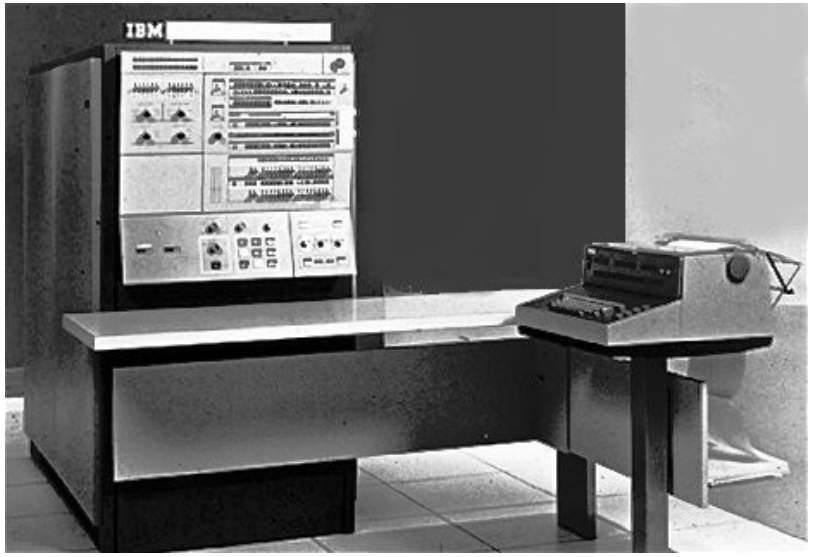
TRANSISTOR

3.3 Tercera generación (1964-1970)

- En esta etapa comienzan a utilizarse los **circuitos integrados**.
- Estos circuitos están formados por miles de transistores conectados entre sí e insertados en un solo chip.
- Con esta nueva tecnología los ordenadores reducen aún más su tamaño y aumentan su velocidad de cálculo, pudiendo ejecutar varios programas a la vez.
- Máquinas de la tercera generación son: IBM 360, la *CDC 6000*, la *UNIVAC 1108* y *1110*, *Honeywell DPS*, entre otras.



CIRCUITO INTEGRADO



IBM 360

3.4 Cuarta generación (1970-1980)

- En 1972 surgen los ***microprocesadores***.

- Este mayor nivel de integración hace posible reducir aún más el tamaño de los ordenadores.



MICROPROCESADOR



3.5 Quinta generación (a partir de 1981)

- En 1981 IBM construyó el primer ordenador personal (PC-basado en el sistema operativo MS-DOS) a partir del cual se revolucionó el mercado informático.

- La utilización de circuitos con mayor nivel de integración, la bajada de los precios y el continuo aumento de prestaciones y servicios generalizan la difusión del ordenador .



- En nuestros días existe una gran variedad de computadoras en el mercado.

CONTINUO AVANCE

- Se desarrollan nuevos modelos de PC más potentes y rápidos (8088,80286,80386,80486,Pentium,Core)

  Ver la evolución de los AMD en http://es.wikipedia.org/wiki/Microprocesador

- Se generaliza el uso de los Portátiles

- Se amplían las posibilidades del ordenador y aparece el concepto de MULTIMEDIA



Se potencia la comunicación entre ordenadores: INTRANET e INTERNET

Superordenador actual- *Noticia El País - 22/06/2011*

- Un superordenador japonés capaz de realizar más de ocho cuatrillones de operaciones por segundo (ocho petaflops) es el nuevo número uno de la lista Top500 de clasificación internacional de las máquinas más veloces.

- El líder se llama ***K computer***, es un desarrollo conjunto de Fujitsu y la fundación Riken y con él Japón regresa a la cabecera de la lista Top500, desde que fue destronado del número uno en 2004.

- Por primera vez, los diez primeros ordenadores de Top500 están en el rango de los petaflops y Estados Unidos tiene cinco, mientras que Japón tiene dos, China otros dos y Francia, uno. La máquina española mejor situada es la Magerit, de la Universidad Politécnica de Madrid ,que ocupa el puesto 136, y la del Centro de Supercomputación de Barcelona, el 171.

- **PETAFLOPS: Un flop es una medida de rendimiento, y "peta" significa 10^15 , así que un petaflop simplemente significa que una computadora puede realizar 1,000,000,000,000,000 operaciones de punto flotante.**

- El ranking de los ordenadores más rápidos se basa en la velocidad alcanzada por cada máquina ejecutando una aplicación denominada Linpack, desarrollada para solucionar determinados sistemas de ecuaciones.

- La lista Top500 se empezó a realizar en 1993 y se actualiza dos veces al año (en junio y en septiembre).

- El más rápido, el *K computer*, aún en fase de configuración y tiene 68.544 CPU (unidad central de procesamiento), con un rendimiento de 8.162 petaflops. Cuando su configuración esté completa, en 2012, debe lograr un rendimiento de 10 petaflops, según se ha diseñado, informa Fujitsu en un comunicado.





4. REPRESENTACIÓN DE LA INFORMACIÓN EN EL ORDENADOR

REPRESENTACIÓN DE LA INFORMACIÓN

- Para que un ordenador pueda llevar a cabo el procesamiento de información, debe ser capaz de poder representar esos datos mediante algún método.

- Ejemplo: los humanos utilizamos los sonidos y los caracteres escritos para representar y comunicar la información.

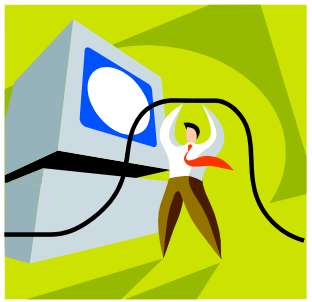
- Los pioneros en el diseño de computadores se enfrentaron a este problema cumpliendo las siguientes condiciones:

  - Poder representar cualquier tipo de información (números, letras, etc).
  - Ser compatibles con los circuitos
  - Representar unívocamente la información (sin confusiones).

- Las válvulas de vacío en la que se basaban los primeros computadores funcionaban como un interruptor: podían dejar pasar la corriente o se podía cortar para que no circulase corriente.

- Ante esto, nos encontramos con un sistema que solamente trabaja con dos estados posibles: hay corriente circulando o no.

- En la actualidad los ordenadores siguen utilizando el mismo sistema para representar información. Esta es representada mediante 2 símbolos (0,1) que se corresponden con los dos estados eléctricos que pueden adoptar los circuitos que forman el ordenador.



- Así un ordenador es capaz de manejar caracteres:
  - Numéricos (0,1,2,…,9)
  - Alfabéticos (A..Z,a..z)
  - Especiales ( ) / & $

- Para ello es necesario una correspondencia entre dichos símbolos y los que usa el ordenador (0,1). De esta forma todos los datos empleado por el ordenador estarán codificados en binario.

- Esta codificación está basada en el sistema de numeración binario.

4.1 SISTEMAS DE NUMERACIÓN

Es un conjunto de símbolos y reglas que se utilizan para la representación de datos numéricos.

1 Sistema decimal

2 Sistema binario

4.1.1 SISTEMA DECIMAL

- El sistema decimal es el que empleamos normalmente
- B_{10} {0,1,2..9}
- Un número mayor se representa con una combinación posicional de los anteriores
- El valor de cada dígito está asociado al de una potencia de base 10 y un exponente igual a la posición que ocupa el dígito (contando desde la derecha)menos uno.
- Ejemplo:
- $3284_{10} = 3x10^3 + 2x10^2 + 8x10^1 + 4x10^0$
- $4578_{10} = 4 * 10^3 + 5 * 10^2 + 7* 10^1 + 8*10^0$

4.1.2 SISTEMA BINARIO

- B_2 {0,1}
- Cada cifra o dígito de un numero representado en este sistema se denomina BIT (binary digit). El BIT representa la cantidad más pequeña de información que puede manejar un sistema informático.
- Para cantidades de información más grandes se utilizan los múltiplos del BIT:

1 BYTE = 8 bits

- EJEMPLO : 00100_2

1Kilobyte (KB)	1024 bytes
1 Megabyte (MB)	1024 Kilobytes
1Gigabyte (GB)	1024 Megabytes
1 Terabyte (TB)	1024 Gigabytes

Conversión de Binario a Decimal y viceversa

Ponemos un 1(activamos) en aquellas posiciones cuyo suma del valor decimal que representan me dan el número que busco.

…	64	32	16	8	4	2	1
…	2^6	2^5	2^4	2^3	2^2	2^1	2^0

12 en binario es igual a 1100

23 en binario es igual a 10111

EJERCICIOS



- 1. Crea una tabla de conversión de los 16 primeros números decimales (empezando por el 0) al sistema binario (utilizando 4 bits).
- 2. Pasa a binario los siguientes números

  199, 35, 85, 64, 76, 89
- 3. Dados los números binarios 01001000 y 01000100, indica cuál es mayor. ¿Es necesario convertir los números al sistema decimal para compararlos?
- 4. ¿Cuántos disquetes de 3 ½ (su capacidad es de 1,44 MB) podrías copiar en un disco de 20 GB?
- 5. Si la x en un lector de o grabador de CDs equivale a 150Kb/s, ¿Cuántos Megabytes puede leer, en un minuto, una lectora cuya velocidad es de 54x?

EJERCICIOS



- 6. Pasa a decimal los siguientes números binarios

011100101
100100
000100010
110111101
1100100
010101
10101010

4.2 Representación de caracteres



Código ASCII

¿Qué es un código de caracteres?

Como ya se ha indicado el ordenador necesita tener los datos codificados en forma binaria, es decir, convertidos en 0 y 1; por tanto, todos los caracteres (letras, números y símbolos) deben disponer de su correspondiente codificación binaria, los que da lugar al denominado código de caracteres.

Este código de caracteres representa cada carácter mediante un número binario constituido por una serie dígitos menor o igual a ocho

Existen distintos códigos de caracteres, siendo el más utilizado el ASCII. En este sistema a cada carácter se le asigna un número decimal comprendido entre 0 y 255 que, una vez convertido al sistema de numeración binario, nos da el código del carácter.

ASCII 7 bits → $2^7 = 128$ caracteres diferentes para representar

ASCII 8 bits → $2^8 = 256$ caracteres diferentes para representar

0		32		64	@	96	`	128	Ç	160	á	192	└	224	α
1	☺	33	!	65	A	97	a	129	ü	161	í	193	┴	225	ß
2	☻	34	"	66	B	98	b	130	é	162	ó	194	┬	226	Γ
3	♥	35	#	67	C	99	c	131	â	163	ú	195	├	227	π
4	♦	36	$	68	D	100	d	132	ä	164	ñ	196	─	228	Σ
5	♣	37	%	69	E	101	e	133	à	165	Ñ	197	┼	229	σ
6	♠	38	&	70	F	102	f	134	å	166	ª	198	╞	230	µ
7	•	39	'	71	G	103	g	135	ç	167	º	199	╟	231	τ
8	◘	40	(	72	H	104	h	136	ê	168	¿	200	╚	232	Φ
9	○	41	)	73	I	105	i	137	ë	169	⌐	201	╔	233	Θ
10	◙	42	*	74	J	106	j	138	è	170	¬	202	╩	234	Ω
11	♂	43	+	75	K	107	k	139	ï	171	½	203	╦	235	δ
12	♀	44	,	76	L	108	l	140	î	172	¼	204	╠	236	∞
13	♪	45	-	77	M	109	m	141	ì	173	¡	205	═	237	ø
14	♫	46	.	78	N	110	n	142	Ä	174	«	206	╬	238	ε
15	☼	47	/	79	O	111	o	143	Å	175	»	207	╧	239	∩
16	►	48	0	80	P	112	p	144	É	176	░	208	╨	240	≡
17	◄	49	1	81	Q	113	q	145	æ	177	▒	209	╤	241	±
18	↕	50	2	82	R	114	r	146	Æ	178	▓	210	╥	242	≥
19	!!	51	3	83	S	115	s	147	ô	179	│	211	╙	243	≤
20	¶	52	4	84	T	116	t	148	ö	180	┤	212	╘	244	⌠
21	§	53	5	85	U	117	u	149	ò	181	╡	213	╒	245	⌡
22	▬	54	6	86	V	118	v	150	û	182	╢	214	╓	246	÷
23	↨	55	7	87	W	119	w	151	ù	183	╖	215	╫	247	≈
24	↑	56	8	88	X	120	x	152	ÿ	184	╕	216	╪	248	°
25	↓	57	9	89	Y	121	y	153	Ö	185	╣	217	┘	249	∙
26	→	58	:	90	Z	122	z	154	Ü	186	║	218	┌	250	·
27	←	59	;	91	[	123	{	155	¢	187	╗	219	█	251	√
28	∟	60	<	92	\	124	¦	156	£	188	╝	220	▄	252	n
29	↔	61	=	93	]	125	}	157	¥	189	╜	221	▌	253	2
30	▲	62	>	94	^	126	~	158	₧	190	╛	222	▐	254	■
31	▼	63	?	95	_	127	⌂	159	ƒ	191	┐	223	▀	255	

CODIGO ASCII

EJERCICIOS

EJERCICIOS



- 1. ¿Cuántos bits ocuparía tu nombre completo?
- 2. ¿Cómo se transmitiría en binario la palabra Informática?
- 3. ¿Cómo se transmitiría en binario tu nombre?
- 4. ¿Cuántos caracteres diferentes se pueden representar utilizando el sistema de numeración binario con 3 dígitos?

  ¿ Y con 8 dígitos ?

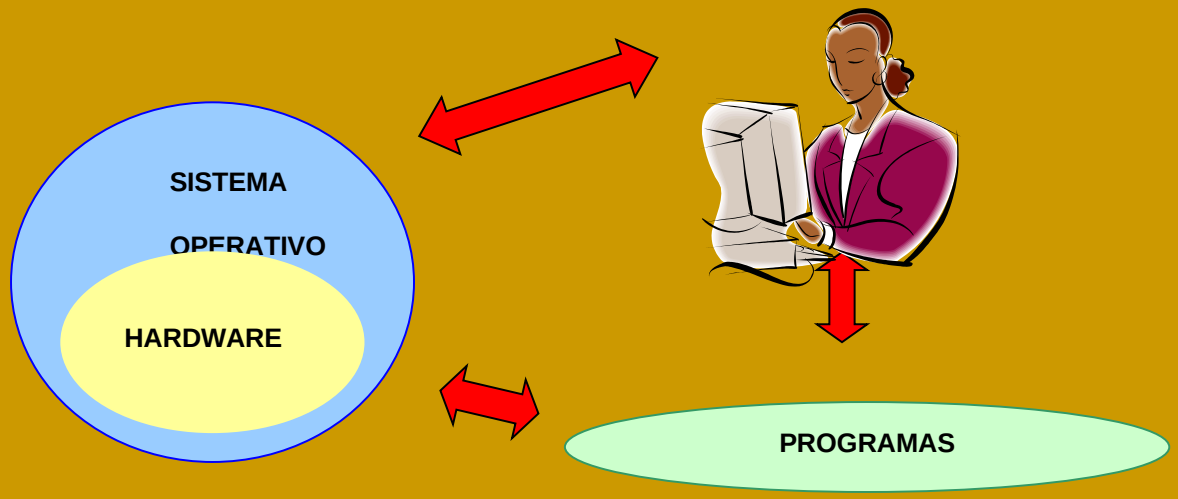
5. CONCEPTOS BÁSICOS

HARDWARE Y SOFTWARE

HARDWARE: Conjunto de componentes físicos que forman el ordenador

SOFTWARE: Conjunto de instrucciones necesarias para que el Hardware pueda funcionar

SISTEMA OPERATIVO: Está formado por un conjunto de programas que permiten administrar, gestionar y facilitar el uso de los recursos de un ordenador.



- HARDWARE





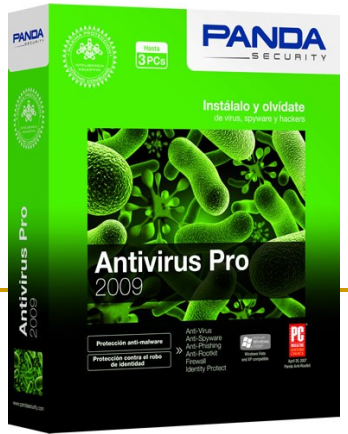
- SISTEMA OPERATIVO





- SOFTWARE





FUNCIONAMIENTO GENERAL DE UN ORDENADOR



		Horas
1	INTRODUCCIÓN	1+4
2	PROCESOS Y THREADS	9
3	GESTIÓN DE MEMORIA	9
4	ENTRADA / SALIDA	

1. Introducción
Curso: 20/21

- ¿Qué sistemas operativos nos suenan?
- ¿Qué es un sistema operativo?
- Evolución de los sistemas operativos
- Recordando aspectos hardware
- Tipos de sistemas operativos
- Conceptos de sistemas operativos (tipo UNIX)
- Llamadas al sistema
- Estructura de los sistemas operativos

¿Qué sistemas operativos nos suenan?

Linux, MacOS, Novell Netware, Solaris, UNIX, Windows, ...

www.netmarketshare.com

Clientes_Desktop Junio_2020





2020	Win10	Win8	Win7	WinXP	Linux	Mac	Chrome OS	Mobile
May	60.1%	3.1%	7.2%	0.1%	4.9%	11.9%	0.4%	12.3%
April	60.1%	3.2%	7.4%	0.1%	4.8%	12.4%	0.4%	11.8%
March	60.6%	3.2%	8.5%	0.1%	5.4%	11.1%	0.4%	10.8%
February	59.1%	3.5%	9.8%	0.2%	5.9%	9.9%	0.0%	11.4%
January	58.1%	3.6%	10.6%	0.2%	6.4%	9.7%	0.4%	11.2%

www.w3schools.com/browsers

¿Qué sistemas operativos nos suenan?

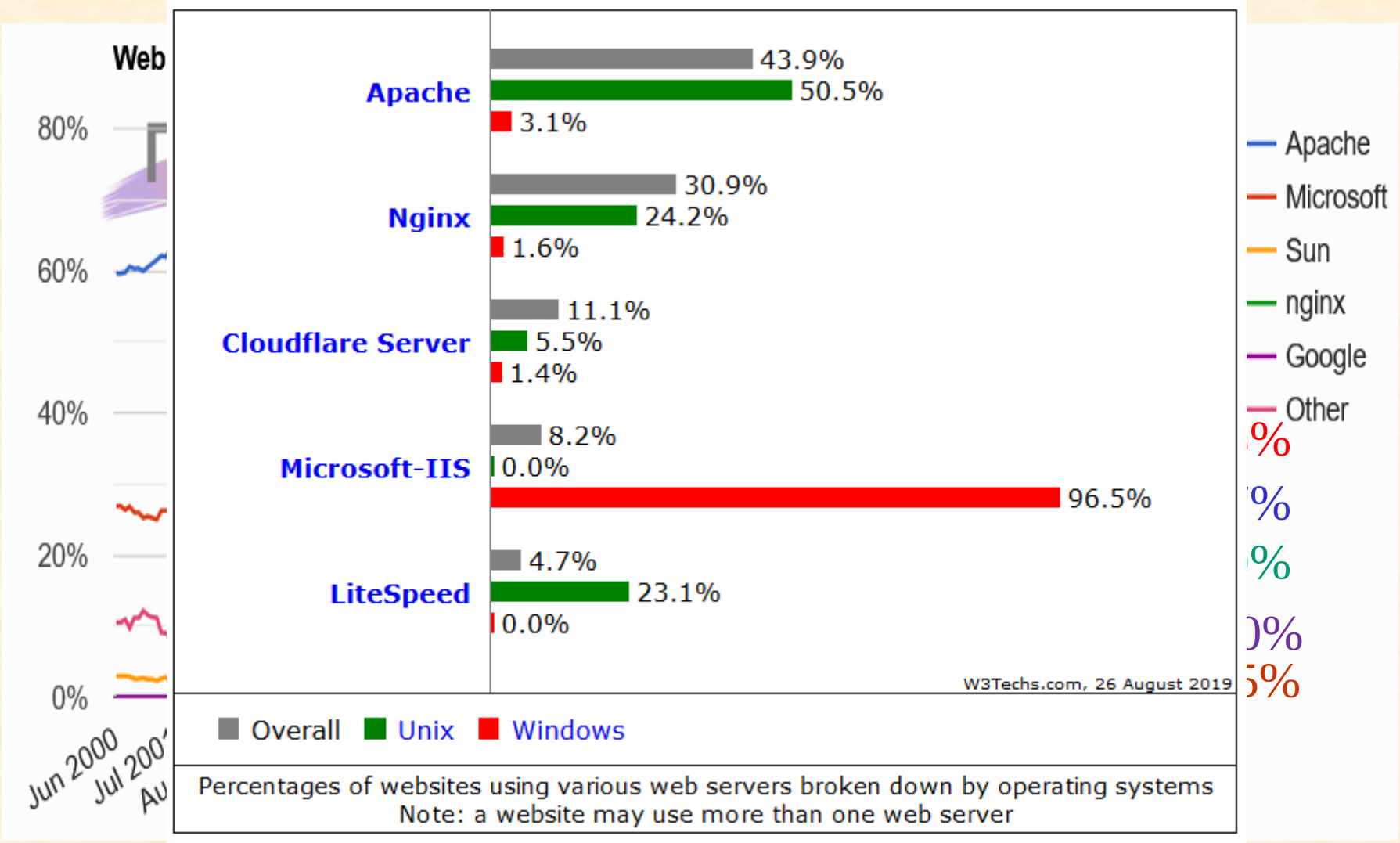
Servidores Web Activos: news.netcraft.com



¿Qué sistemas operativos nos suenan?

Servidores Web Activos: news.netcraft.com

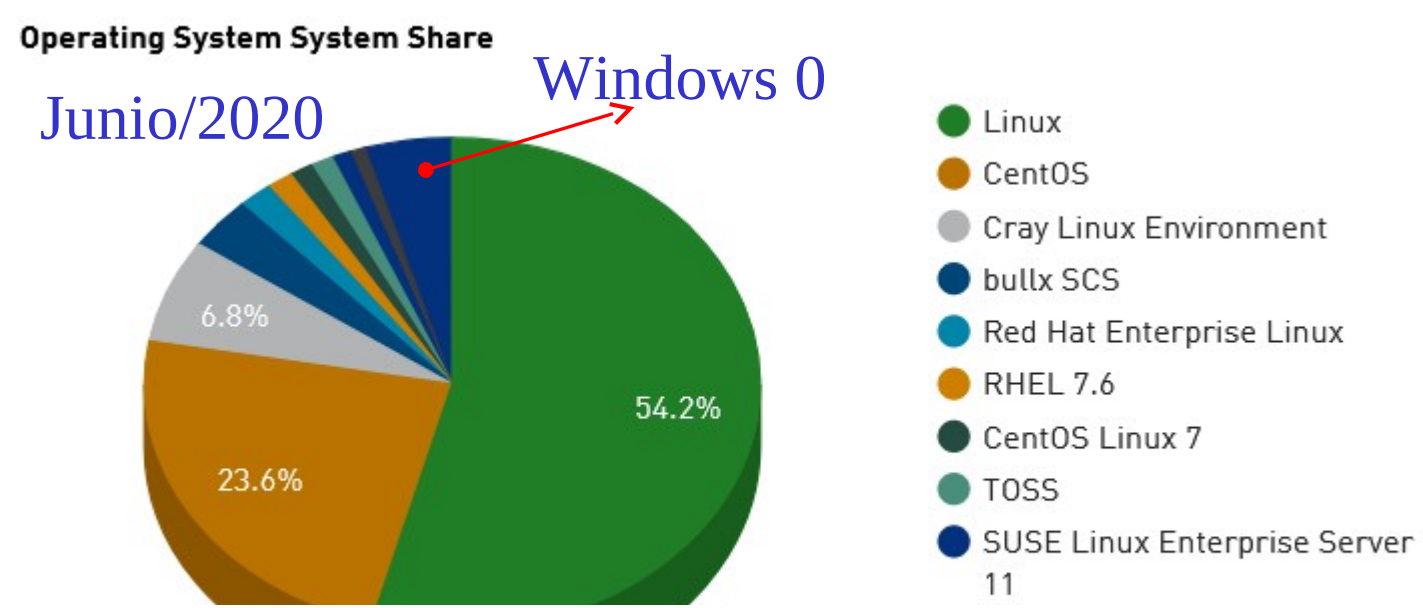




¿Qué sistemas operativos nos suenan?

Linux

www.top500.org ¿Qué S.O. usan?



Harmony OS: la alternativa Huawei a Android debutará en los televisores Honor Vision

13/08/2019





¿Qué sistemas operativos nos suenan?

Mobile Operating System Market Share Worldwide

June 2015 - June 2020

Edit Chart Data



Android

iOS

Huawei se queda con Android, su sistema Harmony OS no está pensado para «smartphones» 23/08/2019

80% 64% 48% 32% 16% 0%

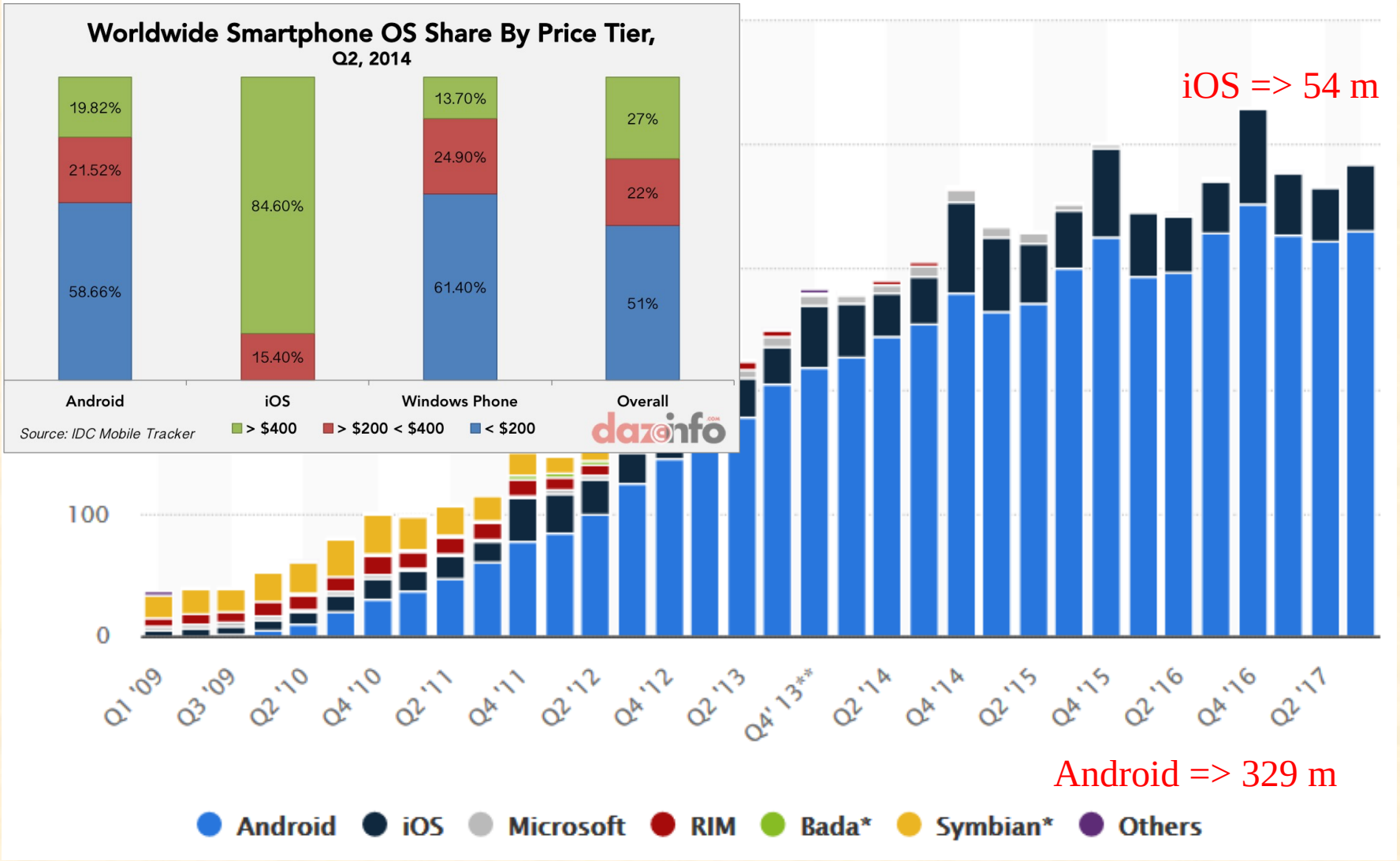
Jan 2016 Sept 2016 Jan 2017 Sept 2017 Jan 2018 Sept 2018 Jan 2019 Sept 2019 Jan 2020

Android iOS Unknown Nokia Unknown Series 40 Windows Samsung BlackBerry OS Other (dotted)

gs.statcounter.com

¿Qué sistemas operativos nos suenan?

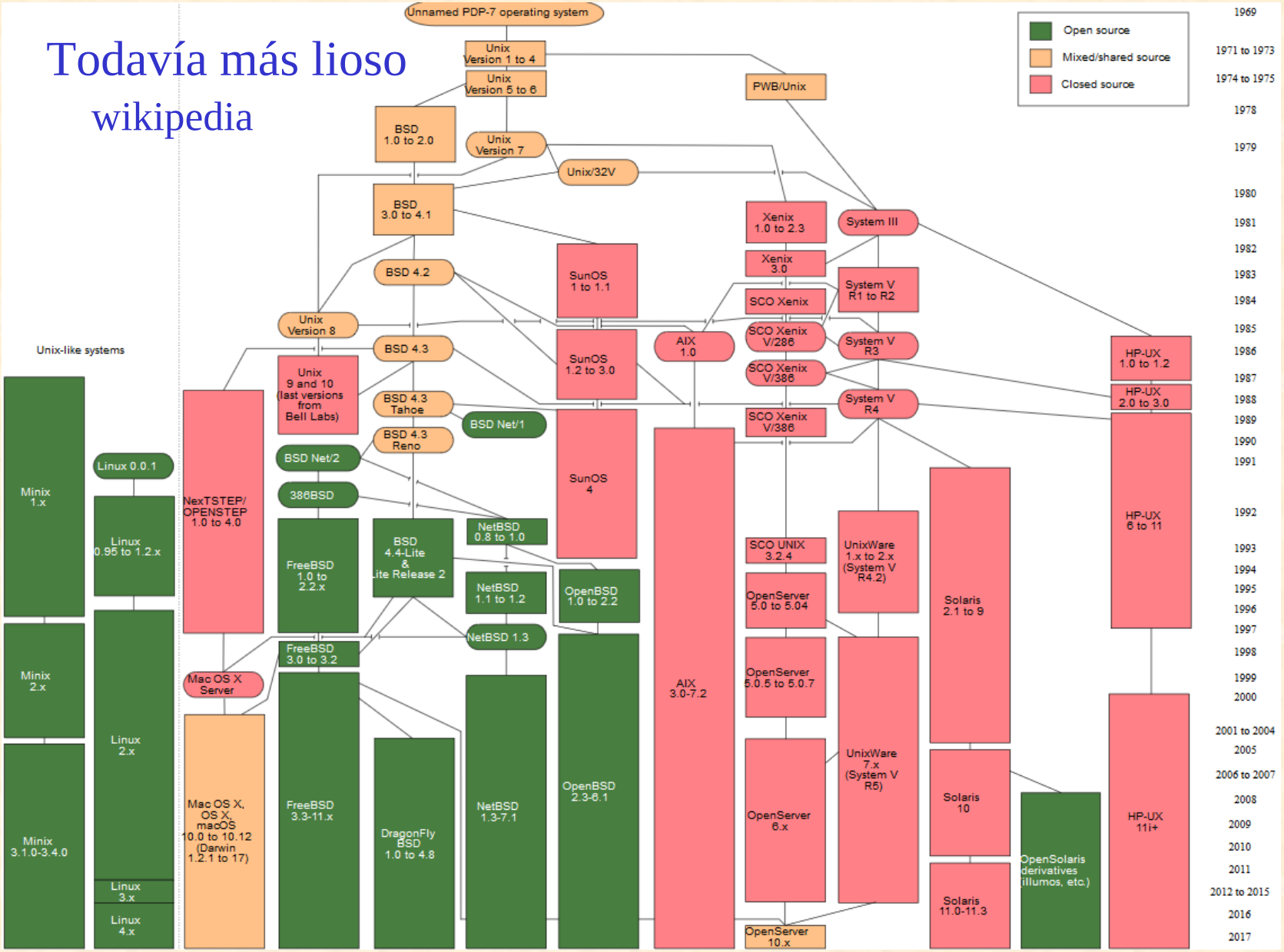
https://www.statista.com/statistics/266219/global-smartphone-sales-since-1st-quarter-2009-by-operating-system/



¿Qué sistemas operativos nos suenan?

Todavía más lioso

wikipedia



¿Qué es un sistema operativo?



Máquina virtual

Top Down





Bottom Up

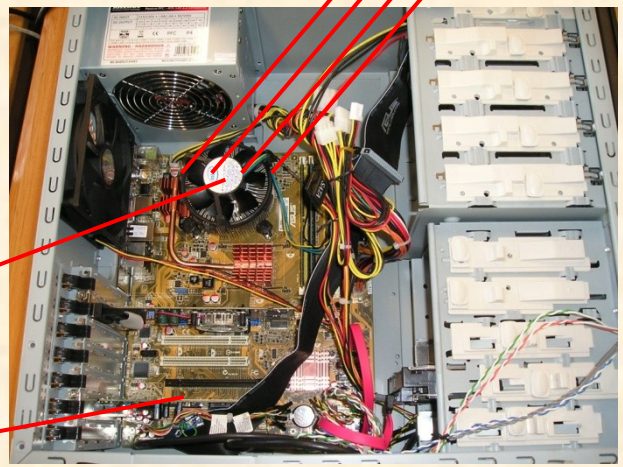


Administrador de recursos



- Escritorio
- Ventanas
- Carpetas

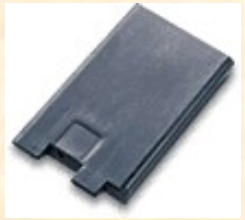
4 núcleos



Baterias







Memoria

RAM, L1, L2, L3

¿Qué es un sistema operativo?









¿Qué es un sistema operativo? ¿Dónde se ubica?

Programas de aplicación

Programas del sistema

Gestión Bancaria	Reserva de Billetes	Juegos
Compila dores	Editores	Montador

Llamadas al sistema

Intérprete de comandos

S.O.

S E R V I C I O S

Lenguaje máquina

MicroArquitectura

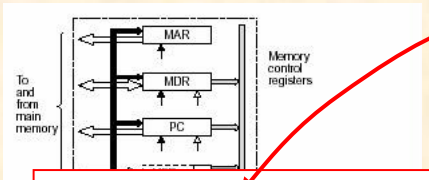
Dispositivos físicos

% sleep 10; bell

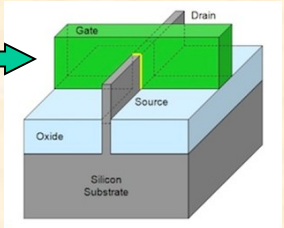
Software

Acceso a periféricos

Hardware




```
seg = 0;
repeat
   alarm(1);
   pause( );
   seg++;
   printf (seg);
until (seg == 60);
```



¿Qué es un sistema operativo? Aspectos esenciales

- Administrador de recursos — CPU's, MP, Discos, ...
- Máquina virtual — ¿Esperar 5 segundos?

```
* registros de la pastilla 68230
reg_cprH    equ     $a00027
reg_cprM    equ     $a00029
reg_cprL    equ     $a0002b

reg_tcr     equ     $a00021
reg_tivr    equ     $a00023
reg_tsr     equ     $a00035

espera      move.w  4(sp),seg
            move.l  256,oldinttimer
            move.l  #int68230,256
            move.b  #$03,reg_cprH
            move.b  #$d0,reg_cprM
            move.b  #$90,reg_cprL
            move.b  #64,reg_tivr
            move.b  #$a1,reg_tcr
            move.w  #$2000,sr
```

```
move.w  #5,SP
bsr espera
```

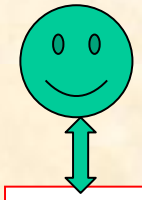
Sin S.O.

¿Error?

```
espera1     tst.w       seg
            bne         espera1
            clr.b       reg_tcr
            move.l      oldinttimer,256
            rts

oldinttimer ds.l        1
seg         ds.w        1

int68230    subi.w      #1,seg
            clr.b       reg_tsr
            move.b      #$a0,reg_tcr
            move.b      #$a1,reg_tcr
            rte
```



¡ Portabilidad !

sleep (5) ← Con S.O.

i386. arm, mips, powerpc, …

Evolución de los sistemas operativos (Leer libro)

1945-1955	1955-1965	1965-1980	1980..
Programación Cableada	Programación Lenguaje Máquina	Multiprogramación Tiempo compartido	
Interacción	Distanciamiento	Interacción	





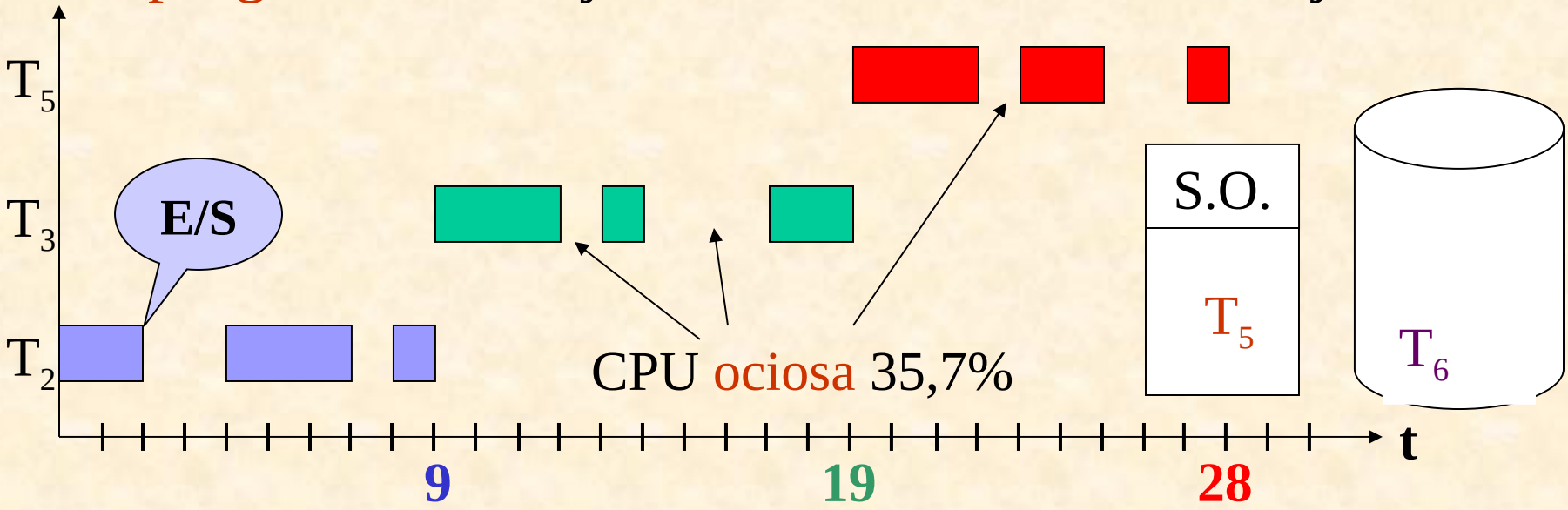
Van apareciendo más y mejores herramientas

- Ensambladores, montadores, cargadores
- Bibliotecas de funciones comunes
- Drivers
- Compiladores (Fortran, Cobol)

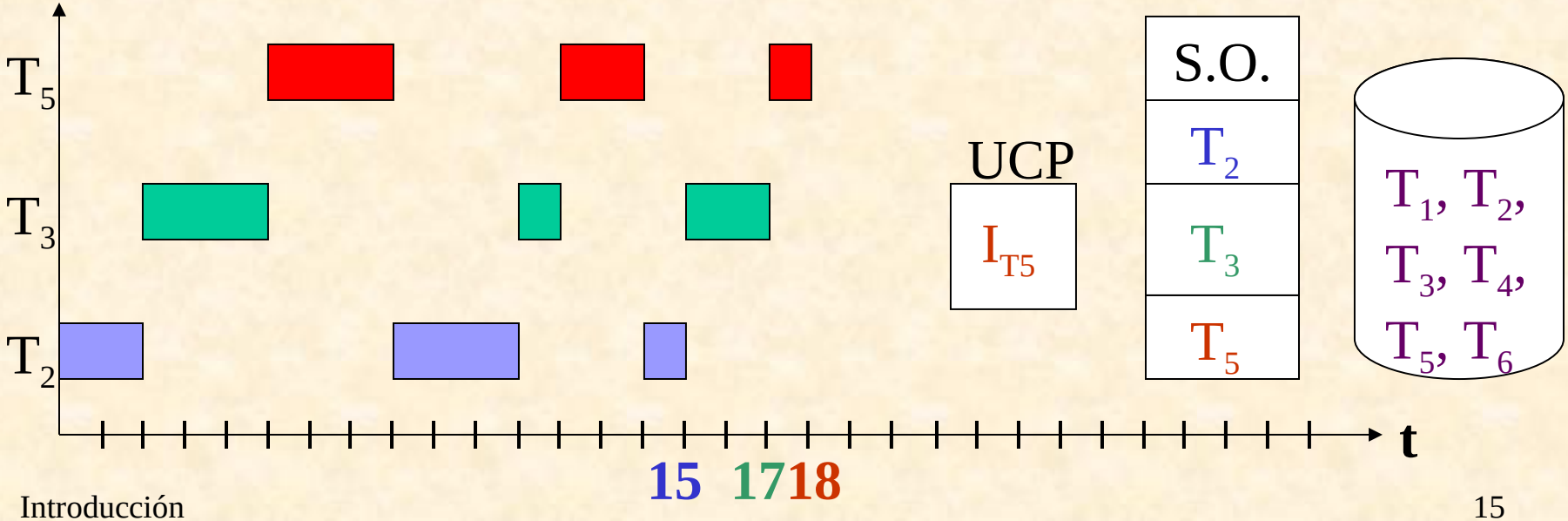
Soporte - automatización

- Procesamiento por lotes (Batch)
- Secuenciación automática de trabajos (monitor) JCL
- Proceso OFF-LINE
- SPOOL
- Multiprogramación y Tiempo compartido

Monoprogramación: Ejecución secuencial de trabajos



Multiprogramación: Ejecución simultánea de trabajos



Multiprogramación: Ejecución simultánea de trabajos

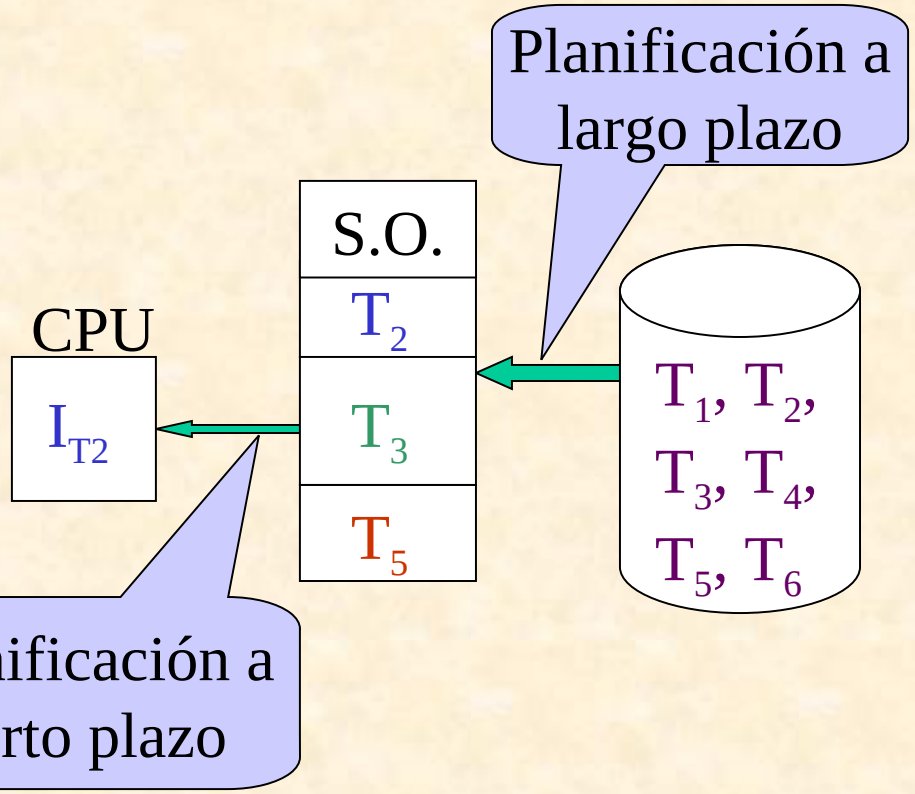
Grandes ventajas:

- Mejor aprovechamiento de CPU 64,3% ➔ 100%
- Soluciones más elegantes (si el problema es en esencia paralelo)
- Ejecución más rápida (si hay más de una CPU)

Algunos inconvenientes:

- Gestión de la CPU
- Gestión de la memoria

  Protección

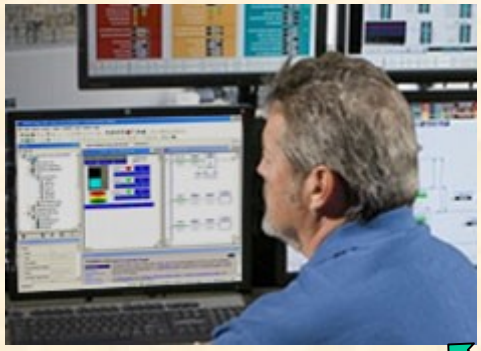


Multiprogramación: Ejecución simultánea de trabajos

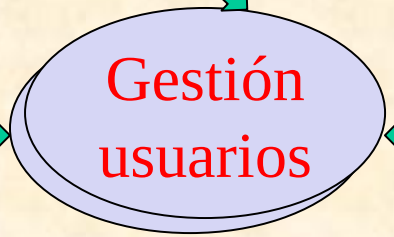
Solución más sencilla y eficiente si inherentemente paralelo

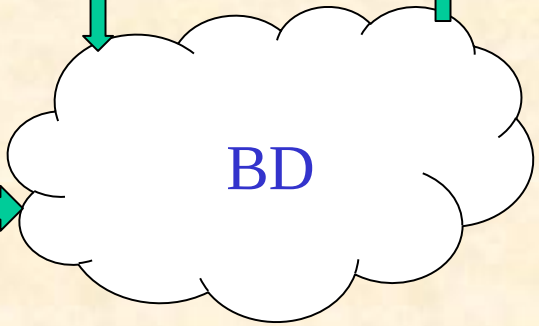
¿ Un único proceso para el control de la fábrica ?

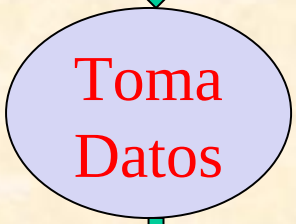


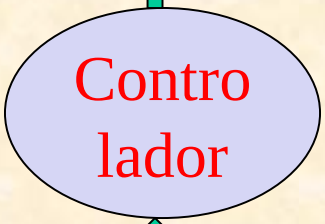








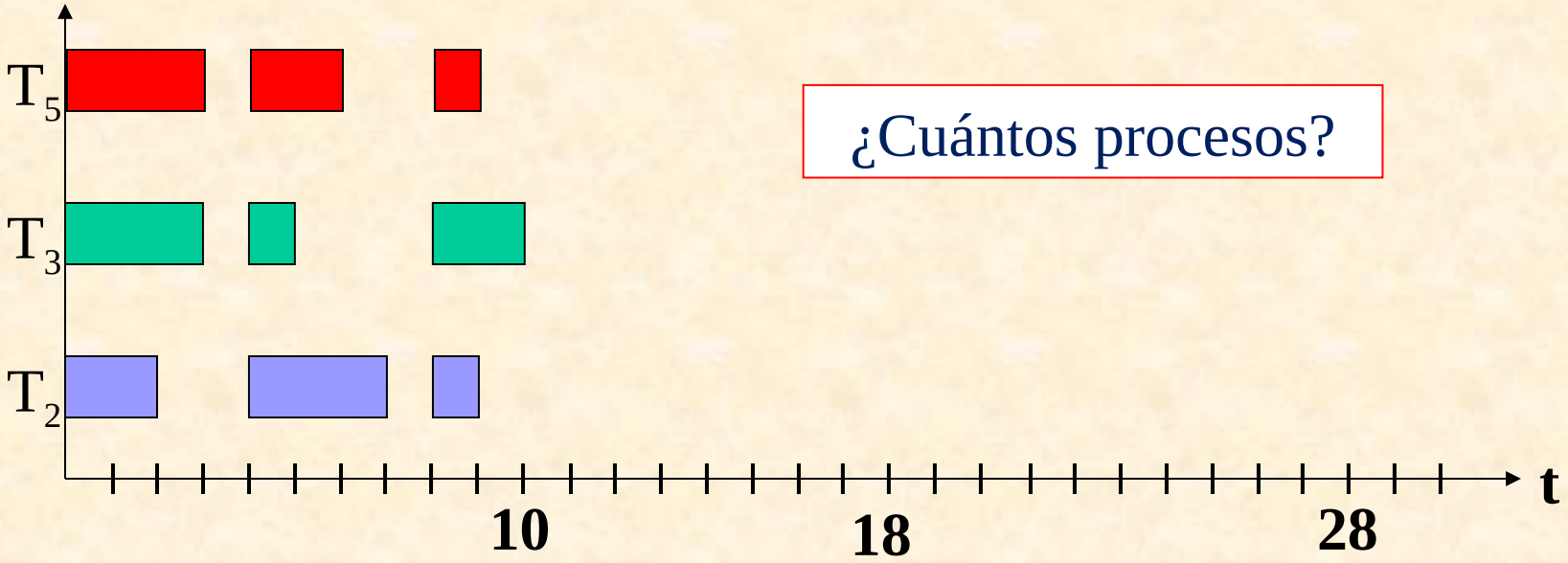




¿Programación concurrente?

Multiprogramación: Ejecución simultánea de trabajos

Aumento cuantitativo de las prestaciones



¿Cuántos procesos?

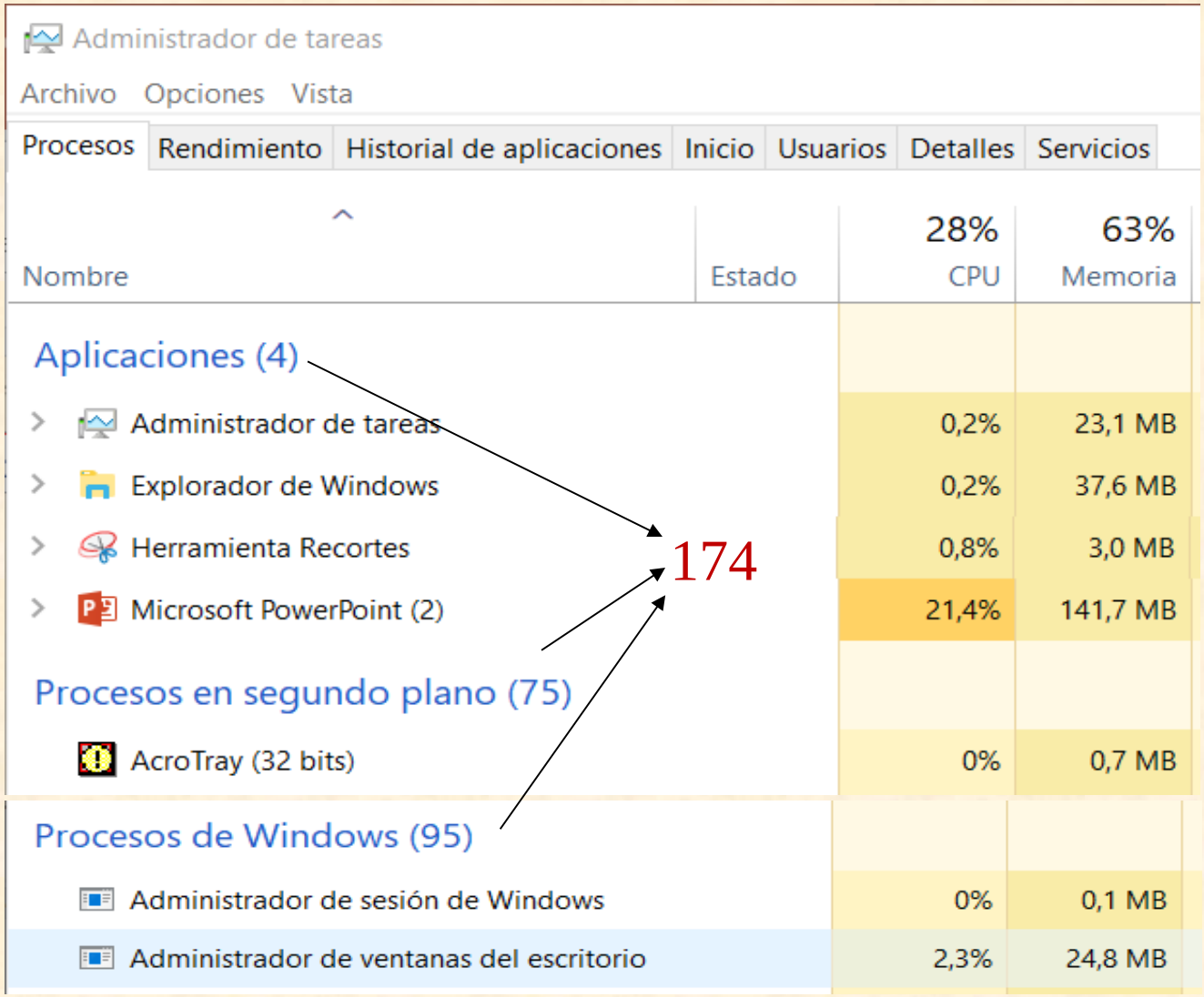
Ejecución paralela

Ejecución pseudo paralela

Ejecución secuencial

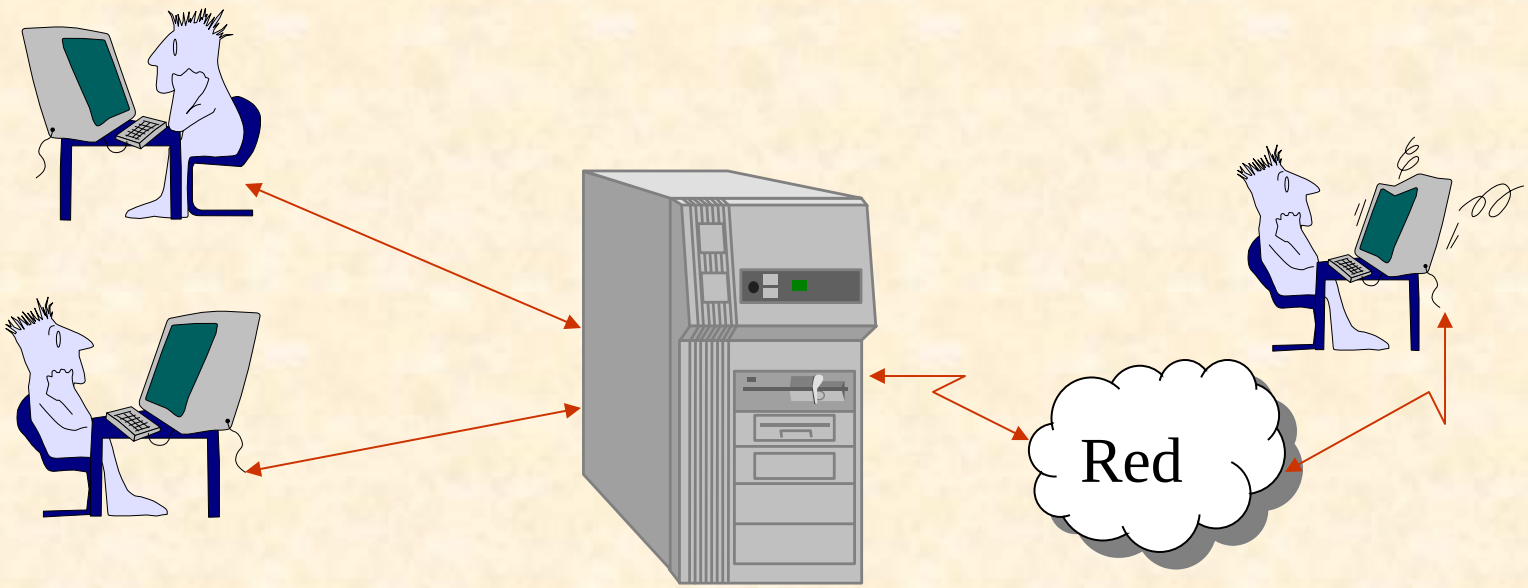
Multiprogramación: Ejecución simultánea de trabajos

¿Cuántos procesos?



Tiempo compartido: Sistema multiprogramado interactivo

Aprovechar la potencia de un servidor por varios usuarios a la vez



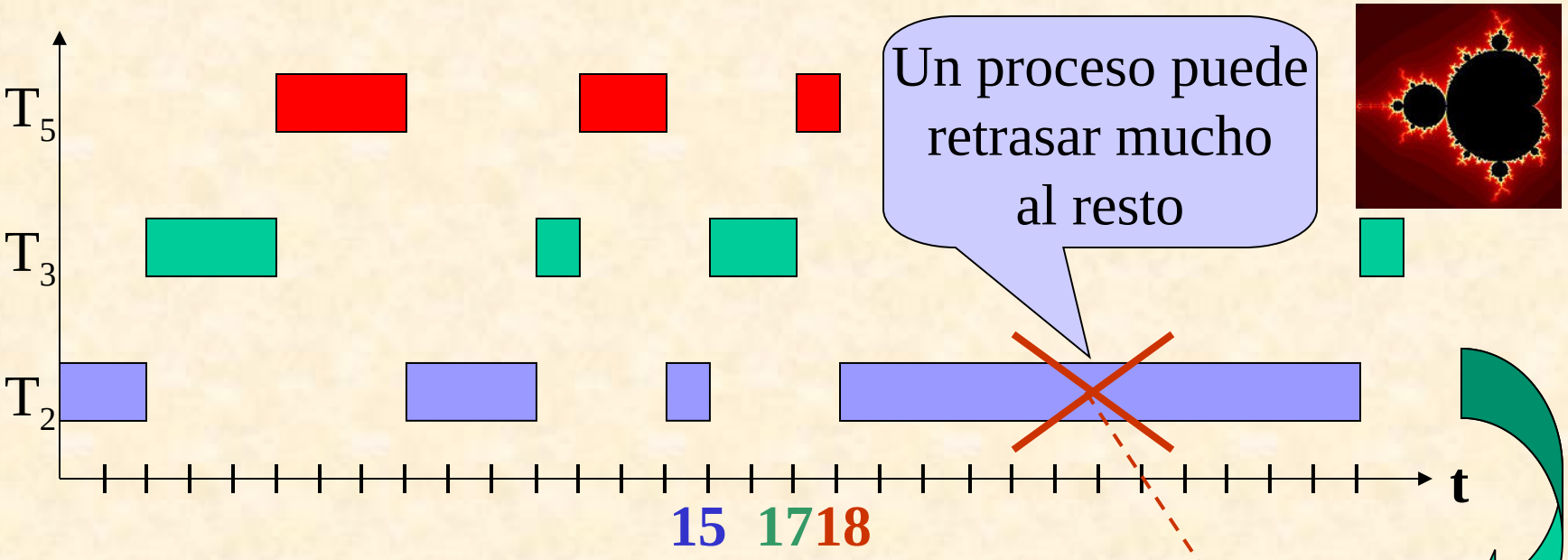
Orígenes MULTICS: Querían dar servicio a toda una ciudad

¿Cada vez menos importante?

Futuro ¿Grandes redes de PC's y compartición de recursos?

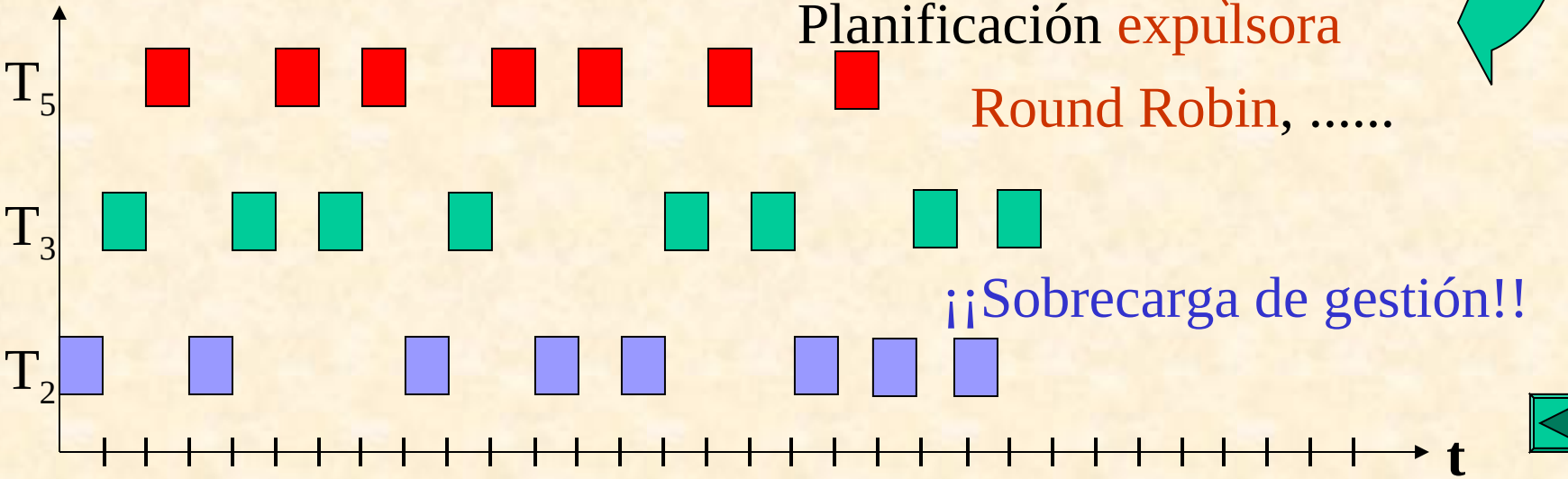
Distinguir entre multiusuario y tiempo compartido

Multiprogramación: No ofrece sensación de interacción



Planificación expulsora

Round Robin, ...



¡¡Sobrecarga de gestión!!

¿Seguro?

Tipos de sistemas operativos



Mainframe

- Batch, transacciones, tiempo compartido
- OS/390, UNIX y Linux

Servidores

- Servicios de impresión, ficheros, web
- UNIX, Windows 2000 server, Linux

Multiprocesador

- UNIX, Encore (RSX-II)

Ordenadores personales

Tiempo real

- Hard y soft: QNX www.qnx.com

Empotrados | Computadoras de bolsillo

- Palm OS, Windows CE, Symbian OS

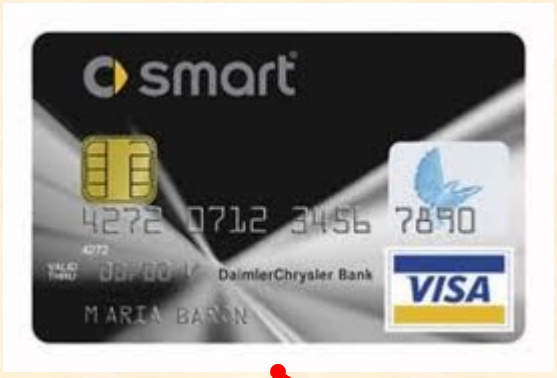
Tarjetas inteligentes

- SCOT, TB ¿Java Card Technology?

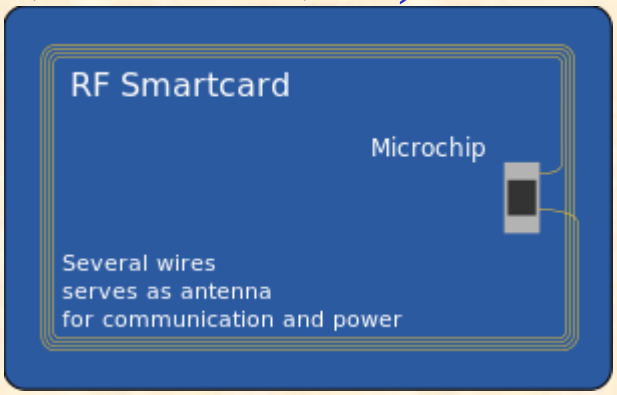


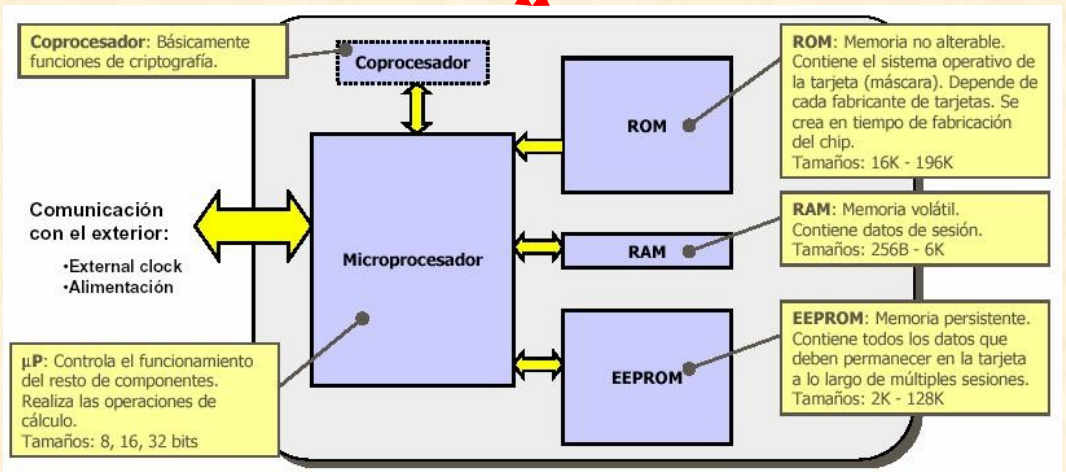
Tipos de sistemas operativos

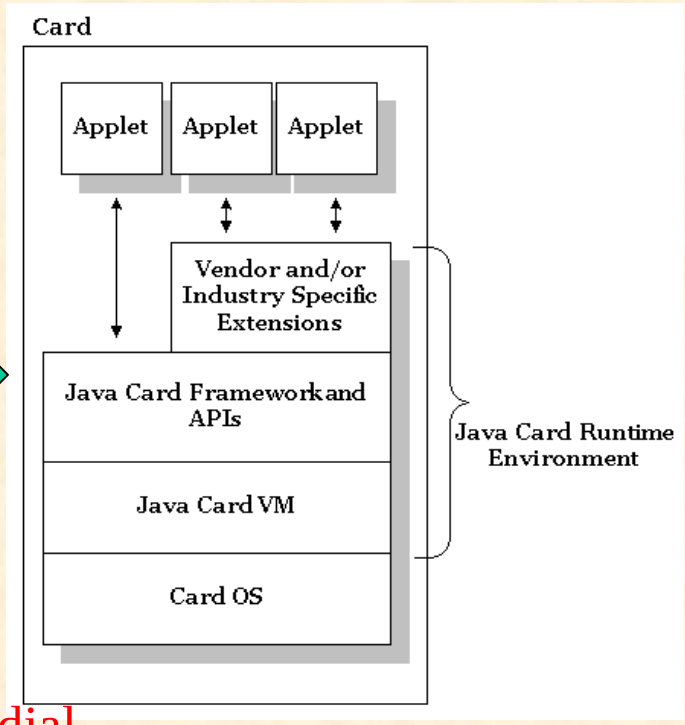
- Tarjetas inteligentes (MULTOS, TIBC, JavaCard, …)









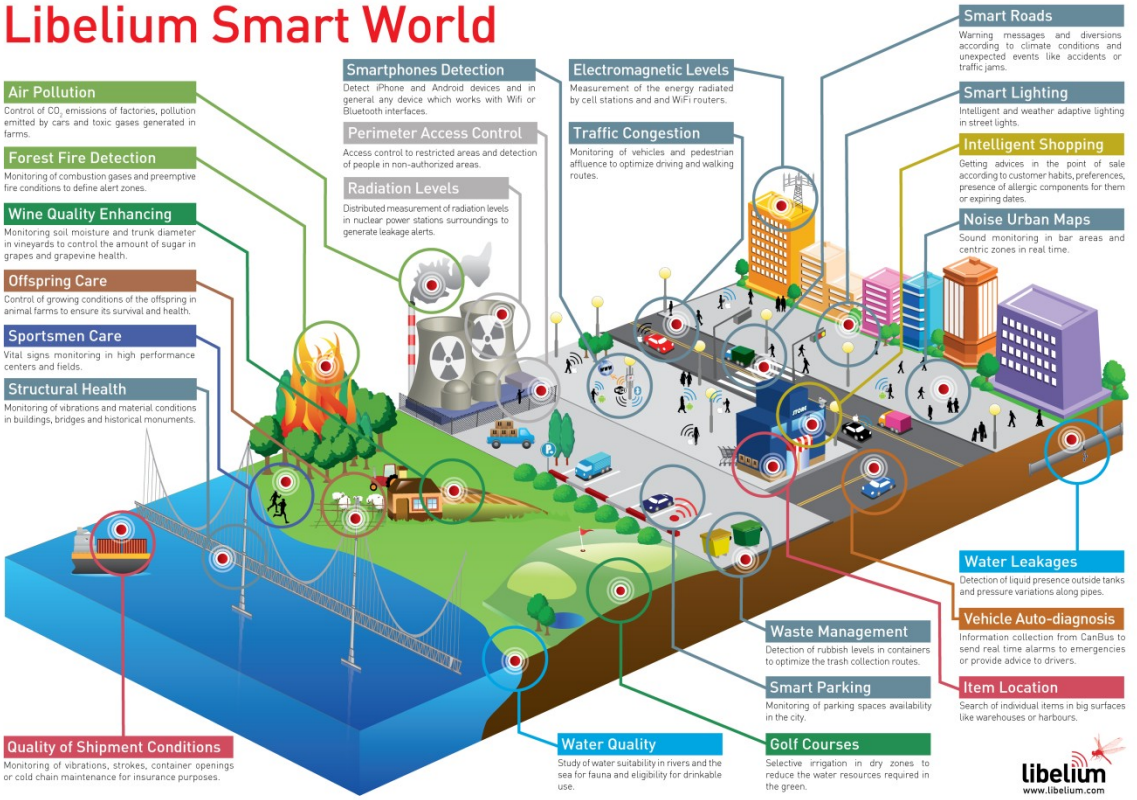


¿ Se venderán 9.715 millones en 2016 ?
2017 => 7.550 millones de $ a nivel mundial

Tipos de sistemas operativos

- Redes de sensores inalámbricos





TinyOS

tasks

Contiki

protothreads

The Open Source OS for the Internet of Things

Conceptos de sistemas operativos "UNIX": Proceso

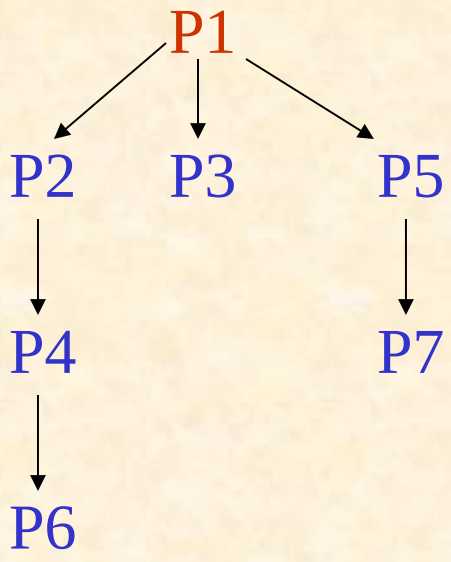
¿Qué es un proceso? ⇒ Programa en ejecución

¿Qué lo caracteriza? ⇒ Contexto del proceso

- Vector de estado → Evoluciona
  - Registros (PC, SP, ...)
  - Memoria (Estática, Dinámica)
  - Archivos en uso
- Código
- Identificador

→ Fijo

¿Número fijo de procesos o variable?



¿Quién es P1?

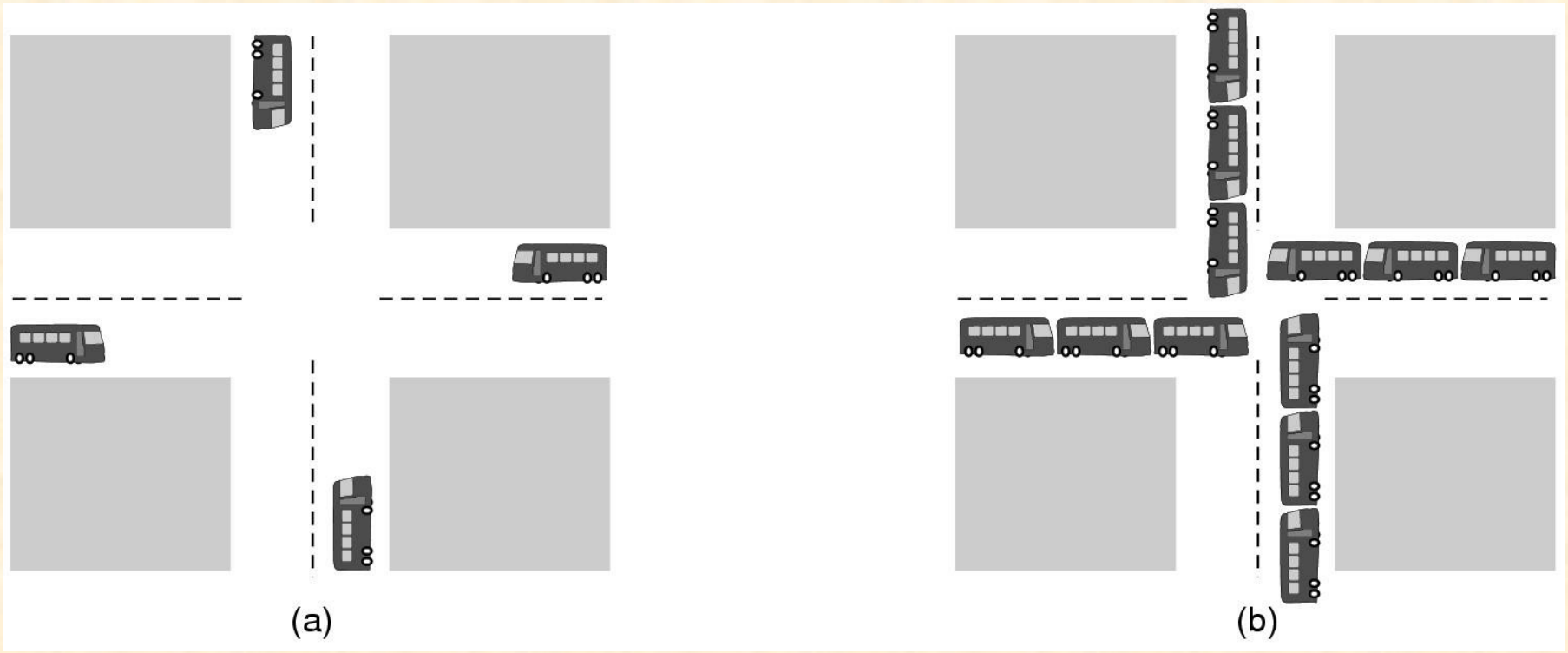
El Padre de todos (primer proceso del sistema). En sistemas tipo Unix, es el proceso init.

Conceptos …: Proceso: operaciones y algunos atributos

- Operaciones
  - Crear un proceso (relación padre-hijo)
  - Terminar un proceso
  - Comunicación y sincronización entre procesos
  - Esperar a que termine uno de los hijos
  - Establecer, tratar y enviar señales

- Un proceso tiene varios identificadores
  - *pid*: el propio proceso
  - *uid*: el usuario que lanzó el proceso ¿No existe?
  - *gid*: el grupo al que pertenece el *uid*

Conceptos …: Interbloqueo entre procesos



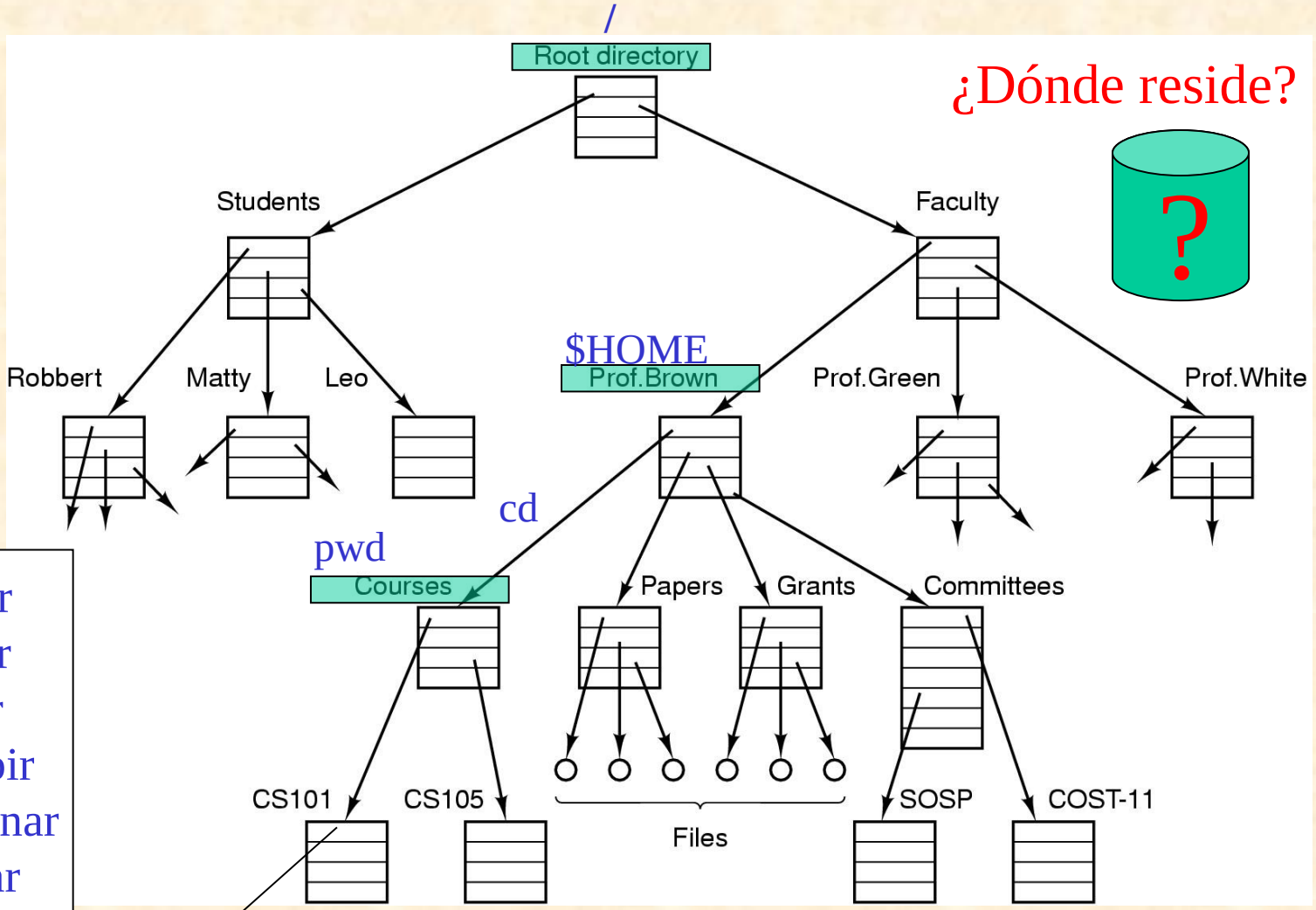
Se producen por la competencia entre procesos por conseguir los recursos del sistema

Conceptos …: Gestión de memoria y E/S

- Gestión de memoria
  - Asignación y liberación de memoria
  - Mecanismos de protección
  - Gestión de huecos y compactación
  - Gestión de la memoria virtual (paginación, segmentación)

- Gestión de dispositivos de E/S
  - Teclados, monitores, impresoras, discos, ...
  - Software de E/S independiente del dispositivo (protección, buffering, esquema de nombrado, tamaño de bloque, ...)
  - Software de entrada salida dependiente del dispositivo (drivers)

Conceptos …: Sistema de ficheros: estructura jerárquica



Crear
Abrir
Leer
Escribir
Posicionar
Cerrar
Borrar

/Faculty/Prof.Brown/Courses/CS101/notas.txt

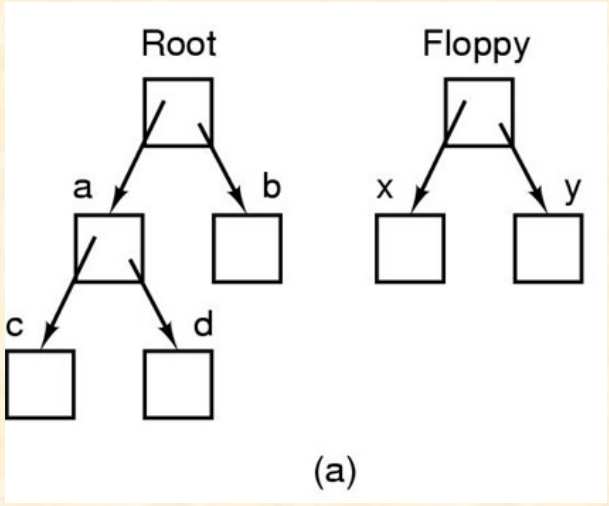
CS101/notas.txt

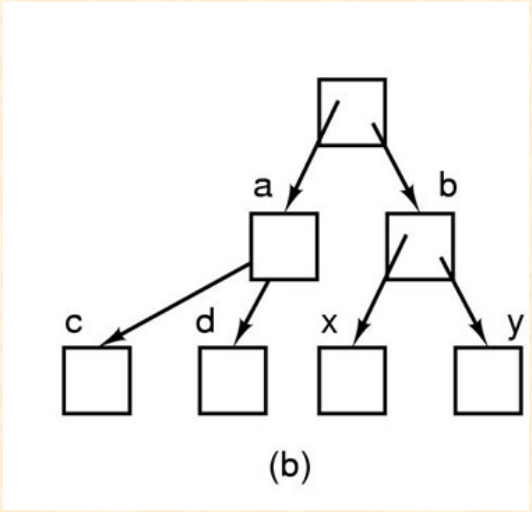
Conceptos …: Sistema de ficheros: ¿varios dispositivos?

Sistema de ficheros montado

¿Windows? Unidades A: C: D:

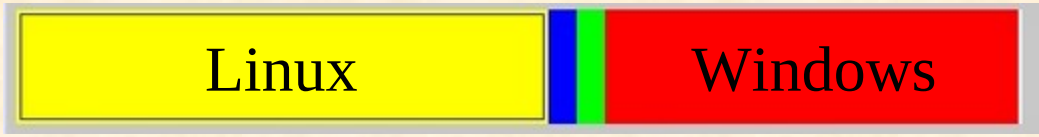






¿Varios S.F. en un mismo dispositivo?

Particiones



Antes de montar

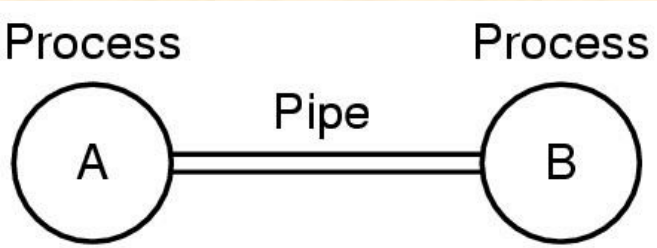
- Los ficheros en floppy son inaccesibles

Después de montar el floppy en b,

- Los ficheros en floppy son parte del sistema de ficheros

Conceptos …: Sistema de ficheros: tipos de ficheros

- Ficheros ordinarios
- Ficheros especiales
  - Los dispositivos de E/S se ven como ficheros.
    - Especial de bloques (por ejemplo, discos, /dev/hd01)
    - Especial de caracteres (por ejemplo, impresora /dev/lpt)
- Pipes
  - Un pseudofichero
  - Dos descriptores: para leer y para escribir
  - Comunican procesos



Conceptos …: Intérprete de comandos

Shell

- Intérprete de comandos
- No es parte del sistema operativo
- sh, csh, ksh, bash
- Crea procesos para ejecutar comandos
- Gestiona las redirecciones (entrada/salida) y pipes

```
cat file1 file2 file3 | sort > /dev/lp
```

- Ejecución en background

```
cat file1 file2 file3 | sort > /dev/lp &
```

Seguridad

- Bits rwx de los ficheros (propietario, grupo, otros)

```
rwx r-- ---
```



Conceptos …: Intérprete de comandos

$ ps –ef

```
root      2708     1  0 Sep07 ?        00:00:00 /usr/sbin/atd
root      2730     1  0 Sep07 ?        00:00:00 /sbin/dhcdbd --system
root      2742     1  0 Sep07 ?        00:00:00 libvirt_qemud --system --daemon
nobody    2879  2742  0 Sep07 ?        00:00:01 dnsmasq --keep-in-foreground --strict
root      2880     1  0 Sep07 ?        00:01:39 /usr/bin/python /usr/sbin/yum-update
avahi     2892     1  0 Sep07 ?        00:00:07 avahi-daemon: running [pc1.local]
avahi     2894  2892  0 Sep07 ?        00:00:00 avahi-daemon: chroot helper
68        2906     1  0 Sep07 ?        00:00:00 hald
root      2907  2906  0 Sep07 ?        00:00:00 hald-runner
68        2918  2907  0 Sep07 ?        00:00:00 hald-addon-keyboard: listening on /de
68        2919  2907  0 Sep07 ?        00:00:00 hald-addon-keyboard: listening on /de
68        2920  2907  0 Sep07 ?        00:00:00 hald-addon-keyboard: listening on /de
root      2935  2907  0 Sep07 ?        00:00:00 /usr/libexec/hald-addon-cpufreq
68        2936  2907  0 Sep07 ?        00:00:00 hald-addon-acpi: listening on acpi ke
root      2962  2907  0 Sep07 ?        00:00:01 hald-addon-storage: polling /dev/scd
root      3036     1  0 Sep07 ?        00:00:00 /usr/sbin/smartd -q never
root      3046     1  0 Sep07 tty2     00:00:00 /sbin/mingetty tty2
root      3047     1  0 Sep07 tty3     00:00:00 /sbin/mingetty tty3
root      3048     1  0 Sep07 tty4     00:00:00 /sbin/mingetty tty4
root      3049     1  0 Sep07 tty5     00:00:00 /sbin/mingetty tty5
root      3050     1  0 Sep07 tty6     00:00:00 /sbin/mingetty tty6
root      3051     1  0 Sep07 ?        00:00:00 /usr/sbin/gdm-binary -nodaemon
root      3128  3051  0 Sep07 ?        00:00:00 /usr/sbin/gdm-binary -nodaemon
root      3132     1  0 Sep07 ?        00:00:00 /usr/sbin/gdm-binary -nodaemon
root      3133  3128  0 Sep07 tty7     00:00:45 /usr/bin/Xorg :0 -br -audit 0 -auth /
gdm       3147  3128  0 Sep07 ?        00:00:00 /usr/libexec/gdmgreeter --gtk-module=
gdm       3150     1  0 Sep07 ?        00:00:00 /usr/libexec/at-spi-registryd
gdm       3152     1  0 Sep07 ?        00:00:00 /usr/libexec/bonobo-activation-serve
root      3211     1  0 Sep07 tty1     00:00:00 /sbin/mingetty tty1
root      4619  2535  0 10:03 ?        00:00:00 sshd: pcarazo [priv]
pcarazo   4622  4619  0 10:03 ?        00:00:00 sshd: pcarazo@pts/0
pcarazo   4624  4622  0 10:03 pts/0    00:00:00 -bash
pcarazo   4917  4624  0 10:12 pts/0    00:00:00 ps -ef
[pcarazo@pc1 ~]$
```

Conceptos …: Intérprete de comandos

$ ps –ef | cut –d “ “ –f 1

```
root
root
root
nobody
root
avahi
avahi
68
root
68
68
68
root
68
root
root
root
root
root
root
root
root
root
root
root
gdm
gdm
gdm
root
root
pcarazo
pcarazo
pcarazo
pcarazo
[pcarazo@pc1 ~]$
```

$ ps –ef | cut –d “ “ –f 1 | grep root

```
root
root
root
root
root
root
root
root
root
root
root
root
root
root
root
root
root
root
root
root
root
root
root
root
root
root
root
root
root
root
root
root
root
root
[pcarazo@pc1 ~]$
```

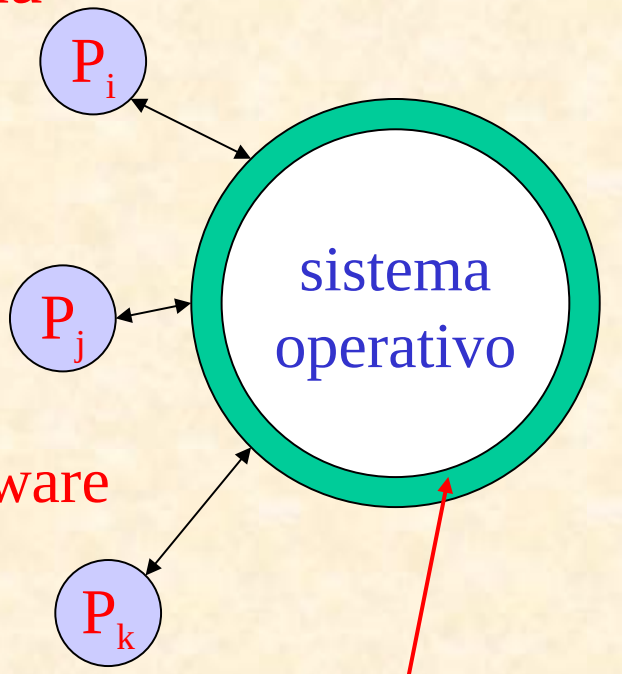
```
[pcarazo@pc1 ~]$ ps -ef | cut -d " " -f 1 | grep root | wc
    106     106     530
[pcarazo@pc1 ~]$
```

Llamadas al sistema

Gestión Bancaria	Reserva de Billetes	Juegos
Compila dores	Editores	Montador
Llamadas al sistema S.O.	Intérprete de comandos	
S E R V I C I O S		
Lenguaje máquina		
MicroArquitectura		
Dispositivos físicos		

Software

Hardware



¿POSIX

WIN32?

Llamadas (genéricas) al sistema operativo

- Control de Procesos:
  - Cargar y ejecutar programas
  - Crear y terminar procesos
  - Leer y poner atributos
  - Esperar un tiempo
  - Esperar un evento
- Gestión de Ficheros:
  - Crear, Borrar
  - Abrir, Cerrar
  - Posicionarse
  - Leer, Escribir
  - Leer y poner atributos
- Manejo de Dispositivos:
  - Solicitar y liberar dispositivo
  - Posicionarse
  - Leer y escribir
  - Leer y establecer atributos
  - Bloquear
- Información y mantenimiento:
  - Leer y poner fecha y hora
  - Pedir estado del sistema
  - Leer y poner atributos de procesos, ficheros o dispositivos

Llamadas (concretas) al sistema operativo (POSIX)

Distinguir: SERVICIOS ↔ INTERFAZ ↔ IMPLEMENTACIÓN

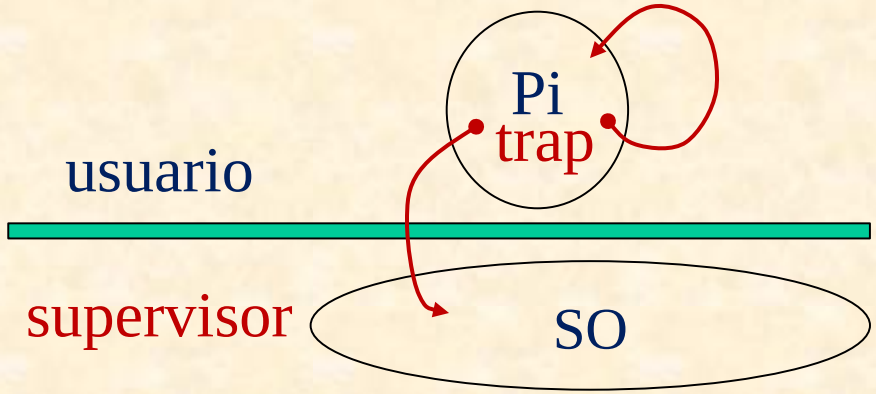
Algunas llamadas de Gestión de Procesos

Fork ⇒ Permite a un proceso (padre) crear un proceso (hijo)

Getpid ⇒ Permite a un proceso conocer su identificación PID

INTERFAZ ¿CÓMO INVOCARLAS? ¡No siempre!

- Llamada a bajo nivel ⇒ Ensamblador (TRAP)



Llamadas (concretas) al sistema operativo (POSIX)

Distinguir: SERVICIOS ↔ INTERFAZ ↔ IMPLEMENTACIÓN

Algunas llamadas de Gestión de Procesos

Fork ⇒ Permite a un proceso (padre) crear un proceso (hijo)

Getpid ⇒ Permite a un proceso conocer su identificación PID

INTERFAZ ¿CÓMO INVOCARLAS?

- Llamada a bajo nivel ⇒ Ensamblador (TRAP)
- Llamada a alto nivel:

C PIIPOO Ada

package unix is

class definition unix;

man fork | getpid (Bibliotecas de llamadas al sistema "libc")

man getpid

System Calls getpid(2)

NAME

getpid, getpgrp, getppid, getpgid – get process, process group, and parent process IDs

SYNOPSIS

```
#include <unistd.h>
int getpid ()
int getpgrp ()
.....
```

Otras llamadas relativas a procesos
pid = fork( ) pid = waitpid(pid, &statloc, options) s = execve(name, argv, environp) exit (status)

DESCRIPTION

The getpid() function returns the process ID of the calling process.

The getpgrp() function returns the process group ID of the calling process.

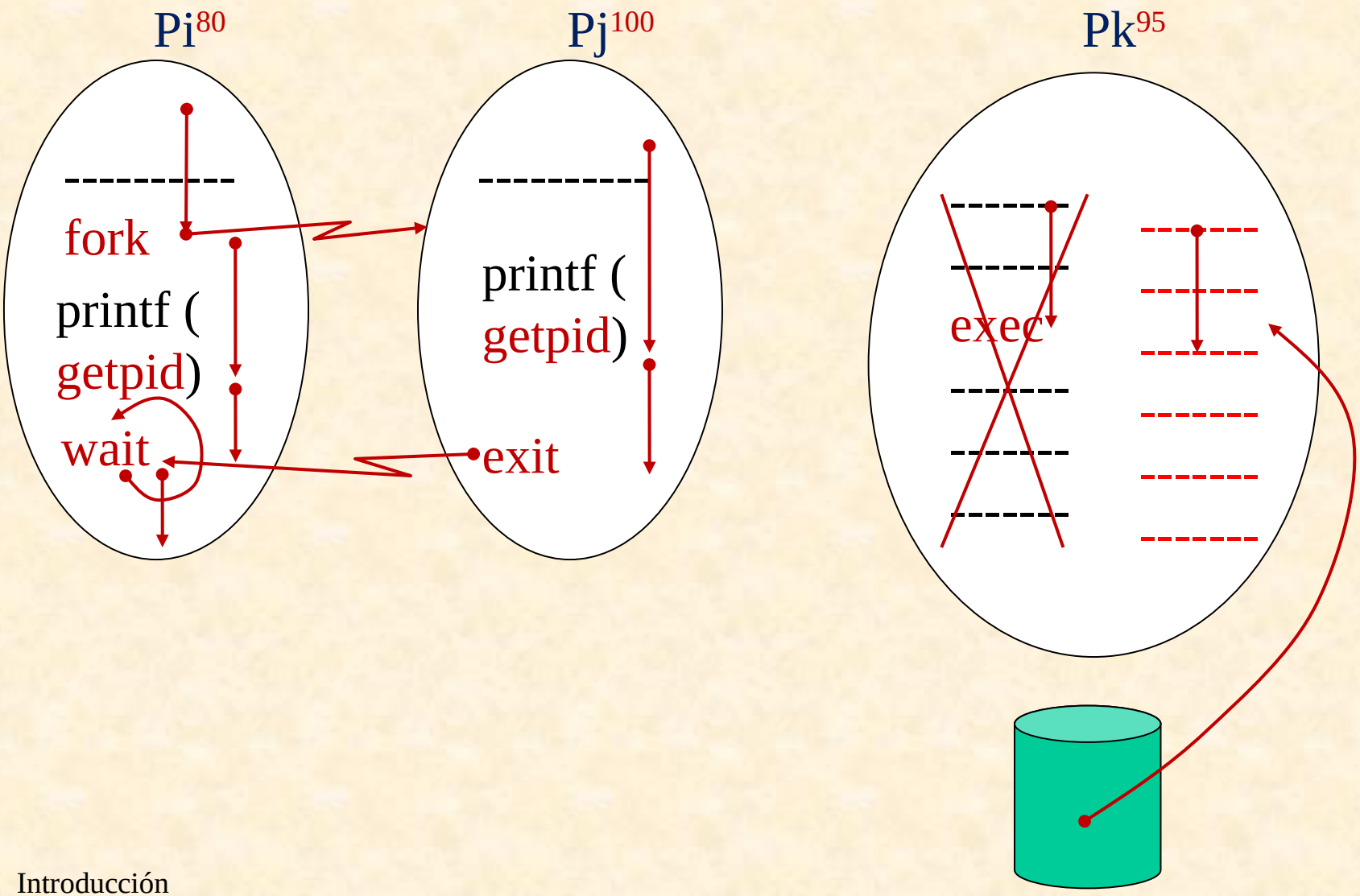
.....

SEE ALSO

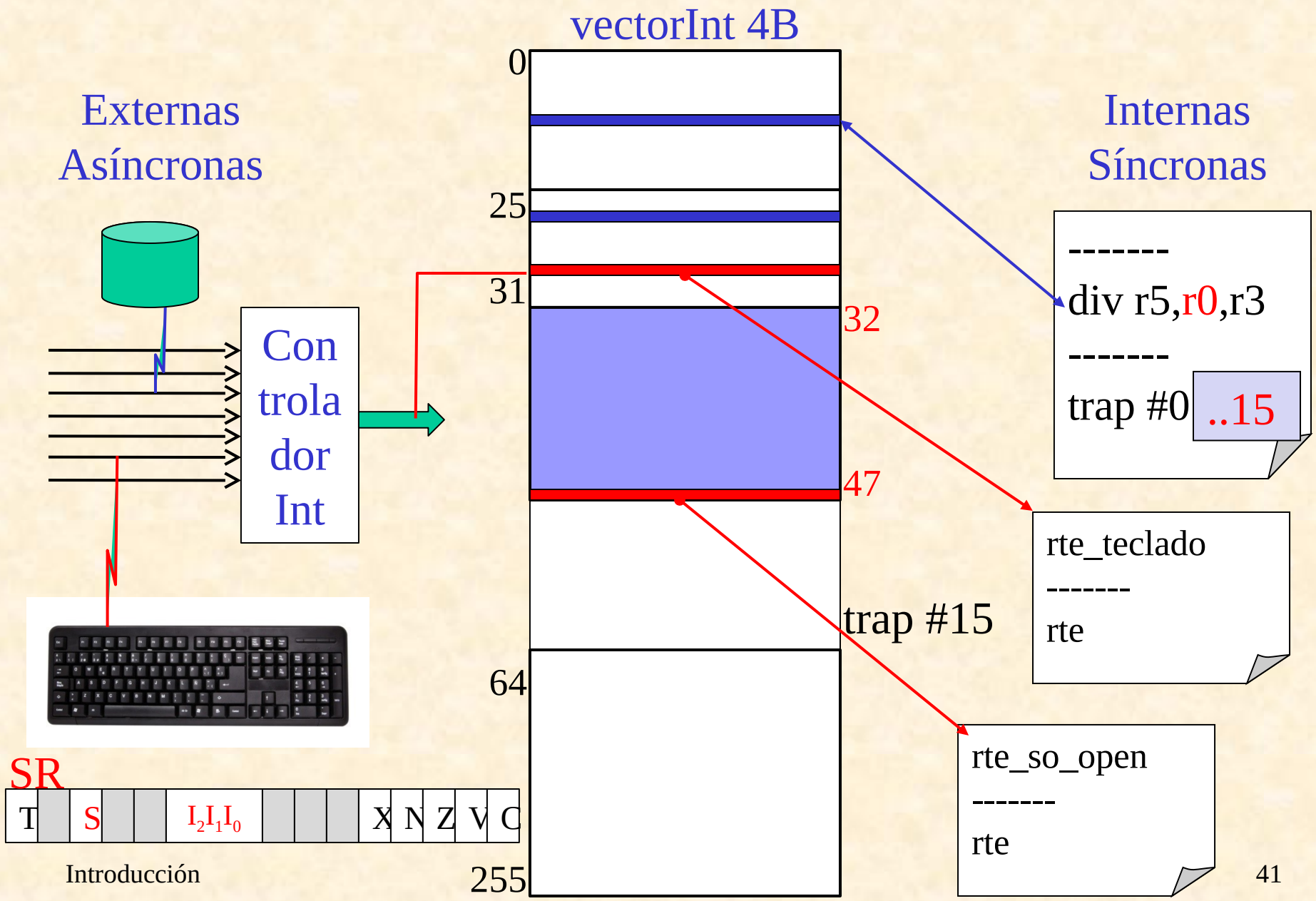
intro(3), exec(2), fork(2), getsid(2), setpgid(2), ..., signal(3C).

ejemplo de uso

System Calls: fork, getpid, wait, exit, exec, … ⇨ TRAP

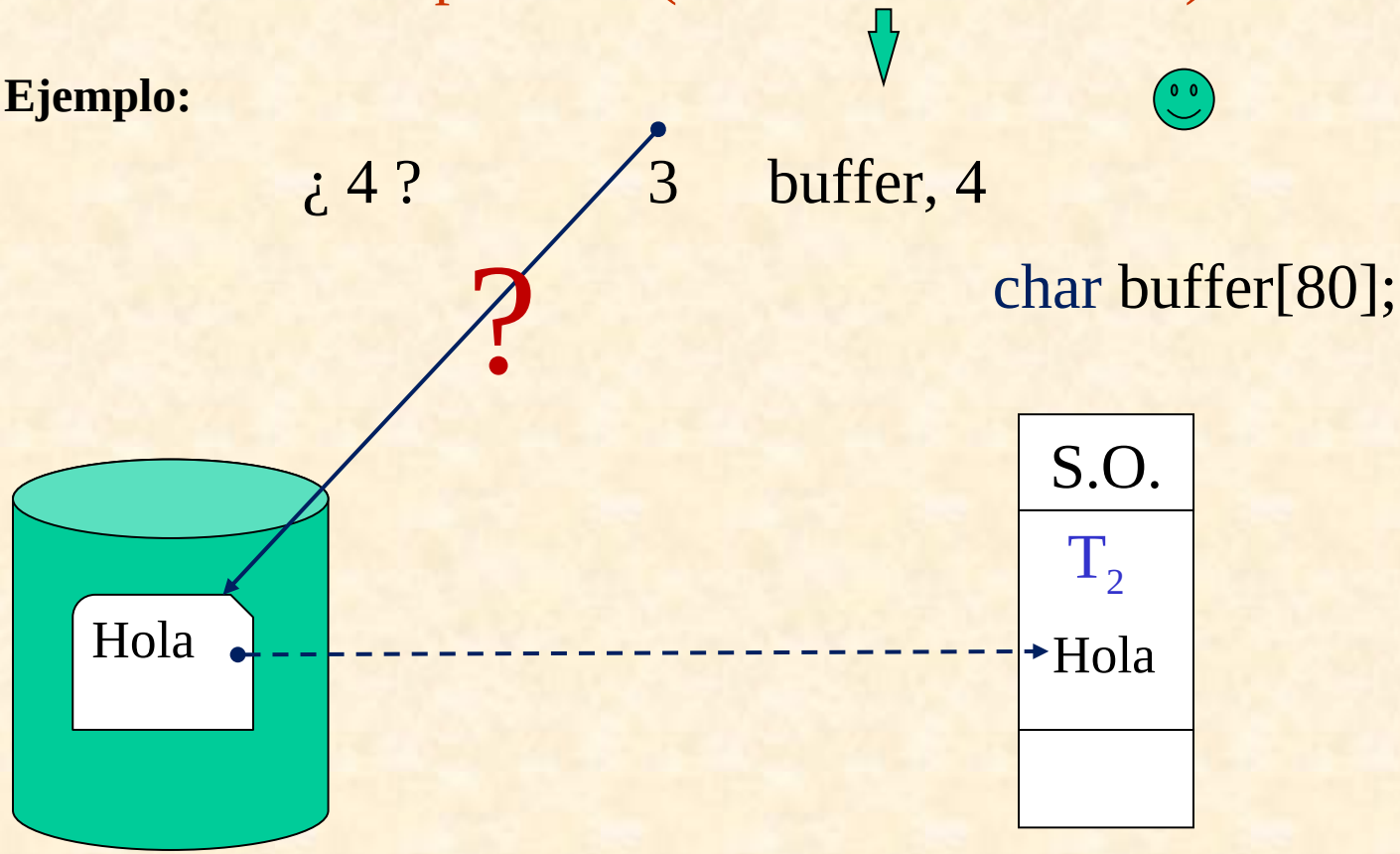


Recordatorio sobre interrupciones | excepciones



Llamadas al sistema operativo (“C” vs ensamblador)

Ejemplo:



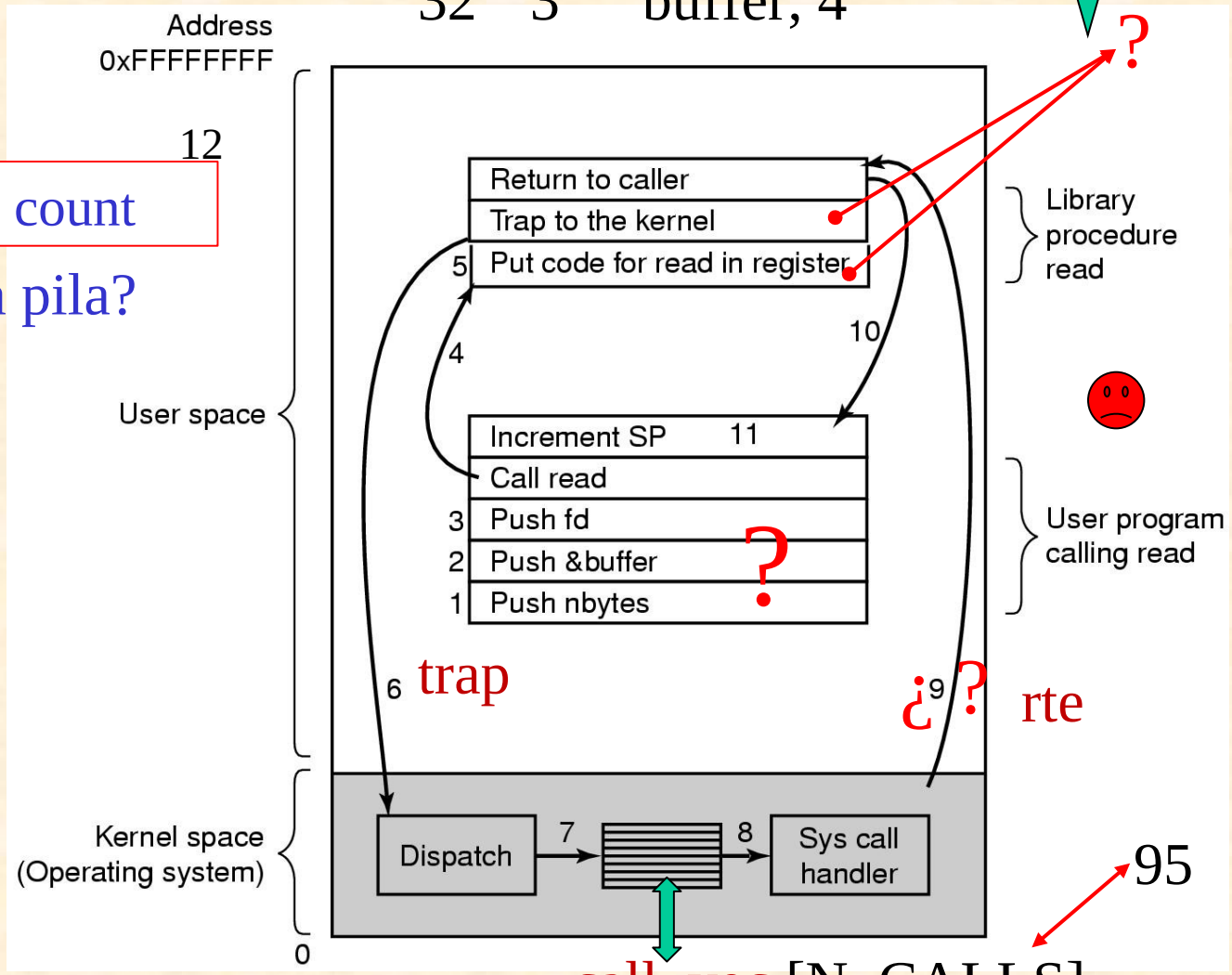
fd = open (“hola.txt”, …)

0 ➔ Entrada estándar
1 ➔ Salida estándar
2 ➔ Salida estándar de errores

Llamadas al sistema operativo ("C" vs ensamblador)

Ejemplo: ¿ ?

32 3 buffer, 4



12

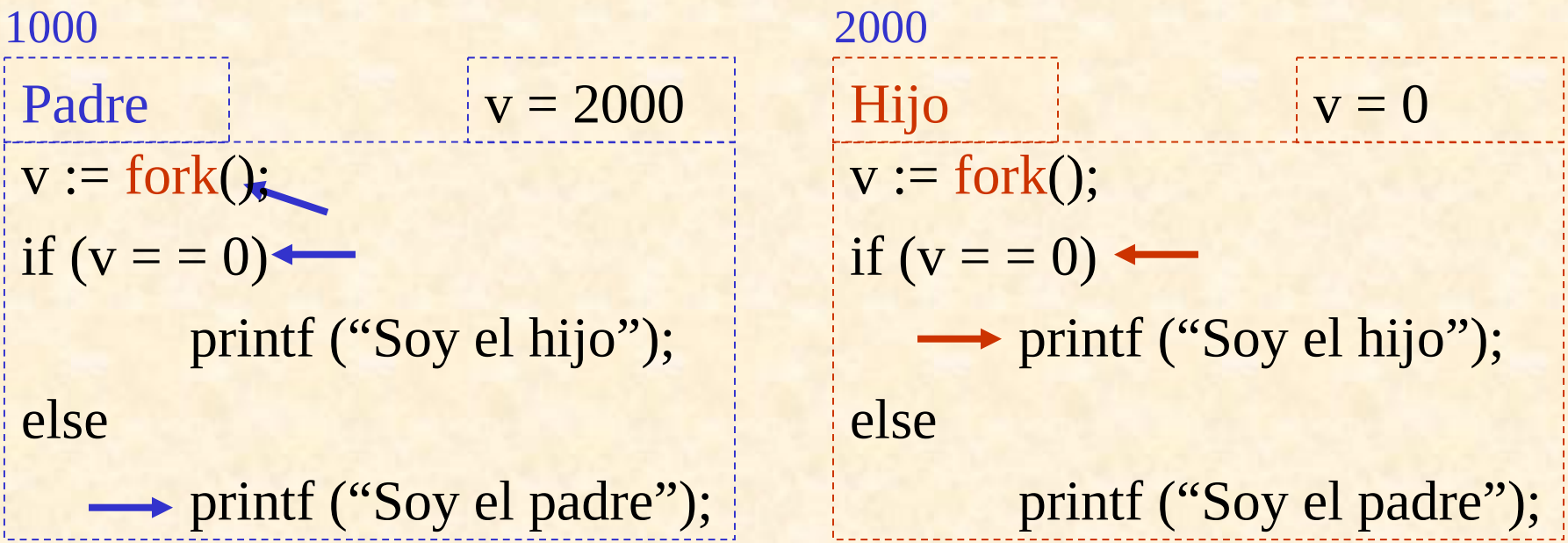
move Ri, count

¿En la pila?

SR
PC'
PC
3
$F3A1
4

Veamos un ejemplo de utilización de: fork y getpid

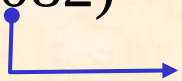
int fork () ⇒ Crea un proceso hijo. Al padre le devuelve el PID del hijo creado y al hijo le devuelve un 0



Veamos un ejemplo completo (padreHijo):

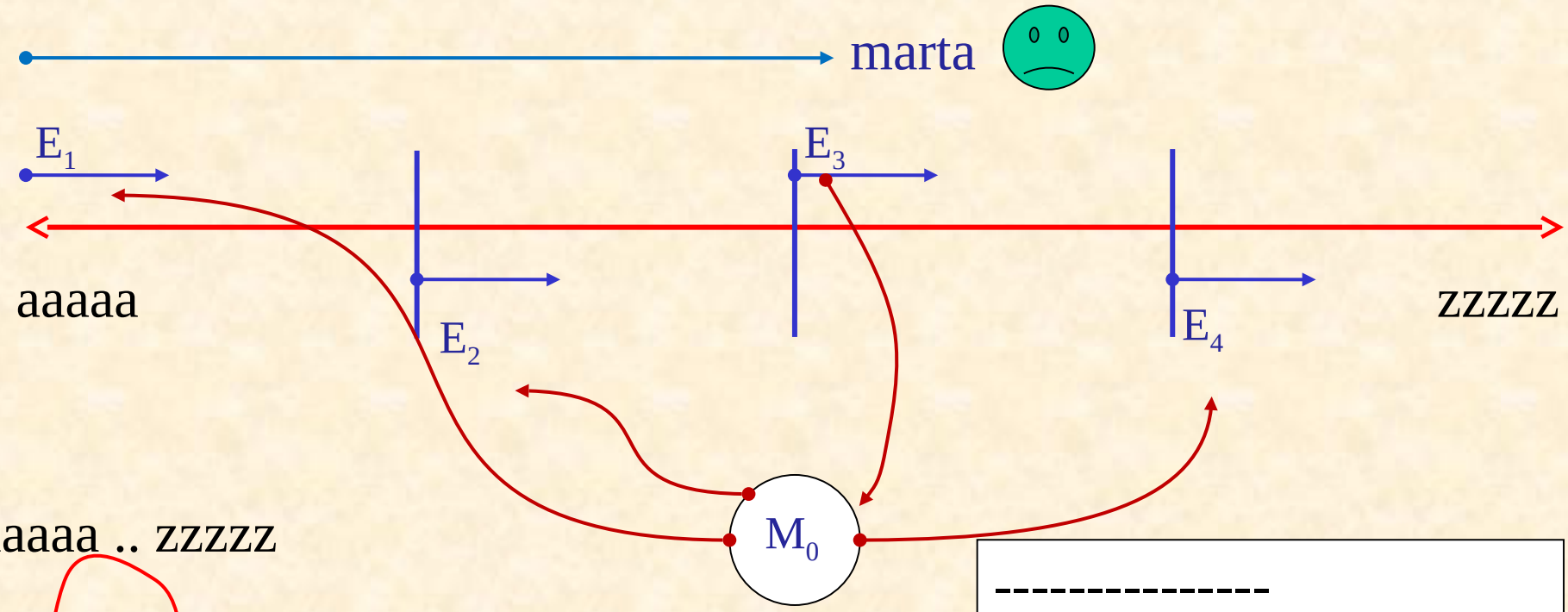
```
#include <stdio.h>
main () {
    int v;
    v = fork();
    if (v = = -1) /* No se pudo crear el proceso */
        printf (“Error al internar crear el proceso\n”);
    else if (v = = 0)      /* Proceso hijo */
        printf (“Hijo (%d)\n”, getpid());
    else                   /* Proceso padre */
        printf (“Padre (%d) e Hijo (%d) \n”, getpid(), v);
}

/* Ejemplo de salida del programa anterior */
Padre (26081) e Hijo (26082)
Hijo (26082)
```



¿Seguro? ¿Más Hijos?

• Descifrar una clave basada en "*crypt*" *¿ 4 núcleos ?*



aaaaa .. zzzzz

```
if (crypt   (claveCruda, "aa")
          == claveCifrada)
    encontrada
else  siguiente (claveCruda)
```

```
-------------
for (i=0; i<4; i++) {
    pidHijo = fork();
    if (pidHijo == 0)
          esclavo (i+1);
}
-------------
```

```
crypt ("abrir", "aa") => "aaHUVtmrCqHAw"
```

Mi primer ejecutor de comandos

leer (comando)
ejecutar (comando)
?

%echo hola
%sleep 5

%ejecutor
>padreHijo

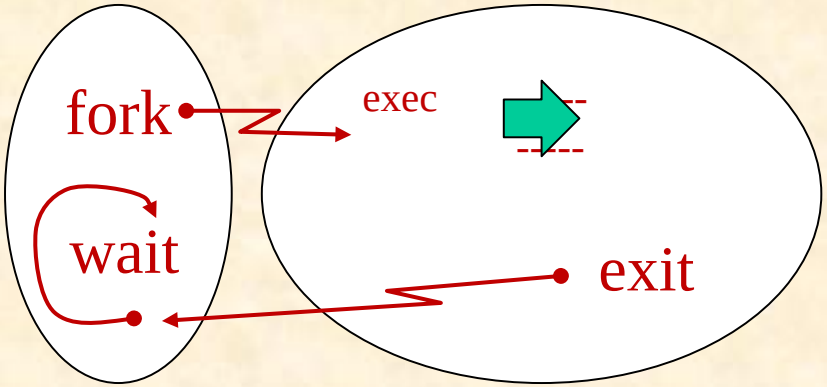
>ls

>0
% -1 ➔ Cualquiera

leer (comando)
execve (comando)
?

Padre mutado

Ejecutable del comando



pid = waitpid(*pid*, &statloc, *options*)
s = execve(name, argv, environp)
exit (status)

Ejecutable (comando)

Mi primer ejecutor de comandos

```
#include <stdio.h>
main ( ) {
    int pid, estado; char programa[20];
    printf (“>”);
    scanf(“%s”, programa);
    while (programa[0] != ‘0’) {
        if (fork( ) = = 0) {
            estado = execl (programa, 0);
            printf (“Error en %s => %i\n”,
                                programa, estado);
            exit (1);
        } else {
            pid = wait (&estado);
            printf (“>”);
            scanf (“%s”, programa);
}   }   }
```

```
%ejecutor
>padreHijo
------------
>ls
------------
>0
%
```

```
pid = waitpid(pid, &statloc, options)
s = execve(name, argv, environp)
exit (status)
```

Llamadas al sistema (Ficheros)

¿Posición? ⇒ 0

¿No existe? ⇒ ¡Se crea!

int open(nombre, modo) ⇒ Descriptor de Fichero

read (df, ...), write (df, ...), lseek (df, ...)

offset

where

SEEK_SET
SEEK_CUR
SEEK_END

fstat (df, ...), stat (nombre)

close (df)

ls -l

usuario

otros

-rw-r----- 1 pcarazo 2062 14321 May 20 13:52 practica1.c

grupo

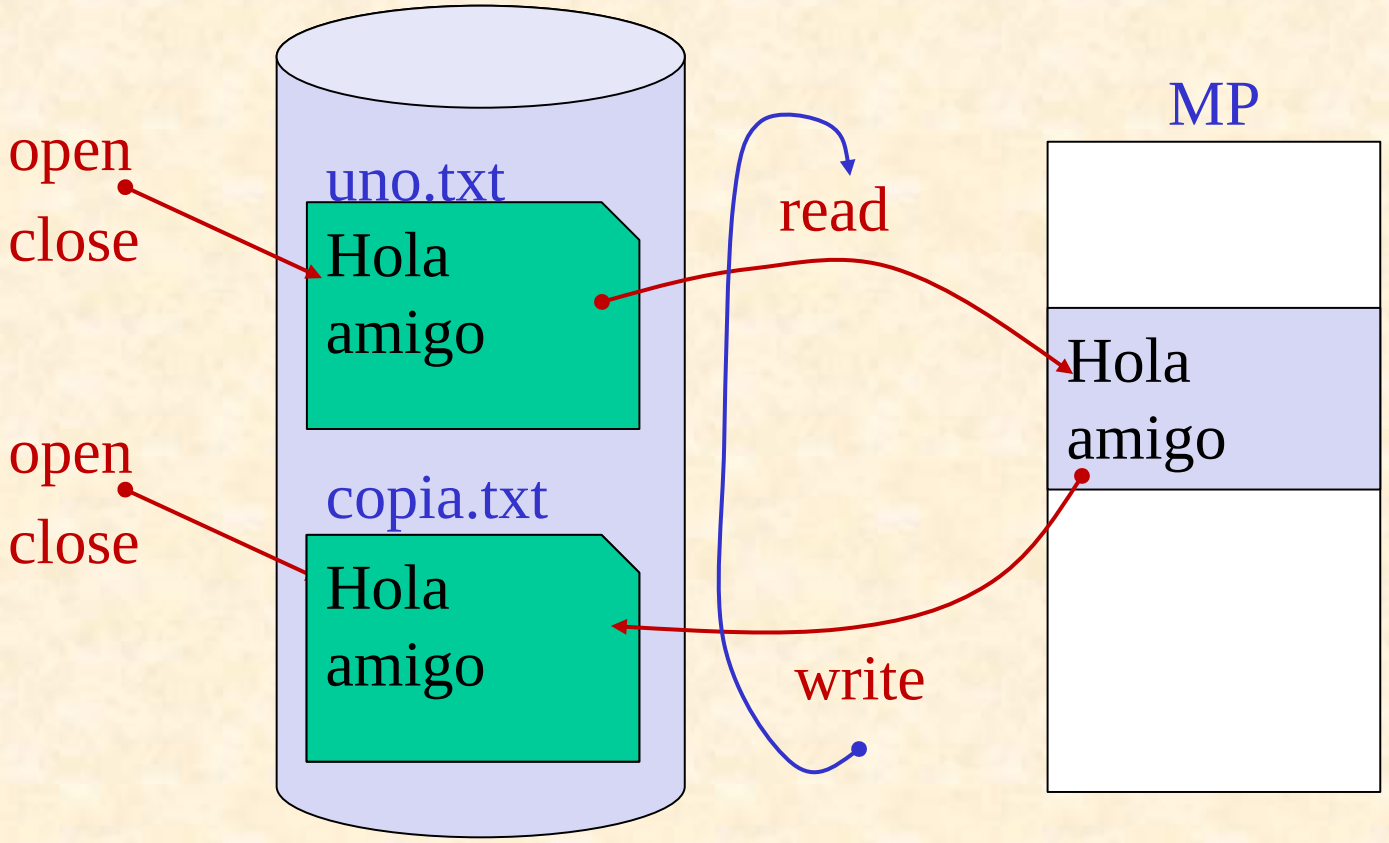
Descripción

Propietario, Protecciones, Tiempos, Tamaño, Tipo, etc.

¿ Sencillo ?

Llamadas al sistema (Ficheros: ejemplo de uso)

cp /usr/pepe/uno.txt /tmp/copia.txt



Llamadas al sistema (Ficheros: ejemplo de uso)

cp /usr/pepe/uno.txt /tmp/copia.txt

```
dforg = open ("/usr/pepe/uno.txt", O_RDONLY, 0);
dfdst = open ("/tmp/copia.txt",
                O_WRONLY | O_CREAT | O_TRUNC, 0600);
do {
      leidos = read (dforg, &buf, BLOQUE);
      write (dfdst, &buf, leidos);
} while (leidos == BLOQUE);
close (dforg); close (dfdst);
```



110 000 000

¿4096?

Llamadas al sistema (Directorios)

hardLink
¡ No para directorios !

Directory and file system management

Call	Description
s = mkdir(name, mode)	Create a new directory
s = rmdir(name)	Remove an empty directory
s = link(name1, name2)	Create a new entry, name2, pointing to name1
s = unlink(name)	Remove a directory entry
s = mount(special, name, flag)	Mount a file system
s = umount(special)	Unmount a file system

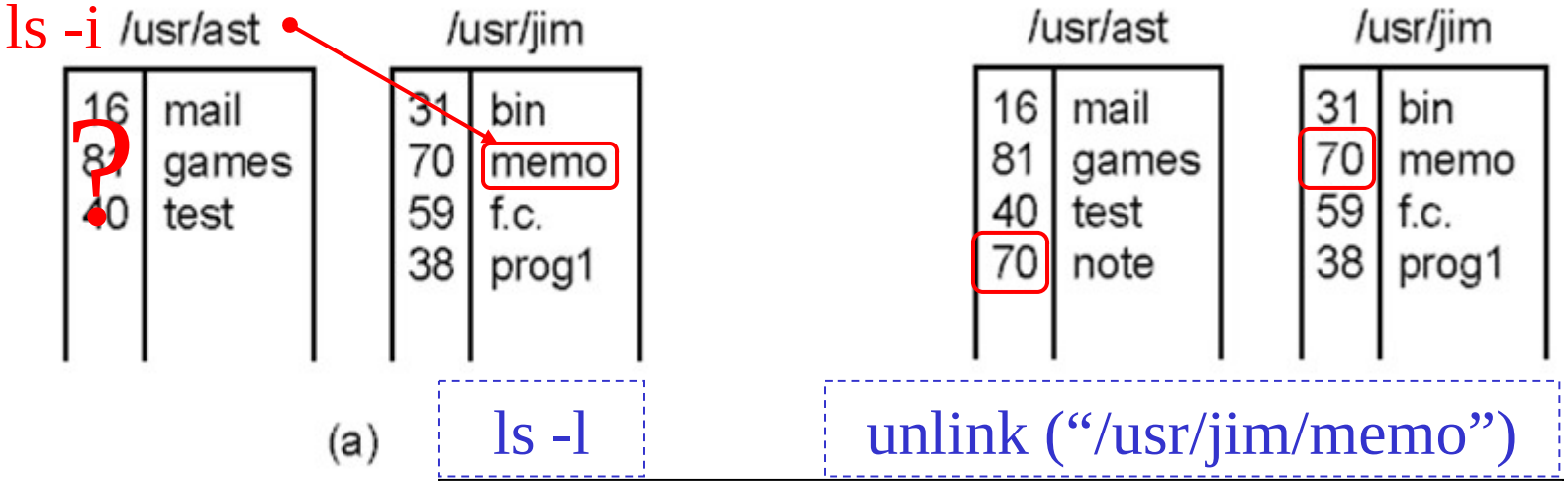
%rm –rf miDir

elPocero ?

?

?

link ("/usr/jim/memo", "/usr/ast/note")

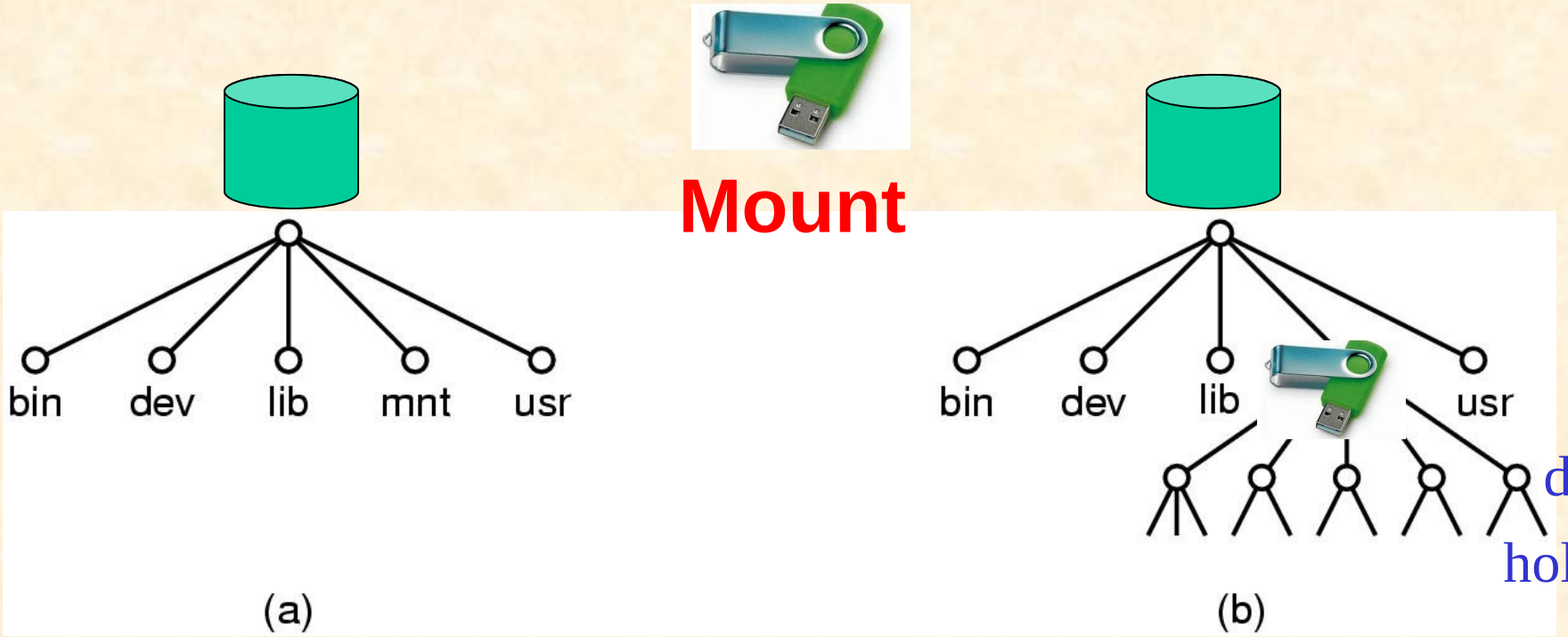


ls -l

unlink ("/usr/jim/memo")

70 -rw-r----- **2** *pcarazo* 2062 14321 May 20 13:52 note

Llamadas al sistema (Mount)



(a) Sistema de ficheros antes de mount (“/dev/sdb1”, “/mnt/”, 0)

(b) Sistema de ficheros después de la llamada al sistema

/etc/fstab

```
/dev/sda1 / ext4 rw 0 0
proc /proc proc rw 0 0
sysfs /sys sysfs rw 0 0
devpts /dev/pts devpts rw,gid=5,mode=620 0 0
tmpfs /dev/shm tmpfs rw,rootcontext="system_u:object_r:tmpfs_t:s0" 0 0
/dev/sda3 /usuarios ext4 rw 0 0
```

/etc/mtab

Llamadas al sistema (Otras)

Miscellaneous

Call	Description
s = chdir(dirname)	Change the working directory
s = chmod(name, mode)	Change a file's protection bits
s = kill(pid, signal)	Send a signal to a process
seconds = time(&seconds)	Get the elapsed time since Jan. 1, 1970

SIGUSR1

PRÁCTICA

int ➔ 32 bits ➡ 4.294.967.296 seg

Hace unos días ➔ 1.598.029.674

gettimeofday
struct timeval
tv_sec
tv_usec

Efecto 2106

¿Hasta cuándo?

Antes x86_64

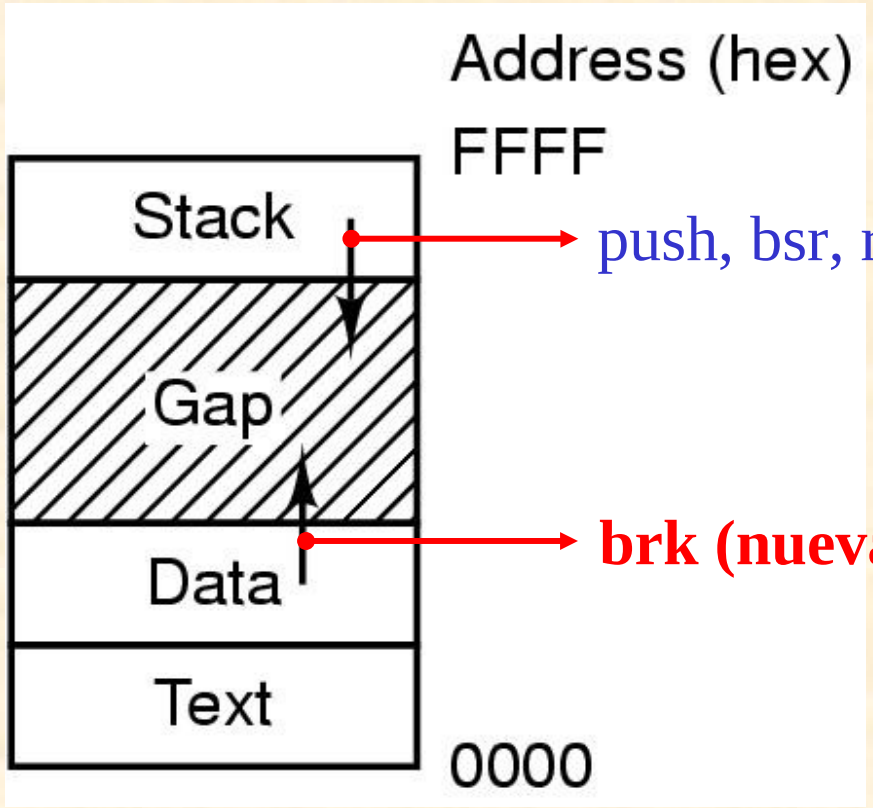


Llamadas al sistema (brk)

¿Estructura de un proceso en memoria?

Código } Fijo

Datos, Pila } Variable



push, bsr, rts

¡ No interviene el S.O. !

brk (nuevaDireccion)

Llamadas al sistema

¡¡¡ INDIVISIBLES !!!

- fork
- getpid
- wait
- exec
- exit

- open
- read
- write
- lseek
- fstat
- stat
- close

- mkdir
- rmdir
- link
- unlink
- ***mount***
- ***umount***

- chdir
- chmod
- kill
- time
- brk

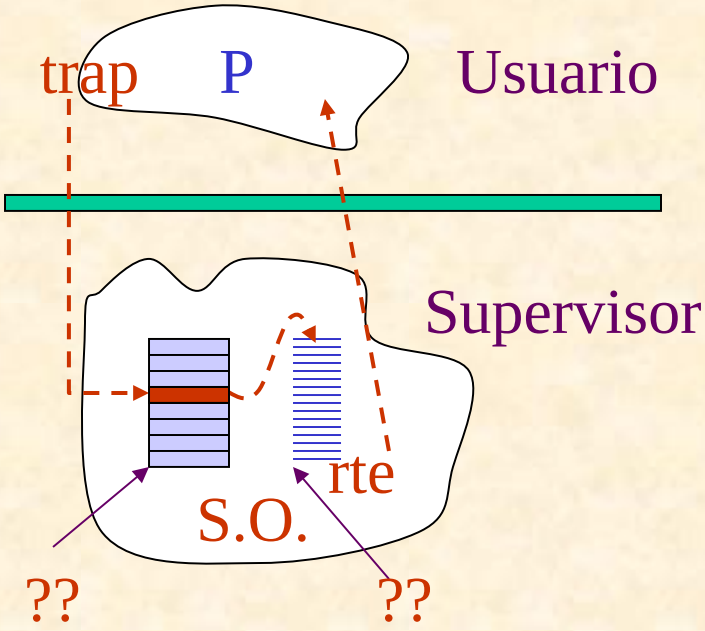
- pipe
- dup

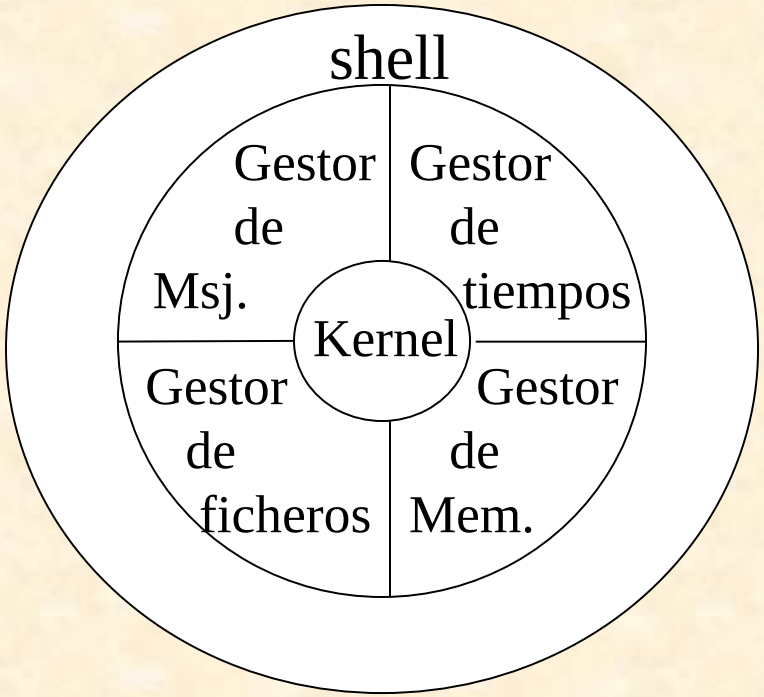
- sigaction
- alarm
- pause



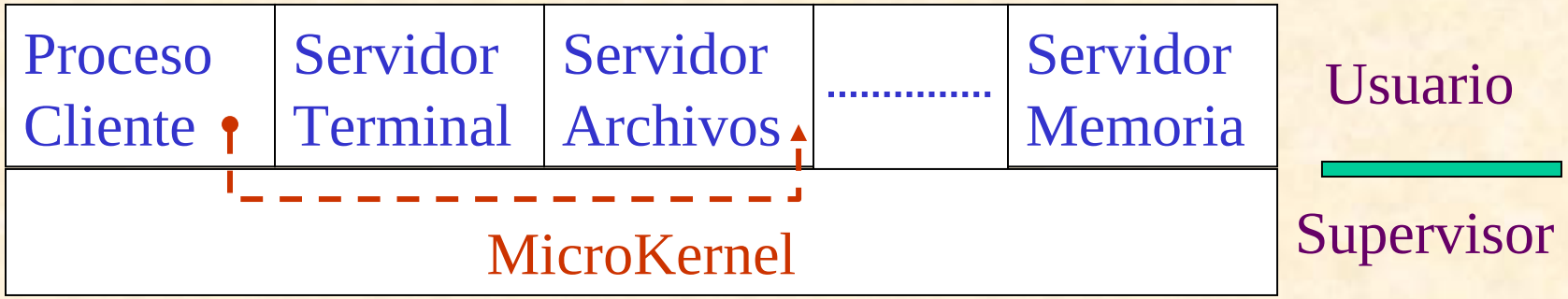
Estructura de los sistemas operativos

- Monolítico
- Con cierta estructura

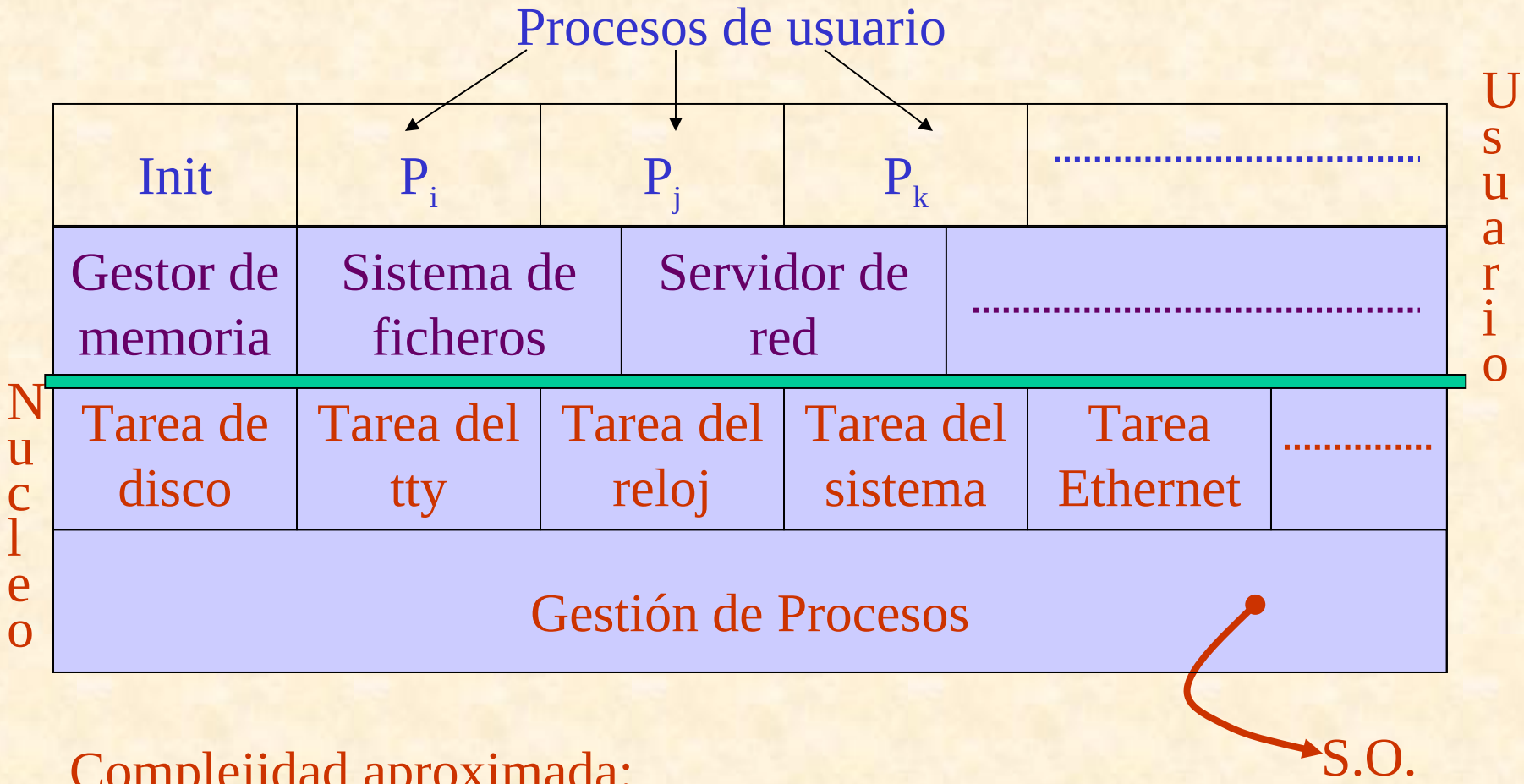




- Modelo Cliente/Servidor



Estructura interna de MINIX (por capas)



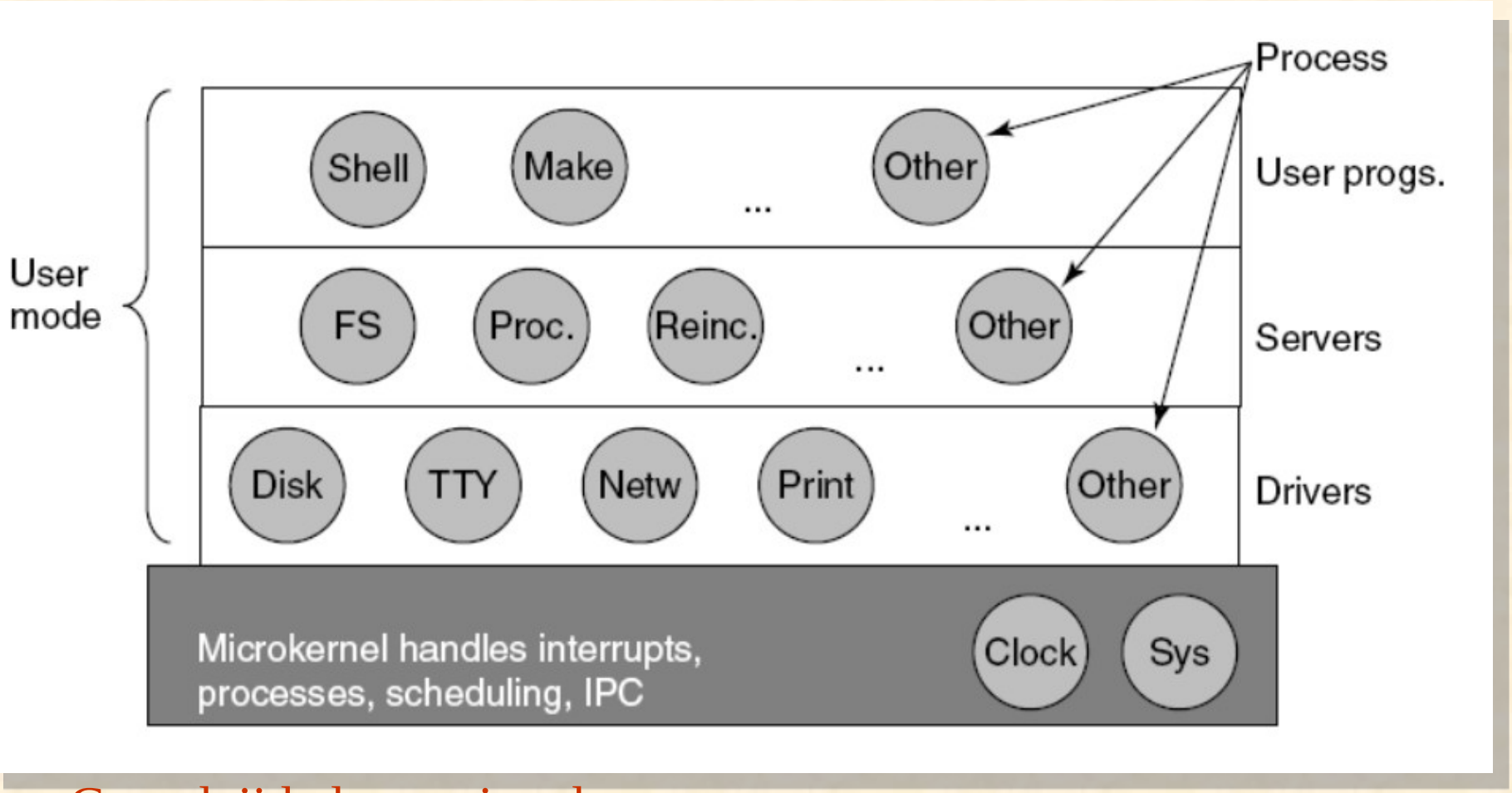
Complejidad aproximada:

Tareas ⇒ reloj (5); disco (12); tty (25)

Servidores ⇒ memoria (47); ficheros (102)

Counting Source Line of Code (SLOC) www.dwheeler.com/sloc

Estructura interna de MINIX 3



Complejidad aproximada:

- Microkernel ⇒ 3.200 líneas en C y 800 asm
- Drivers ⇒ disco (12); tty (30)
- Servidores ⇒ ficheros (111)

Volúmen de código del *kernel* de Linux

21 October 2008, 15:50 « previous | next »

Kernel Log: More than 10 million lines of Linux source files

www.h-online.com

2.6.27

```
SLOC    Directory
3301081 drivers

1258638 arch

544871  fs
376716  net
356180  sound
320078  include
74503   kernel
36312   mm
32729   crypto
25303   security
24111   scripts

17065   lib
10723   block
7616    Documentation
5227    ipc
2622    virt
2287    init
1803    firmware
833     samples
493     usr
0       top_dir
```

```
Totals grouped by language (dominant language first):
ansic:       6168175 (96.39%)
asm:          212699 (3.32%)
perl:           6672 (0.10%)
cpp:            3302 (0.05%)
yacc:           2954 (0.05%)
sh:             2715 (0.04%)
lex:            1820 (0.03%)
python:          424 (0.01%)
lisp:            218 (0.00%)
pascal:          116 (0.00%)
awk:              96 (0.00%)
```

Volúmen de código de distribuciones de S.O.

wikipedia

Year	Operating System	SLOC (Million)
1993	Windows NT 3.1	4-5[1]
1994	Windows NT 3.5	7-8[1]
1996	Windows NT 4.0	11-12[1]
2000	Windows 2000	more than 29[1]
2001	Windows XP	45[2]
2003	Windows Server 2003	50[1]
2015	Windows 10	80 ??

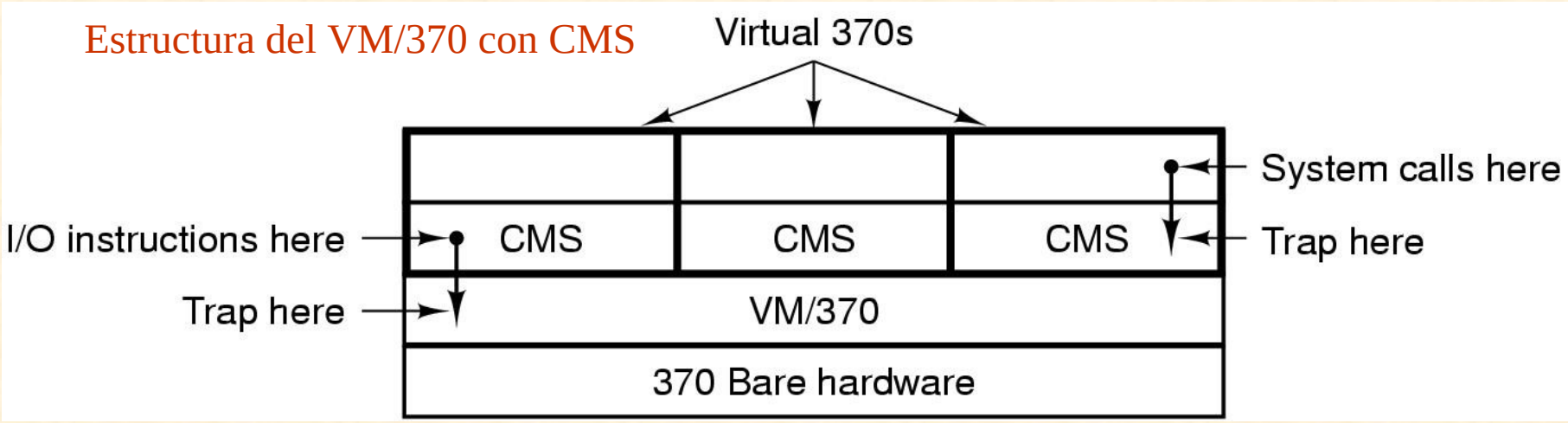
MSDOS
1981
4.000 asm

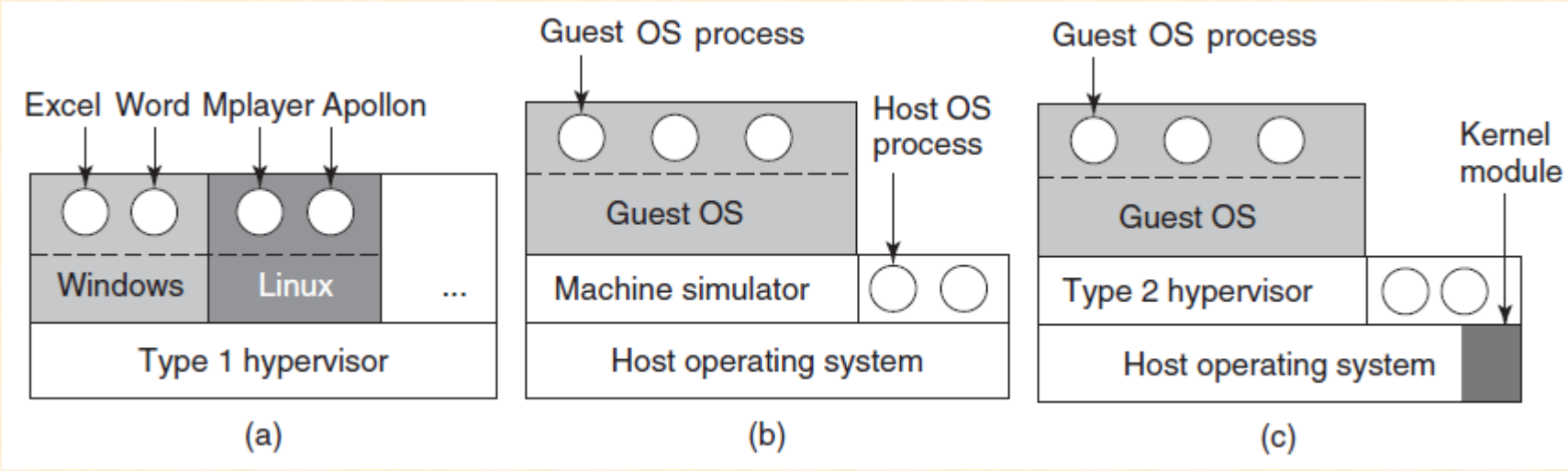
Año	Operating System	SLOC (Million)
2000	Debian 2.2	55-59[4][5]
2002	Debian 3.0	104[5]
2005	Debian 3.1	215[5]
2010	Debian 4.0	283[5]
2011	Debian 5.0	324[5]
2009	OpenSolaris	9.7
2011	FreeBSD	8.8
2004	Mac OS X 10.4	86[6][n 1]
2003	Linux kernel 2.6.0	5.2
2009	Linux kernel 2.6.29	11.0
2010	Linux kernel 2.6.32	12.6[7]
2011	Linux kernel 2.6.35	13.5[8]
2012	Linux kernel 3.6	15.9[9]
2015	Linux Kernel 4.2	>20[8]

MINIX 1 1987 11.800C + 800asm
MINIX 2 1997 62.200
MINIX 3 2004 *4.000* + ¿28.864? + …

Estructura de los sistemas operativos (Máquinas virtuales)

Estructura del VM/370 con CMS





Taller Diagramas de Flujo

Resuelva los siguientes ejercicios usando los diagramas de flujo y aplicando el proceso completo, análisis, construcción y prueba.

Ejercicio 1 – Algoritmo minutos en horas

Realice un algoritmo que determine cuantos minutos hay en 5 horas.

Ejercicio 2 – Algoritmo sumatoria enteros

Desarrolle un algoritmo que lea los primeros 300 números enteros y determine cuántos de ellos son impares; al final deberá indicar su sumatoria.

Ejercicio 3 – Algoritmo números naturales

Desarrolle un algoritmo que permita realizar la escritura de los primeros 100 números naturales utilizando la estructura Mientras (While)

Ejercicio 4 – Algoritmo Tiempo

Desarrolle un algoritmo que permita determinar a partir de un número de días, ingresado por pantalla, ¿Cuántos años, meses, semanas y días; constituyen el número de días proporcionado utilizando la estructura Mientras o While?

Calidad del software

Juan Manuel Fernández Peña

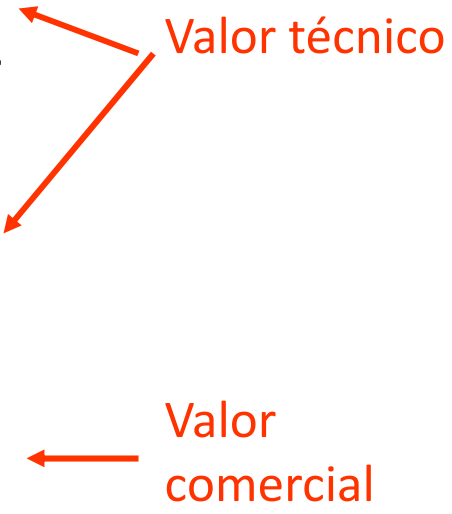
2011

Modelos de Calidad

Modelos de Calidad del Software

Tres tipos de modelos importantes:

- Calidad del producto: propiedades del producto según usuario y según desarrollador
- Calidad del proceso: actividades que influyen en calidad del producto
- Calidad en uso: relación del producto con el ambiente donde se le emplea



Modelos de calidad

- En este curso se tratan los basados en el proceso
- Son los dominantes en la industria

Calidad basada en el proceso

Se busca analizar las actividades del proceso que más influyen en la calidad del producto.

Se modela el proceso para analizarlo mejor.

Se pueden hacer preguntas como:

- ¿dónde y cuándo se puede hallar un tipo de defecto?
- ¿cómo hallar los defectos antes?
- ¿existen actividades alternas que proporcionen mayor calidad?

Algunos modelos basados en proceso

Modelos de madurez:

- CMM (Capability Madurity Model) y CMMI (CMM Integrated)
- ISO 15504 SPICE (Software Process Improvement and Capability dEtermination)
- ISO 9000
- NYSE NMX-I-059/02 (Moprosoft y Evalprosoft) Norma Mexicana

ISO 9000

Familia de estándares, Comité Técnico 176 de ISO

Estándar en más de 80 países.

ISO 9000-1 a ISO 9000-4 son relativas a Estándares de Admon. De Calidad y Aseguramiento de Calidad.

ISO 9000-3: Guía para aplicación de ISO 9001, Desarrollar, Proveer, Instalar y Mantener software para computadora.

ISO 9001: Sistemas de Calidad. Modelo para asegurar calidad en diseño, desarrollo, producción, instalación y servicios de software.

ISO 9000

Organizado en partes:

Requerimientos: Deben cumplirse necesariamente. Dice QUÉ debe hacerse, pero no CÓMO

Directrices:

- Recomendaciones: deberían cumplirse
- Sugerencias: Podrían cumplirse

ISO 9000

Conceptos de Calidad que tocan.

A) Una organización debe alcanzar y sostener la calidad de un producto o servicio de modo que satisfaga en forma continua las necesidades explícitas e implícitas del comprador.

B) Una organización debe brindar confianza en su propia administración de que la calidad intentada será alcanzada y sostenida.

C) Una organización debe proporcionar confianza al comprador de que la calidad propuesta ha sido o será alcanzada en el producto o servicio ofrecido. Si se requiere por contrato, debe haber demostración.

ISO 9000

Algunos aspectos de ISO 9001.

Requiere que la política de calidad sea definida, documentada, extendida, implementada y mantenida. Deben definirse responsabilidades y autoridad de todos los que participen en especificar, lograr y monitorear calidad.

Compra de productos: deben conformarse con requerimientos.

Material de proveedores: verificados, controlados y mantenidos.

Distribución y modificación de documentos, controladas.

Productos identificados y trazables durante su proceso de desarrollo, entrega e instalación.

Registrar estados e inspecciones.

Registros de calidad colectados, mantenidos y disponibles.

ISO 9000:2000

algunos cambios

- Enfoque en procesos
- Menos papeleo
- Alta dirección
- Comunicación interna y con el cliente
- Menos énfasis en producción de bienes
- Medición y seguimiento de información
- Mejora contínua
- Busca adaptarse a PYMES

ISO 9000:2000

Principios de gestión

- Enfoque en el cliente
  - cumplir y superar sus requerimientos
- Liderazgo
  - Crear ambiente adecuado en organización
- Participación del personal
- Enfoque basado en procesos
- Enfoque de sistemas a la gestión
- Mejora continua
- Basado en hechos para toma de decisiones
- Relaciones mutuamente beneficiosas con proveedores

ISO 9000:2000

Enfoque en procesos

- Modelo de proceso:
  - Planear -> Hacer -> Verificar -> Actuar

- Áreas de procesos:
  - Sistema de gestión de calidad
  - Responsabilidad de Alta Dirección
  - Gestión de recursos
  - Realización del producto
  - Medición, análisis y mejora

ISO 9000:2000

Pasos

- Evaluar necesidades y metas de organización
- Obtener información
- Nombrar consultor
- Toma de conciencia y formación
- Análisis de brecha
- Revisión o definición de procesos
- Suministrar personal
- Establecer cronograma
- Redactar manual de calidad
- Realizar auditoría
- Solicitar certificación
- Realizar evaluaciones

Familia de normas

- **ISO 9000:2000,** ***Quality management systems – Fundamentals and vocabulary (Sistemas de gestión de la calidad – Fundamentos y vocabulario)***
- **ISO 9001:2000,** ***Quality management systems - Requirements (Sistemas de gestión de la calidad – Requisitos)***
- **ISO 9004:2000,** ***Quality management systems – Guidelines for performance improvements (Sistemas de gestión de la calidad – Directrices para la mejora del desempeño)***
- **ISO/DIS 19011,** ***Guidelines on quality and/or environmental management systems auditing (Directrices sobre auditorías de sistemas de gestión de calidad y/o ambiental)***

Normas adicionales

- ISO 10006:1997, *Quality management-Guidelines to quality in project management (Gestión de la Calidad – Directrices para la calidad en gestión de proyectos)*
- ISO 10007:1995, *Quality management- Guidelines for Configuration Management (Gestión de la Calidad – Directrices para la Gestión de la Configuración)*
- ISO 10012-1:1992, *Quality assurance requirements for measuring equipment*
- ISO 10012-1:1992, *Quality assurance requirements for measuring equipment*
- ISO/TR 10014:1998, *Guidelines for managing the economics of quality (Directrices para la Gestión de la Economía de la Calidad).*
- ISO/TR 10017:1999, *Guidance on statistical techniques for ISO 9001:1994 (Guía sobre*
- *Técnicas Estadísticas para ISO 9001:1994).*

9001

- Determina procesos necesarios en la organización
- Determina secuencia e interacción de procesos
- Criterios y métodos necesarios para la operación y control de los procesos
- Asegurar disponibilidad de recursos e información para operación y seguimiento de procesos
- Realizar seguimiento, medición y análisis
- Implementar acciones para lograr objetivos y mejorar los procesos

9001

- Controlar la documentación, tener proceso de control
- Controlar registros asociados con los procesos
- Personal: asegurarse su competencia, capacitación, registro de sus características

ISO/IEC 90003:2004

Estándar específico para Software

- Reemplaza ISO 9000-3: 1997
- La parte de requerimientos es igual a la de ISO 9001:2000; las directrices son específicas.
- Se aplica a productos y servicios de software
- Productos en mercado o de soporte a organización
- Productos que forman parte de contratos con otras organizaciones
- Software embebido

IEC: International Electrotechnical Commission

ISO/IEC 90003:2004

Algunos aspectos específicos

Recursos

- Asegurarse de proveeer recursos de calidad: personal, ambiente, infraestructura
- Personal: asegurar su correcta
  - Experiencia
  - Formación
  - Entrenamiento
  - Habilidades
- Definir niveles de competencia
  - Definir requerimientos de entrenamiento

ISO/IEC 90003:2004

Algunos aspectos específicos

Calidad del producto

- Objetivos, requerimentos, riesgos
- Elegir un modelo de ciclo de vida adecuado
- Comunicación permanente con cliente; tener representante
- Definir entradas y salidas de procesos
- Verificación y validación, incluyendo revisiones y pruebas de unidad y sistema
- Administración de la configuración y seguimiento de cambios
- Control de adquisición de componentes
- Medición
- Control de producción del software
- Monitorear y preservar activos incluyendo componentes

ISO 15504: SPICE

ISO 15504

Software Process Improvement and Capacity Determination

Ante el aumento de métodos de estimar capacidad y de evaluar procesos, se necesita método más manejable a nivel de proceso, de proyecto. Además se requiere poder comparar evaluaciones.

Nace en Inglaterra, en defensa. Aprox. 1995

Se creó como vía alterna a proceso de estandarización.

Relación con el estándar ISO/IEC 12207.

ISO 15504

Software Process Assessment (SPA)

Examen disciplinado de los procesos usados por una organización frente a un conjunto de criterios para determinar la capacidad de tales procesos de realizarse dentro de metas de calidad, costo y programación. El propósito es caracterizar la práctica actual, identificar fortalezas y debilidades y la habilidad del proceso de controlar o evitar causas significativas de baja calidad, costo o rendimiento programado.

ISO 15504

Propósitos

- aplicable a mejoramiento de procesos y a determinar capacidad
- aplicable a diferentes dominios, necesidades y tamaño de organización
- no supone estructura organizacional, filosofía administrativa, modelo e ciclo de vida, tecnologías de software o método de desarrollo
- usa criterios objetivos y prefiere cuantitativos
- salida en forma de perfiles comparables (en vez de número o pasa/falla)

ISO 15504

Contexto

unidad organizacional con actividad coherente y metas coherentes

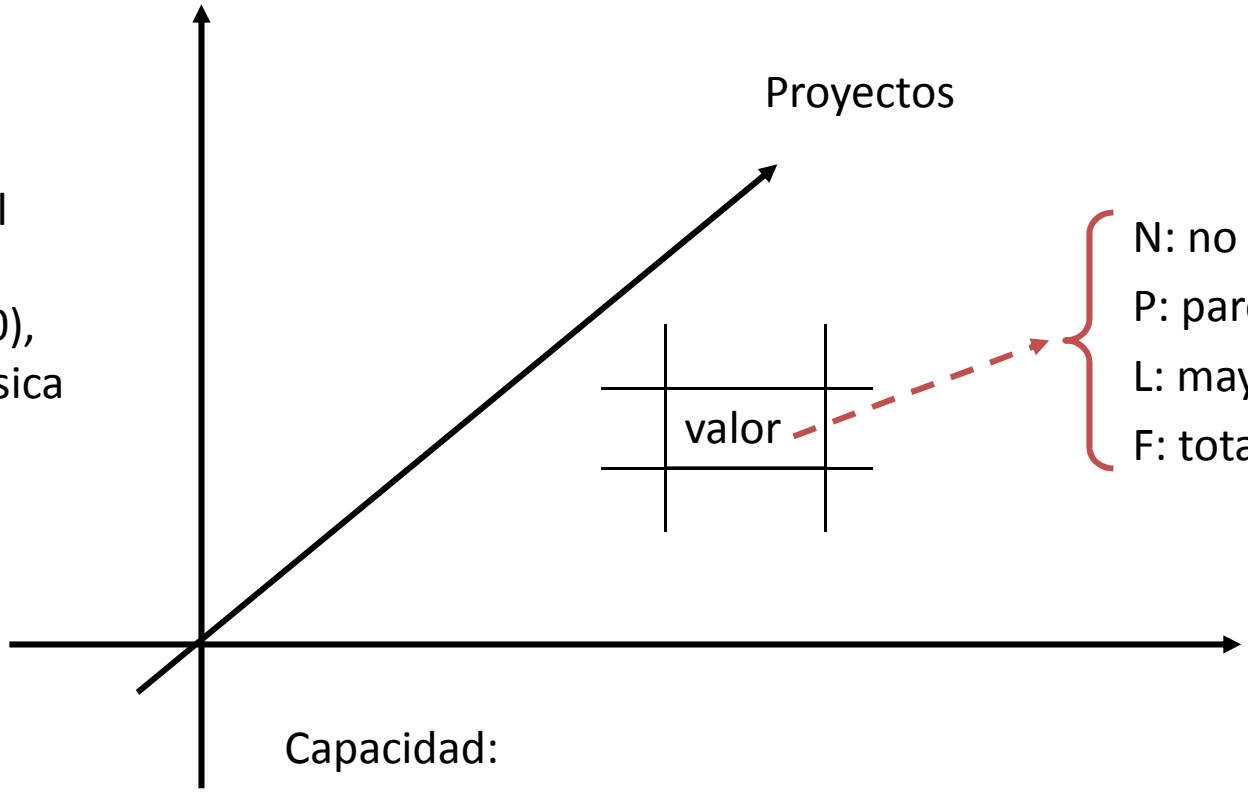
Etapas

- Preparación: alcance, metas del negocio, procesos a evaluar, instancias de proceso
- Recolección de datos: expertos; entrevistas, discusiones, análisis de documentos, herramientas
- Análisis de datos, asignar niveles, preparar salida
- Retroalimentación de resultados

Evaluación

Funcional:

categoría del proceso (5), proceso (<10), actividad básica



Atributos de procesos (9)

Niveles de capacidad (6)

ISO/IEC 15504

Categorías de proceso:

CUS	servicios al cliente
ENG	desarrollo directamente
SUP	soporte a todos los procesos
MAN	administración de procesos
ORG	de la organización que apoyan

ISO/IEC 15504

Ejemplo: procesos de desarrollo:

1. Requerimientos y diseño del sistema
2. Requerimientos del software
3. Diseño del software
4. Implementación del diseño
5. Integración y prueba del software
6. Integración y prueba del sistema
7. Mantenimiento del software y el sistema

ISO/IEC 15504

Niveles de capacidad:

0. Incompleto
1. Realizado
2. Administrado
3. Establecido
4. Predecible
5. Optimización

Atributos de proceso:

1.1 Process performance

2.1 Performance management

2.2 Work product management

3.1 Process definition

3.2 Process resource

4.1 Process measurement

4.2 Process control

5.1 Process change

5.2 Continuous improvement

ISO/IEC 15504

Fragmento de perfil

proceso

categoría

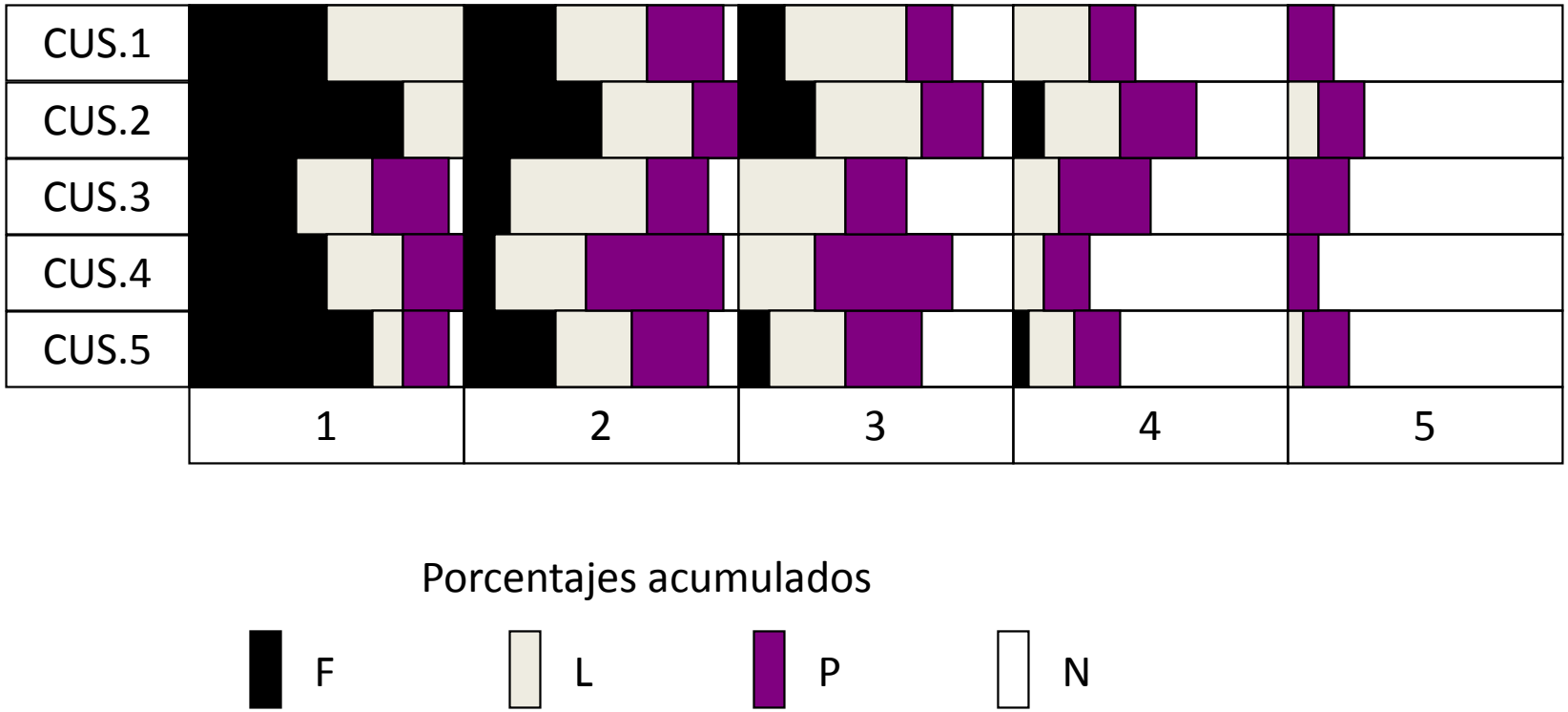
CUS.1	F	F	F	L	P	P	N	N	N
CUS.2	F	F	P	L	N	N	P	N	N
CUS.3	L	L	L	P	P	N	N	N	N
CUS.4	L	L	P	L	L	N	N	N	N
CUS.5	P	P	L	P	N	P	P	P	N
Atributos de proceso →	1.1	2.1	2.2	3.1	3.2	4.1	4.2	5.1	5.2
Nivel de capacidad →	1	2		3		4		5	

N: no realizado P: parcial

L: mayoritario F: total

ISO/IEC 15504

Fragmento de perfil gráfico



N: no realizado P: parcial L: mayoritario F: total

CMMI

CMMI: CMM Integrado

- Creado como un marco (framework) para varias disciplinas relacionadas:
  - Ingeniería de sistemas
  - Ingeniería de software
  - Desarrollo integrado de productos y procesos
  - Control de proveedores

  - No se requieren usar todas.
  - Se espera agregar otras más adelante

CMMI: CMM Integrado

- Dos tipos de modelos:
  - Continuo: útil para evaluaciones diferenciadas por proceso y comparaciones detalladas; permite migración de EIA/IS 731 (Industria eléctrica); permite comparación con ISO/IEC 15504
  - Por niveles: útil para comparación agregada; da resultado global que puede compararse con otras empresas; ayuda a migrar desde SW-CMM

CMMI: CMM Integrado

- Cada modelo tiene cuatro áreas:
  - Gestión de procesos
  - Gestión de proyectos
  - Soporte
  - Ingeniería
- Tiene metas específicas
- Tiene prácticas específicas

CMMI: CMM Integrado

Ingeniería de sistemas:

- Desarrollo de sistemas totales con o sin software. Transforma requerimientos del cliente en producto que resuelva sus problemas y soporte durante su ciclo de vida.

Ingeniería de Software:

- Enfoque sistemático, disciplinado y cuantificable al desarrollo, operación y mantenimiento de software

CMMI: CMM Integrado

Desarrollo integrado de productos y procesos:

- Enfoque sistemático que logra la colaboración a tiempo de los principales involucrados a través de la vida del producto. Debe usarse junto a un área de ingeniería.

Control de proveedores:

- Análisis de fuentes y monitoreo de proveedores antes de que entreguen los productos; sólo si es crítica la adquisición.

CMMI: CMM Integrado

Área de proceso:

Conjunto de prácticas relacionadas en un área que, al realizarse, satisfacen un conjunto de metas consideradas importantes para lograr mejoras significativas en el área.

CMMI: CMM Integrado

Cada área de proceso:

- Componentes requeridas:
  - Metas específicas
  - Metas genéricas (soporte)
- Componentes esperadas:
  - Prácticas específicas
  - Prácticas genéricas

CMMI: CMM Integrado

Niveles de capacidad:

0. Incompleto
1. Realizado
2. Administrado
3. Definido
4. Administrado cuantitativamente
5. Optimizante

CMMI: CMM Integrado

Nivel 0 Incompleto:

Una o más metas no se satisfacen; puede realizarse parcialmente o no realizarse del todo.

CMMI: CMM Integrado

Nivel 1 Realizado:

Todas las metas específicas se cumplen;
permite y soporta la producción de productos de salida bien identificados a partir de productos de entrada bien identificados.

Realiza prácticas básicas.

CMMI: CMM Integrado

Nivel 2 Administrado:

Además de ejecutarse, se planeó y se ejecutó de acuerdo a política; emplea gente hábil, recursos adecuados y salidas controladas; involucrados participan; monitoreado, controlado y revisado; evaluado frente a descripción de proceso.

Se satisfacen otrs metas como costo, calendario y aspectos de calidad.

Realiza prácticas avanzadas.

CMMI: CMM Integrado

Nivel 3 Definido:

Se define a partir de procesos estandarizados de la empresa, usando guías de adaptación. (Uso de estándares, procedimientos y descripción de proceso).

Proceso definido: propósito; entradas; criterios de entrada; actividades; papeles; medidas; pasos de verificación; salidas; criterios de salida.

CMMI: CMM Integrado

Nivel 4 Administrado cuantitativamente:

Se le controla usando métodos cuantitativos, especialmente con técnicas estadísticas.

Calidad y rendimiento del proceso sujetos a metas cuantitativas.

CMMI: CMM Integrado

Nivel 5 Optimizante:

Se cambia y adapta para satisfacer objetivos de negocios relevantes, actuales y proyectados.

Mejora continua analizando causas de variación en procesos.

CMMI: CMM Integrado

Áreas de proceso

Administración de procesos

- Enfoque de procesos organizacionales
- Definición de procesos organizacionales
- Entrenamiento organizacional
- Rendimiento de procesos organizacionales
- Innovasción y despliegue organizacionales

CMMI: CMM Integrado

Áreas de proceso

Administración de proyectos

- Planeación
- Monitoreo y control
- Administración de acuerdos con proveedores
- Administración de proyectos integrada
- Gestión de riesgo
- Control integrado de equipos
- Administración integrada de proveedores
- Administración cuantitativa del proyecto

MOPROSOFT (NYSE NMX-I-059/02)

Estándar
NMX-I-059-NYCE 2005

- **NYSE NMX-I-059/01** definición de conceptos y productos
- **NYSE NMX-I-059/02** requisitos de procesos
- **NYSE NMX-I-059/03** guía de implantación de procesos
- **NYSE NMX-I-059/04** para la evaluación de procesos (EvalProsoft)

Antecedentes

Problemática

- El 90% de las empresas desarrolladoras de software son micro y pequeña industria.
- Las empresas:
  - Son volátiles.
  - Cuentan con pocos recursos.
  - Tienen procesos no estandarizados, que dependen del personal que los ejecuta.
- Luchando por sobrevivir.
  - Buscan mejorar la calidad de sus productos a través de la mejora de sus procesos.

Programa Nacional para la Industria de Software en México

- En 2002 la Secretaría de Economía (SE) inició el Programa para el Desarrollo de la Industria de Software (PROSOFT)
- Objetivo:
  - Fortalecer a la industria de software en México

Estrategias del PROSOFT

1. Promover exportaciones y la atracción de inversiones
2. Educación y formación de personal competente
3. Contar con un marco legal promotor de la industria
4. Desarrollar el mercado interrno
5. Fortalecer a la industria local
6. Alcanzar niveles internacionales en capacidad de procesos
7. Promover la construcción de infraestructura física y de telecomunicaciones

Estrategia 6 (marzo 2002)

6. Alcanzar niveles internacionales en capacidad de procesos.

   6.1Definición de un modelo de procesos y de evaluación apropiado para la industria de software mexicana.

   6.2 Formación de instituciones de capacitación y asesoría en mejora de procesos.

   6.3 Apoyo financiero para la capacitación y la evaluación de capacidad de procesos.

   ...

MoProsoft: estructura interna

Características deseadas del Modelo del Procesos para la Industria de Software (MoProSoft)

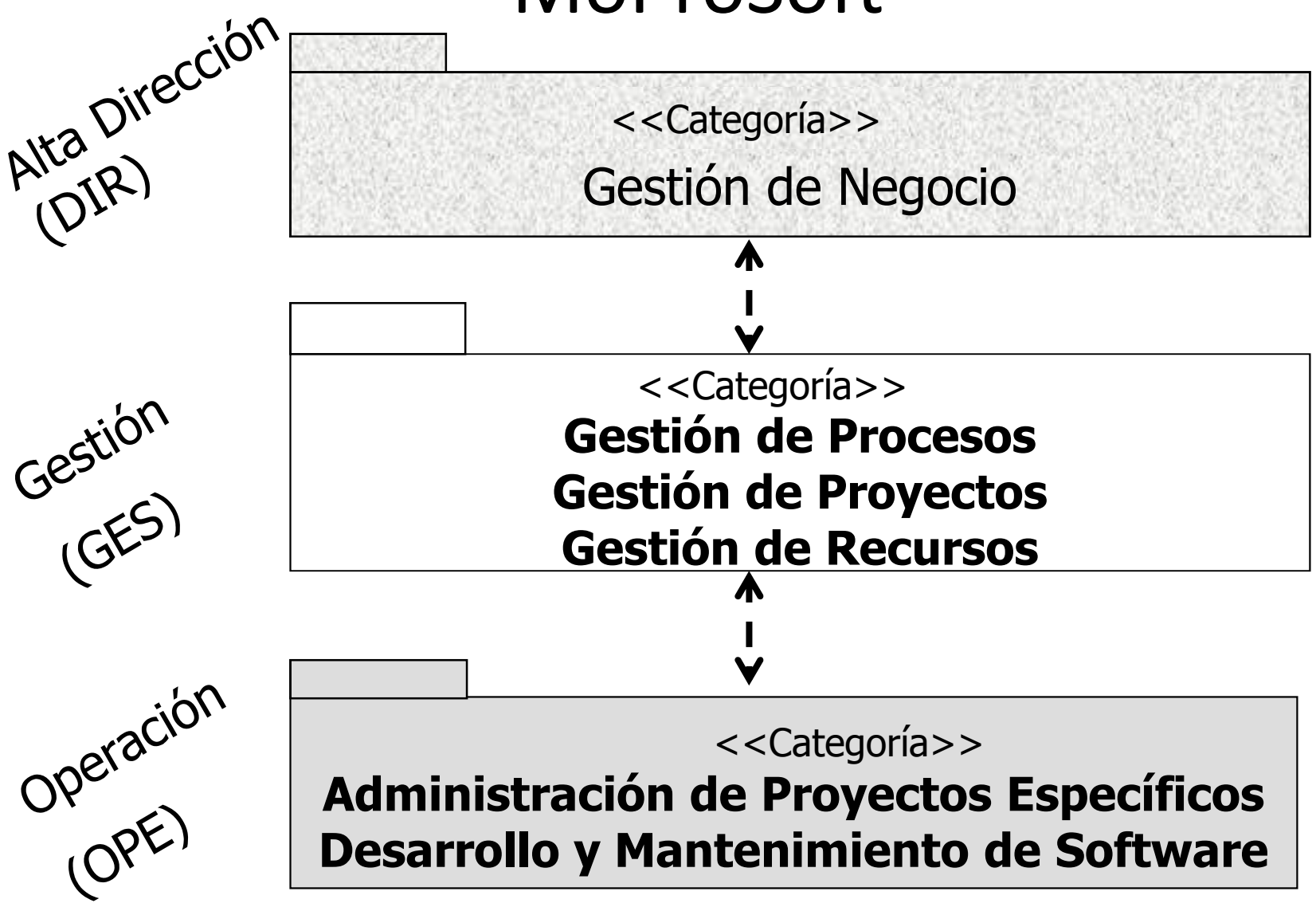
1. Específico para el desarrollo y mantenimiento de software.
2. Fácil de entender (comprensible).
3. Definido como un conjunto de procesos.
4. Práctico y fácil de aplicar, sobre todo en organizaciones pequeñas.

5. Orientado a mejorar los procesos para contribuir a los objetivos del negocio.

   (no simplemente ser un marco de referencia de certificación).
6. Debe de tener un mecanismo de evaluación o certificación.

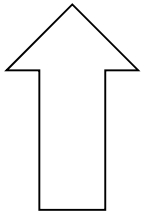
   (que indique un estado real de una organización durante un periodo de vigencia específico).
7. Aplicable como norma mexicana.

2.1 Categoría de Procesos de MoProSoft



Procesos de Operación







Administración de Proyectos Específicos

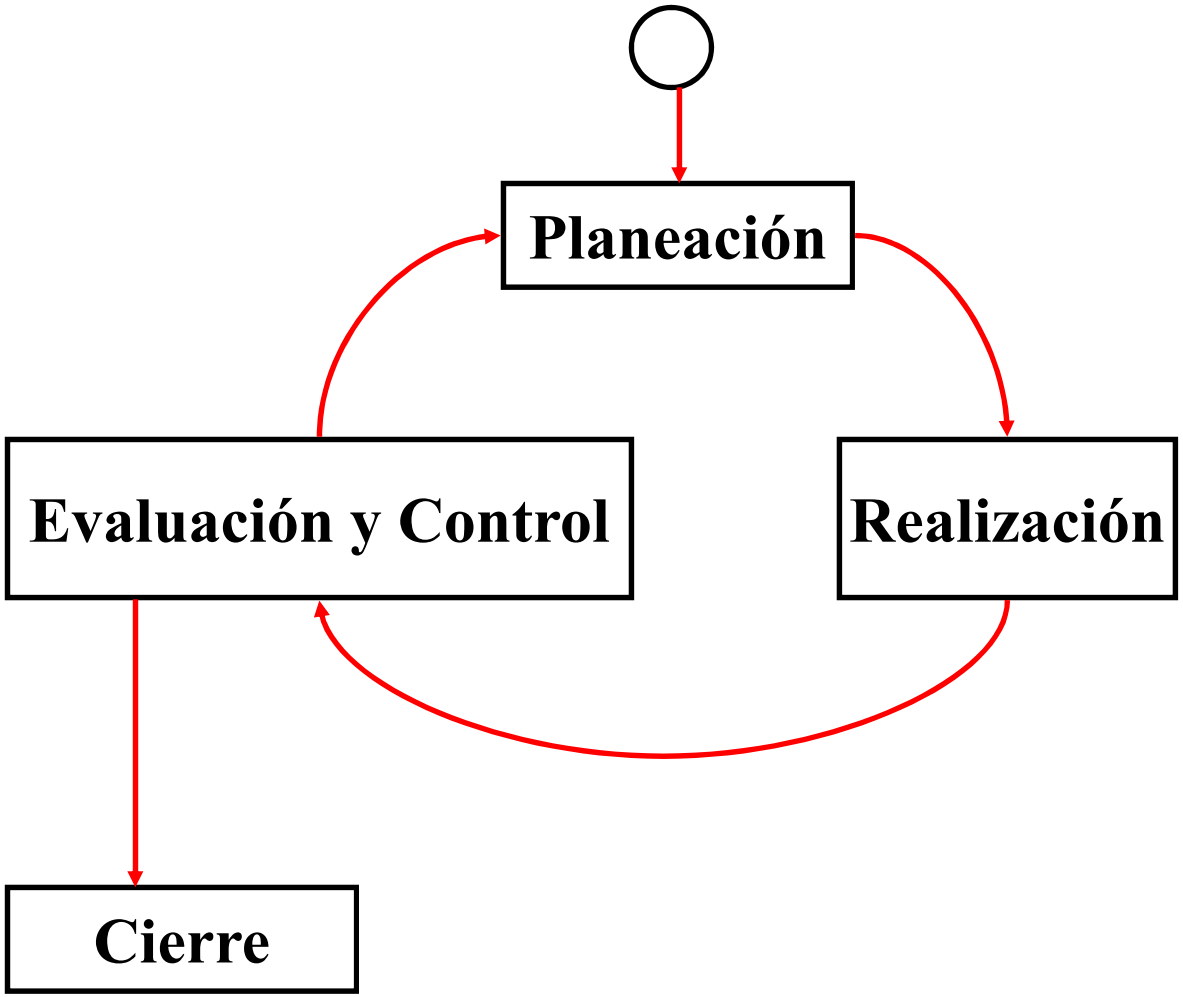


- Propósito:

  Establecer y llevar a cabo sistemáticamente las actividades que permitan cumplir con los objetivos de un proyecto en tiempo y costo esperados.

Administración de Proyectos Específicos

OPE



Desarrollo y Mantenimiento de Software



- Propósito:
  - Es la realización sistemática de las actividades de análisis, diseño, construcción, integración y pruebas de productos de software nuevos o modificados cumpliendo con los requerimientos especificados.

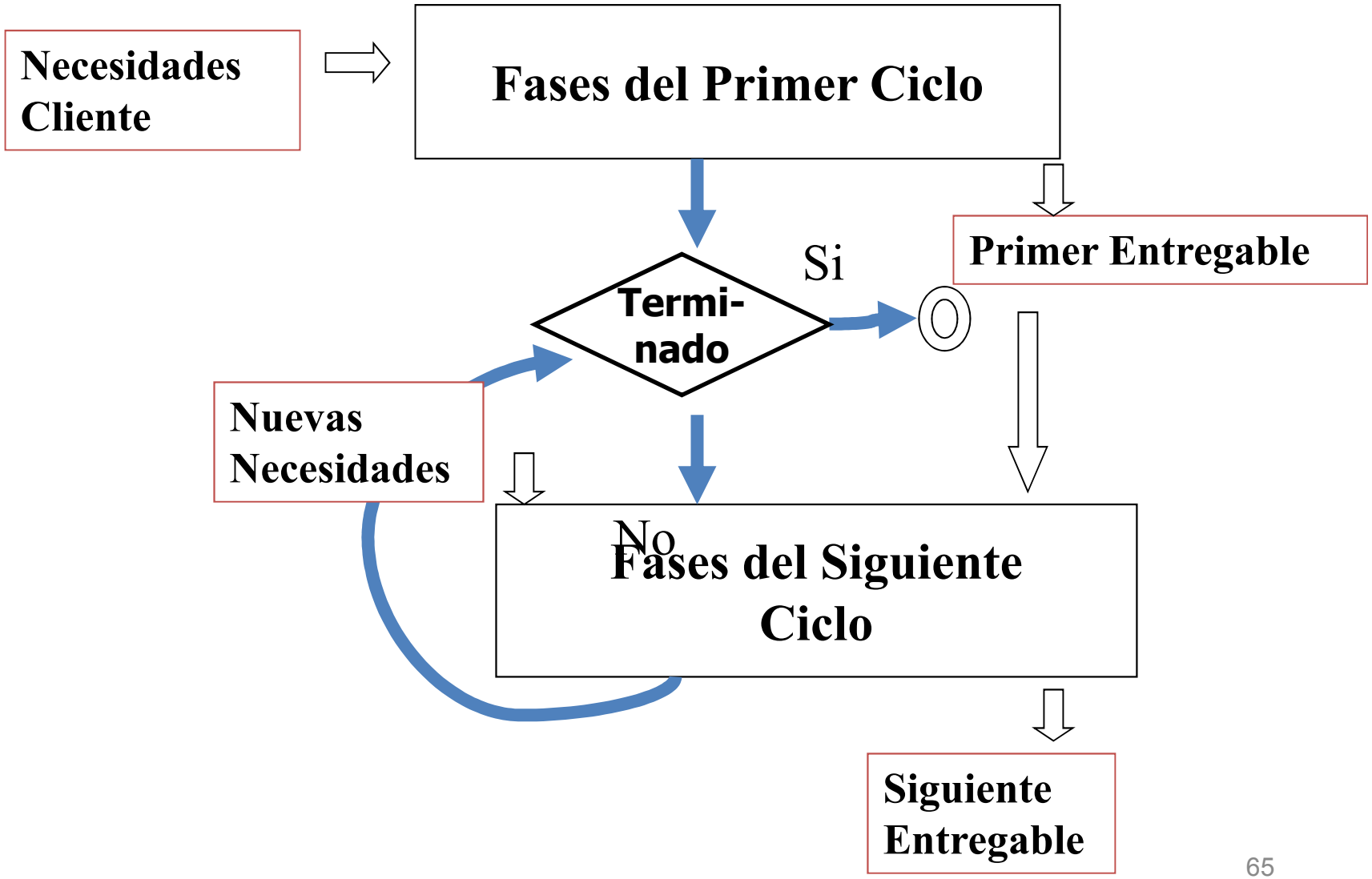
Proceso de Desarrollo y Mantenimiento de Software

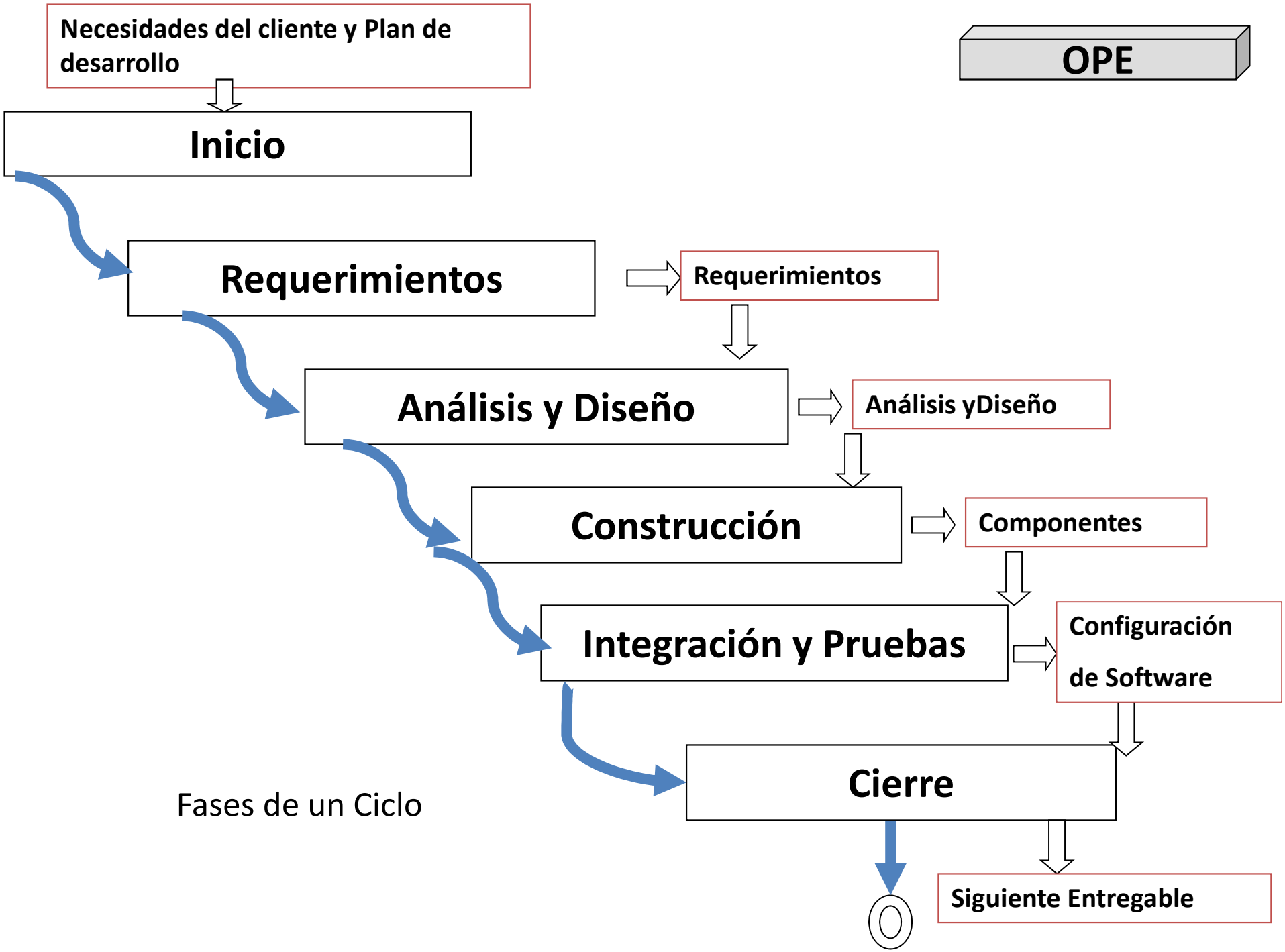
Flujos de trabajo

- Ciclos de Desarrollo
- Fases de un Ciclo
- Actividades de una Fase

Ciclos de Desarrollo

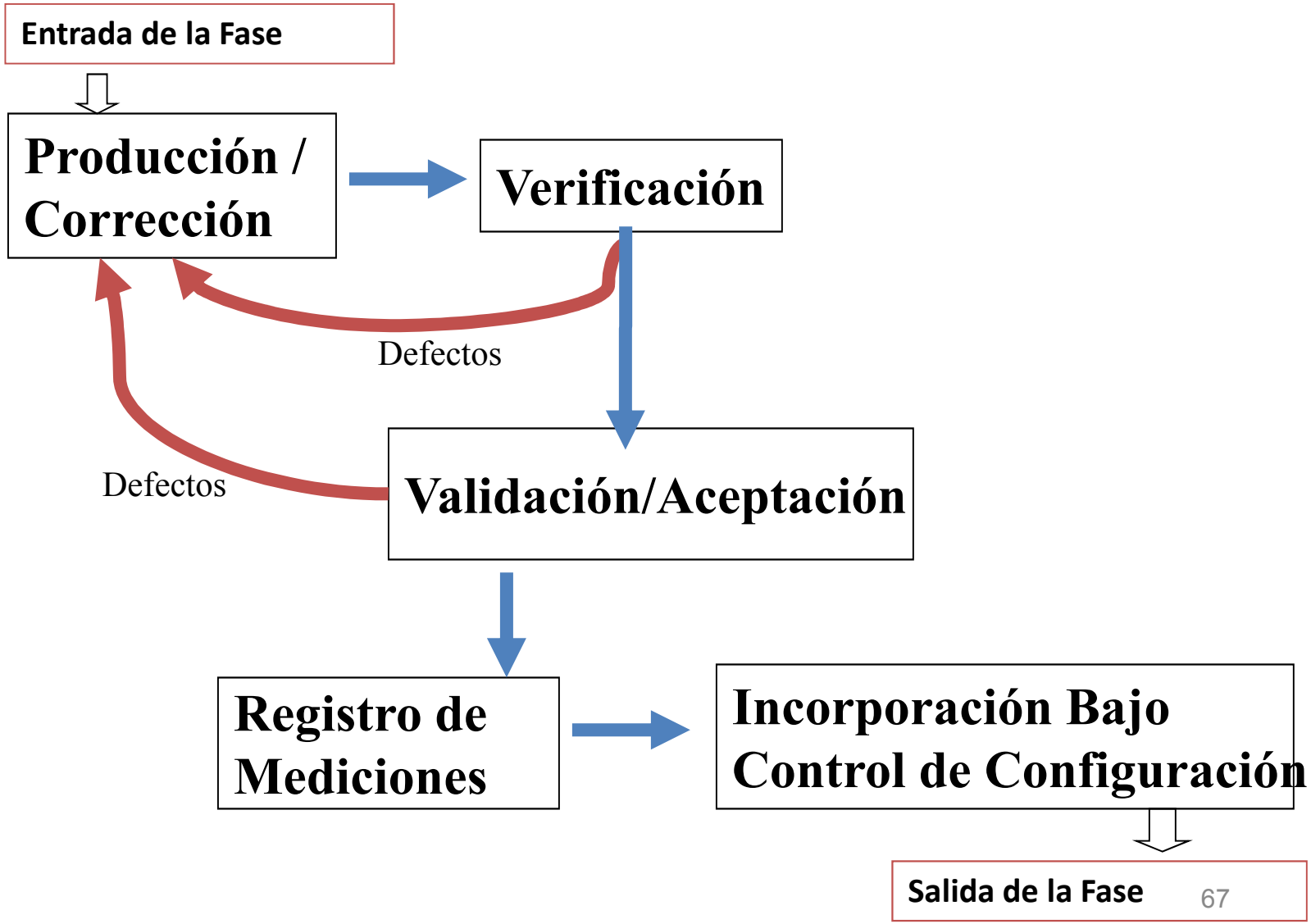
OPE





Fases de un Ciclo

Actividades de una Fase



2. MoProSoft (Patrón de Procesos)

Patrón de procesos

1. **Definición general de proceso**
2. **Prácticas**
3. **Guías de ajuste**

1. Definición general de proceso

1. Proceso
2. Categoría
3. Propósito
4. Descripción
5. Objetivos
6. Indicadores
7. Metas cuantitativas
8. Responsabilidad y autoridad
9. Subprocesos (opcional)
10. Procesos relacionados
11. Entradas
12. Salidas
13. Productos internos
14. Referencias bibliográficas

2. Prácticas

1. Roles involucrados y capacitación
2. Actividades
3. Diagrama de flujo de trabajo (en UML)
4. Verificaciones y validaciones
5. Incorporación a la Base de Conocimiento
6. Recursos de Infraestructura
7. Mediciones
8. Capacitación
9. Situaciones excepcionales
10. Lecciones aprendidas

3. Guías de ajuste

1. Identificación de la Guía
2. Descripción de la guía.

Descripción de posibles modificaciones al proceso que no deben afectar los objetivos del mismo

3. Uso del modelo de procesos.
Si **no hay** procesos establecidos.

1. Definir las metas cuantitativas de acuerdo a las estrategias de la organización.
2. Revisar los nombres de los roles y los productos (entradas, salidas o internos) y en su caso sustituirlos por los que se acostumbran en la organización.
3. Para cada producto definir el estándar de documentación cumpliendo con las características mencionadas en la descripción del producto.
4. Definir los recursos de infraestructura de cada proceso.

3. Uso del modelo de procesos.
Si **no hay** procesos establecidos.

5. Analizar si las mediciones de cada proceso son aplicables dentro del contexto de organización y en su caso modificarlas.
6. Usar las guías de ajuste para adecuar el proceso en función de las estrategias de la organización.
7. Posteriormente sustituir las guías de ajuste del modelo por las guías que apliquen en la organización.
8. Definir métodos, técnicas o procedimientos específicos para las actividades, tareas, verificaciones y validaciones.

3. Uso del modelo de procesos. Si ya se cuenta con procesos establecidos

- Establecer la correspondencia entre estos procesos y el modelo MOPROSOFT para identificar las coincidencias y discrepancias.
- La organización debe analizar las discrepancias y planear las actividades de ajuste de los procesos para lograr la cobertura completa de MoProSoft.

Implantación y mejora continua

- La organización debe establecer la estrategia de implantación de los procesos definidos. Puede decidir probarlos en proyectos

EvalProsoft

- La evaluación del cumplimiento de Moprosoft se hace de manera similar al estándar ISO 15504 (SPICE), usando los procesos definidos en Moprosoft.

- Cada concepto se califica con:
  - N: no se cumple (0 a 15%)
  - P: se cumple Parcialmente (más de 15 y hasta 50%)
  - A: se cumple Ampliamente (más de 50 hasta 85%)
  - C: se cumple Completamente (más del 85%)

Calificaciones de referencia para alcanzar un nivel

Nivel / Calificación mínima Atributo	1	2	3	4	5
Realización del proceso	A	C	C	C	C
Administración de la realización	-	A	C	C	C
Administración del producto de trabajo	-	A	C	C	C
Definición del proceso	-	-	A	C	C
Implantación del proceso	-	-	A	C	C
Medición del proceso	-	-	-	A	C
Control del proceso	-	-	-	A	C
Innovación del proceso	-	-	-	-	A
Optimización del proceso	-	-	-	-	A

Aseguramiento de la calidad

Tomado de Pressman, 5ª Ed., cap 8

2011

Aseguramiento y verificación

- La calidad del software debe asegurarse a todo lo largo del proyecto. Busca garantizar que las cosas se hacen bien desde un principio, no como algo que se añade al final.
- La verificación se realiza cuando van concluyendo etapas, generalmente asociada con pruebas. En ese momento no se puede cambiar mucho, aunque se puede corregir defectos.

Aseguramiento de calidad

- También llamada Garantía de calidad
- Consiste en auditoría y funciones de información de la gestión
- Permite informar los datos necesarios sobre la calidad del producto

Costos de calidad

- Prevención
  - Costos para prevenir problemas, evitar defectos
- Evaluación
  - Costos de evaluar productos
- Fallos
  - Costos derivados de los defectos, especialmente los residuales que llegan al cliente

Costos de calidad

- Prevención
  - Planificación de calidad
  - Revisiones técnicas formales
  - Equipo de pruebas
  - Formación

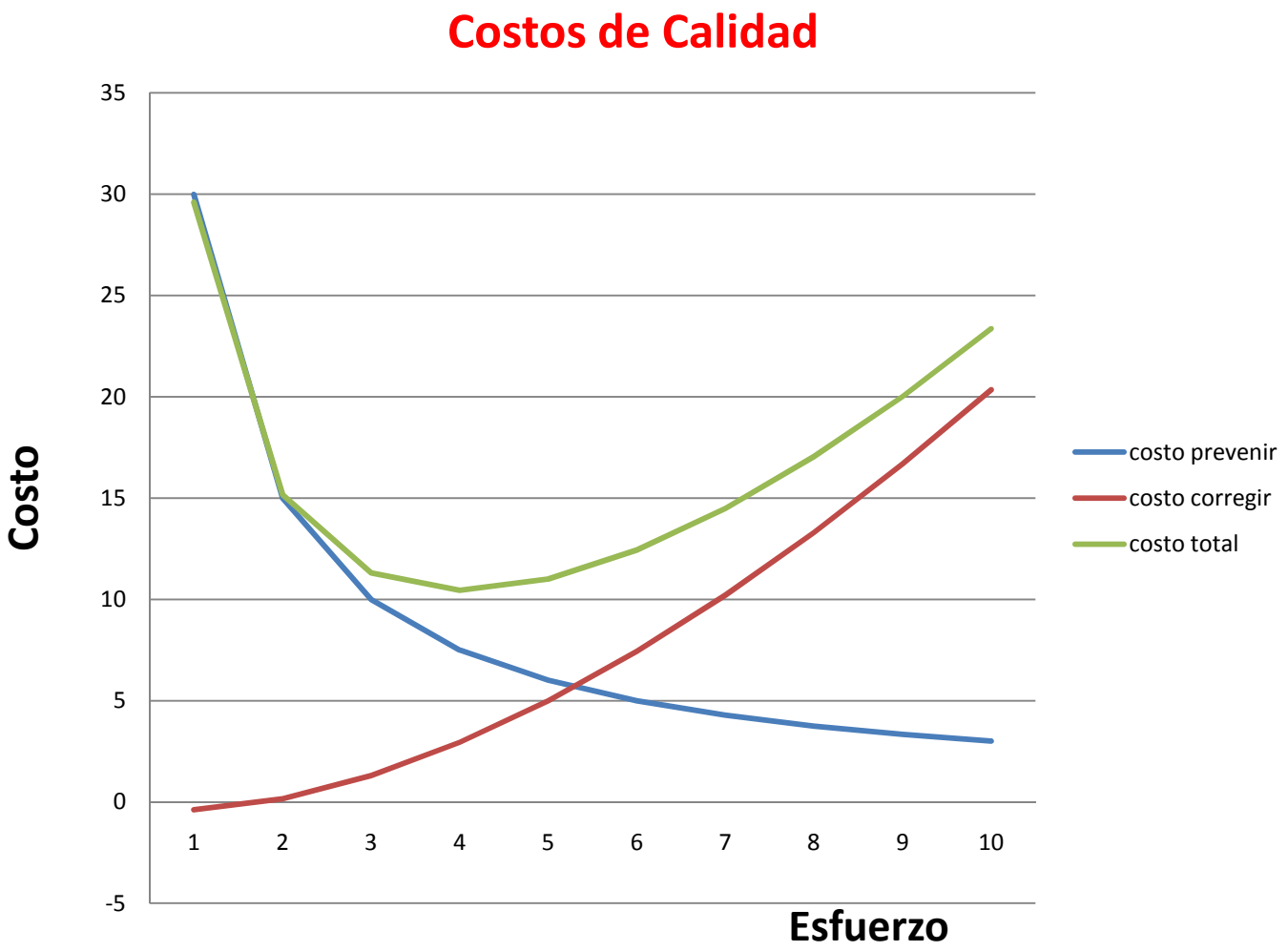
Costos de calidad

- Evaluación
  - Inspección en el proceso y entre procesos
  - Calibración y mantenimiento de equipos
  - Realización de Pruebas

Costos de calidad

- Fallos
  - Internos (se identifican antes de liberar el producto)
    - Retrabajo debido a revisión
    - Reparación de defectos
    - Análisis de las modalidades de los fallos
  - Externos (después de entregado)
    - Resolución de quejas
    - Devolución y sustitución de productos
    - Soporte de línea de ayuda
    - Trabajo de garantía

Costos de Calidad



Actividades de aseguramiento de calidad

- Establecer plan de aseguramiento de calidad
- Participar en el desarrollo de la descripción del proceso de software
- Revisión de actividades de ingeniería de software, para verificar su ajuste al proceso definido
- Auditoría de los procesos de software, para verificar su ajuste al proceso definido
- Asegurar que las desviaciones del trabajo y los productos se documenten y manejen de acuerdo a procedimiento establecido
- Registrar lo que no se ajuste a los requerimientos y avisar a superiores

Plan de aseguramiento de calidad

- Atributos relevantes para el proyecto
- Evaluaciones a realizar
- Auditorías y revisiones a realizar
- Estándares aplicables
- Procedimientos para información y seguimiento de problemas
- Documentos producidos por el grupo de aseguramiento de calidad
- Realimentación de información proporcionada al equipo de desarrollo

Revisiones

- Se aplican en diversos momentos del desarrollo, para detectar errores y defectos
- Verifican que los productos intermedios cumplan lo que se espera de ellos
- Existen muchas modalidades, desde charla informal, hasta presentación formal a clientes.
- Varias formas: recorridos (walkthrough), inspecciones, revisiones técnicas formales

RTF

- La llevan a cabo ingenieros de software y otros como apoyo, según se requiera
- Objetivos:
  - Descubrir errores en la función, la lógica o la implementación del software
  - Verificar que el software bajo revisión alcanza los requerimientos
  - Verificar que el software ha sido representado de acuerdo a estándares predefinidos
  - Conseguir software desarrollado de manera uniforme
  - Hacer los proyectos más manejables
- Además:
  - Sirve de entrenamiento de personal joven
  - Promueve seguridad y continuidad (sirve para prevenir algunos riesgos)

RTF

- Se convocan entre tres y cinco personas, entregándoles materiales necesarios
- Cada uno prepara por anticipado, unas dos horas de trabajo
- La reunión se centra en aspectos específicos y reducidos de productos, no en personas
- La duración de la reunión debe ser menor a dos horas
- Se incluye al autor, para presentar el producto

RTF

- Al final se decide:
  - Se acepta como está
  - Se rechaza el producto
  - Se acepta pero sujeto a cambios (generalmente por defectos menores; no requiere otra revisión)
- Se registra:
  - Lista de sucesos
  - Informe sumario

RTF

- Lista de sucesos
  - Identifica áreas problemáticas
  - Sirve como lista de comprobación para hacer correcciones

RTF

- Informe sumario: responde a
  - Qué se revisó
  - Quiénes lo revisaron
  - Qué se descubrió
  - Cuáles son las conclusiones
- Características:
  - Página simple (puede tener anexos)
  - Se almacena en el registro histórico del proyecto
  - Se envía a líder de proyecto y otros

Directrices para RTF

1. Revisar el producto, no al productor (cuidar interacciones, tono de la reunión)
2. Fijar agenda y mantenerla (no dejar divagar)
3. Limitar debate e impugnaciones
4. Enunciar áreas de problemas, pero no intentar resolverlos
5. Tomar notas escritas
6. Limitar número de participantes e insistir en preparación (a veces excluyen al que no lo hace)
7. Desarrollar lista de comprobación para cada producto a revisar
8. Disponer de recursos y agenda
9. Entrenar a los revisores
10. Repasar revisiones anteriores

Costo beneficio

- Las reuniones cuestan (horas de trabajo, que se agregan al esfuerzo total del proyecto)
- Ganancia: hallar defectos antes de pruebas y evitando rehacer trabajo; ahorro en costos
- Otra ganancia: evidencia de la calidad a lo largo del proceso

Ejemplo

- Se muestran fragmentos de listas de cotejo empleadas como guía para realizar revisiones técnicas formales en la Especialización en Ingeniería de Software.
- En ese programa los alumnos desarrollaban software comenzando con Áncora y siguiendo con el Proceso Unificado (RUP), implementando en Delphi.
- Para cada revisión se cruzaban los proyectos: un equipo revisa el avance de otro y viceversa.
- Antes de la revisión se enviaba el material para preparar la tarea

Análisis

Proyecto: _____

Autor: _____

Revisó: _____

Fecha: _____

Escala: 5: todas(os); 4: casi todos(as); 3: aproximadamente la mitad; 2: casi ninguno(a); 1: ninguno(a); 0: no puedo calificar

I. Con paquetes de análisis, modelo de casos de uso

Atributo observado	**Concepto**	5	4	3	2	1	0
Conforme	Cada paquete contiene al menos un caso de uso						
Completo	(Todos asignados) Cada caso de uso del modelo de casos de uso está asignado a algún paquete						
Conforme	(No repetición) Cada caso de uso está asignado a **un solo paquete**						
Correcto	Cada caso de uso en paquetes corresponde a uno del modelo de casos de uso						
Conforme	Los casos de uso agrupados en un paquete muestran cohesión						
Conforme	Los paquetes muestran poco acoplamiento entre sí						
Conforme	Las relaciones entre paquetes se indican punteadas						
Conforme	Las relaciones entre paquetes van de capa específica a capa general						

Análisis

III Con la tabla de Riesgos

Atributo observado	**Concepto**	5	4	3	2	1	0
Conforme	Todos los campos de la tabla están llenos						
Realista	Todos los riesgos son plausibles						
Conforme	La columna Impacto indica las partes del proyecto que serán afectadas						
Realista	La asignación de responsabilidad es aceptable						
Conforme	La columna Contingencia expresa una acción a tomar cuando ocurre el riesgo						
Realista	La acción expresada en la columna Contingencia es aceptable						
	Cada riesgo tiene asociada la persona o grupo encargadas del monitoreo						
Razonable	El conjunto de riesgos cubre todas las expectativas razonables						

DISEÑO
I. Con casos uso diseño y casos uso análisis

Atributo observado	Concepto	5	4	3	2	1	0
Correcto	Cada caso de uso de diseño corresponde a uno de análisis						
Completo	Cada caso de uso de análisis corresponde a uno de diseño						
Completo	Cada clase de análisis corresponde o está incluida en una clase de diseño						
Conforme	Cada clase de diseño tiene sus atributos y métodos						
Conforme	Los atributos y métodos de cada clase se orientan al lenguaje de programación elegido						
Completo	Cada diagrama de colaboración (análisis) corresponde a algún diagrama de secuencia						
Conforme	Cada clase empleada en un diagrama de secuencia existe entre las clases de diseño						
Conforme	Cada interacción en un diagrama de secuencia tiene dirección y es de línea sólida						
Conforme	Si existen líneas punteadas, cada una corresponde a un regreso del control						
Conforme	Cada interacción en un diagrama de secuencia tiene el nombre del método correspondiente en la clase destino						
Correcto	Cada método empleado en un diagrama de secuencia existe en alguna clase de diseño						
Correcto	Cada método de cada clase de diseño se emplea en al menos un diagrama de secuencia						
Correcto	En cada caso de uso de diseño, si existen restricciones, corresponden a restricciones de análisis						

Pruebas

I. Con bitácora, casos de uso de diseño y casos de prueba

Atributo observado	Concepto	5	4	3	2	1
Conforme	Cada caso de prueba tiene entrada y salida esperada					
Conforme	En cada caso de prueba la entrada corresponde a parejas (variable, valor) o acción en teclado o ratón					
Conforme	En cada caso de prueba la salida corresponde a parejas (variable, valor) o a un mensaje					
Conforme	En cada caso de prueba que tenga condiciones de entrada, estas son proposiciones lógicas					
Conforme	En cada caso de prueba que tenga condiciones de salida, estas son proposiciones lógicas					
Conforme	Cada caso de prueba indica a qué caso de uso corresponde					
Completo	Cada renglón de la bitácora corresponde al menos con un caso de prueba					
Completo	Cada caso de uso corresponde al menos con un caso de prueba positivo y uno negativo					

CMMI

Marca externa

GC-F-004v.04



¿QUÉ ES CMMI?

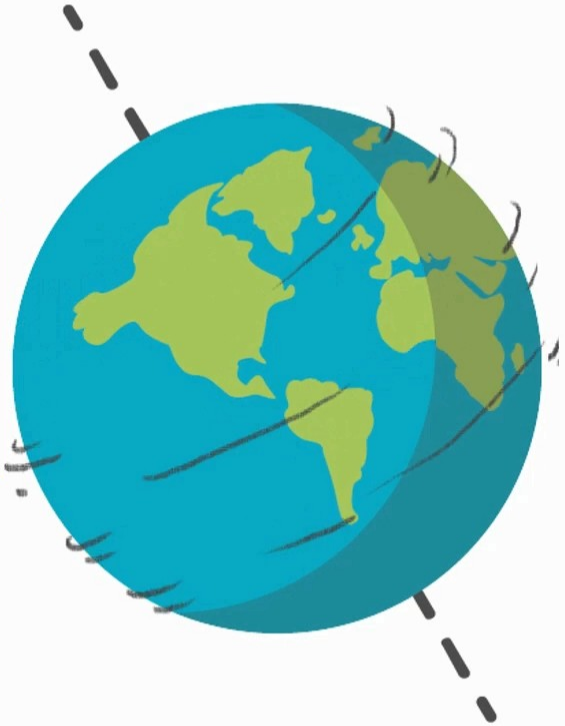
CAPABILITY MATURITY MODEL INTEGRATION (MODELO DE MADUREZ DE CAPACIDADES DE INTEGRACIÓN)

CMMI ES UN MODELO QUE CONTIENE LAS MEJORES PRÁCTICAS Y QUE PROVEE A LAS ORGANIZACIONES DE AQUELLOS ELEMENTOS QUE SON ESENCIALES PARA QUE LOS **PROCESOS DE NEGOCIO** DE LAS MISMAS SEAN EFECTIVOS.



CCMI NACE A FINALES DE LOS AÑOS 80

EN LA INDUSTRIA MILITAR DE EU Y AMÉRICA LATINA



SU PRIMERA FINALIDAD INICIAL FUE LA DE **EVALUAR LOS PROCESOS** DE SOFTWARE MÁS CRÍTICOS.



1991

SE CREÓ POR PRIMERA VEZ EL MODELO CMMI PARA SOFTWARE

LA PROLIFERACIÓN DE NUEVOS MODELOS DIO LUGAR A CONFUSIÓN, POR LO QUE EL GOBIERNO DECIDIÓ FINANCIAR UN PROYECTO DE 2 AÑOS A FIN DE CREAR UN SOLO MARCO EXTENSIBLE PARA

INGENIERÍA DE SISTEMAS

INGENIERIA DE SOFTWARE

Y EL DESARROLLO DE PRODUCTOS



ANTECEDENTES

1984

El congreso del gobierno Americano aprueba la creación de un organismo de investigación para el desarrollo de modelos de mejora en el Desarrollo de sistemas de software

1985

SEI (Software Engineering Institute) empieza a trabajar en un marco de madurez de procesos para evaluar a las empresas productoras de software.

SEI publica la versión 1.0 del modelo de madurez de capacidades para el software (SW-CMM, Capability Maturity Model for Software)

1991

1993

SEI publica la version 1.1 de SW-CMM

Publicación de la version 1.2

1997

2000

SW-CMM fue integrado y relevado por el nuevo modelo CMMI

2005

Limite para migrar de SW-CMMI a CMMI

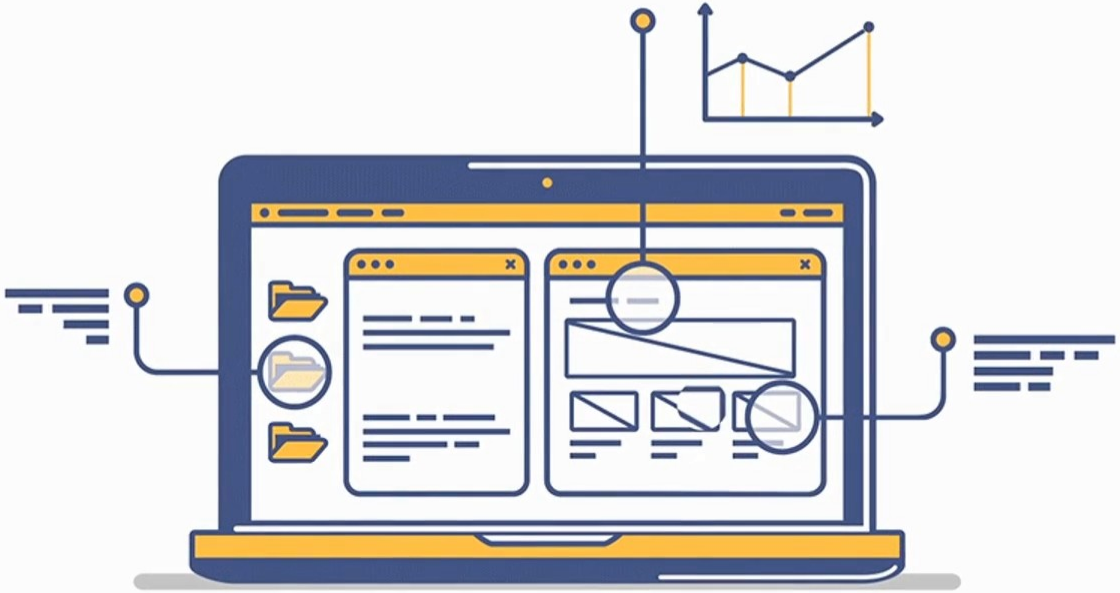
2006

Se libera la version CMMI para el Desarrollo (DEV-CMMI)

Se libera CMMI para la adquisición. (ACQ-CMMI)

2007

Actualidad



EL MODELO CMMI SE USA PARA EVALUAR EL NIVEL DE MADUREZ DE UNA COMPAÑÍA EN TÉRMINOS DE DESARROLLO INFORMÁTICO.

CMMI TIENE COMO OBJETIVO:

ALENTAR A LAS COMPAÑÍAS PARA QUE MONITOREEN Y MEJOREN CONTINUAMENTE SUS PROCESOS, Y EVALÚEN EL NIVEL DE MADUREZ DE LOS MISMOS EN UNA ESCALA DE CINCO NIVELES ESTABLECIDA POR EL CMMI.

EN LA ACTUALIDAD HAY DOS ÁREAS DE INTERÉS CUBIERTAS POR LOS MODELOS CMMI

DESARROLLO

ADQUISIÓN

OBJETIVOS CMMI



EL MODELO TIENE

4 ÁREAS DE CONOCIMIENTO



ING.SISTEMAS

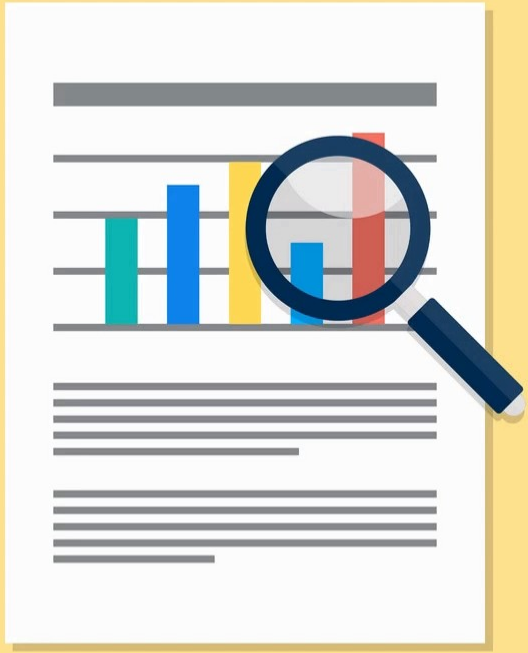
ACUERDOS CON PROVEEDORES

ING.SOFTWARE

IPPD

CARACTERÍSTICAS

LAS PRÁCTICAS QUE DEBEN SER REALIZADAS POR CADA ÁREA CLAVE DE PROCESO ESTÁN ORGANIZADAS EN 5 CARACTERÍSTICAS COMUNES, LAS CUALES CONSTITUYEN PROPIEDADES QUE INDICAN SI LA IMPLEMENTACIÓN Y LA INSTITUCIONALIZACIÓN DE UN PROCESO CLAVE ES EFECTIVO, REPETIBLE Y DURADERO.



ESTAS 5 CARACTERISTICAS SON:



01 COMPROMISO DE LA REALIZACIÓN

02 LA CAPACIDAD DE REALIZACIÓN

03 LAS ACTIVIDADES REALIZADAS

04 LAS MEDICIONES Y EL ANÁLISIS

05 LA VERIFICACIÓN DE LA IMPLEMENTACIÓN

DESVENTAJAS

- FALTA DE ADECUACIÓN AL ENFOQUE DE SERVICIOS QUE EXPERIMENTA EL SECTOR DE LAS TI (PROCESOS DE DESARROLLO DE PRODUCTOS DE SOFTWARE)
- ALTO ESFUERZO DE IMPLEMENTACIÓN QUE EXIGE
- EL PROCESO DE EVALUACIÓN ES MUY COSTOSO EN TIEMPO Y ESFUERZO.
- LA COMPLEJIDAD DE LA EVALUACIÓN PUEDE ATENTAR CONTRA LA DEFINICIÓN DE OBJETIVOS CONCRETOS DE LA EMPRESA



VENTAJAS

- REDUCCIÓN DEL COSTE DE DESARROLLO.
- LOCALIZACIÓN Y RESOLUCIÓN DE DEFECTOS.
- MEJORA EN LA FIABILIDAD DE LA PLANIFICACIÓN.
- **AUMENTO DE LA PRODUCTIVIDAD.**
- REDUCCIÓN DE LOS TRABAJOS DERIVADOS DE CORRECCIONES.
- AUMENTO DE LA EFECTIVIDAD SOBRE LA PLANIFICACIÓN REALIZADA.
- **MEJORA EN LA CALIDAD DEL PRODUCTO.**
- SE PUEDE APLICAR A MÁS DE UNA DISCIPLINA.
- MEJOR ATENCIÓN A LAS ÁREAS DE INGENIERÍA
- MEJORA DE LA COMUNICACIÓN PARA QUE CADA PARTICIPANTE CUMPLA CON SUS RESPONSABILIDADES
- MEJORA LA PLANIFICACIÓN
- **LOS CLIENTES VIVEN MÁS INFORMADOS**



¿EN QUE SE DIFERENCIA

CMMI

DE OTROS MODELOS



SE DIFERENCIA DE OTROS MODELOS POR EL HECHO DE ESTAR BASADO EN PRÁCTICAS AJUSTABLES A **CUALQUIER** DOMINIO DE PRODUCCIÓN



Y POSEER UN ENFOQUE GLOBAL E INTEGRADO DE LA ORGANIZACIÓN, CON EL PROPÓSITO DE

ALCANZAR LOS OBJETIVOS DEL NEGOCIO.



CMMI PERMITE A EMPRESAS COMPLEJAS COMPUESTAS POR VARIAS ÁREAS DE NEGOCIO INSTAURAR DE UNA FORMA MÁS SENCILLA UN SISTEMA DE **ASEGURAMIENTO DE LA CALIDAD**.

EL CCMI TIENE **DOS REPRESENTACIONES**, ESTAS REPRESENTACIONES PERMITEN A LA ORGANIZACIÓN PERSEGUIR DIFERENTES OBJETIVOS DE MEJORA.

LA PRESENTACIÓN Y ORGANIZACIÓN DE LA INFORMACIÓN ES DIFERENTE PARA CADA UNA, SIN EMBARGO, EL CONTENIDO ES EL MISMO.



POR ETAPAS (STAGED)

MIDEN LA MADUREZ DE UN GRUPO DE PROCESOS DENTRO DE UNA ORGANIZACIÓN





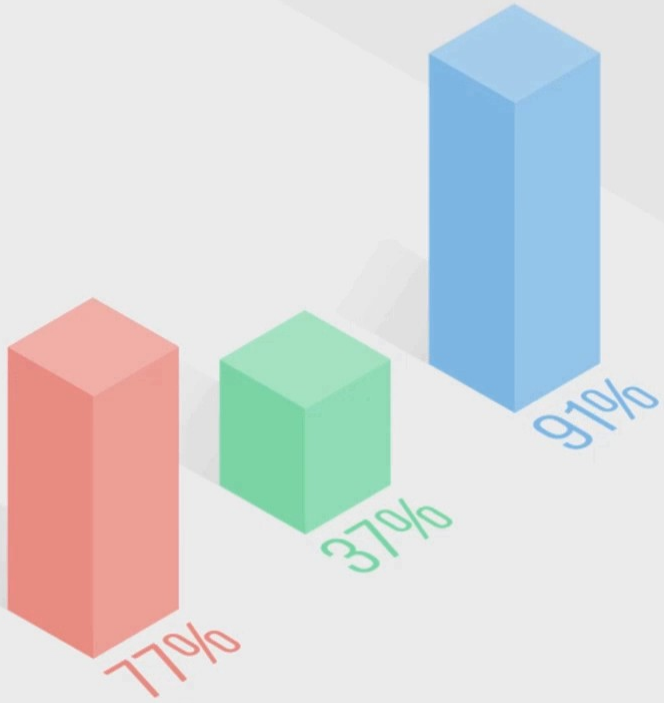


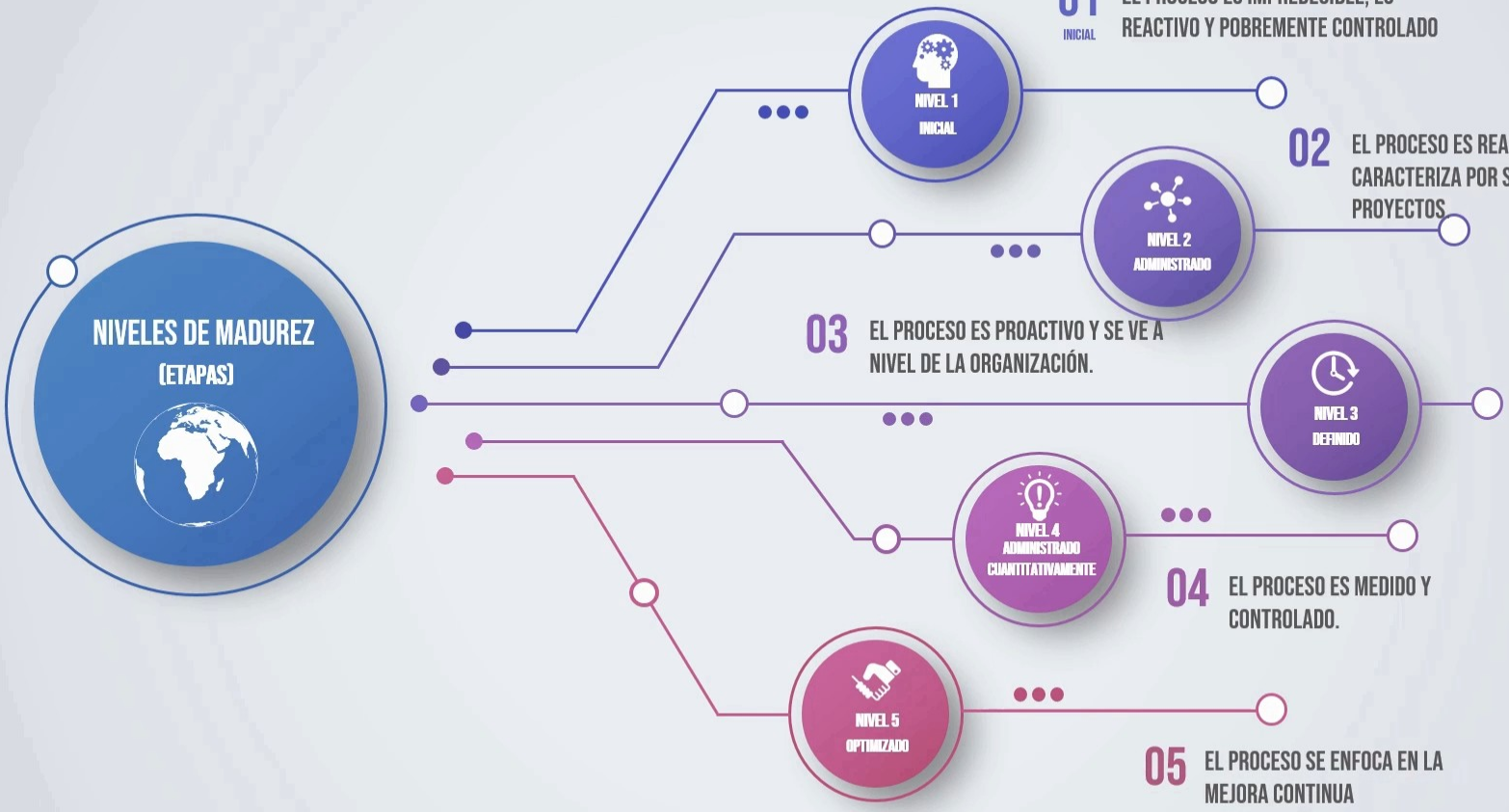


CONTINUO

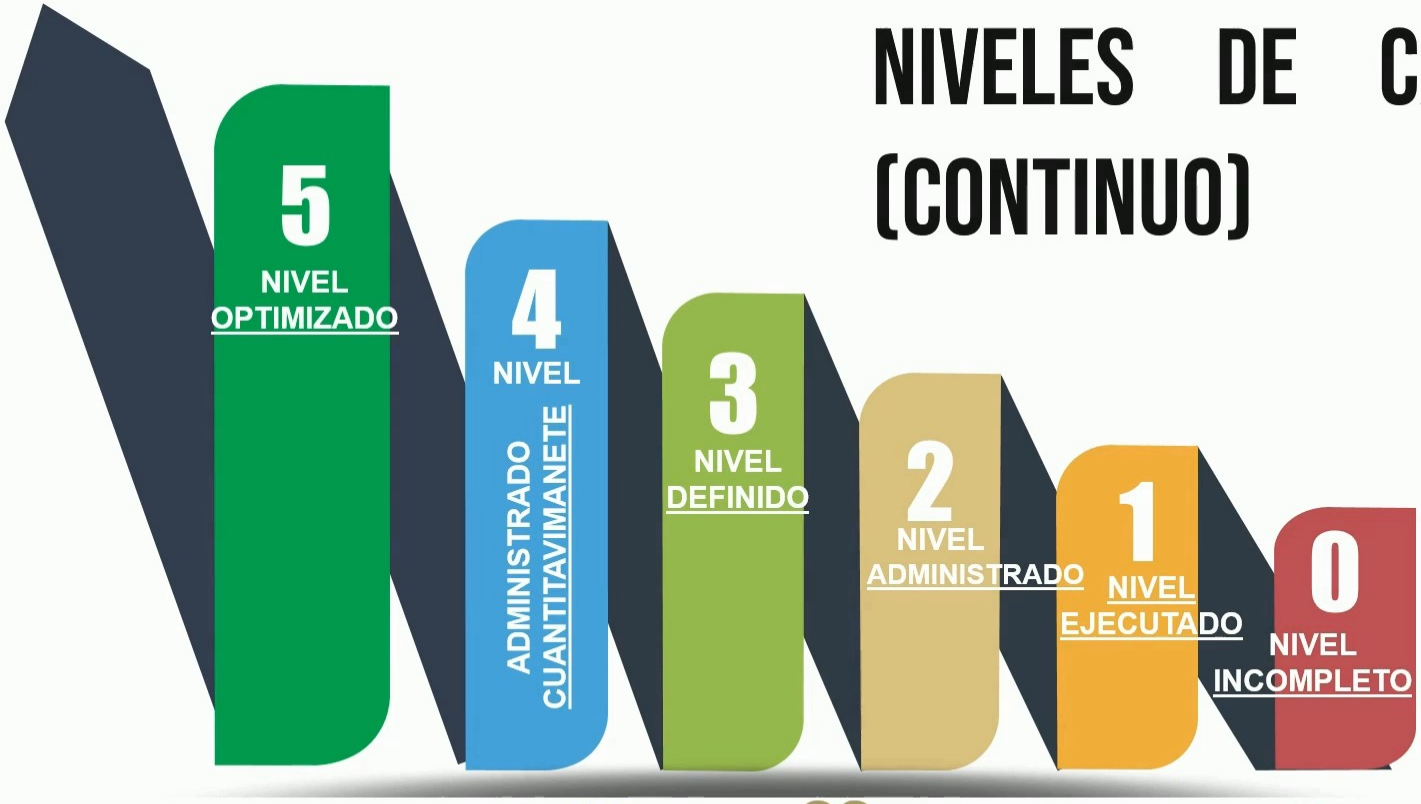
MIDEN LA MADUREZ DE UN PROCESO EN PARTICULAR DENTRO DE UNA ORGANIZACIÓN

5 NIVELES DE CAPACIDAD




NIVELES DE MADUREZ
(ETAPAS)
NIVEL 1
INICIAL
NIVEL 2
ADMINISTRADO
NIVEL 3
DEFINIDO
NIVEL 4
ADMINISTRADO
CUANTITATIVAMENTE
NIVEL 5
OPTIMIZADO
01
INICIAL
EL PROCESO ES IMPREDECIBLE, ES REACTIVO Y POBREMENTE CONTROLADO
02
EL PROCESO ES REACTIVO Y SE CARACTERIZA POR SU APLICACIÓN A PROYECTOS
03
EL PROCESO ES PROACTIVO Y SE VE A NIVEL DE LA ORGANIZACIÓN.
04
EL PROCESO ES MEDIDO Y CONTROLADO.
05
EL PROCESO SE ENFOCA EN LA MEJORA CONTINUA

NIVELES DE CAPACIDAD (CONTINUO)



05 OPTIIMIZADO
El proceso se enfoca en la mejora continua.

04 ADMINISTRADO CUANTITATIVAMENTE
El proceso es medido y controlado.

03 DEFINIDO
El proceso es proactivo y se ve a nivel de la organización.

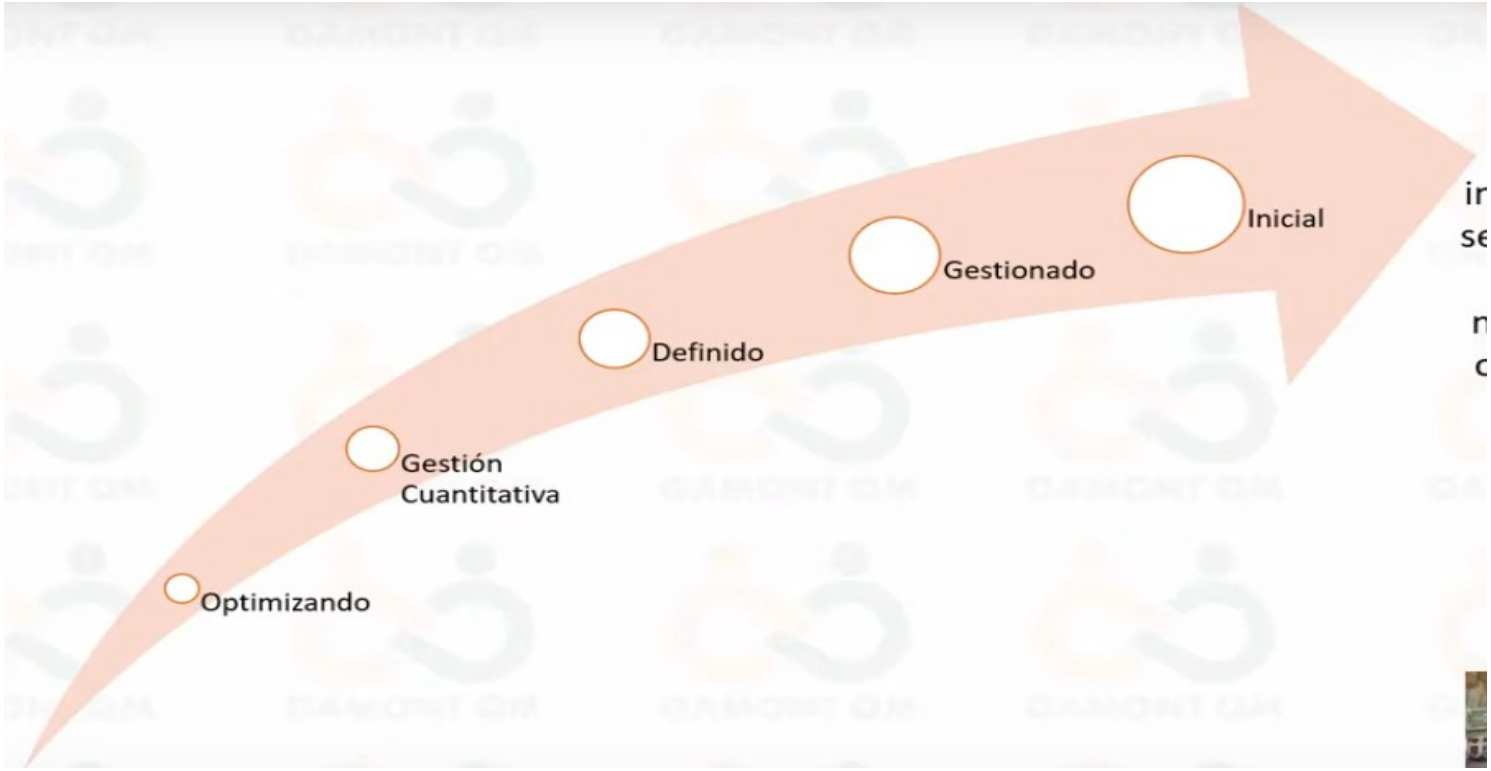
02 ADMINISTRADO
El proceso es reactivo y se caracteriza por su aplicación a proyectos.

01 EJECUTADO
El proceso se ejecuta y se producen productos basados en productos de entrada identificados.

0 INCOMPLETO
El proceso no se ejecuta o se hace parcialmente.

NIVEL DE MADUREZ Y EL RIESGO

GRAVEDAD DEL IMPACTO DE EVENTOS NEGATIVOS



POSIBILIDAD DE OCURRENCIA DE EVENTOS NEGATIVOS

EMPRESAS CERTIFICADAS CON CMMI MUNDIALMENTE



EMPRESAS CERTIFICADAS CON CMMI MUNDIALMENTE



EMPRESAS CERTIFICADAS CON CMMI MUNDIALMENTE



EN 2007 EL PROCESO DE EVALUACIÓN INCLUYÓ LA EVALUACIÓN DE LOS PROYECTOS DE IBM EN APPLICATION SERVICES MOSTRANDO EL CONTINUO LIDERAZGO Y ÉXITO DE IBM EN LA ENTREGA DE SERVICIOS SUPERIORES A SUS CLIENTES A NIVEL MUNDIAL

IBM® INDIA®



MÉTODO DE DETERMINAR EL NIVEL DE MADURACIÓN

Nivel de Madurez	Idea descriptiva	Grupo	Individuo	Concepto Clave
Inicial	El Sistema triunfa por genialidades	El Equipo de Fútbol de Llanero al Real Madrid	El Empleado del Mes	Los Super Héroes, los Centuriones
Gestionado	El éxito se basa en el cumplimiento de las órdenes (cadena de mando)	La armada, el ejército	El Soldado de Infantería	Los Generales del ejército
Definido	El cumplimiento de las reglas y procedimientos	La Burocracia Funcional, las dictaduras ilustradas	El Burócrata Ilustrado	Gestión por Funciones
Gestión Cuantitativa	La organización Científica, basada en el uso del Método)	El Reloj Suizo, jerárquico mecaniza	El Gestor de Proceso	Gestión por Procesos, no por funciones
Optimizando	El sistema sociocultural que se adapta y evoluciona en su contexto, como el cuerpo humano	Holocracia, equipos empoderados	El Gestor de Valor	Organismo complejo y biológico (ser vivo)



G R A C I A S

Línea de atención al ciudadano: 018000 910270
Línea de atención al empresario: 018000 910682

@SENAcomunica

www.sena.edu.co