



# **UNIDAD DE TRABAJO 1**

# **TEORÍA DE SISTEMAS OPERATIVOS**



1

# INTRODUCCIÓN

# INTRODUCCIÓN

La primera pregunta que nos debemos hacer es, ¿qué es un ordenador?



# DEFINICIÓN DE ORDENADOR

## Definición:

Máquina electrónica capaz de almacenar información y tratarla automáticamente mediante operaciones matemáticas y lógicas controladas por programas informáticos

Analicemos en detalle esta definición....

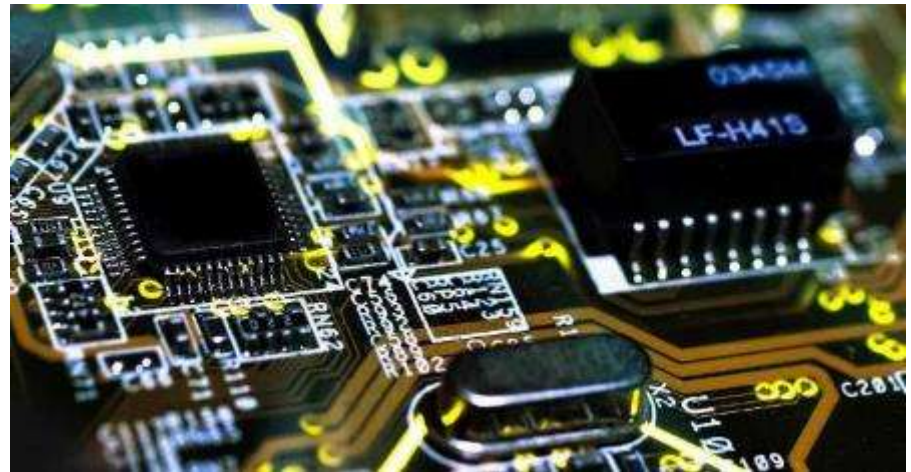


# DEFINICIÓN DE ORDENADOR

**Máquina electrónica** capaz de almacenar información y tratarla automáticamente mediante operaciones matemáticas y lógicas controladas por programas informáticos

Hace referencia al **hardware** del ordenador (componentes físicos).

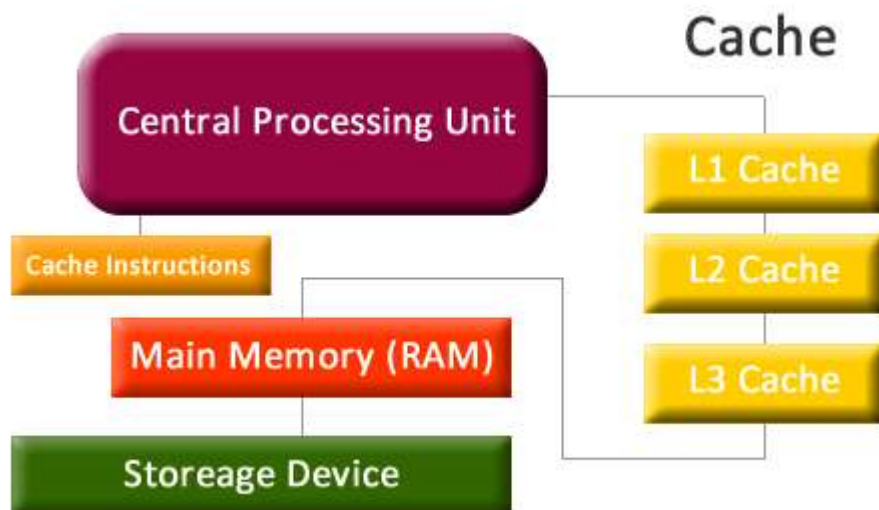
Debe funcionar mediante medios electrónicos.



# DEFINICIÓN DE ORDENADOR

Máquina electrónica **capaz de almacenar información** y tratarla automáticamente mediante operaciones matemáticas y lógicas controladas por programas informáticos

Debe disponer de una memoria que le permita almacenar los datos con los que trabaje.

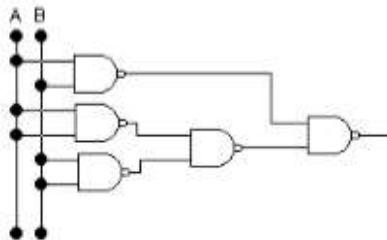


La memoria puede tener muchas formas: registros, RAM, discos duros, CDs, ....

# DEFINICIÓN DE ORDENADOR

Máquina electrónica capaz de almacenar información y **tratarla automáticamente mediante operaciones matemáticas y lógicas** controladas por programas informáticos

Al nivel más bajo el ordenador solo realiza operaciones lógicas (**Álgebra de Boole**) que se enmascaran con sucesivas **abstracciones**.



```

.DOSSEG
.MODEL SMALL
.STACK 1024

Two EQU 2
.DATA
VarB DB ?
VarW DW 1010b
VarW2 DW 257
VarD DD 0AFFFFh
S DB "Hello I",0
.CODE
main: MOV AX,DGROUP
      MOV DS,AX
      MOV [VarB],42
      MOV [VarD],-7
      MOV BX,Offset[S]
      MOV AX,[VarW]
  
```

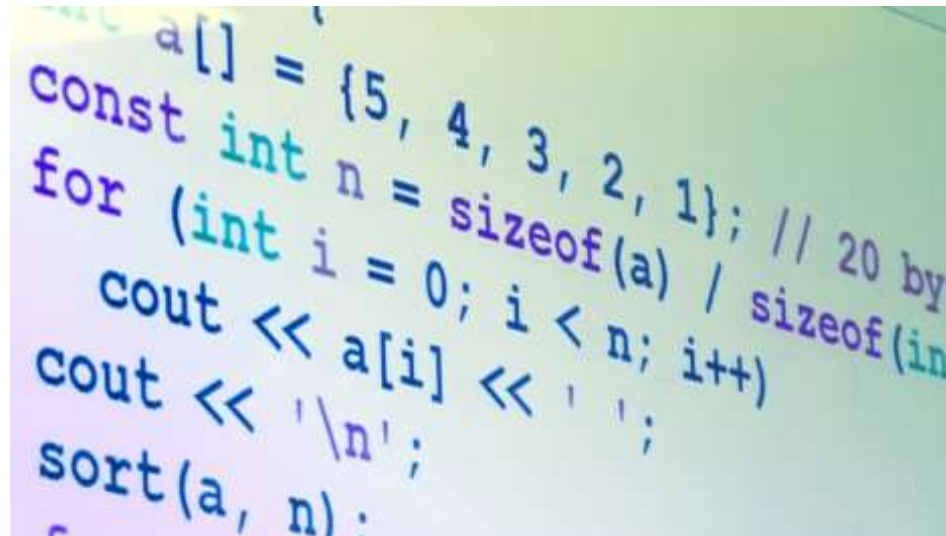
```

25
26 import java.util.*;
27 import java.lang.*;
28 import java.util.Random;
29
30 class Rextester
31 {
32     public static void main(String args[])
33     {
34         // l: lanzamientos; m: fr. muestreo
35         int l = 10000000, m = 5000;
36         // f: favorables; b=0; C; b=1; X
37         int f = 0, b = 0, r, n = 0;
38         double fr; // fr: frec. rel
39
40         Random g = new Random();
41         for(int i=0;i<(l/m);i++) {
42             for(int j=0;j<m;j++) {
43                 r = g.nextInt(2);
44                 if(r == b) f++;
45             }
46             n+=m; fr = (double)f/n;
47             System.out.println(f + "/" + n +
48
49
  
```

# DEFINICIÓN DE ORDENADOR

Máquina electrónica capaz de almacenar información y tratarla automáticamente mediante operaciones matemáticas y lógicas **controladas por programas informáticos**

Lo que diferencia a un ordenador de otras máquinas es que es **programable**, es decir, puede realizar diferentes tareas.



```
int a[] = {5, 4, 3, 2, 1}; // 20 bytes
const int n = sizeof(a) / sizeof(int);
for (int i = 0; i < n; i++)
    cout << a[i] << ' ';
cout << '\n';
sort(a, n);
```



# DEFINICIÓN DE SISTEMA OPERATIVO

## ¿Qué es un sistema operativo?

Es un programa o conjunto de programas que actúa como **intermediario entre el usuario y el hardware** del ordenador, gestionando los recursos y optimizando su uso.

Además, proporciona al usuario una **abstracción** de la máquina subyacente más fácil de manejar y programar.

# FUNCIONES DEL SISTEMA OPERATIVO

Las funciones más importantes de un sistema operativo son:

- Extender la máquina
- Administrar los recursos

# EL S.O. COMO EXTENSIÓN DE LA MÁQUINA

Si vemos un ordenador a nivel de lenguaje máquina, su **arquitectura** (*conjunto de instrucciones, organización de memoria, E/S y estructura de bus*) es primitiva y complicada de programar.

```

0040764E . 8945 F4      MOV DWORD PTR SS:[LOCAL.3],EAX
00407651 . 837D F4 00    CMP DWORD PTR SS:[LOCAL.3],0
00407655 . 74 16         JE SHORT 0040766D
00407657 . 8B4D F4      MOV ECX,DWORD PTR SS:[LOCAL.3]
0040765A . 33D2         XOR EDX,EDX
0040765C . 8A11         MOV DL,BYTE PTR DS:[ECX]
0040765E . 85D2         TEST EDX,EDX
00407660 . 74 07         JE SHORT 00407669
00407662 . B8 01000000  MOV EAX,1
00407667 . EB 47         JMP SHORT 00407680
00407669 > 33C0         XOR EAX,EAX
0040766B . EB 43         JMP SHORT 00407680
0040766D > A1 14AE4200  MOV EAX,DWORD PTR DS:[42AE14]
00407672 . 25 00800000  AND EAX,00008000
00407677 . 85C0         TEST EAX,EAX
00407679 . 74 07         JE SHORT 00407680

```

# EL S.O. COMO EXTENSIÓN DE LA MÁQUINA

Por ejemplo, las operaciones de lectura y escritura de un **disco duro a bajo nivel** requiere operaciones como:

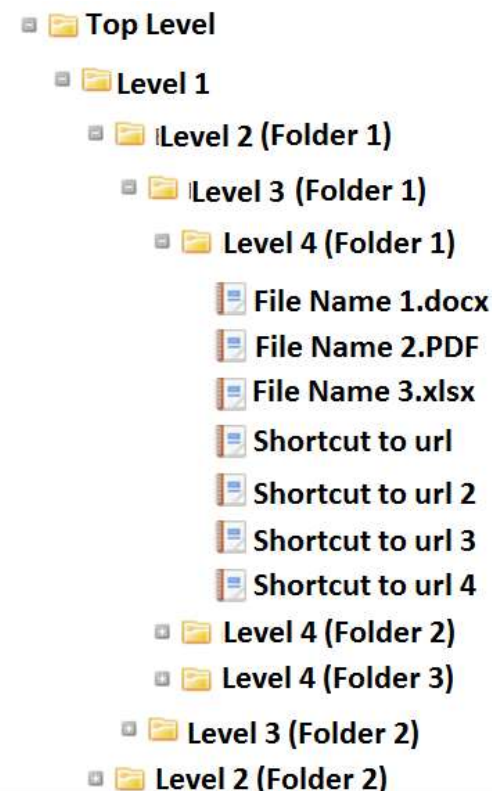
- Estructura lógica del disco
- Estado del motor
- Modo de grabación empleado
- Etc...



# EL S.O. COMO EXTENSIÓN DE LA MÁQUINA

En este caso una **abstracción** típica sería:

- El disco contiene una colección de archivos con nombre
- Cada archivo puede abrirse para lectura/escritura, operar con él y luego cerrarse
- Los detalles internos (*si la grabación tiene algún tipo de comprobación de errores o si el motor está encendido o apagado*) no deberán aparecer en la abstracción.





# EL S.O. COMO EXTENSIÓN DE LA MÁQUINA

El sistema operativo es quien muestra esta abstracción al usuario.

Se extiende a todos los aspectos complicados de la máquina: interrupciones, temporizadores, administración de memoria, ....

Sirve para facilitar el trabajo del usuario con la máquina.

# EL S.O. COMO EXTENSIÓN DE LA MÁQUINA

Con el tiempo las abstracciones se han ido haciendo cada vez más accesibles para el usuario:

```

Enter today's date (m-d-y): 08-04-81

The IBM Personal Computer DOS
Version 1.00 (C)Copyright IBM Corp 1981

A>dir *.com

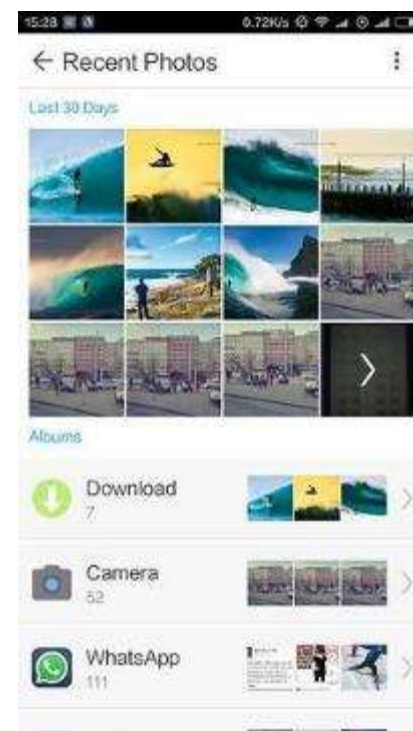
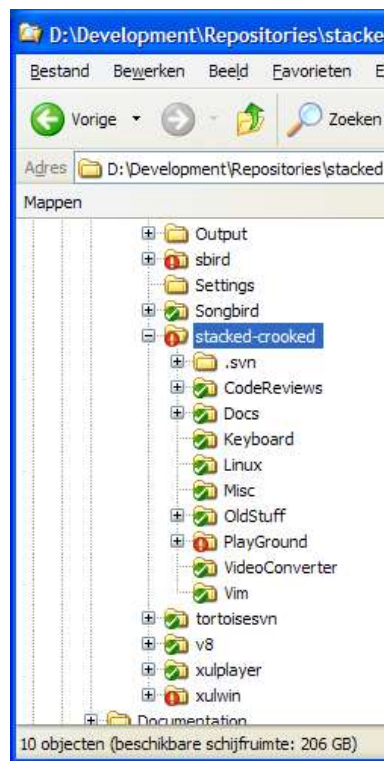
```

File Name	Attributes	Size	Date	Time
IBMBIO	COM	1920	07-23-81	
IBMDOS	COM	6400	08-13-81	
COMMAND	COM	3231	08-04-81	
FORMAT	COM	2560	08-04-81	
CHKDSK	COM	1395	08-04-81	
SYS	COM	896	08-04-81	
DISKCOPY	COM	1216	08-04-81	
DISKCOMP	COM	1124	08-04-81	
COMP	COM	1620	08-04-81	
DATE	COM	252	08-04-81	
TIME	COM	250	08-04-81	
MODE	COM	860	08-04-81	
EDLIN	COM	2392	08-04-81	
DEBUG	COM	6049	08-04-81	
BASIC	COM	10880	08-04-81	
BASICA	COM	16256	08-04-81	

```

A>_

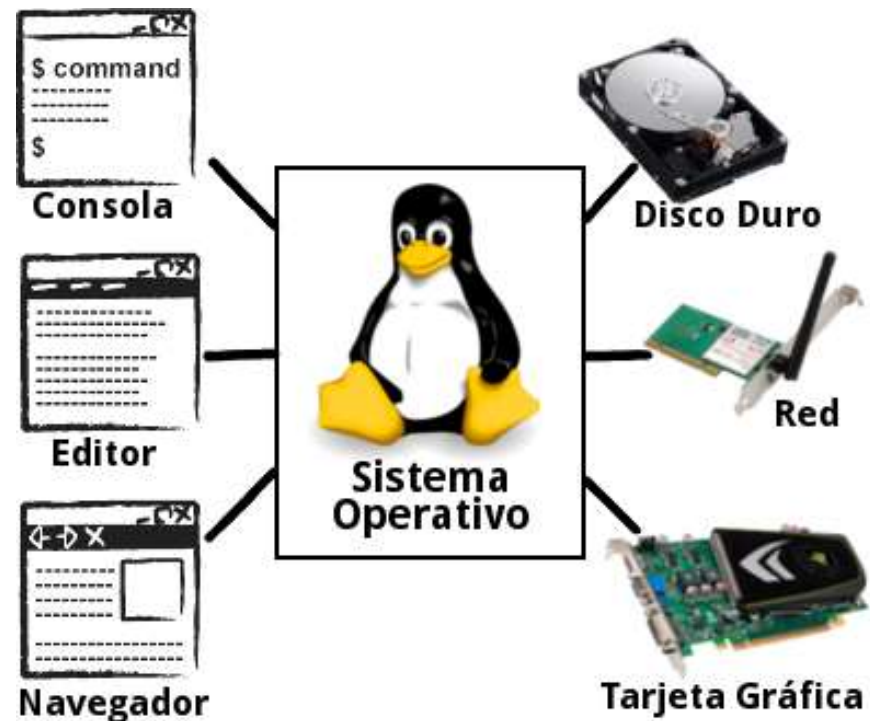
```



# EL S.O. COMO GESTOR DE RECURSOS

Los ordenadores constan de procesadores, memorias, temporizadores, discos, ratones, interfaces de red, ...

Cada programa y usuario necesita acceder a esos recursos y **deben repartírselos.**



# EL S.O. COMO GESTOR DE RECURSOS

Es tarea del sistema operativo **efectuar un reparto ordenado y controlado de todos los recursos** entre los diferentes programas que compiten por obtenerlos.

Algunas funciones en este aspecto son:

- Saber quién está usando qué recurso
- Conceder solicitudes de recursos
- Dar cuenta del uso de recursos
- Mediar entre solicitudes de diferentes programas y usuarios en conflicto
- Evitar que un usuario o programa interfiera en un recurso adjudicado a otro programa.

# OBJETIVOS DEL SISTEMA OPERATIVO

La realización de ambas funciones (extensión de la máquina y gestión de los recursos) se hace teniendo en cuenta tres objetivos:

- Comodidad
- Eficiencia
- Capacidad de evolución



# OBJETIVOS DEL SISTEMA OPERATIVO: COMODIDAD

## COMODIDAD

La máquina debe mostrarse de una forma que sea familiar para el usuario.

Para ello hace uso de abstracciones, que enmascaran las interioridades de la máquina.

# OBJETIVOS DEL SISTEMA OPERATIVO: EFICIENCIA

## EFICIENCIA

El SO no solamente tiene que gestionar los recursos, sino que debe hacerlo de forma eficiente.

Los recursos son un bien limitado y en un sistema hay múltiples procesos pugnando por ellos.

Un reparto eficiente garantizará un alto aprovechamiento de los mismos.

# OBJETIVOS DEL SISTEMA OPERATIVO: EFICIENCIA

Según el tipo de recurso, el reparto puede ser de dos formas:

- **Multiplexaje en el tiempo:**
  - Hay recursos que únicamente pueden ser utilizados por un único proceso en un instante determinado.
  - Ejemplos: procesador, conexión de red,...
  - El sistema operativo decide qué proceso puede utilizarlo en cada instante.
  - Normalmente alterna tan rápidamente entre procesos que da sensación de simultaneidad.

# OBJETIVOS DEL SISTEMA OPERATIVO: EFICIENCIA

- **Multiplexaje en el espacio:**
  - En este caso, varios procesos pueden usar simultáneamente el proceso, por lo que se debe dividir en partes.
  - Ejemplos: memoria RAM, disco duro,...
  - El sistema operativo se encargará
    - Asignar una parte a cada proceso
    - Asegurarse de que un proceso no pueda acceder al espacio asignado a otro proceso.

# OBJETIVOS DEL SISTEMA OPERATIVO: EVOLUCIÓN

## FACILIDAD DE EVOLUCIÓN

El sistema operativo debe estar en continua evolución:

- **Actualizaciones de hardware y nuevos tipos de hardware:** tiene que ser capaz de reconocer nuevos tipos de hardware e integrarlos en el sistema.

Esto lo consigue mediante los drivers o controladores.

- **Nuevos servicios:** los sistemas operativos están continuamente añadiendo nuevas funcionalidades.

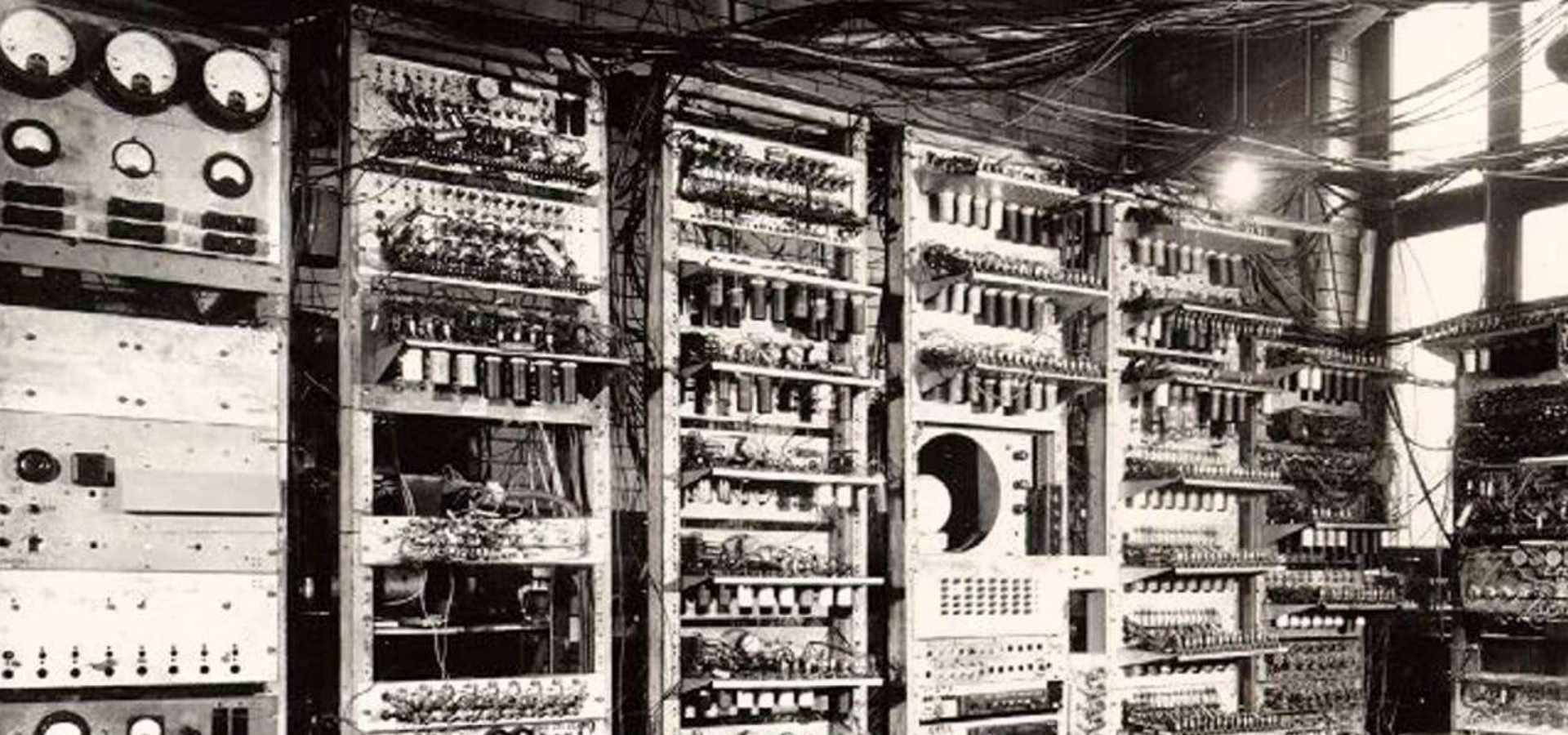


# OBJETIVOS DEL SISTEMA OPERATIVO: EVOLUCIÓN

- **Correcciones:** un SO es un programa con millones de líneas de código, por lo que invariablemente tendrá fallos y agujeros de seguridad.

Estos ***bugs*** se solucionarán a medida que se descubran mediante **parches** y **actualizaciones**.





# 2 HISTORIA DE LOS SISTEMAS OPERATIVOS

# HISTORIA DE LOS ORDENADORES

En la **historia de los ordenadores** podemos hablar de generaciones.

- **Era pre-informática** (hasta 1945): primeras máquinas de calcular
- **Primera generación** (1945-1955): Tubos de vacío y tableros
- **Segunda generación** (1955-1965): Transistores y sistemas por lotes
- **Tercera generación** (1965-1980): Circuitos integrados y multiprogramación
- **Cuarta generación** (1980-1995): Computadoras personales
- **Quinta generación** (1995-hoy): Redes e inteligencia artificial

# PRE-INFORMÁTICA (-1940)

## Ábaco

Aprox. siglo XX aC

- Instrumento que sirve para efectuar operaciones aritméticas sencillas (sumas, restas y multiplicaciones).
- Es considerado el precursor de la calculadora digital moderna.





# PRE-INFORMÁTICA (-1940)

## Ábaco neperiano (John Napier)

1617

- Tablero con reborde en el que se colocarán las varillas neperianas (tiras de madera, metal o cartón grueso) para realizar las operaciones de multiplicación, división o raíz cuadrada.





# PRE-INFORMÁTICA (-1940)

## Pascalina (Blaise Pascal)

1642

- Primera calculadora que funcionaba a base de ruedas y engranajes.
- Las ruedas representaban el sistema decimal de numeración.
- Fue diseñado por Pascal a los 19 años de edad.

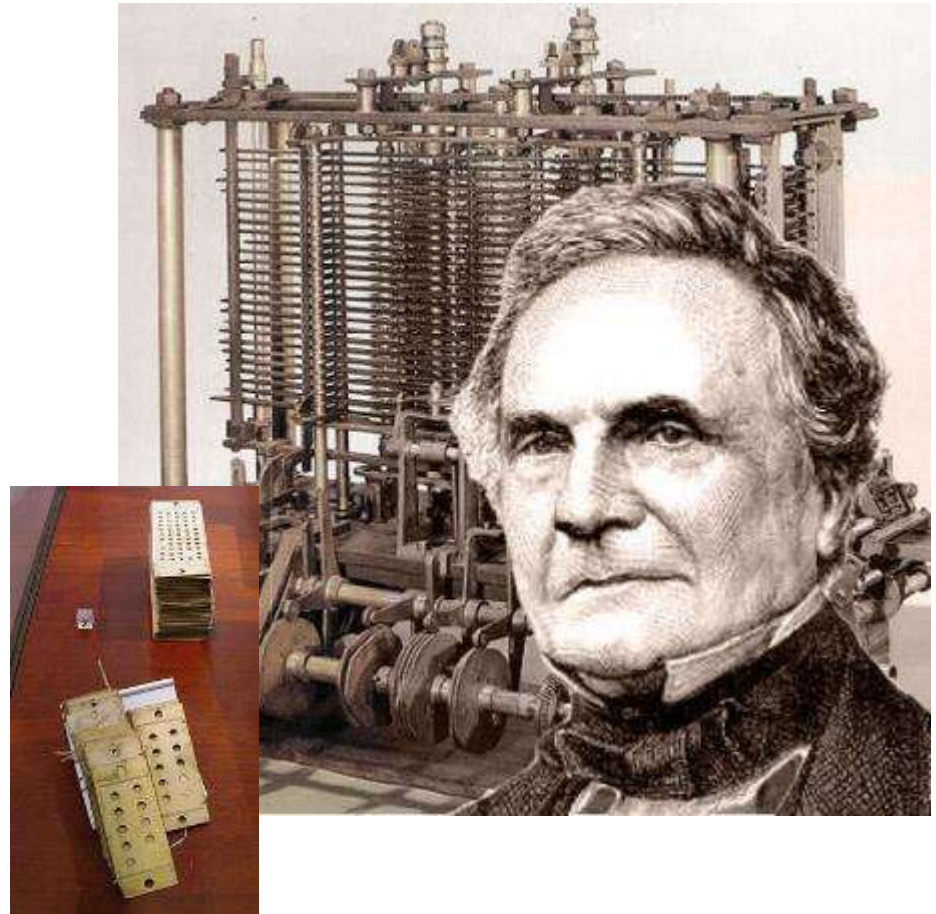


# PRE-INFORMÁTICA (-1940)

## Máquina analítica (Charles Babbage)

1642

- Se puede considerar el primer **ordenador de propósito general** de la historia ya que contaba con **unidad aritmético lógica, unidad de control y memoria**. Los datos se introducían mediante tarjetas perforadas.



# PRE-INFORMÁTICA (-1940)

- Nunca llegó a fabricarse por diversos motivos:
  - El gobierno retiró los fondos para su construcción.
  - Babbage tenía un carácter inquieto que le hacía saltar de proyecto en proyecto.
  - La tecnología de la época era incapaz de construir las piezas con la precisión requerida para el proyecto.

# PRE-INFORMÁTICA (-1940)

- Algunas características:
  - Funcionaba a vapor y mediría 30x10 m.
  - Trabajaba en base 10
  - La memoria podía almacenar 1.000 números de 40 dígitos (16.2 KB)
  - La ALU podía realizar operaciones aritméticas, comparaciones y raíces cuadradas.

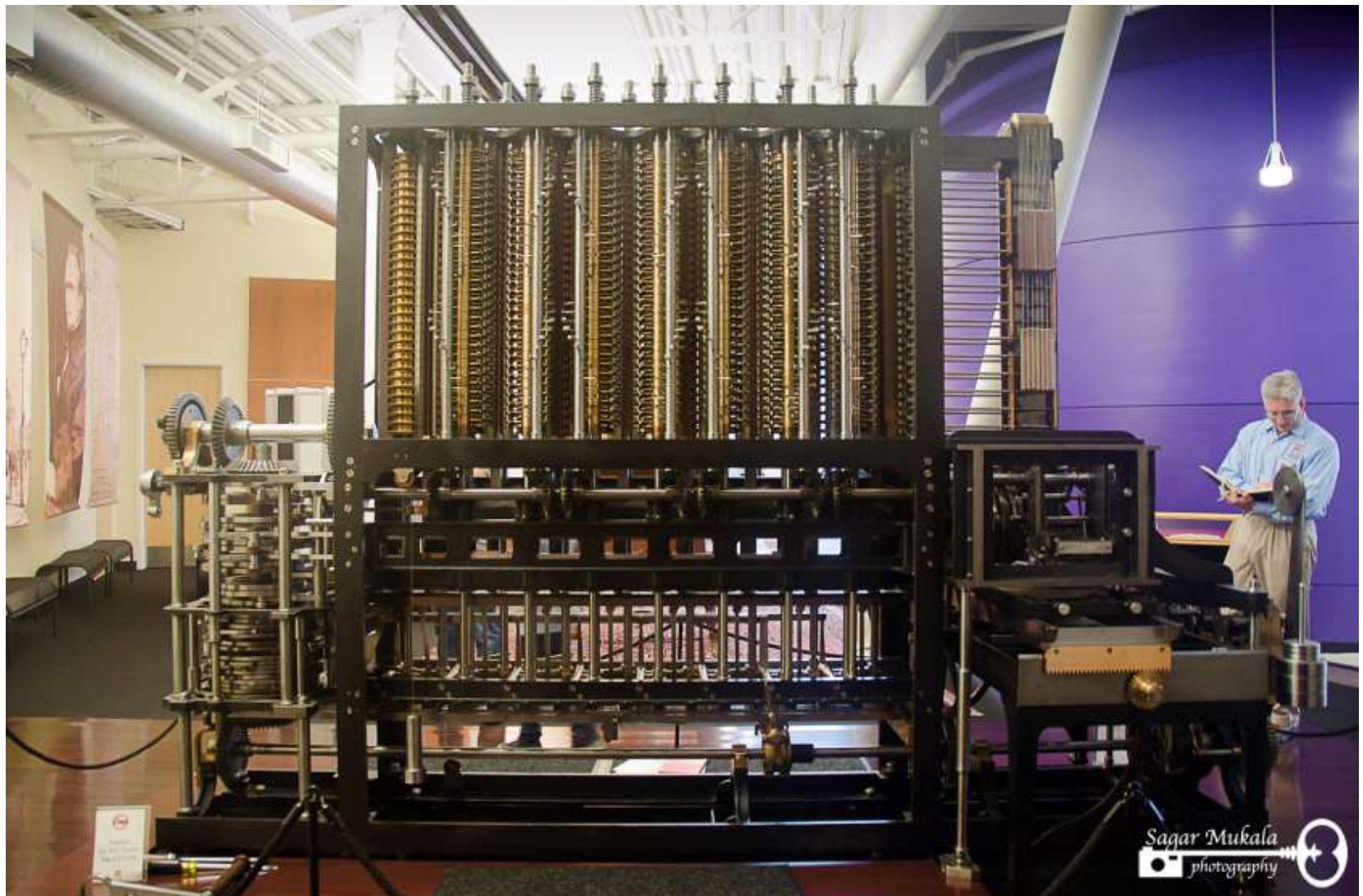
# PRE-INFORMÁTICA (-1940)

- Aunque la máquina nunca llegó a fabricarse, el hijo de Babbage, *Henry Prevost Babbage*, construyó en 1910 una parte de la ALU (*mill*) que actualmente se puede ver en el Museo de Ciencias de Londres.
- Allí también se puede ver un modelo funcional de la **máquina diferencial de Babbage**.





# PRE-INFORMÁTICA (-1940)



# PRE-INFORMÁTICA (-1940)

- Babbage fue ayudado por **Ada King, Condesa de Lovelace** e hija del poeta inglés Lord Byron.
- Las notas realizadas por Ada Lovelace se consideran el software de la máquina analítica.





# PRE-INFORMÁTICA (-1940)

- En una nota Ada describe un algoritmo para la máquina analítica para calcular los números de Bernuilli (en el que utilizaba dos bucles) y está considerado el primer algoritmo específicamente diseñado para ser ejecutado por un ordenador.
- Por ello **Ada es considerada la primera programadora de la historia.**

Diagram for the computation by the Engines of the Numbers of Bernoulli. See Note G. (page 777 of seq.)

Diagram for the computation by the Engines of the Numbers of Bernoulli. See Note G. (page 777 of seq.)

Number of Operations: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100, 101, 102, 103, 104, 105, 106, 107, 108, 109, 110, 111, 112, 113, 114, 115, 116, 117, 118, 119, 120, 121, 122, 123, 124, 125, 126, 127, 128, 129, 130, 131, 132, 133, 134, 135, 136, 137, 138, 139, 140, 141, 142, 143, 144, 145, 146, 147, 148, 149, 150, 151, 152, 153, 154, 155, 156, 157, 158, 159, 160, 161, 162, 163, 164, 165, 166, 167, 168, 169, 170, 171, 172, 173, 174, 175, 176, 177, 178, 179, 180, 181, 182, 183, 184, 185, 186, 187, 188, 189, 190, 191, 192, 193, 194, 195, 196, 197, 198, 199, 200, 201, 202, 203, 204, 205, 206, 207, 208, 209, 210, 211, 212, 213, 214, 215, 216, 217, 218, 219, 220, 221, 222, 223, 224, 225, 226, 227, 228, 229, 230, 231, 232, 233, 234, 235, 236, 237, 238, 239, 240, 241, 242, 243, 244, 245, 246, 247, 248, 249, 250, 251, 252, 253, 254, 255, 256, 257, 258, 259, 260, 261, 262, 263, 264, 265, 266, 267, 268, 269, 270, 271, 272, 273, 274, 275, 276, 277, 278, 279, 280, 281, 282, 283, 284, 285, 286, 287, 288, 289, 290, 291, 292, 293, 294, 295, 296, 297, 298, 299, 300, 301, 302, 303, 304, 305, 306, 307, 308, 309, 310, 311, 312, 313, 314, 315, 316, 317, 318, 319, 320, 321, 322, 323, 324, 325, 326, 327, 328, 329, 330, 331, 332, 333, 334, 335, 336, 337, 338, 339, 340, 341, 342, 343, 344, 345, 346, 347, 348, 349, 350, 351, 352, 353, 354, 355, 356, 357, 358, 359, 360, 361, 362, 363, 364, 365, 366, 367, 368, 369, 370, 371, 372, 373, 374, 375, 376, 377, 378, 379, 380, 381, 382, 383, 384, 385, 386, 387, 388, 389, 390, 391, 392, 393, 394, 395, 396, 397, 398, 399, 400, 401, 402, 403, 404, 405, 406, 407, 408, 409, 410, 411, 412, 413, 414, 415, 416, 417, 418, 419, 420, 421, 422, 423, 424, 425, 426, 427, 428, 429, 430, 431, 432, 433, 434, 435, 436, 437, 438, 439, 440, 441, 442, 443, 444, 445, 446, 447, 448, 449, 450, 451, 452, 453, 454, 455, 456, 457, 458, 459, 460, 461, 462, 463, 464, 465, 466, 467, 468, 469, 470, 471, 472, 473, 474, 475, 476, 477, 478, 479, 480, 481, 482, 483, 484, 485, 486, 487, 488, 489, 490, 491, 492, 493, 494, 495, 496, 497, 498, 499, 500, 501, 502, 503, 504, 505, 506, 507, 508, 509, 510, 511, 512, 513, 514, 515, 516, 517, 518, 519, 520, 521, 522, 523, 524, 525, 526, 527, 528, 529, 530, 531, 532, 533, 534, 535, 536, 537, 538, 539, 540, 541, 542, 543, 544, 545, 546, 547, 548, 549, 550, 551, 552, 553, 554, 555, 556, 557, 558, 559, 560, 561, 562, 563, 564, 565, 566, 567, 568, 569, 570, 571, 572, 573, 574, 575, 576, 577, 578, 579, 580, 581, 582, 583, 584, 585, 586, 587, 588, 589, 590, 591, 592, 593, 594, 595, 596, 597, 598, 599, 600, 601, 602, 603, 604, 605, 606, 607, 608, 609, 610, 611, 612, 613, 614, 615, 616, 617, 618, 619, 620, 621, 622, 623, 624, 625, 626, 627, 628, 629, 630, 631, 632, 633, 634, 635, 636, 637, 638, 639, 640, 641, 642, 643, 644, 645, 646, 647, 648, 649, 650, 651, 652, 653, 654, 655, 656, 657, 658, 659, 660, 661, 662, 663, 664, 665, 666, 667, 668, 669, 670, 671, 672, 673, 674, 675, 676, 677, 678, 679, 680, 681, 682, 683, 684, 685, 686, 687, 688, 689, 690, 691, 692, 693, 694, 695, 696, 697, 698, 699, 700, 701, 702, 703, 704, 705, 706, 707, 708, 709, 710, 711, 712, 713, 714, 715, 716, 717, 718, 719, 720, 721, 722, 723, 724, 725, 726, 727, 728, 729, 730, 731, 732, 733, 734, 735, 736, 737, 738, 739, 740, 741, 742, 743, 744, 745, 746, 747, 748, 749, 750, 751, 752, 753, 754, 755, 756, 757, 758, 759, 760, 761, 762, 763, 764, 765, 766, 767, 768, 769, 770, 771, 772, 773, 774, 775, 776, 777, 778, 779, 780, 781, 782, 783, 784, 785, 786, 787, 788, 789, 790, 791, 792, 793, 794, 795, 796, 797, 798, 799, 800, 801, 802, 803, 804, 805, 806, 807, 808, 809, 810, 811, 812, 813, 814, 815, 816, 817, 818, 819, 820, 821, 822, 823, 824, 825, 826, 827, 828, 829, 830, 831, 832, 833, 834, 835, 836, 837, 838, 839, 840, 841, 842, 843, 844, 845, 846, 847, 848, 849, 850, 851, 852, 853, 854, 855, 856, 857, 858, 859, 860, 861, 862, 863, 864, 865, 866, 867, 868, 869, 870, 871, 872, 873, 874, 875, 876, 877, 878, 879, 880, 881, 882, 883, 884, 885, 886, 887, 888, 889, 890, 891, 892, 893, 894, 895, 896, 897, 898, 899, 900, 901, 902, 903, 904, 905, 906, 907, 908, 909, 910, 911, 912, 913, 914, 915, 916, 917, 918, 919, 920, 921, 922, 923, 924, 925, 926, 927, 928, 929, 930, 931, 932, 933, 934, 935, 936, 937, 938, 939, 940, 941, 942, 943, 944, 945, 946, 947, 948, 949, 950, 951, 952, 953, 954, 955, 956, 957, 958, 959, 960, 961, 962, 963, 964, 965, 966, 967, 968, 969, 970, 971, 972, 973, 974, 975, 976, 977, 978, 979, 980, 981, 982, 983, 984, 985, 986, 987, 988, 989, 990, 991, 992, 993, 994, 995, 996, 997, 998, 999, 1000.

# PRE-INFORMÁTICA (-1940)

- Funcionamiento de la máquina de Babbage:
  - <https://www.youtube.com/watch?v=0anlyVGeWOI>

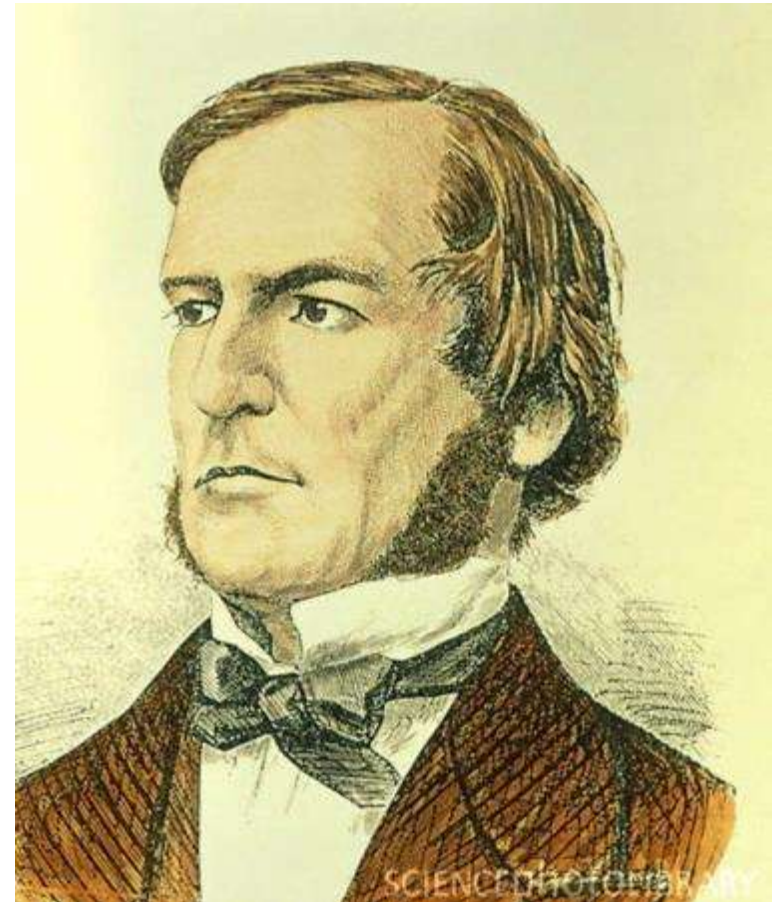


# PRE-INFORMÁTICA (-1940)

## Álgebra de Boole

1847

- En 1847 George Boole enunció el álgebra de Boole que define el modo de realizar cálculos y comparaciones en binario.
- Actualmente todos los ordenadores fundamentan su funcionamiento en el álgebra de Boole



# PRE-INFORMÁTICA (-1940)

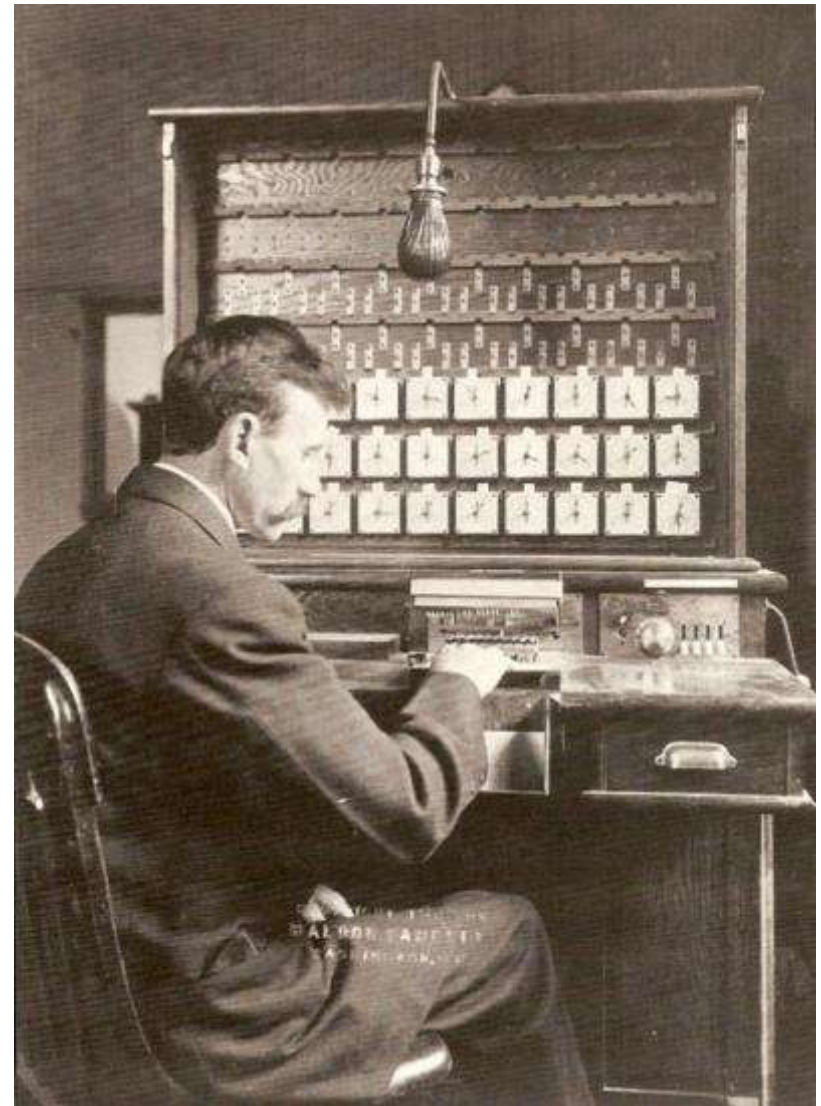
## Máquina tabuladora de Hollerith

1890

- El censo americano de 1880 había necesitado **7 años** de análisis.
- Para simplificar esa tarea Herman Hollerith construyó una máquina tabuladora o censadora.
- Las preguntas del censo se podían contestar con opciones binarias por lo que Hollerith utilizó tarjetas de 80 columnas con 2 posiciones.
- El censo de 1890, con 62.622.250 habitantes, se analizó en solo **6 semanas**.

# PRE-INFORMÁTICA (-1940)

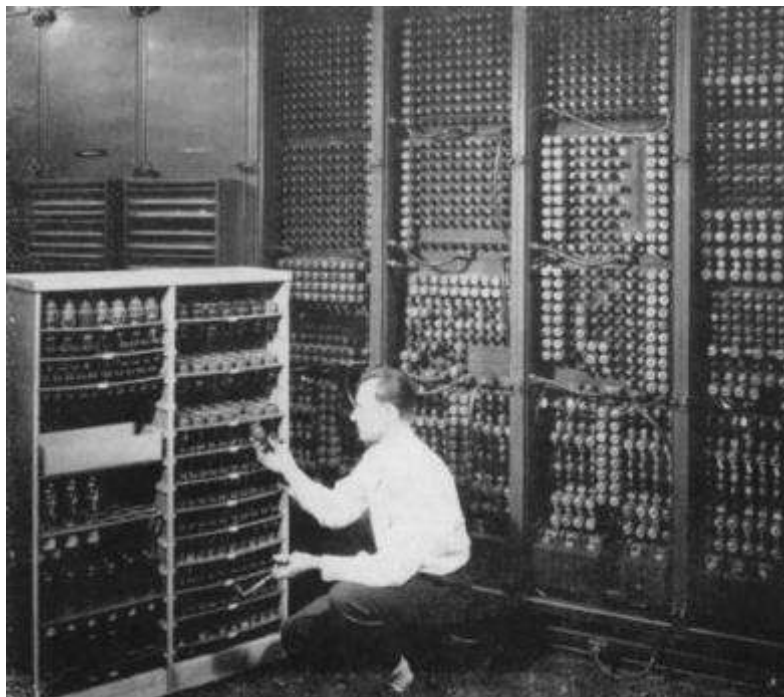
- Se considera el primer hombre en realizar un **tratamiento automático de la información**.
- Hollerith fundó en 1896 la empresa **Tabulating Machine Company** que tras fusionarse con otras dos empresas pasó a llamarse **International Bussiness Machines**, la actual IBM.





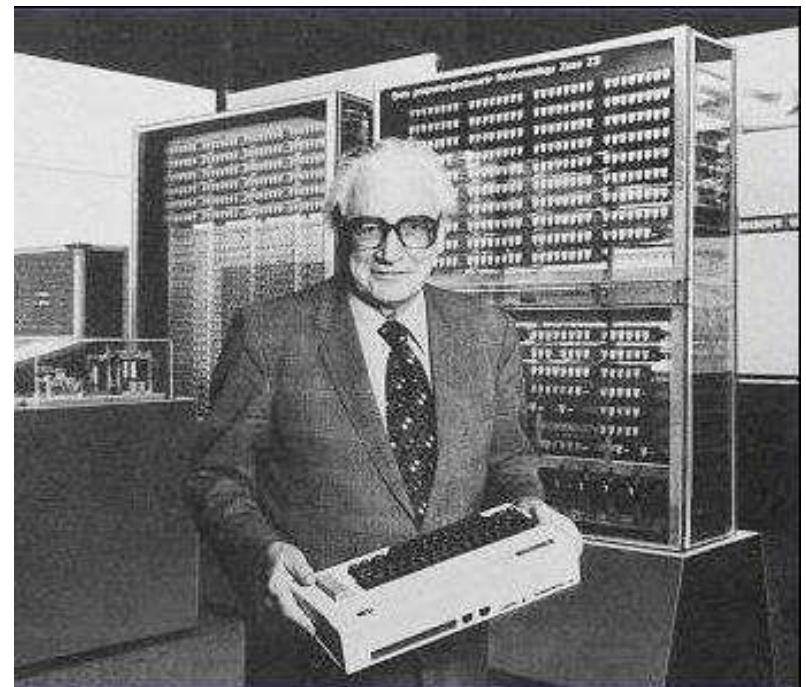
# PRIMERA GENERACIÓN (1940-1955)

A mediados de 1940 aparecen las primeras máquinas calculadoras.



Replacing a bad tube meant checking among ENIAC's 19,000 possibilities.

**ENIAC**

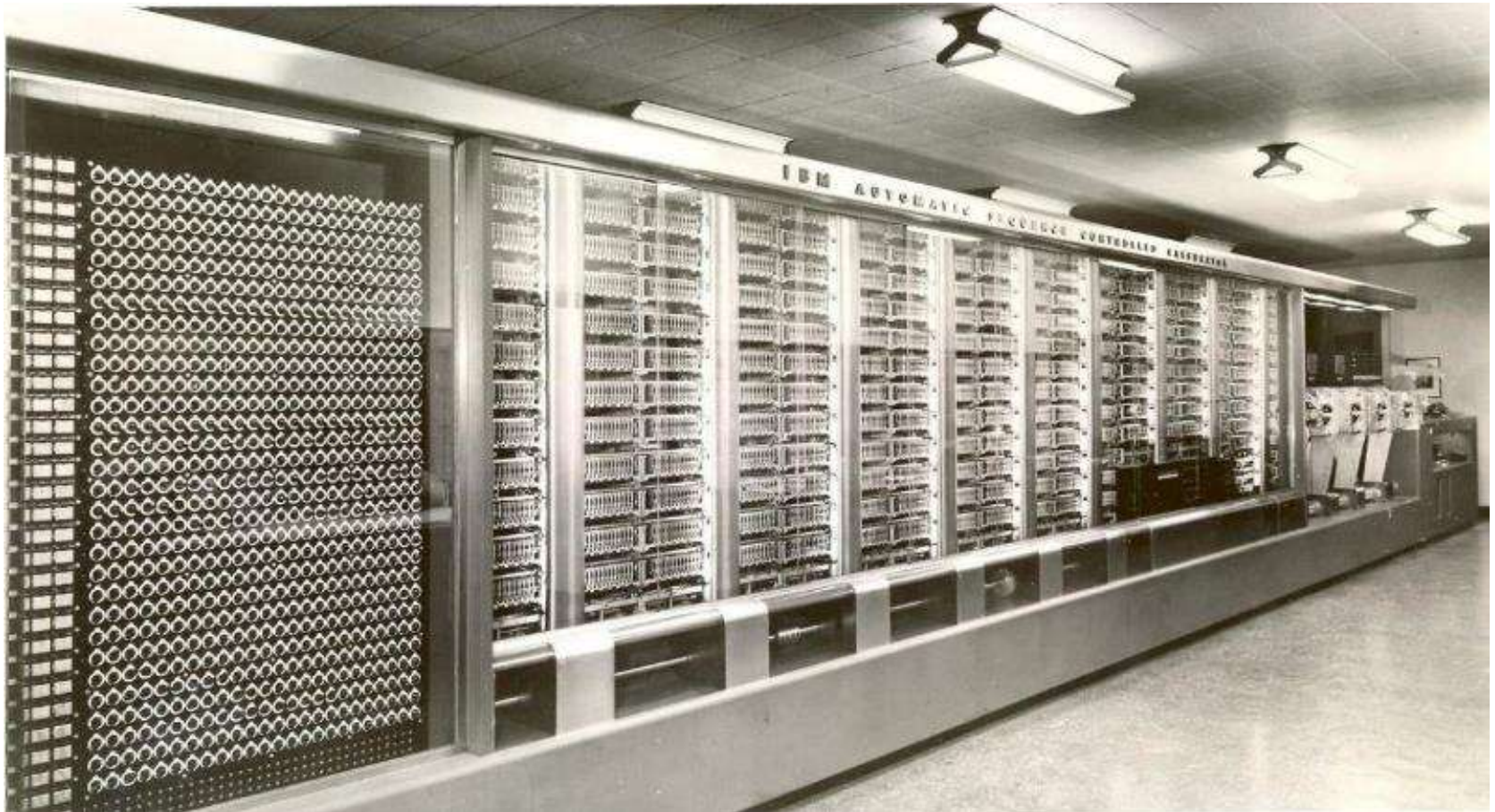


**Z1 (Konrad Zuse)**

# PRIMERA GENERACIÓN (1940-1955)

## HARVARD MARK 1

1944





# PRIMERA GENERACIÓN (1940-1955)

Como ejemplo, algunas características de la MARK I son:

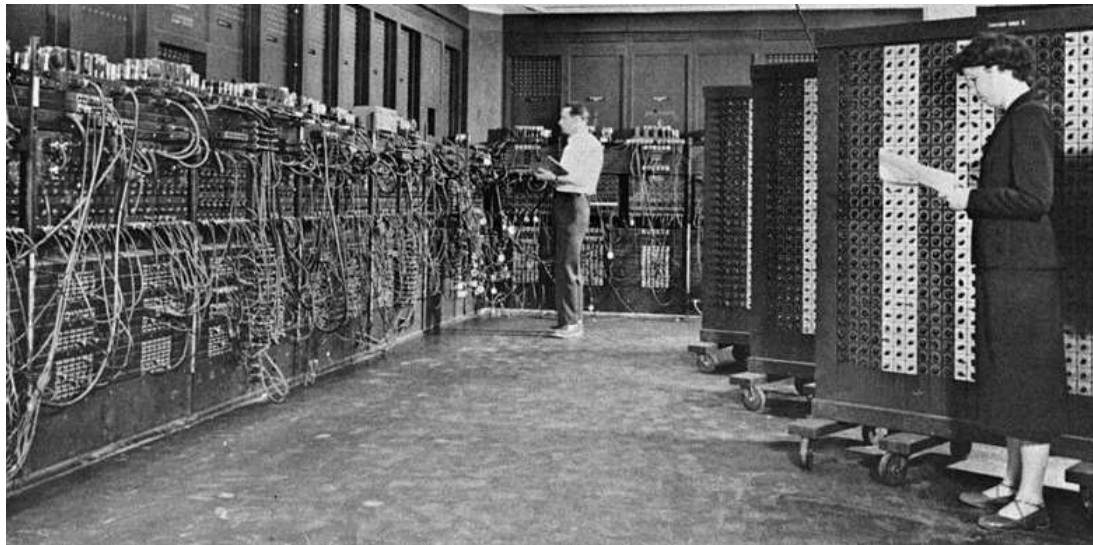
- Estaba basado en la máquina analítica de Babbage
- Construida con transmisores, interruptores, ejes de rotación y embragues. Enviaba señales electromagnéticas para mover partes mecánicas.
- Medía 16 m. de largo, 2,4 m. de alto y 61 cm de ancho.
- Disponía de 765.000 componentes electrónicos y 18 km de cables.
- Costó entre 250k y 500k dólares
- Funcionaba mediante un motor eléctrico de 4 kW de potencia

# PRIMERA GENERACIÓN (1940-1955)

## ENIAC

1946

- El **ENIAC** (*Electronical Numerical Integrator and Computer*) fue desarrollado por Eckert y Mauchly.
- Fue la primera computadora que utilizaba tubos de vacío.



# PRIMERA GENERACIÓN (1940-1955)

- Algunos datos interesantes son:
  - Ocupaba 167 m<sup>2</sup> y pesaba 27 Tm
  - Tenía 17.468 tubos de vacío
  - 7.200 diodos de cristal, 1.500 relés, 70.000 resistencias, 10.000 condensadores y 5 millones de soldaduras
  - Programarla requería cambiar, conectar y reconectar cables y podía llevar semanas
  - Aproximadamente cada hora se fundía un tubo de vacío, obligando a parar la máquina y reemplazarlo.
  - Internamente trabajaba en decimal

# PRIMERA GENERACIÓN (1940-1955)

Estas máquinas tenían componentes mecánicos y electrónicos.

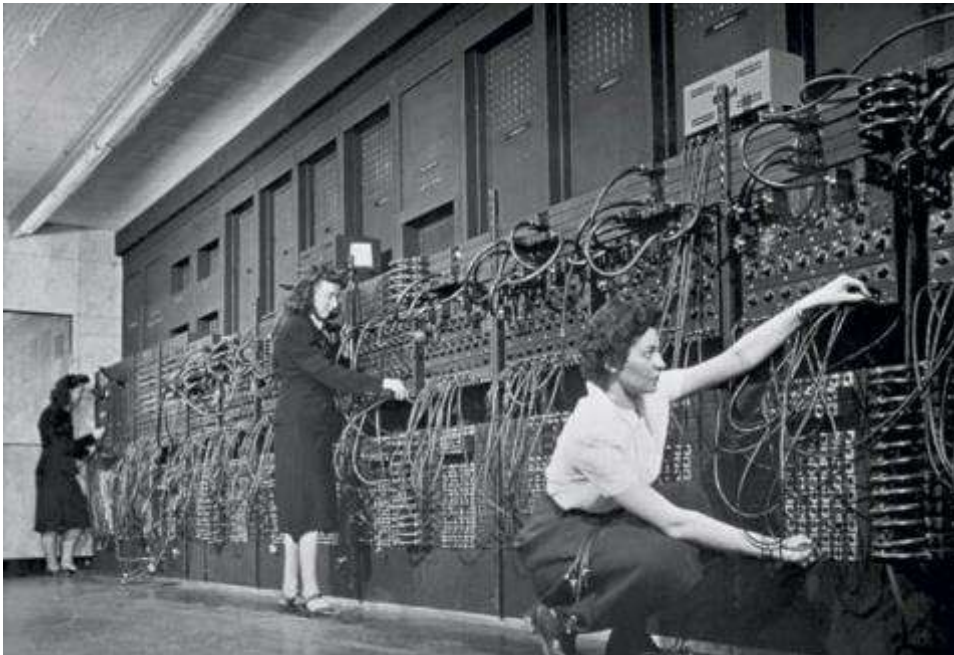
Al principio reveladores mecánicos y posteriormente válvulas de vacío.



# PRIMERA GENERACIÓN (1940-1955)

Un solo grupo de personas diseñaba, construía, programaba, operaba y mantenía cada máquina.

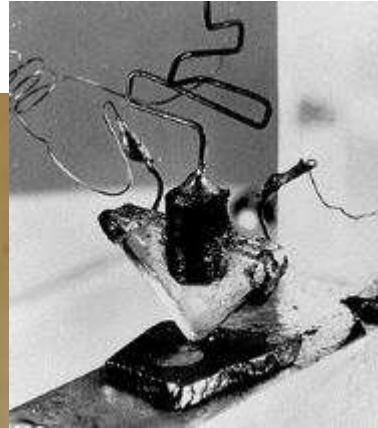
La programación se realizaba en lenguaje máquina absoluto, a menudo alambRANDO tableros.



**En esta generación  
no existían  
lenguajes de  
programación ni  
sistemas  
operativos.**

# SEGUNDA GENERACIÓN (1955-1965)

Aparece el **transistor** como sustituto del tubo de vacío.



Esto permite ordenadores más fiables y pequeños.

200 transistores ocupan lo mismo que un tubo de vacío.

Aparecen los primeros ordenadores **comerciales**



## SEGUNDA GENERACIÓN (1955-1965)

- Aparecen los primeros periféricos: tarjetas perforadas, cinta magnética e impresora.
- Los ordenadores empezaron a llamarse **mainframes** o macrocomputadoras.
- Los ordenadores se utilizaban para cálculos científicos y de ingeniería.



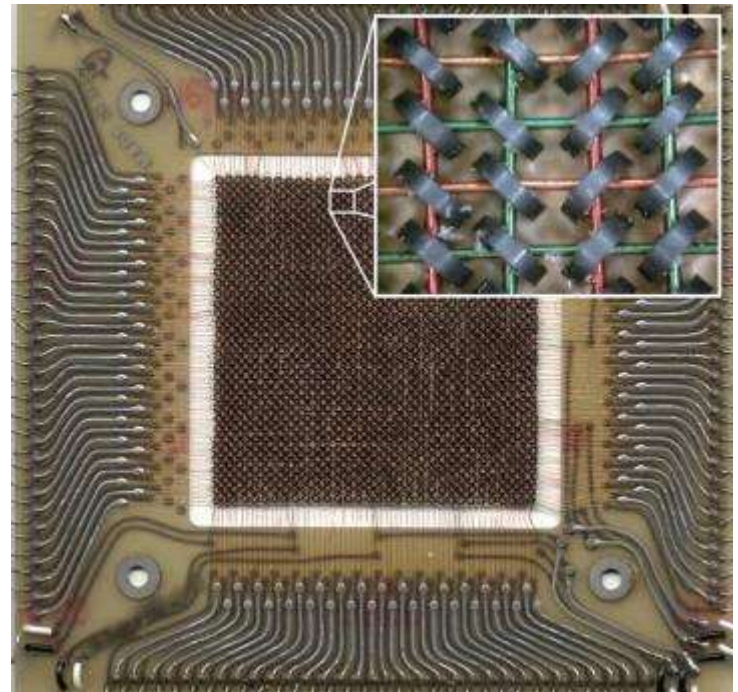


# SEGUNDA GENERACIÓN (1955-1965)

## IBM 7094

1958

- Comercializado por IBM.
- Vendió unas 300 unidades a un precio de 3 millones de dólares.
- Algunos datos:
  - 50.000 transistores
  - Memoria de núcleos de ferrita
  - 200K sumas, 40K multiplicaciones o 30K divisiones por segundo



# SEGUNDA GENERACIÓN (1955-1965)



# SEGUNDA GENERACIÓN (1955-1965)

El sistema operativo para este ordenador era el **IBSYS**.

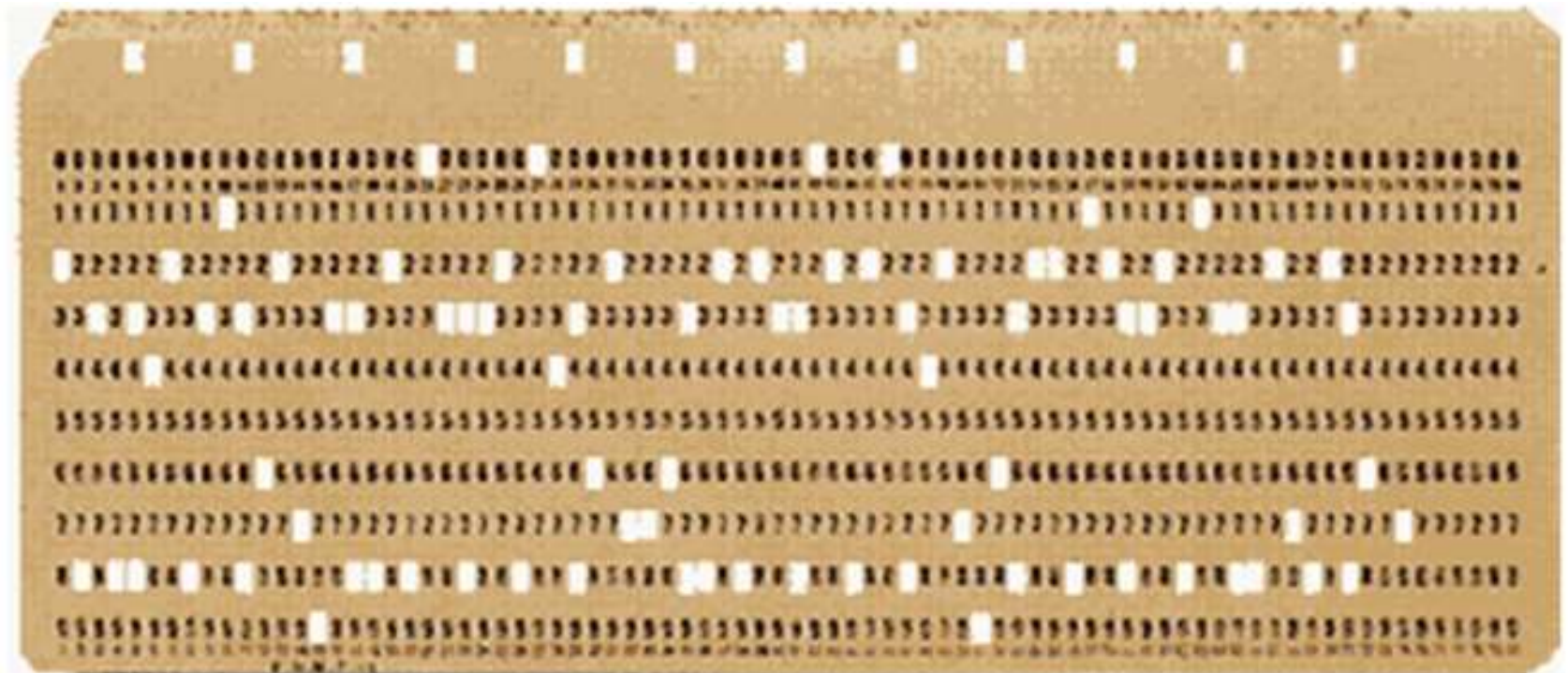
Sus componentes eran:

- Interfaz basada en tarjetas perforadas
- Compilador de FORTRAN y COBOL
- Un ensamblador
- Algunas utilidades





# SEGUNDA GENERACIÓN (1955-1965)



# SEGUNDA GENERACIÓN (1955-1965)

## IBM 7090

1958

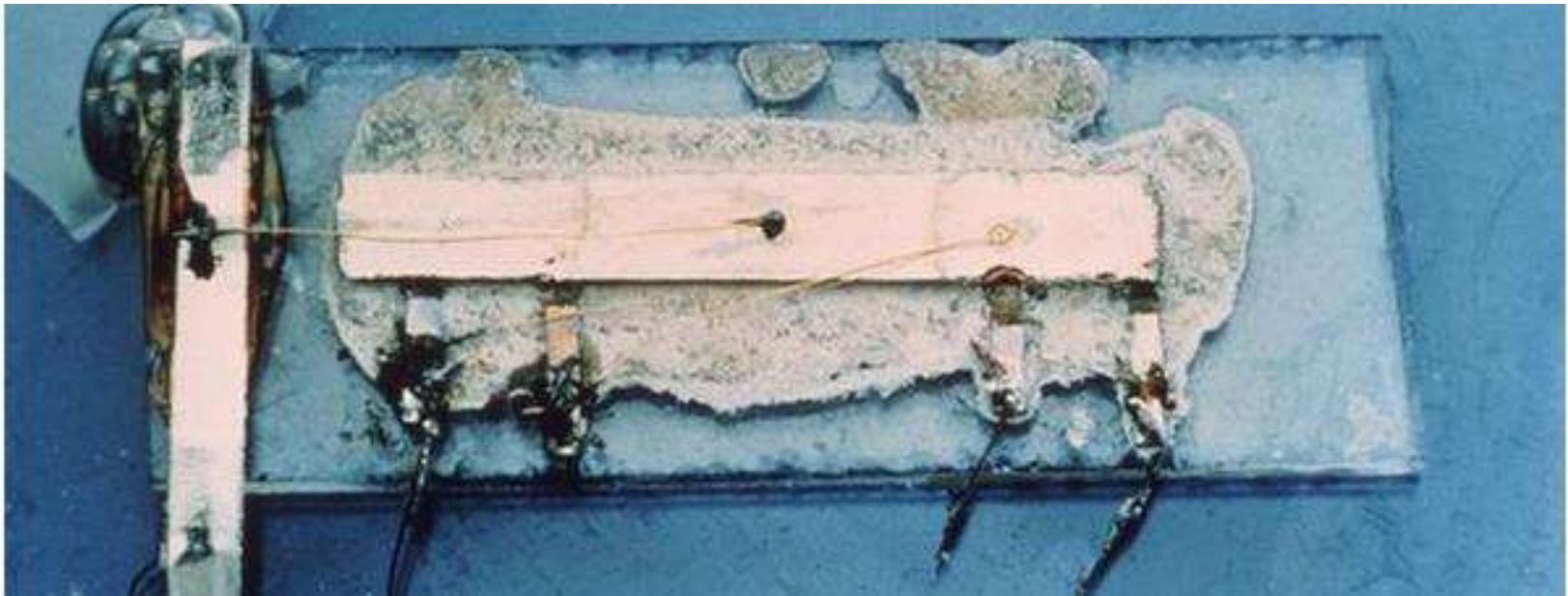
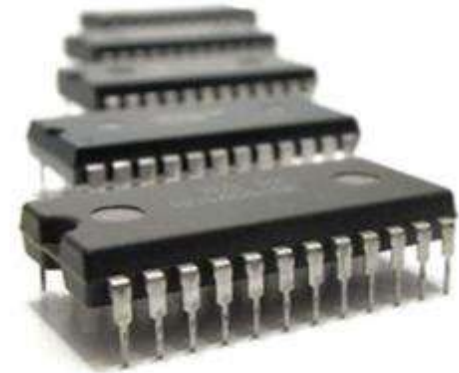
Otro ejemplo es el **FMS** (*Fortran Monitoring System*) para el **IBM 7090**.

En este caso utiliza **cintas** para la entrada y salida.



# TERCERA GENERACIÓN (1965-1980)

- Está marcada por los **circuitos integrados**.
- El primero fue desarrollado en 1958 por **Jack Kilby**.





## TERCERA GENERACIÓN (1965-1980)

- Permitía la integración de grandes cantidades de transistores en un pequeño chip.
- Los ordenadores de esta generación tenían un menor consumo de energía y además mostraron una reducción en el tamaño.
- Surgieron nuevos periféricos para introducir y obtener la información: teclados, monitores,...

# TERCERA GENERACIÓN (1965-1980)

## IBM System 360

1964

- Gama de computadoras con variedad de precios y desempeño.
- Eran compatibles entre todas ellas (en teoría).



## TERCERA GENERACIÓN (1965-1980)

Otro concepto nuevo aparecido en esta generación es el **tiempo compartido**.

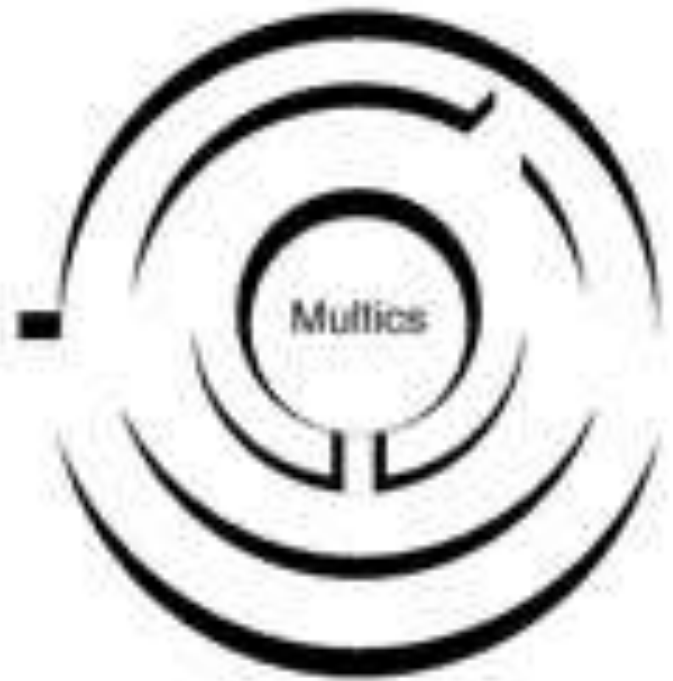
Cada usuario tiene una terminal en línea desde la que pueden trabajar simultáneamente.

El primer sistema operativo que implementaba el tiempo compartido fue el **CTSS** (***Compatible Time Sharing System***)

## TERCERA GENERACIÓN (1965-1980)

El éxito de CTSS llevó a Bell Labs, MIT y General Electric a desarrollar un *servicio de computadora*, una máquina que atendiera a cientos de usuarios.

Este sistema se llamó **MULTICS** (*Multiplexed Information and Computing System*)



# TERCERA GENERACIÓN (1965-1980)

## DEC PDP

1964

- Fueron las primeras **minicomputadoras**.
- La PDP-1 tenía un precio de 12.000\$, menos del 5% de la IBM 7094





## TERCERA GENERACIÓN (1965-1980)

**Ken Thompson**, un científico que había trabajado en el desarrollo de MULTICS halló una DEC-DP 7 que nadie estaba usando y desarrolló una variante de MULTICS para un solo usuario.



El nombre original fue UNICS, aunque posteriormente fue cambiado a **UNIX**

## TERCERA GENERACIÓN (1965-1980)

Tuvo una rápida expansión entre universidades donde una gran cantidad de colaboradores contribuyó a mejorarlo.

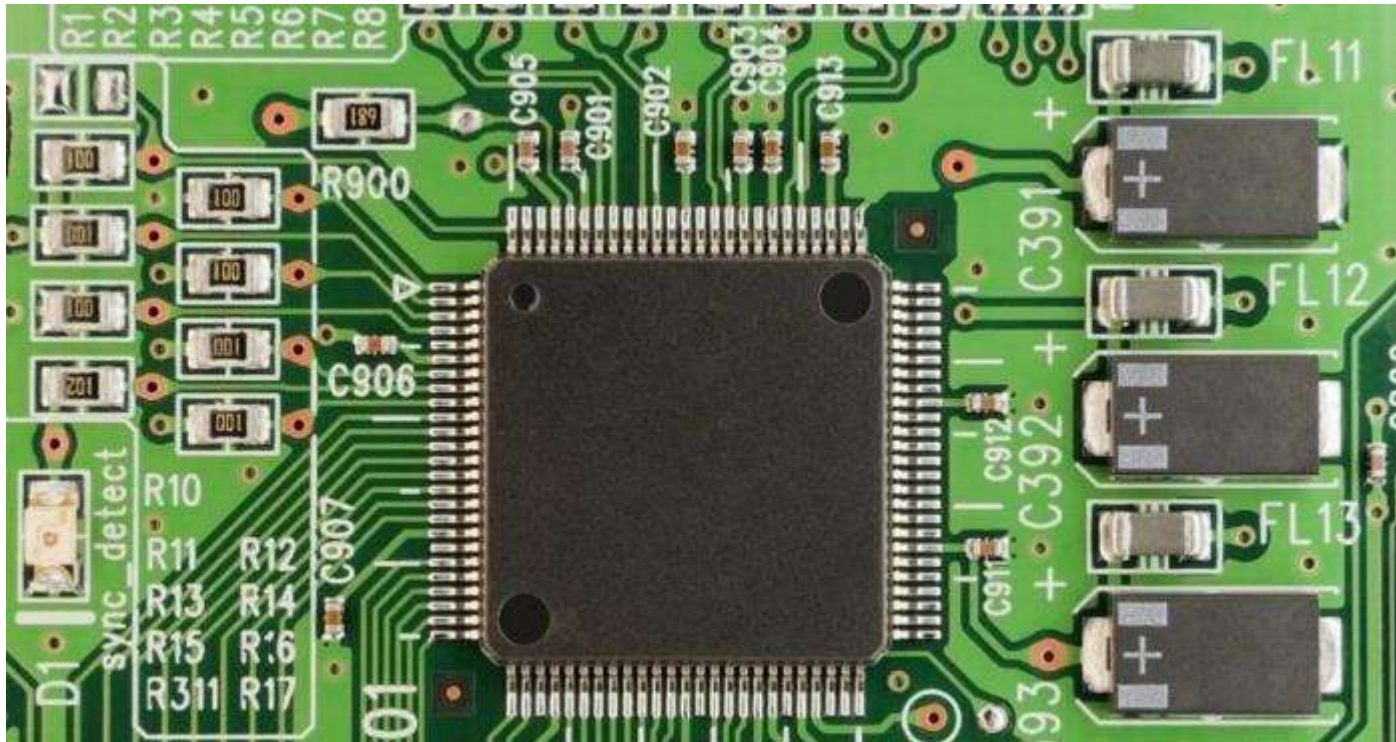
Dos versiones:

- System V de AT&T
- BSD de la Universidad Berkeley



# CUARTA GENERACIÓN (1983-Hoy)

En esta generación ya hay **circuitos integrados a gran escala**, lo que permite una mayor miniaturización de los ordenadores.



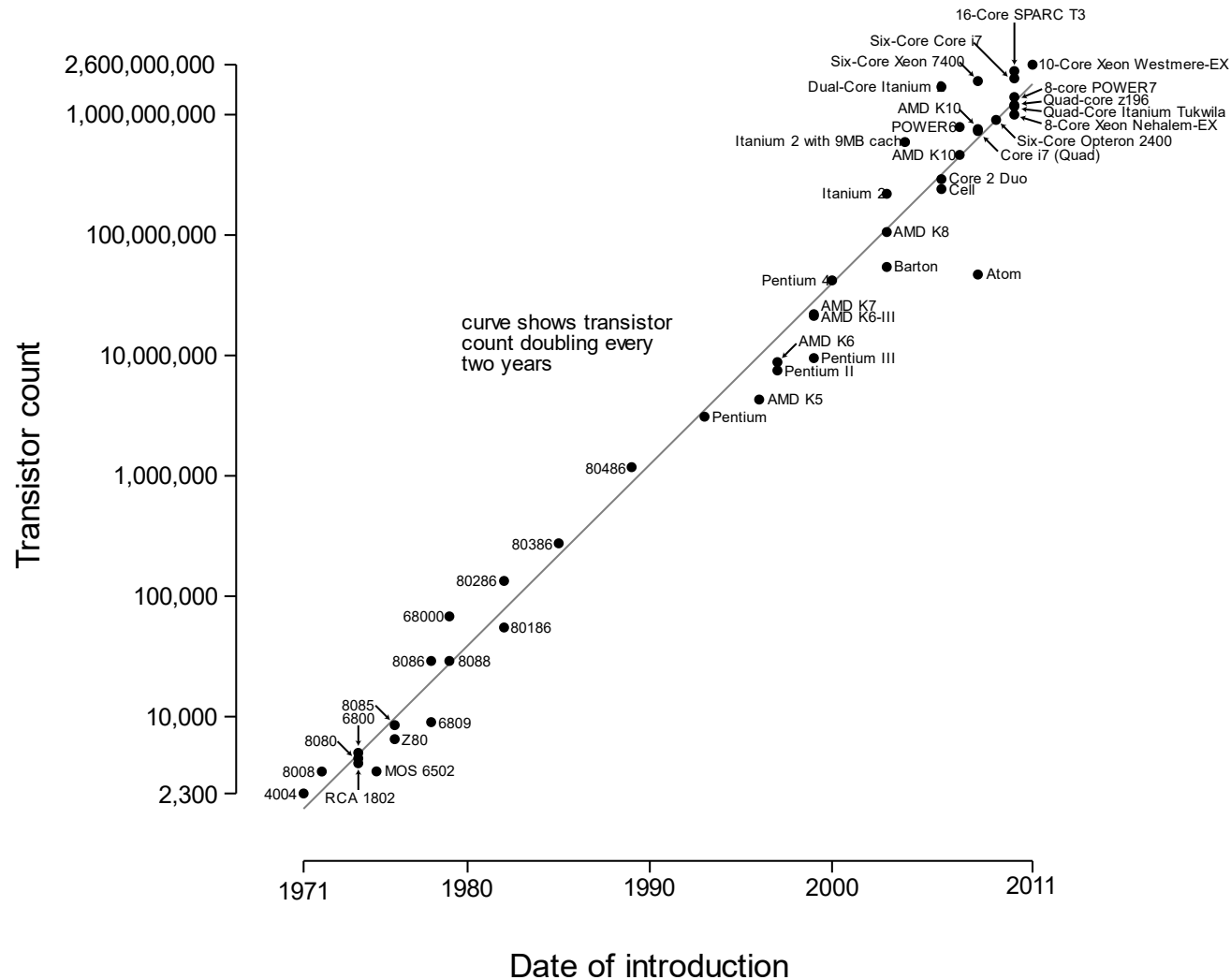
# CUARTA GENERACIÓN (1983-Hoy)

La **escala de integración** cada vez es mayor:

- Intel 8080 (1974): 6000 transistores (6  $\mu\text{m}$ )
- Intel 386 (1985): 275.000 transistores (1.5  $\mu\text{m}$ )
- Pentium II (1997): 7.5 millones de transistores (0.35  $\mu\text{m}$  )
- Core 2 Duo (2006): 410 millones de transistores (45 nm)
- Nehalem – 1st Gen. (2010): 2300 millones (45 nm)
- Ryzen (2017): 4800 millones (14 nm)
- Ryzen 7 3700X (2019): 19200 millones (7 nm)

# CUARTA GENERACIÓN (1983-Hoy)

## Microprocessor transistor counts 1971-2011 & Moore's law





# CUARTA GENERACIÓN (1983-Hoy)

Si en la generación anterior cada empresa podía tener un ordenador, en esta cada persona puede tener su ordenador.

Aparecen los **ordenadores personales**.



# CUARTA GENERACIÓN (1983-Hoy)

## IBM PC

1981

- Fue el ordenador que marcó un hito en esta generación.
- Características:
  - 64 KB de memoria
  - Procesador Intel 8088
  - 4,77 MHz
  - Almacenamiento en cinta de cassette o disquete.



# CUARTA GENERACIÓN (1983-Hoy)

## XEROX ALTO

1973

- Primer ordenador cuyo sistema operativo estaba basado en una GUI (**Interfaz Gráfica de Usuario**) utilizando la metáfora de escritorio.



# CUARTA GENERACIÓN (1983-Hoy)

El primer sistema operativo para micro-computadoras fue el **CP/M (Control Program for Microcomputers)** para el procesador Intel 8080

```

Diskette Drive(s) : 2
Serial Port(s) : 2
Memory (Kb) : 608

A>dir
A: ASM86      CMD : ASSIGN  CMD : CONFIG  CMD : DATA  PFK
A: DDT86      CMD : DSKMAINT CMD : ED      CMD : FUNCTION CMD
A: GENCMD     CMD : HDMAINT  CMD : PIP     CMD : PRINT  CMD
A: SETUP     CMD : STAT     CMD : SUBMIT  CMD : TOD    CMD
A: HELP      CMD : HELP     HLP : GENDEF  CMD :      LIB

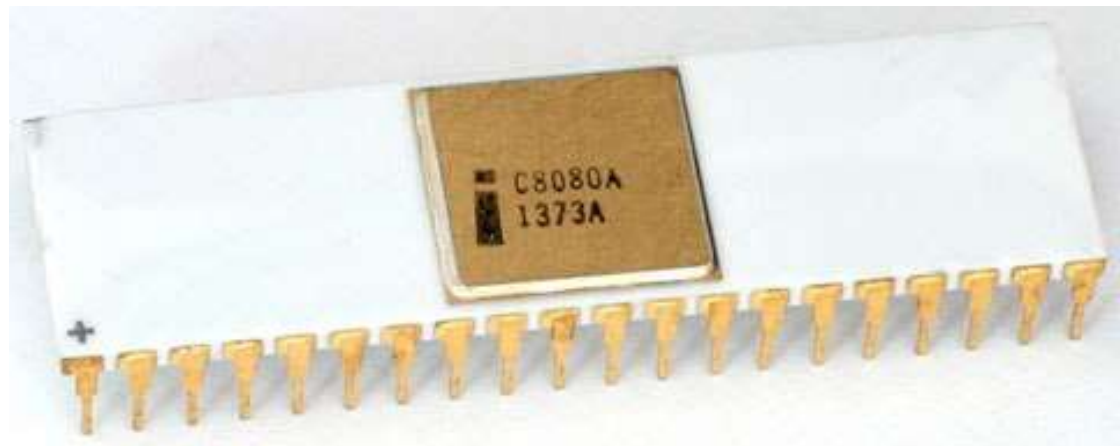
SYSTEM FILE(S) EXIST
A>asm86

CP/M 8086 ASSEMBLER VER 1.1

NO FILE: A:..A86
A>ddt86
DDT86 1.2

-^C
A>
A>du^[D
User 0      0:00:22      Aug. 1, 1983

```



# CUARTA GENERACIÓN (1983-Hoy)

Otro gran sistema operativo de la época es el MS-DOS.

Tiene su origen en el sistema **DOS** de Seattle Computer Products.

**Bill Gates** adquirió DOS y le añadió un intérprete BASIC





# CUARTA GENERACIÓN (1983-Hoy)

```
Starting MS-DOS...

HIMEM is testing extended memory...done.

C:\>C:\DOS\SMARTDRV.EXE /X
C:\>dir

Volume in drive C is MS-DOS_6
Volume Serial Number is 4B77-00E8
Directory of C:\

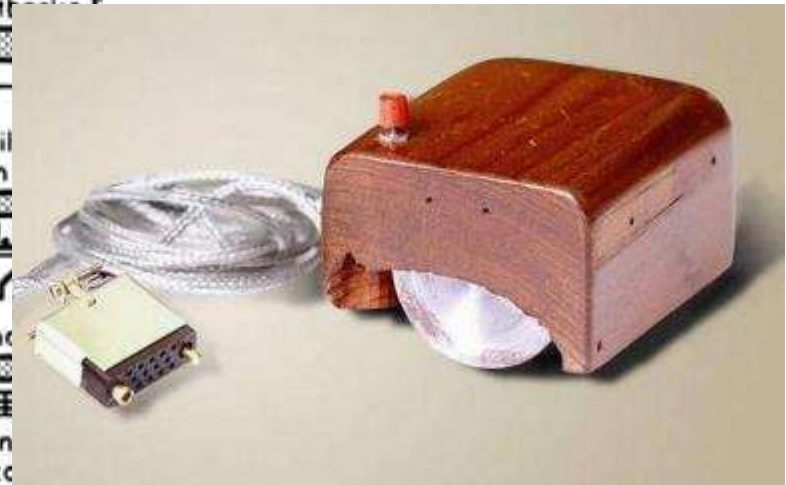
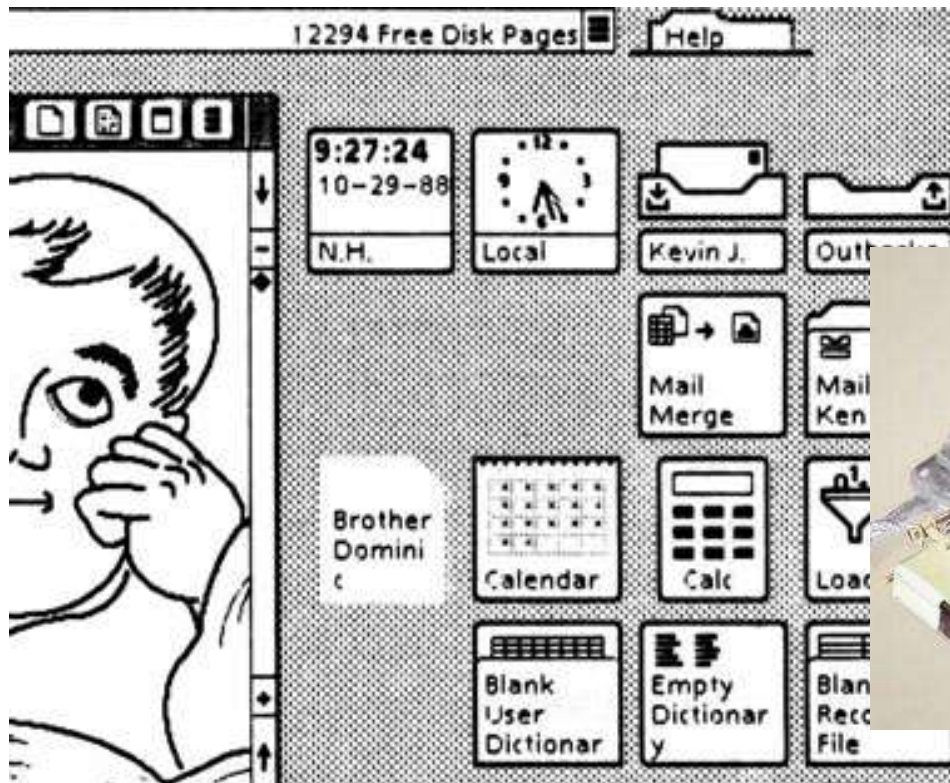
DOS             <DIR>             11-23-17   12:07a
COMMAND  COM           54,645 05-31-94   6:22a
WINA20   386           9,349 05-31-94   6:22a
CONFIG   SYS             71 11-23-17   12:07a
AUTOEXEC BAT          78 11-23-17   12:07a
          5 file(s)             64,143 bytes
                               517,021,696 bytes free

C:\>_
```

Lo relevante de MS-DOS es que estaba incluido en el hardware en lugar de venderlo directamente al usuario.

# CUARTA GENERACIÓN (1983-Hoy)

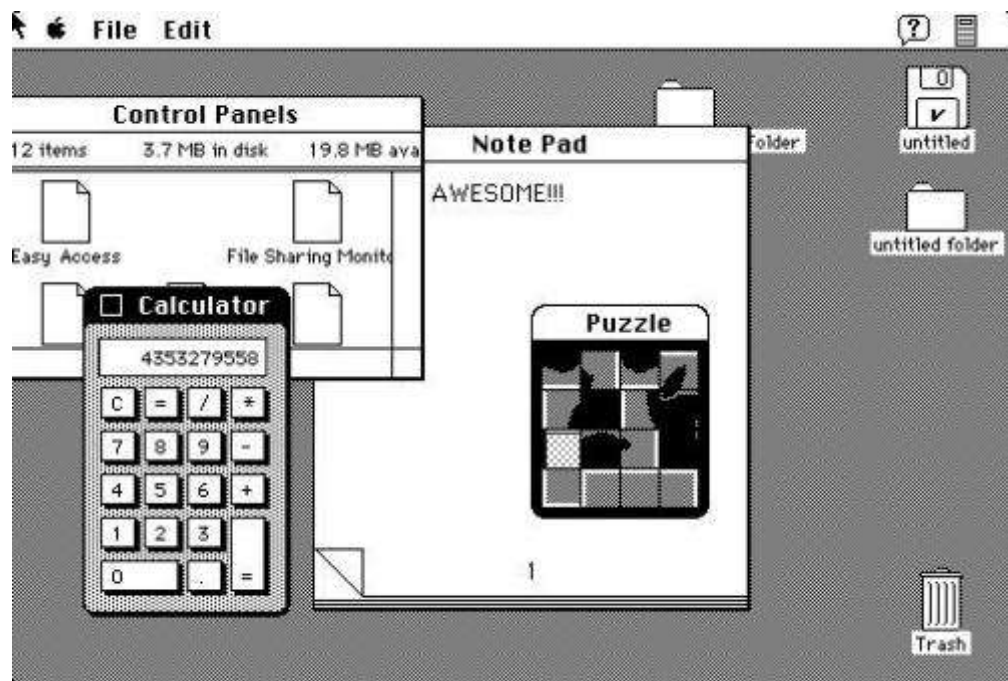
Un hito importante fue la aparición del **ratón** por parte de **Doug Engelbart**, lo que llevó a la aparición de las primeras **interfaces gráficas (GUI)** en Xerox Park



# CUARTA GENERACIÓN (1983-Hoy)

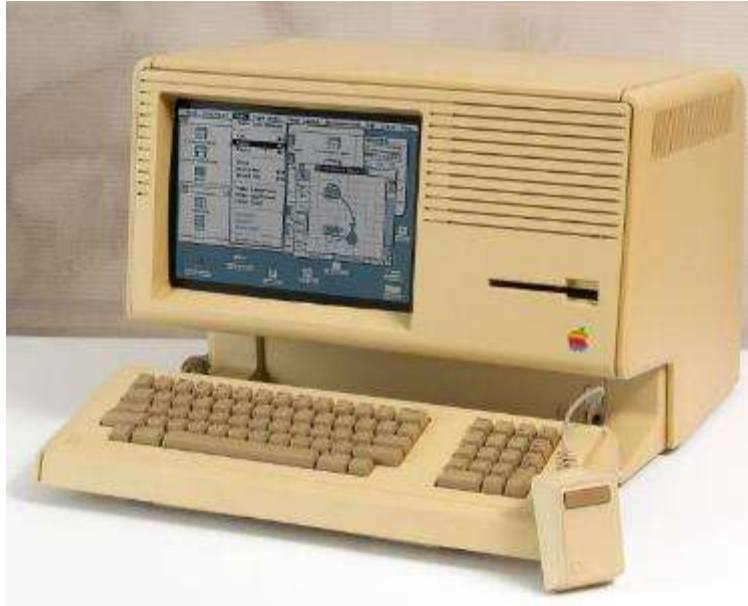
Steve Jobs vio el potencial de la GUI.

Desarrolló un primer ordenador (**Apple Lisa**) que no tuvo mucho éxito por su elevado precio.



Pero su segundo ordenador (**Apple Macintosh**) tuvo un enorme éxito por su menor precio y por ser amigable para el usuario.

# CUARTA GENERACIÓN (1983-Hoy)



## CUARTA GENERACIÓN (1983-Hoy)

El sistema operativo de Apple desde 2001 es **Mac OS X**

El sistema operativo tiene su origen en el **kernel Mach** de BSD, que denominó XNU (X Not Unix)



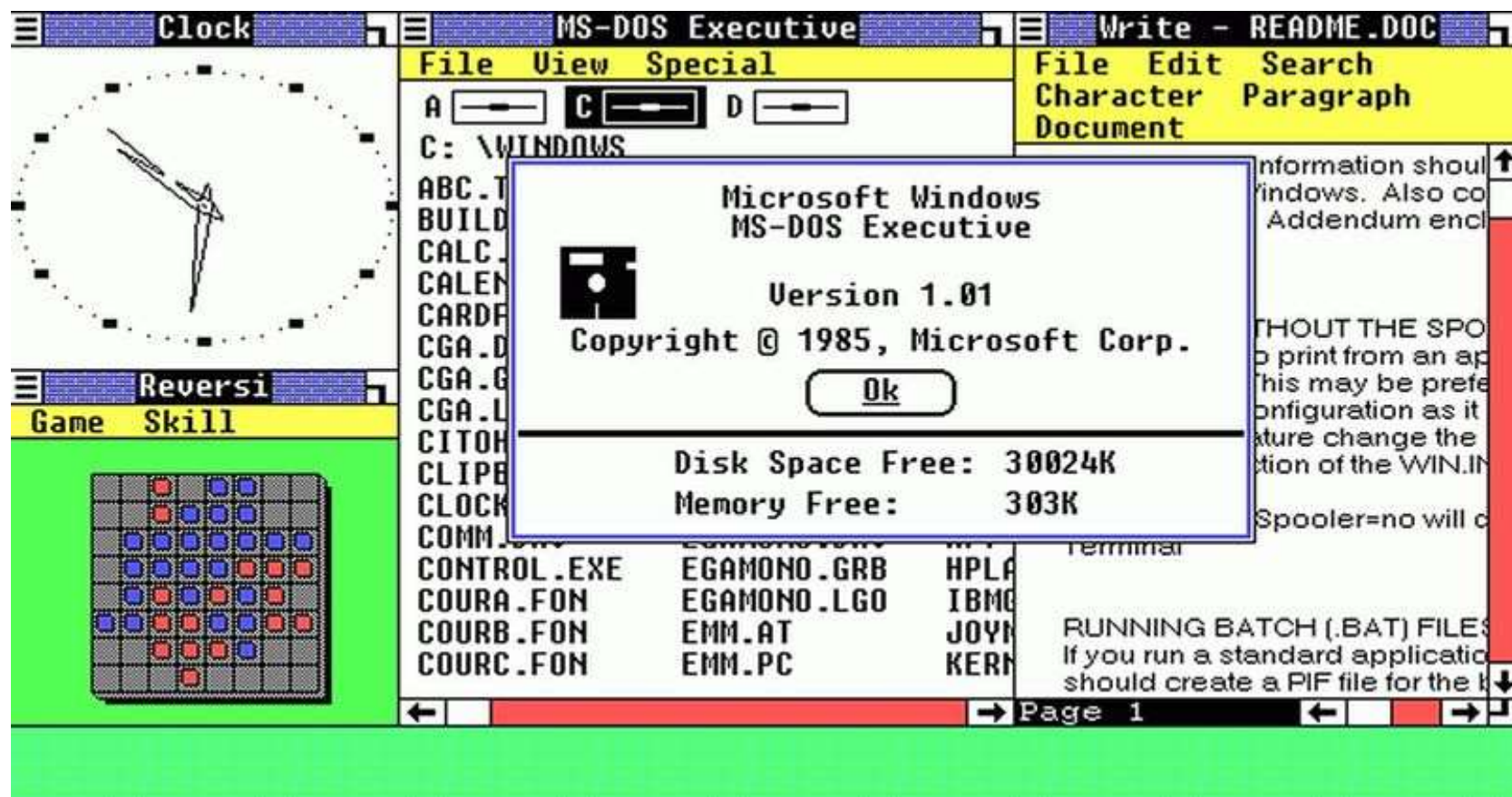
Este sistema se llamó **Darwin**, que es software libre.

**Mac OS X** consiste en el kernel Darwin más la interfaz Aqua, pero con licencia propietaria.



# CUARTA GENERACIÓN (1983-Hoy)

El éxito de Apple llevó a Bill Gates a buscar un sustituto con entorno gráfico para MS-DOS. Lo llamó **Windows**



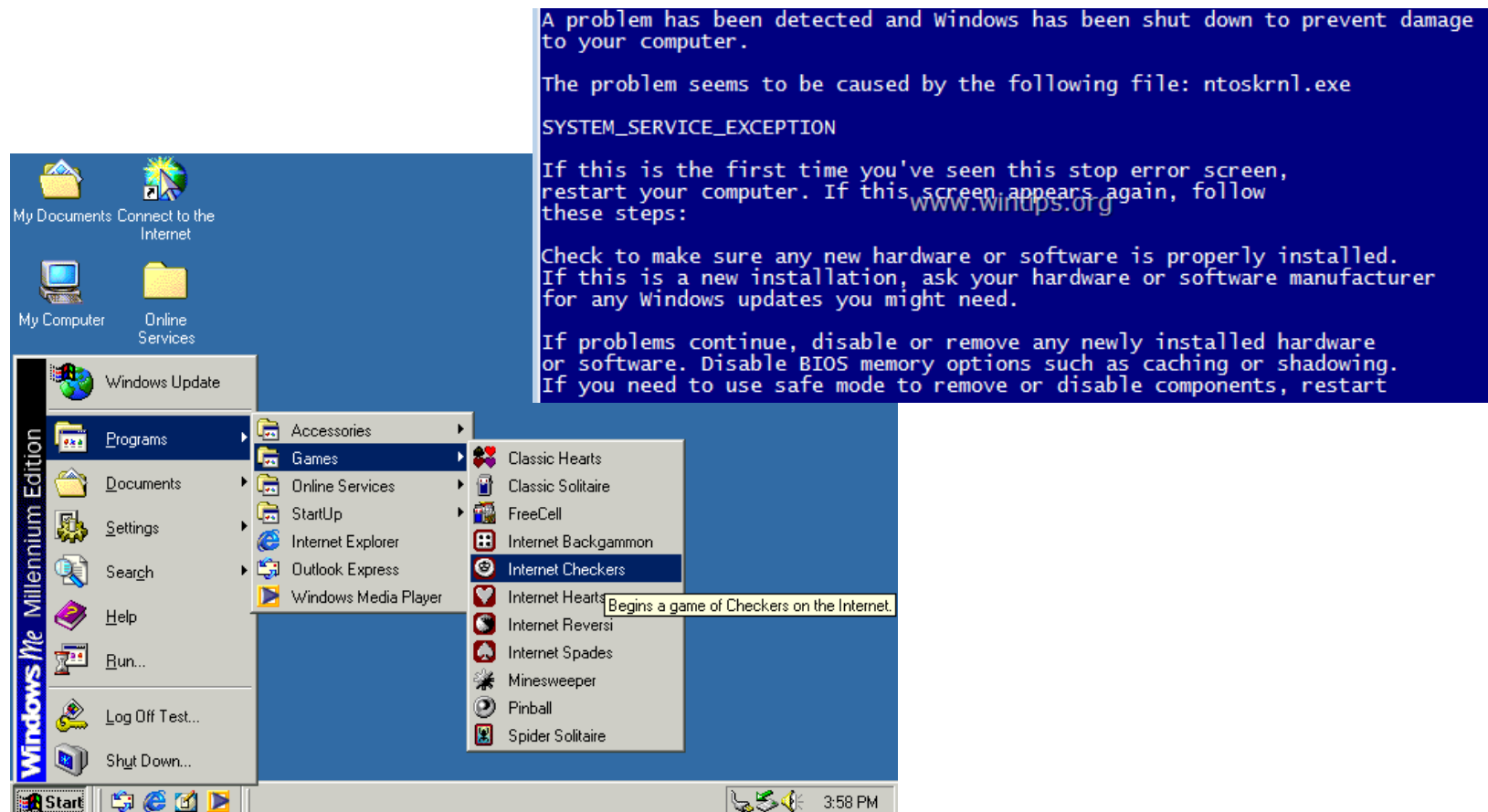
# CUARTA GENERACIÓN (1983-Hoy)

Las primeras versiones no eran un sistema operativo propiamente dicho, sino que eran un *shell* que se ejecutaba sobre MS-DOS



# CUARTA GENERACIÓN (1983–Hoy)

Posteriores versiones fueron **Windows 95**, **Windows 98** o el malogrado **Windows Me**.



# CUARTA GENERACIÓN (1983-Hoy)

Hasta aquí todos los sistemas estaban contruidos sobre MS-DOS

Microsoft decidió lanzar un nuevo sistema que diseñado desde cero para librarse de la dependencia de MS-DOS

Ese sistema fue **Windows NT** que posteriormente dio lugar a **Windows 2000**.



# CUARTA GENERACIÓN (1983-Hoy)

Windows 2000 era un SO para entornos profesionales.

En vista de su éxito y estabilidad, Microsoft lanzó una versión para usuarios domésticos, denominada **Windows XP**.



Indudablemente el mayor éxito de la compañía, con **1000 millones de copias** vendidas hasta 2014



## CUARTA GENERACIÓN (1983-Hoy)

En 2005 Microsoft anunció un nuevo sistema operativo con nombre clave **Longhorn** que iba a ser toda una revolución.

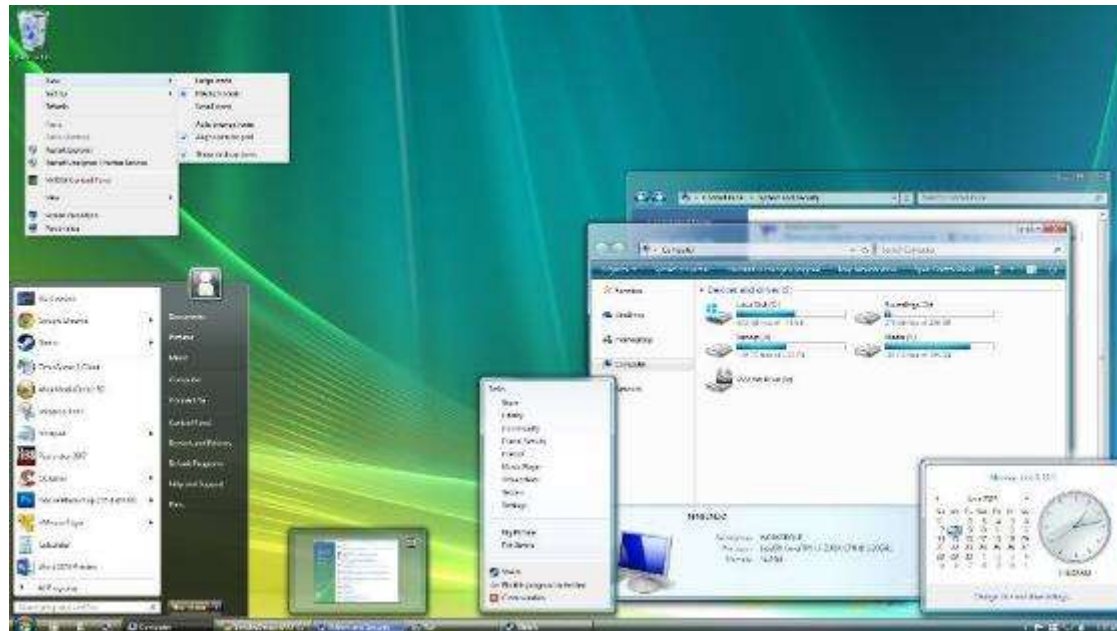
Pronto surgieron problemas en todas esas nuevas funcionalidades que hicieron que el proyecto se retrasase



# CUARTA GENERACIÓN (1983-Hoy)

Finalmente, todo lo que había prometido Microsoft se convirtió en humo

De ahí surgió el otro gran fiasco de Microsoft: **Windows Vista**.



# CUARTA GENERACIÓN (1983–Hoy)

El fracaso de Windows Vista fue rápidamente tapado por **Windows 7**, el sistema que ha sido capaz de reemplazar a Windows XP



# CUARTA GENERACIÓN (1983-Hoy)

El último sistema operativo de Microsoft es Windows 10.

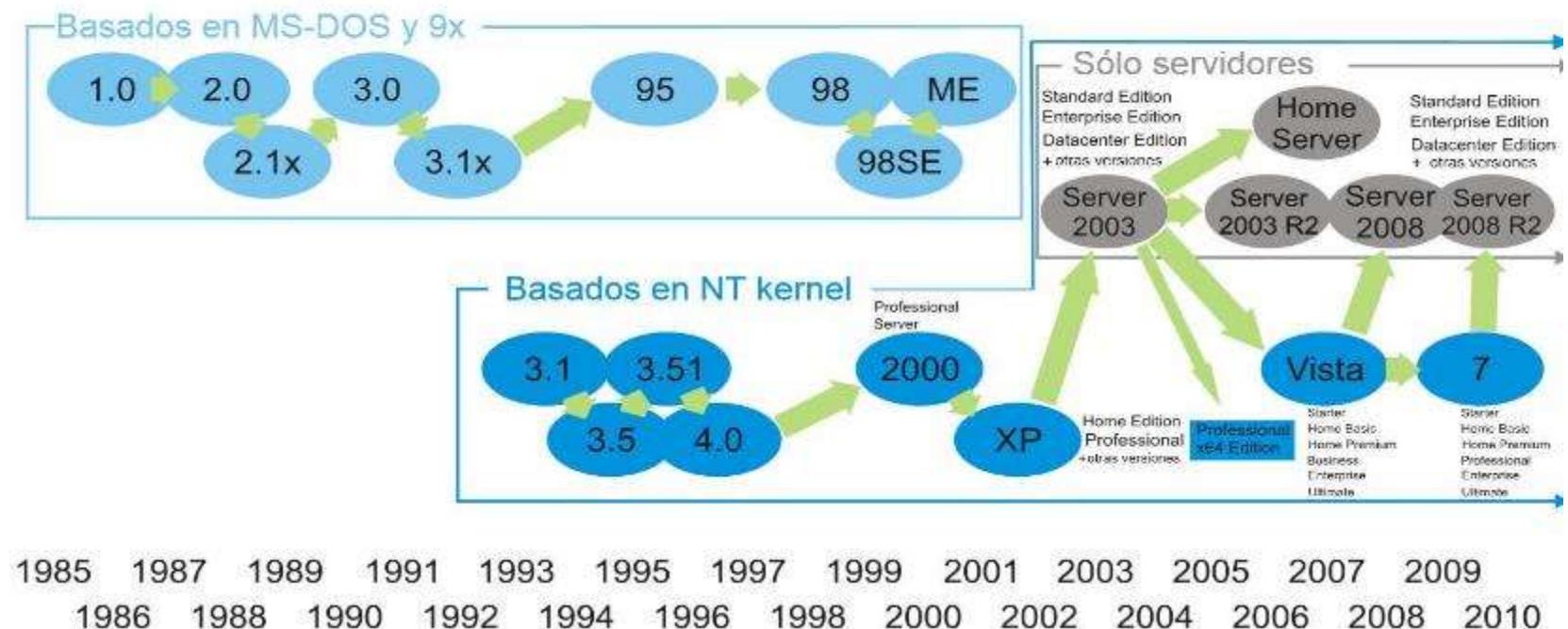


# CUARTA GENERACIÓN (1983-Hoy)

Paralelamente a la familia de oficina Microsoft ha sacado una familia orientada a servidores: Windows Server

## Microsoft Windows

árbol familiar de productos





## CUARTA GENERACIÓN (1983-Hoy)

El desarrollo de **Linux** es paralelo a Windows y tiene su origen en el año 1983.

En este año Richard Stallman acuña el concepto de **Software Libre**.



## CUARTA GENERACIÓN (1983-Hoy)

El software libre contempla cuatro libertades esenciales:

- Libertad de ejecutar un programa, para cualquier propósito.
- Libertad para estudiar como funciona un programa, por lo que es necesario el acceso al código fuente.
- Libertad de redistribuir copias de un programa
- Libertad de distribuir copias de sus versiones modificadas a terceros.

## CUARTA GENERACIÓN (1983-Hoy)

Además, en ese año inició el **proyecto GNU** (acrónimo recursivo de *GNU is not Unix*) con el que pretende crear un sistema operativo similar y compatible con Unix.



## CUARTA GENERACIÓN (1983-Hoy)

En 1985 creó la **Fundación del Software Libre (FSF)** y desarrolló la **Licencia General de GNU (GNU GPL)**



## CUARTA GENERACIÓN (1983-Hoy)

A principios de los 90 tenía mucho software disponible, pero le faltaba lo más importante: el **núcleo** o **kernel**.

Al principio el proyecto GNU contemplaba un núcleo, denominado **GNU Hurd**, aunque su desarrollo nunca llegó a completarse.

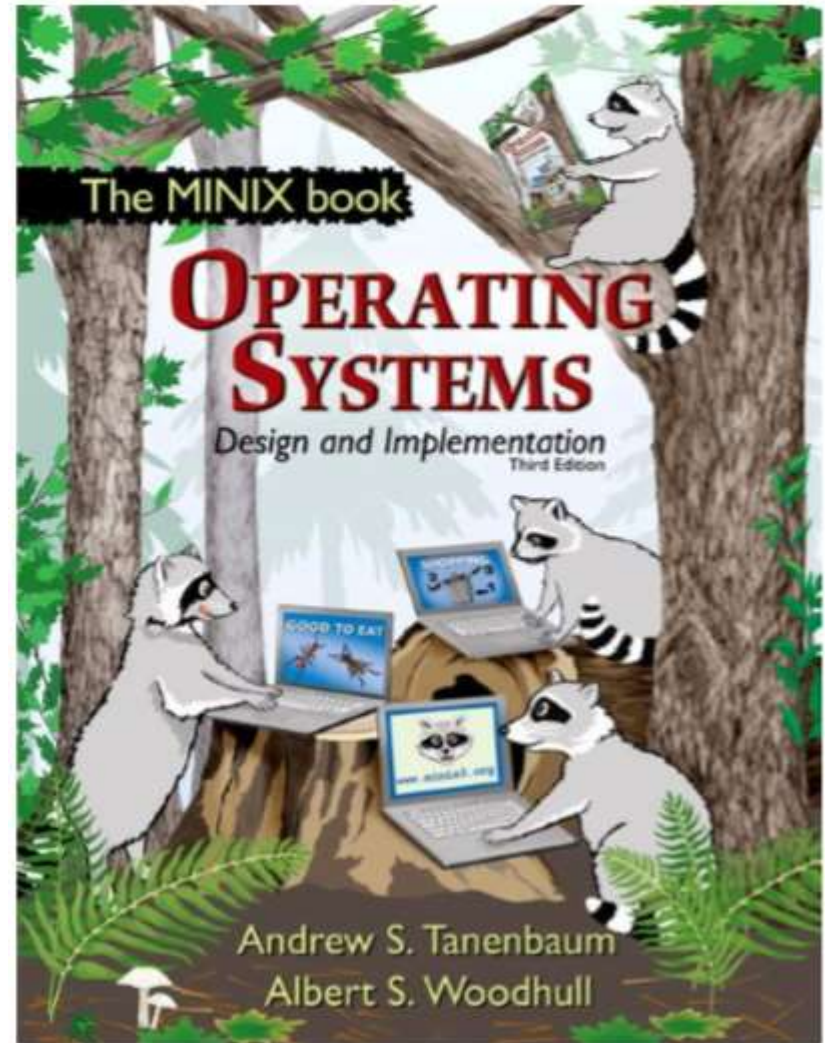
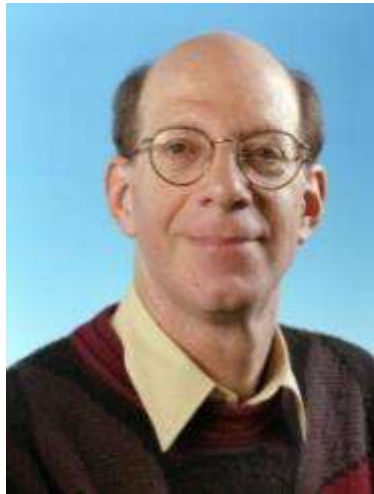
Aquí es donde aparece **Linus Torvals**.





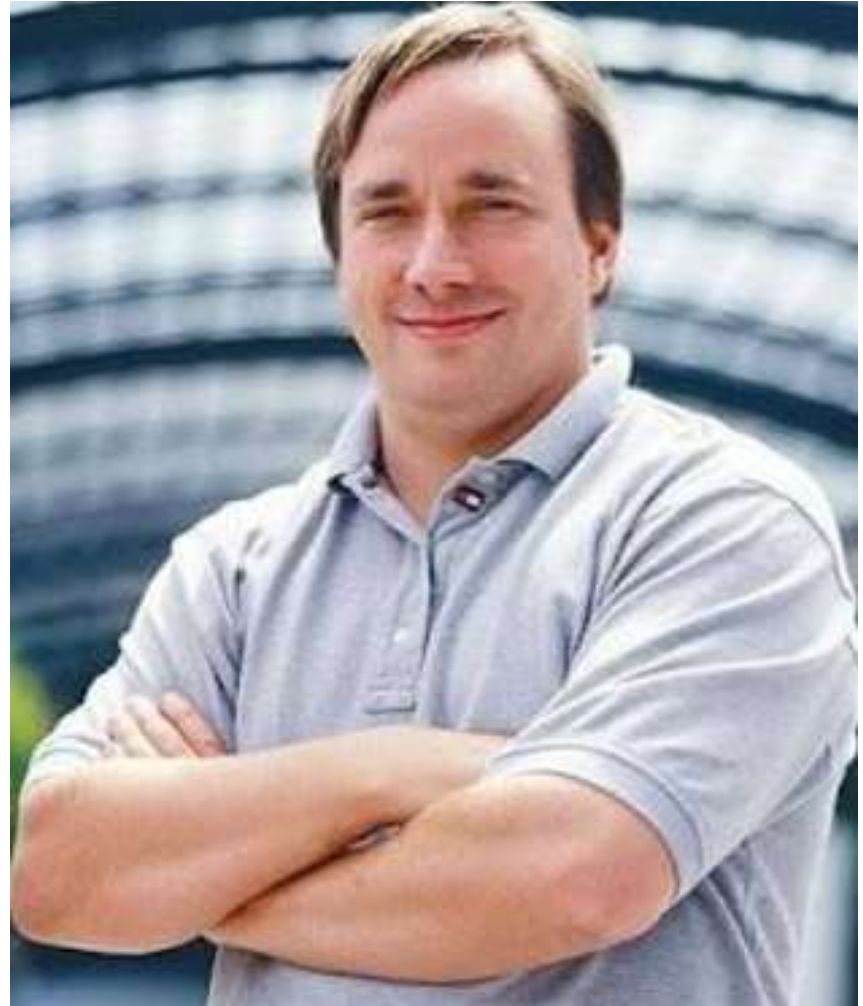
# CUARTA GENERACIÓN (1983-Hoy)

En 1987 el profesor **Andrew S. Tanenbaum** desarrolló un clon de Unix denominado **Minix** muy sencillo para enseñar los fundamentos de SO a sus alumnos.



## CUARTA GENERACIÓN (1983-Hoy)

El 25 de agosto de 1991 publicó un mensaje en la red Usenet anunciando la creación de este núcleo y dejando el proyecto abierto a la colaboración.



## CUARTA GENERACIÓN (1983-Hoy)

Aunque al principio lo publicó con una licencia propia (que restringía la actividad comercial) en 1992 lo vinculó a la licencia **GNU GPL**.

**Linux es solo el núcleo** del sistema operativo, por lo que el nombre correcto del sistema operativo es **GNU/LINUX**



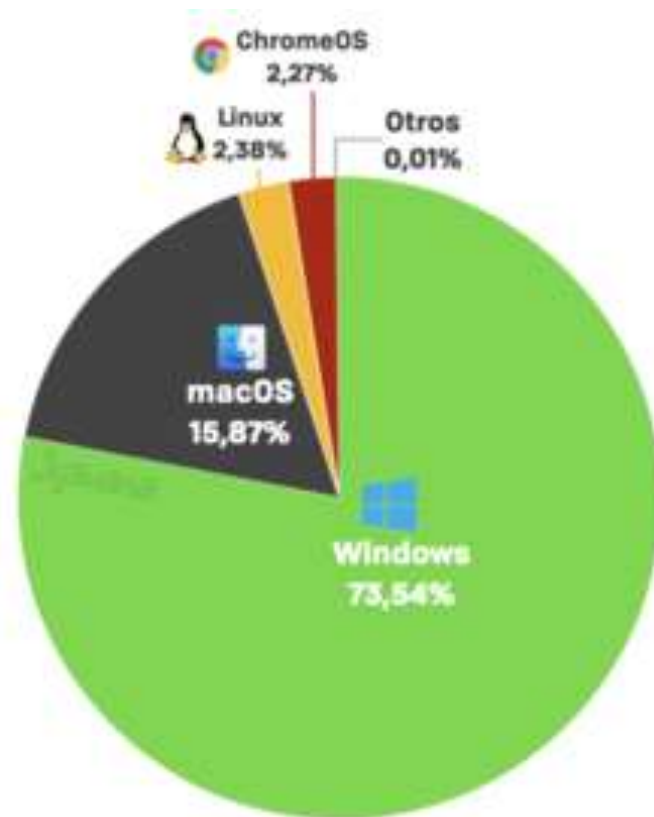
# CUARTA GENERACIÓN (1983-Hoy)

Desde entonces Linux ha crecido y en la actualidad hay cientos de distribuciones diferentes.



# CUARTA GENERACIÓN (1983-Hoy)

Cuota de mercado de los sistemas operativos en mayo de 2020



Cuota de mercado de sistemas operativos de escritorio.  
Datos: Stat Counter Global Stats.

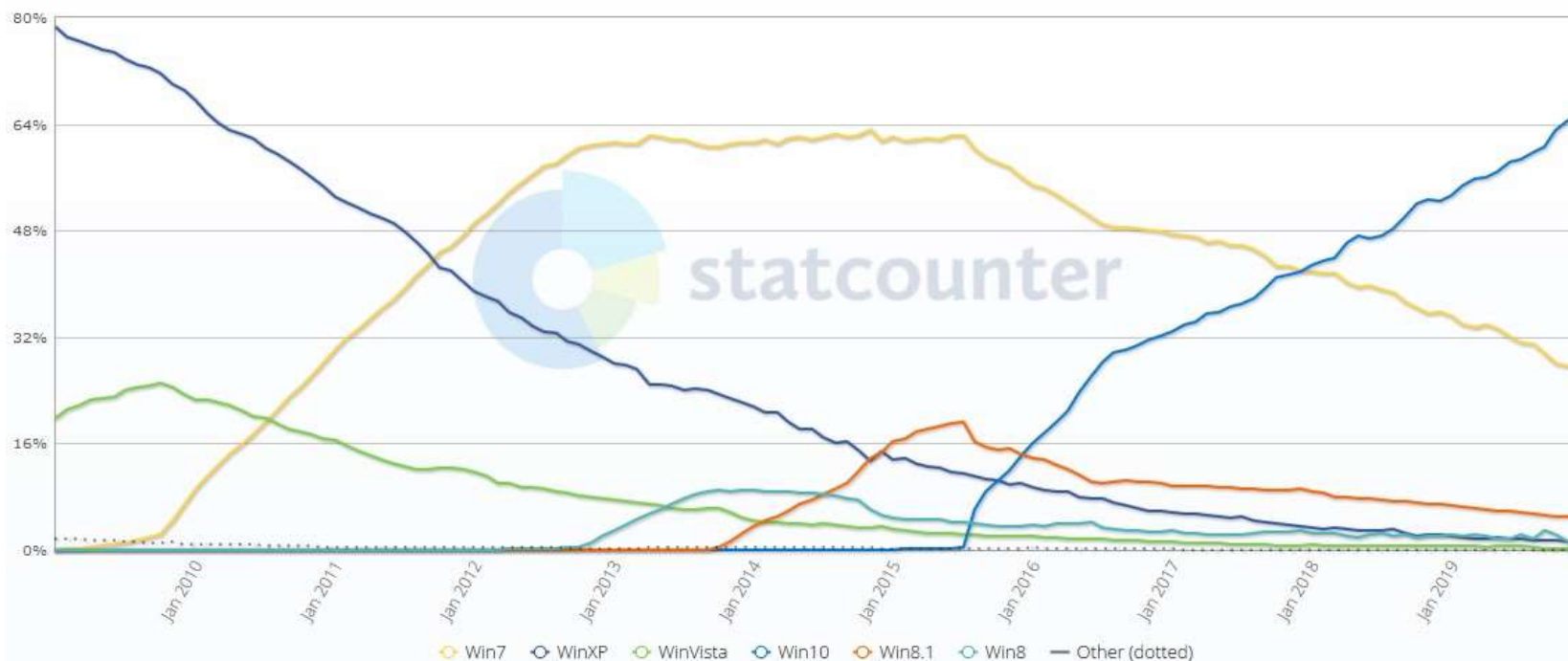


# CUARTA GENERACIÓN (1983-Hoy)

## Cuota de mercado de los sistemas operativos Windows hasta diciembre de 2019

Desktop Windows Version Market Share Worldwide

Jan 2009 - Dec 2019

[Edit Chart Data](#)

# CUARTA GENERACIÓN (1983-Hoy)

Distribución por sistema operativo en **servidores Web**  
(w3techs.com)

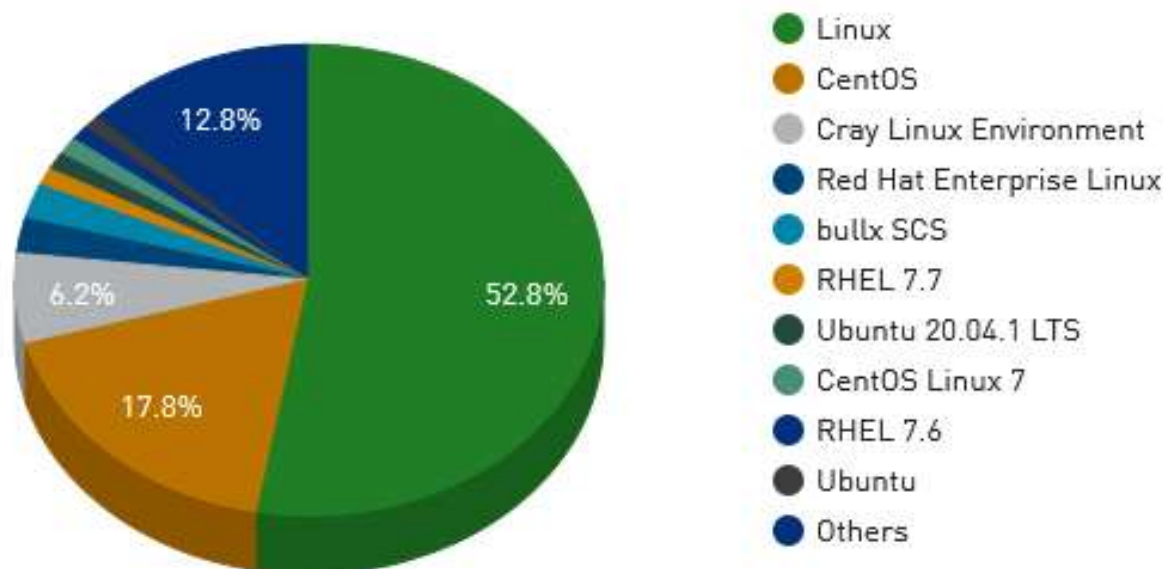
© W3Techs.com	usage	change since 1 August 2020
1. Unix	70.9%	+0.6%
2. Windows	29.1%	-0.6%

percentages of sites

# CUARTA GENERACIÓN (1983-Hoy)

Distribución por sistema operativo en los 500 mayores supercomputadores del mundo en junio de 2020 (top500.org)

Operating System System Share



# CUARTA GENERACIÓN (1983-Hoy)

## Detalle de la gráfica anterior

	Operating System	Count	System Share (%)	Rmax (GFlops)	Rpeak (GFlops)	Cores
1	Linux	264	52.8	613,794,368	1,204,571,501	19,952,748
2	CentOS	89	17.8	299,116,910	569,172,925	7,368,768
3	Cray Linux Environment	31	6.2	213,378,180	312,270,382	5,816,556
4	Red Hat Enterprise Linux	12	2.4	633,345,280	787,755,359	10,785,184
5	bullx SCS	12	2.4	57,010,650	89,353,113	1,979,488
6	RHEL 7.7	6	1.2	41,736,700	84,978,446	1,204,288
7	Ubuntu 20.04.1 LTS	6	1.2	87,277,000	108,063,220	755,904
8	CentOS Linux 7	6	1.2	88,718,610	138,334,625	1,739,472
9	RHEL 7.6	5	1	56,673,970	79,499,650	929,712
10	Ubuntu	5	1	18,058,400	74,741,105	237,744
11	Linux/TOSS	5	1	12,755,500	16,695,897	359,616
12	VEOS	4	0.8	25,003,900	33,846,360	111,360
13	Ubuntu 16.04.3 LTS	4	0.8	66,360,000	98,228,920	629,760
14	SUSE Linux Enterprise Server 11	4	0.8	17,679,850	24,000,510	738,800
15	SUSE Linux Enterprise Server 12 SP1	3	0.6	18,526,020	27,252,925	408,248
16	RHEL 7.4	3	0.6	163,132,460	222,997,778	2,721,492
17	Ubuntu 18.04.01	3	0.6	18,779,000	24,217,959	274,896
18	SLES15 SP2	3	0.6	29,710,800	37,398,530	1,075,200
19	Cray OS	3	0.6	23,114,600	28,200,960	783,360
20	SUSE Linux Enterprise Server 15 SP2	2	0.4	14,035,980	24,834,660	211,392



**3**

# COMPONENTES DE LOS SISTEMAS OPERATIVOS



# COMPONENTES DE UN SISTEMA OPERATIVO

Se suele considerar que el sistema operativo está formado por **tres capas**:

- Núcleo
- Servicios
- Intérprete de mandatos o Shell



# NÚCLEO

El **núcleo** o **kernel** es la parte del sistema operativo que interacciona directamente con el **hardware de la máquina**.

Sus funciones se centran en **gestión de recursos** como:

- Procesador
- Tratamiento de interrupciones
- Funciones básicas de manipulación de memoria.

Los **servicios** se suelen agrupar según su funcionalidad en varios componentes.

Cada uno se encarga de las siguientes funciones:

- Gestión de procesos
- Gestión de memoria
- Gestión de archivos y directorios
- Gestión de E/S
- Gestión de red
- Seguridad

# SERVICIOS – GESTIÓN DE PROCESOS

## Gestión de procesos

**Programa:** es un conjunto de instrucciones para realizar una determinada tarea

**Proceso:** es un programa en ejecución

```
# Ejemplo 3.  
# Haciendo un copy & paste de la función  
# RenameProcess del modulo misc de wicd.  
# /usr/share/pyshared/wicd/misc.py  
def RenameProcess(new_name):  
    """ Renames the process calling the function :  
    if sys.platform != 'linux2':  
        print 'Unsupported platform'  
        return False  
    try:  
        import ctypes  
        is_64 = os.path.exists('/lib64/libc.so.6')  
        if is_64:  
            libc = ctypes.CDLL('/lib64/libc.so.6')  
        else:  
            libc = ctypes.CDLL('/lib/libc.so.6')  
        libc.prctl(15, new_name, 0, 0, 0)  
        return True  
    except:  
        print "rename failed"  
        return False
```



# SERVICIOS – GESTIÓN DE PROCESOS

Cuando un proceso se encuentra en ejecución necesita una serie de recursos, por ejemplo:

- Tiempo de procesador
- Memoria donde almacenar memoria y datos
- Usualmente espacio de almacenamiento en disco
- Comunicación con dispositivos de E/S

En un momento determinado hay múltiples procesos ejecutándose en un ordenador .

Esto se consigue con el **multiplexado**



# SERVICIOS – GESTIÓN DE PROCESOS

**Multiplexado:** asignar a cada proceso un *quantum* de tiempo. Pasado este tiempo es otro proceso quien pasa a ejecutarse. Esto se conoce como **multitarea aparente**.

**El sistema operativo crea, suspende, reanuda y elimina procesos de usuario y del sistema**

# SERVICIOS – GESTIÓN DE MEMORIA

## Gestión de memoria

La memoria principal también se conoce como memoria **RAM** (*Random Access Memory*)

Es un elemento clave en el rendimiento ya que cualquier proceso que se ejecute debe estar en memoria

Es un posible cuello de botella ya que la memoria RAM es mucho más lenta que el procesador



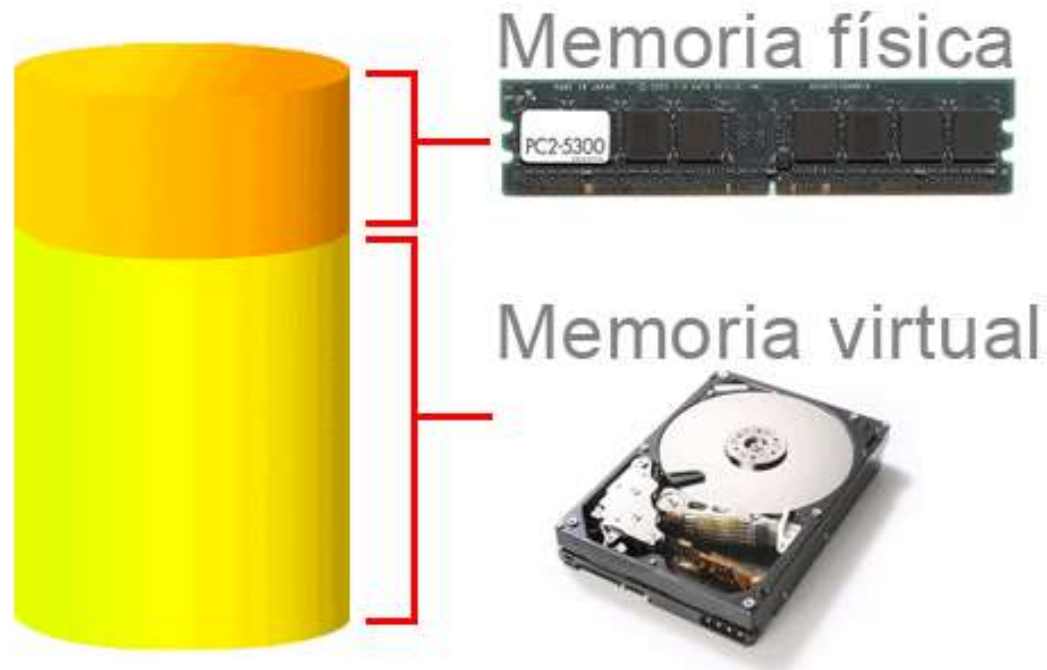
# SERVICIOS – GESTIÓN DE MEMORIA

La RAM se encuentra justo en medio de la **jerarquía de memoria**, que representa el compromiso entre capacidad, velocidad y precio.



# SERVICIOS – GESTIÓN DE MEMORIA

Como la memoria es limitada los sistemas operativos hacen uso de la **memoria virtual**, espacio en disco que gestiona el sistema operativo gestiona como memoria RAM.



# SERVICIOS – GESTIÓN DE MEMORIA

El sistema operativo tiene algoritmos para:

- Decidir qué procesos se cargan en memoria
- Qué direcciones de memoria se asignan a un proceso y se liberan cuando termina
- Qué procesos son cargados de la memoria RAM a la memoria virtual y viceversa.

Además, debe llevar un seguimiento de las posiciones de memoria están siendo utilizadas y cuáles están libres.



## Gestión de ficheros y directorios

### Memoria auxiliar

- Son los dispositivos hardware que permiten almacenar información
- Son **no volátiles**, es decir, la información no se pierde al apagar el ordenador.
- Algunos ejemplos: memorias flash, discos duros, CD-ROM, tarjetas de memoria,...

# SERVICIOS – GESTIÓN DE FICHEROS Y DIRECTORIOS

Se denomina **sistema de ficheros** a la forma que tiene el sistema operativo de organizar los archivos.

Por norma general cada SO tiene su propio sistema de ficheros.

- Windows utiliza NTFS y FAT32
- Linux reconoce gran número de sistemas: ext2, ext3, ReiserFS, ...

# SERVICIOS – GESTIÓN DE FICHEROS Y DIRECTORIOS

Es tarea del sistema operativo:

- Gestionar el espacio libre en el sistema de ficheros
- Asignar espacio en el disco a cada fichero
- Liberar espacio al borrarlo
- Garantizar el acceso seguro a la información
- Garantizar la integridad de los datos.

## Gestión de E/S

El sistema gestiona la comunicación con los periféricos

Hay miles de periféricos de todo tipo y no todos los sistemas operativos los soportan:

- Windows no tiene problemas prácticamente con ninguno
- Max OS X prácticamente solo soporta periféricos fabricados por Apple.
- Linux soporta gran número de periféricos pero sigue habiendo algunos que no reconoce

## SERVICIOS – GESTIÓN DE E/S

Para que un sistema operativo reconozca un periférico necesita un programa llamado **driver** o **controlador**.

Normalmente los propios fabricantes proporcionan los drivers para Windows.

Sin embargo suelen olvidarse de los de Linux, los cuales suelen estar desarrollados por la comunidad.



## Gestión de red

Muchas de las tareas que se realizan en un ordenador requieren una conexión al exterior, bien sea a la red de área local (LAN) o bien a Internet.

Por ejemplo:

- Imprimir en una impresora en red
- Acceder a los ficheros en otro equipo o en la nube (*cloud computing*)
- Visitar páginas web

# SERVICIOS – GESTIÓN DE RED

Algunas de las funciones del sistema operativo respecto a las redes son:

- Abstraer al usuario la topología de red, es decir, el conexionado físico de la red.
- Mostrar recursos remotos de la misma forma que los recursos locales.

## Seguridad

La seguridad en un SO se enfoca desde dos perspectivas:

- **Autenticación y protección de datos:**
  - Los usuarios se deben validar en el sistema proporcionando mecanismos para garantizar la **confidencialidad** de sus datos.

- **Protección de los recursos de los procesos:**
  - Todos los procesos utilizan recursos (memoria, procesador,..)
  - El sistema operativo tiene que repartir equitativamente los recursos y garantizar que ningún proceso pueda acceder a los recursos de otro proceso

# SERVICIOS

Los servicios se ofrecen a través de una **interfaz de llamadas al sistema**.

Ejemplos de interfaces son **Win32** y **POSIX**

# SHELL

En el nivel más alto se encuentra el **Shell** o **intérprete de comandos**, que es la interfaz primaria entre usuario y sistema operativo.

Puede ser textual o gráfico.

```
bash --init-file $BASH_IT/bash_it.sh -i

Horebas@: ~
→ cd workspace
ls
cd jockeyjs
git
Horebas@: ~/workspace
→ ls
cd jockeyjs
git statusAlcatraz      apartmentlist      jackeyjs      friend_newories      staggys
AutolayoutCollectionViewIssue  apartmentlist-ios  keyboard_type  nioajs-css-extension  terminal-screens
Contentment      apartmentlist.sublime-project  keyboard_type  nioajs-css-extension  typer
DynamicBezierUITextView  apartmentlist.sublime-workspace  keyboard_type  nioajs-css-extension  xit-frontent
JustAiduo        bash-it      bash-it-theme-screenshots  resume.rb      where_are_you
TestuserDefaults  WhereinHellAreYou105  drugs-and-booze-check  run_distance
XcodeBacExpander  facebook_event_gcal  scripts

Horebas@: ~/workspace
→ cd jockeyjs

Horebas@: ~/workspace/jockeyjs [master A]
→ git status
# On branch master
# Untracked files:
#   (use "git add <file>..." to include in what will be committed)
#
#       bobby-black.jpg
nothing added to commit but untracked files present (use "git add" to track)

Horebas@: ~/workspace/jockeyjs [master A]
→ cd ~/Desktop
screenshotcapture ba
Horebas@: ~/Desktop
→ screenshotcapture bakke-black.jpg
```







5

# ARQUITECTURA DE LOS SISTEMAS OPERATIVOS

# ARQUITECTURA DE LOS SISTEMAS OPERATIVOS

Según la forma en que se organizan internamente los sistemas operativos hay diversas arquitecturas:

- Monolíticos
- En capas
- Máquinas virtuales
- Exokernels
- Micronúcleos
- Por módulos

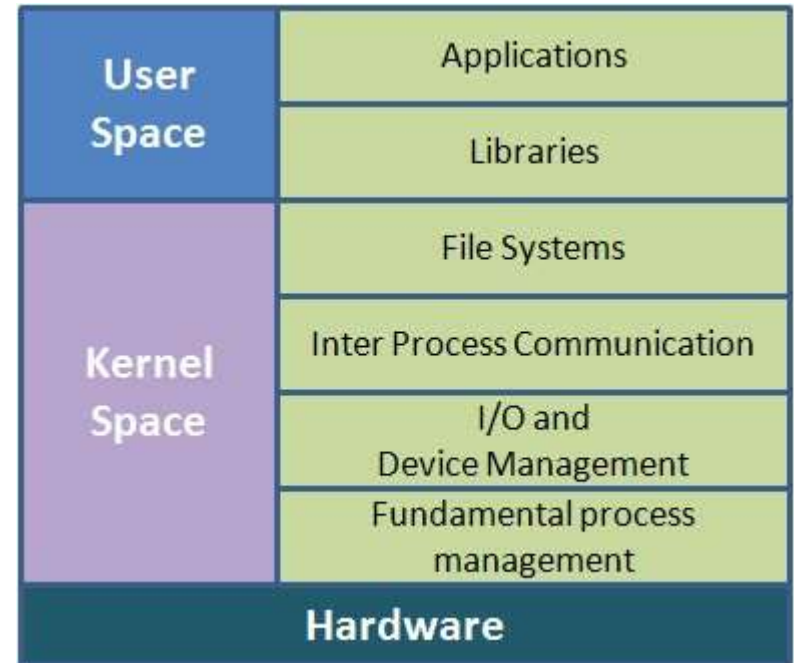
# SISTEMA OPERATIVO MONOLÍTICO

Se caracterizan por una **ausencia de estructura**.

Los procedimientos internos del sistema operativo se pueden llamar entre sí sin ninguna restricción.

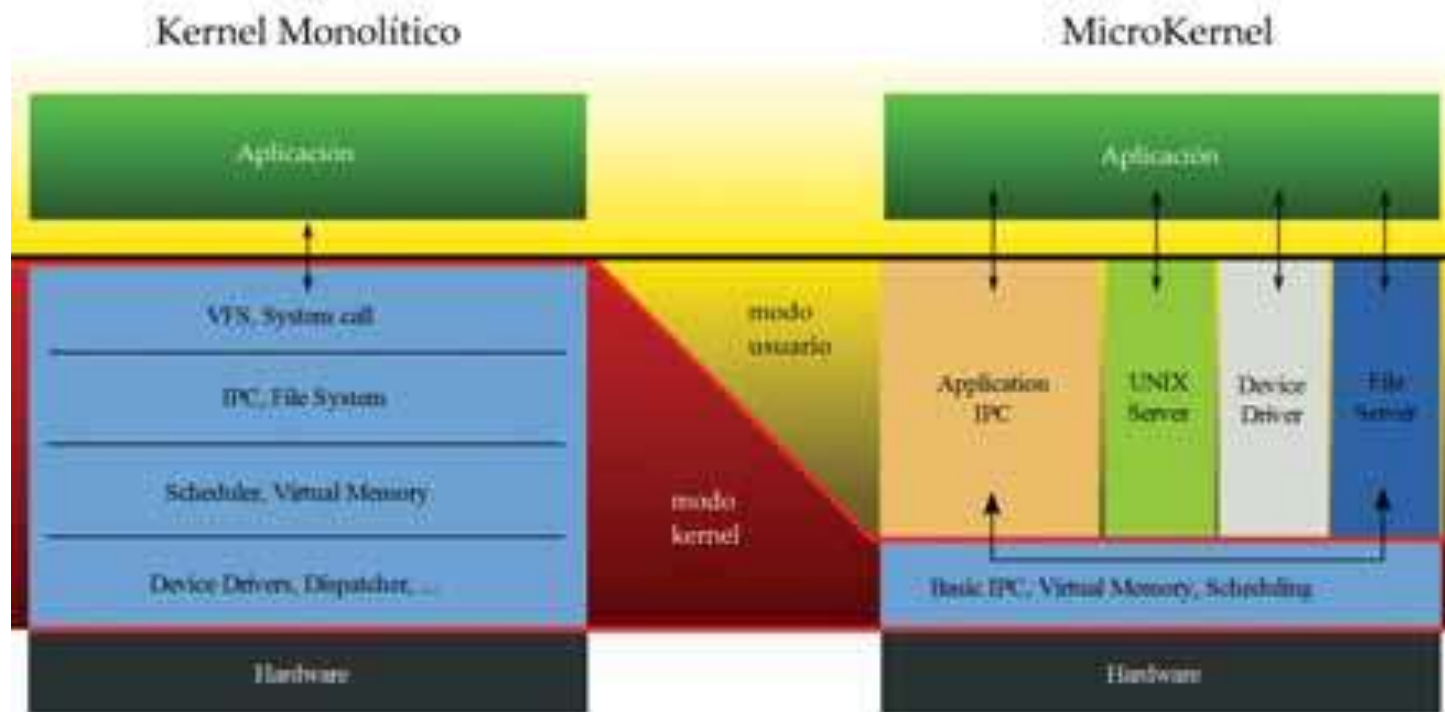
No hay ningún tipo de ocultamiento de la información.

**Ejemplos:** FreeBSD, MS-DOS, familia Windows 9x



# SISTEMA OPERATIVO DE MICROKERNEL

Se quita en lo posible el modo *kernel*, dejando un **micro-kernel** mínimo que provee un conjunto de primitivas o llamadas básicas al sistema.



# SISTEMA OPERATIVO DE MICROKERNEL

Sus ventajas son:

- Reducción de la complejidad
- Descentralización de los fallos

**Ejemplos:** Amoeba, Minix, Hurd o Symbian



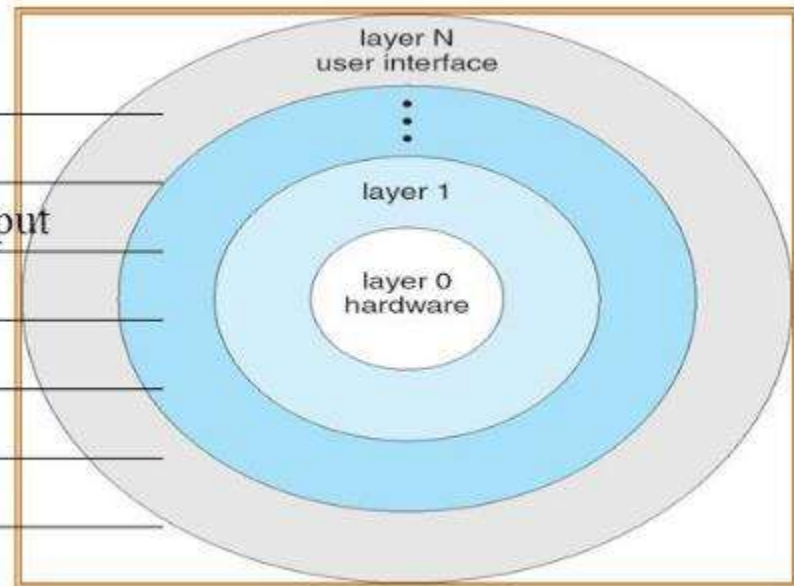
# SISTEMA OPERATIVO POR CAPAS

Se estructura en una jerarquía de capas.

El primer sistema operativo con esta filosofía fue **THE**, de E. W. Dijkstra.

Its six layers are as follows:

- layer 5: user programs
- layer 4: buffering for input and output
- layer 3: Process management
- layer 2: memory management
- layer 1: CPU scheduling
- layer 0: hardware

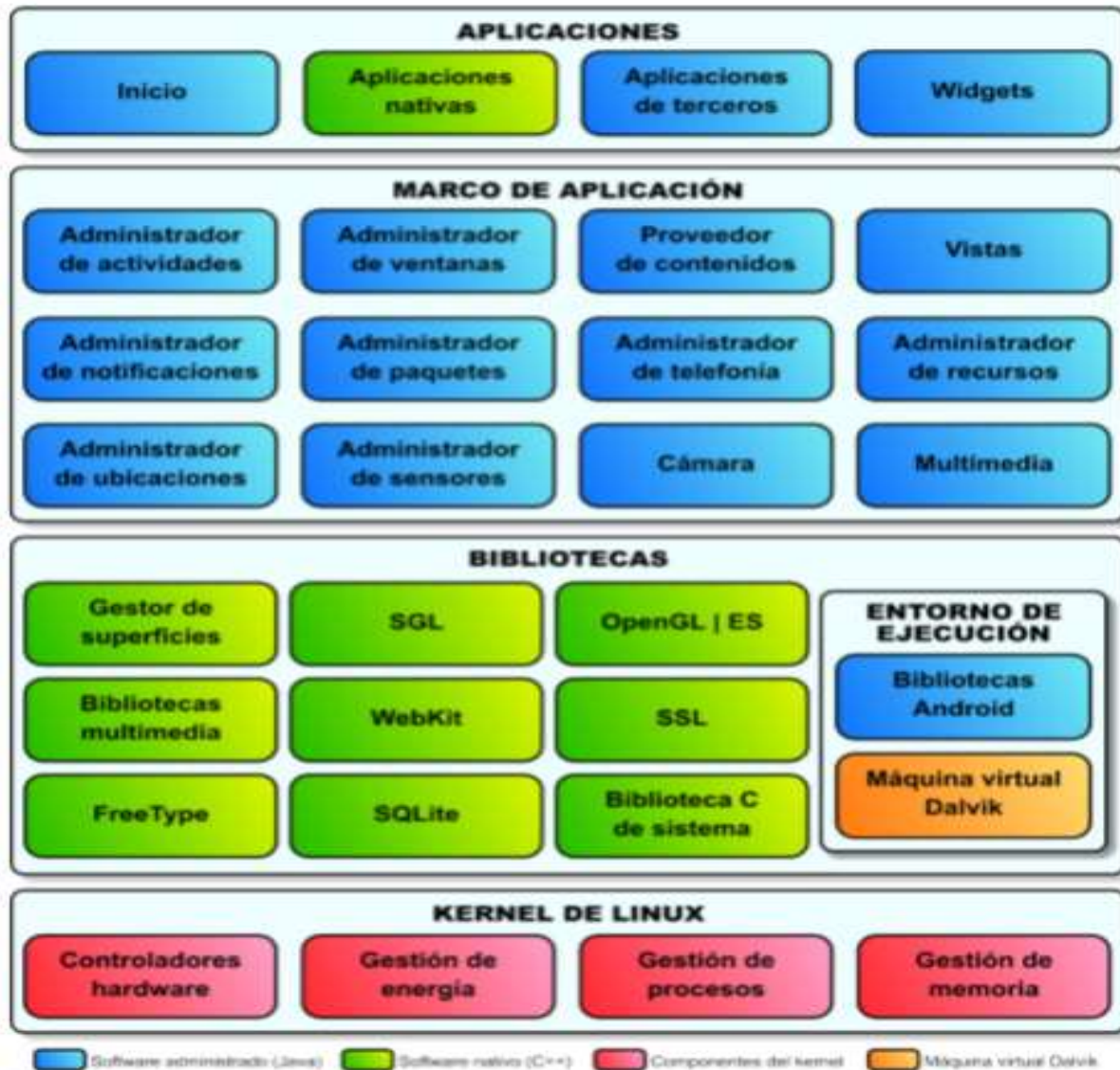


**Ejemplos:** Venus, MULTICS o Android



# SISTEMA OPERATIVO POR CAPAS

Android



# SISTEMA OPERATIVO DE MÁQUINA VIRTUAL

Implementado por primera vez en el **IBM VM/370**

Consiste en separar los dos componentes principales de un sistema de tiempo compartido:

- Multiprogramación
- Una máquina extendida con una interfaz más cómoda que el hardware desnudo



# SISTEMA OPERATIVO DE MÁQUINA VIRTUAL

El corazón del sistema es el **monitor de máquina virtual**.

Este proporciona varias máquinas virtuales a la capa superior que son **copias exactas** del hardware desnudo.

Cada máquina virtual es **idéntica al hardware verdadero**.

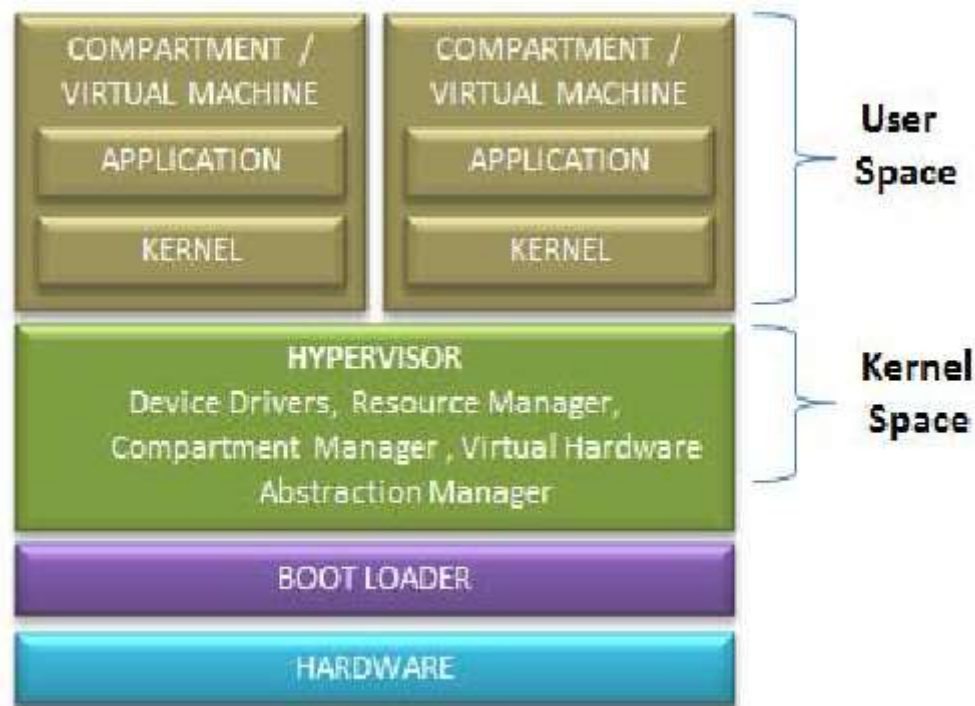
Diferentes máquinas virtuales pueden ejecutar diferentes sistemas operativos.

# SISTEMA OPERATIVO DE MÁQUINA VIRTUAL

Otro ejemplo es el **modo 8086 virtual** disponible en los procesadores de 32 bits de Intel que permite ejecutar antiguos programas de MS-DOS

# SISTEMA OPERATIVO DE EXOKERNEL

Va más allá de los sistemas operativos de máquina virtual, proporcionando a cada usuario un **clon de la computadora real** con un subconjunto de los recursos.



# SISTEMA OPERATIVO MODULAR

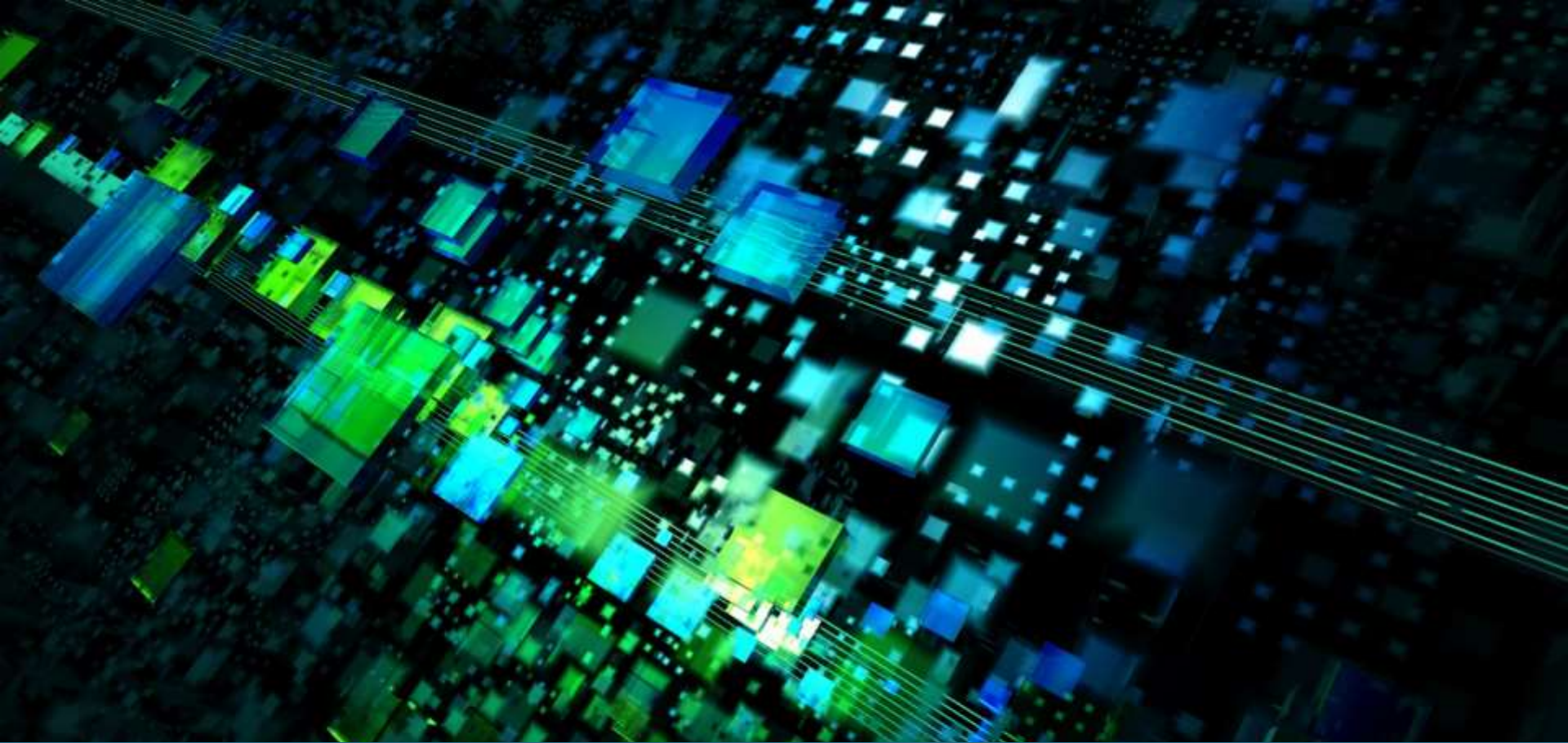
Es el enfoque de la mayoría de sistemas operativos modernos.

El núcleo está formado por **módulos independientes** entre sí.

Los módulos se cargan según se necesiten, ya sea en tiempo de ejecución o durante el arranque.

**Ejemplos:** Linux, Solaris





# 6

# CLASIFICACIÓN DE LOS SISTEMAS OPERATIVOS

# CLASIFICACIÓN DE LOS SISTEMAS OPERATIVOS

Hay diferentes formas de clasificar los sistemas operativos:

- En función del número de usuarios
- En función del número de procesos simultáneos
- En función de los requerimientos temporales
- En función del modo de ofrecer los servicios

# EN FUNCIÓN DEL NÚMERO DE USUARIOS

En este caso pueden ser:

- **Monousuario:** solo soportan el trabajo de un usuario que dispondrá de todos los recursos del sistema. **Ejemplos:** MS-DOS, Windows XP
- **Multiusuario:** son capaces de dar servicio a varios usuarios de manera que compartan los recursos del sistema. **Ejemplos:** UNIX, Linux , Windows Server.

# EN FUNCIÓN DEL NÚMERO DE PROCESOS SIMULTÁNEOS

Pueden ser:

- **Monotarea o monoprogramación:** solo pueden ejecutar una tarea cada vez y cuando terminan pasan a la siguiente.
- **Multitarea o multiprogramación:** se pueden ejecutar varias tareas simultáneamente. La concurrencia es aparente ya que las tareas se van **turnando** en el uso del procesador.

# EN FUNCIÓN DEL NÚMERO DE PROCESADORES SIMULTÁNEOS

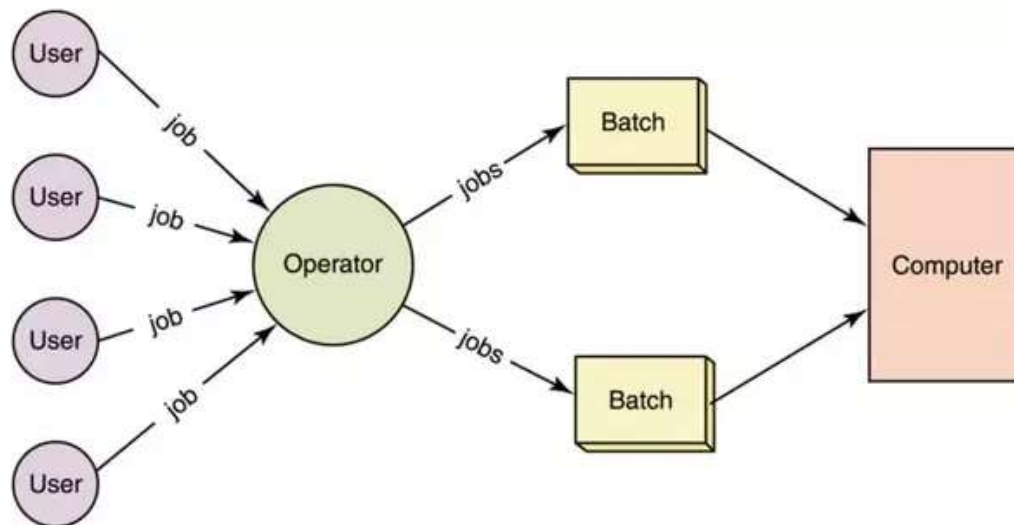
Pueden ser:

- **Monoprocesador:** trabajan con un único procesador sobre el que se van alternando todos los trabajos.
- **Multiprocesador:** trabajan con varios procesadores, por lo que pueden desarrollar varias tareas simultáneamente.

# EN FUNCIÓN DE LOS REQUERIMIENTOS TEMPORALES

Pueden ser:

- **Procesos por lotes (*batch*)**: cada trabajo ocupa totalmente la CPU y cuando finaliza pasa al siguiente trabajo. Permite la ejecución de varios programas sin la supervisión del usuario.





# EN FUNCIÓN DE LOS REQUERIMIENTOS TEMPORALES

- **Procesos con respuesta en tiempo:**
  - **Tiempo real:** muchas operaciones deben cumplirse en plazos estrictos. A su vez pueden ser:
    - *Tiempo real riguroso* si el plazo es ineludible
    - *Tiempo real no riguroso*, si es aceptable no cumplir de vez en cuando algunos plazos.  
Ejemplo: VXWorks o QNX

# EN FUNCIÓN DE LOS REQUERIMIENTOS TEMPORALES

- **Sistemas interactivos:**
  - Deben responder en tiempo las peticiones del usuario, pero el condicionante del tiempo no es tan vital.
  - Utilizan técnicas de multiprogramación para atender varias peticiones de forma simultánea.
  - Ejemplos: Windows, Linux, ...

# EN FUNCIÓN DEL MODO DE OFRECER LOS SERVICIOS

Pueden ser:

- **Sistemas centralizados:**
  - Un ordenador se encarga de todo el trabajo de procesamiento.
  - A él se conectan los terminales desde donde los usuarios envían sus trabajos.
  - Ejemplos: MULTICS, OS/390, universeOS

# EN FUNCIÓN DEL MODO DE OFRECER LOS SERVICIOS



# EN FUNCIÓN DEL MODO DE OFRECER LOS SERVICIOS

- **Sistemas en red:**
  - Mantienen varios equipos conectados entre sí para compartir información y recursos
  - Cada equipo mantiene su autonomía en cuanto a capacidad de proceso
  - **Ejemplos:** Novell Netware, Windows Server, ...

# EN FUNCIÓN DEL MODO DE OFRECER LOS SERVICIOS

- **Sistemas distribuidos:**
  - Diferentes equipos interconectados que se reparten los trabajos de forma transparente para el usuario.
  - **Ejemplos:** Amoeba, Sprite

