

Introducción a los sistemas operativos

Diego Fernández Slezak

Departamento de Computación, FCEyN,
Universidad de Buenos Aires, Buenos Aires, Argentina

Sistemas Operativos, primer cuatrimestre de 2024

(2) Créditos

- Material original de las teóricas preparado por F. Schapachnik.

(2) Créditos

- Material original de las teóricas preparado por F. Schapachnik.
- Ampliado y actualizado por R. Baader, D. Fernández Slezak y S. Yovine.


(2) Créditos

- Material original de las teóricas preparado por F. Schapachnik.
- Ampliado y actualizado por R. Baader, D. Fernández Slezak y S. Yovine.
- Gráficos extraídos de distintas fuentes, la internés, bibliografía de la materia, etc.


(3) Algunas aclaraciones preliminares

- Cosas importantes (tal vez no las únicas): 


(3) Algunas aclaraciones preliminares

- Cosas importantes (tal vez no las únicas): 
- Diapos numeradas.


(3) Algunas aclaraciones preliminares

- Cosas importantes (tal vez no las únicas): 
- Diapos numeradas.
- Su NO pregunta SÍ molesta.


(3) Algunas aclaraciones preliminares

- Cosas importantes (tal vez no las únicas): 
- Diapos numeradas.
- Su NO pregunta SÍ molesta.
- Las siguientes cosas no son equivalentes (de a pares):


(3) Algunas aclaraciones preliminares

- Cosas importantes (tal vez no las únicas): 
- Diapos numeradas.
- Su **NO** pregunta **SÍ** molesta.
- Las siguientes cosas no son equivalentes (de a pares):
 - ① Presenciar esta clase.


(3) Algunas aclaraciones preliminares

- Cosas importantes (tal vez no las únicas): 
- Diapos numeradas.
- Su NO pregunta SÍ molesta.
- Las siguientes cosas no son equivalentes (de a pares):
 - 1 Presenciar esta clase.
 - 2 Leer los apuntes.


(3) Algunas aclaraciones preliminares

- Cosas importantes (tal vez no las únicas): 
- Diapos numeradas.
- Su NO pregunta SÍ molesta.
- Las siguientes cosas no son equivalentes (de a pares):
 - 1 Presenciar esta clase.
 - 2 Leer los apuntes.
 - 3 Presenciar las clases prácticas sobre el tema.


(3) Algunas aclaraciones preliminares

- Cosas importantes (tal vez no las únicas): 
- Diapos numeradas.
- Su NO pregunta SÍ molesta.
- Las siguientes cosas no son equivalentes (de a pares):
 - 1 Presenciar esta clase.
 - 2 Leer los apuntes.
 - 3 Presenciar las clases prácticas sobre el tema.
 - 4 Hacer los talleres.


(3) Algunas aclaraciones preliminares

- Cosas importantes (tal vez no las únicas): 
- Diapos numeradas.
- Su NO pregunta SÍ molesta.
- Las siguientes cosas no son equivalentes (de a pares):
 - 1 Presenciar esta clase.
 - 2 Leer los apuntes.
 - 3 Presenciar las clases prácticas sobre el tema.
 - 4 Hacer los talleres.
 - 5 Hacer las prácticas.


(3) Algunas aclaraciones preliminares

- Cosas importantes (tal vez no las únicas): 
- Diapos numeradas.
- Su **NO** pregunta **SÍ** molesta.
- Las siguientes cosas no son equivalentes (de a pares):
 - 1 Presenciar esta clase.
 - 2 Leer los apuntes.
 - 3 Presenciar las clases prácticas sobre el tema.
 - 4 Hacer los talleres.
 - 5 Hacer las prácticas.
- ¿Cómo sé si entendí los temas?

(3) Algunas aclaraciones preliminares

- Cosas importantes (tal vez no las únicas): 
- Diapos numeradas.
- Su **NO** pregunta **SÍ** molesta.
- Las siguientes cosas no son equivalentes (de a pares):
 - 1 Presenciar esta clase.
 - 2 Leer los apuntes.
 - 3 Presenciar las clases prácticas sobre el tema.
 - 4 Hacer los talleres.
 - 5 Hacer las prácticas.
- ¿Cómo sé si entendí los temas?
 - Los prácticos: si me salen los ejercicios (en un tiempo razonable).

(3) Algunas aclaraciones preliminares

- Cosas importantes (tal vez no las únicas): 
- Diapos numeradas.
- Su **NO** pregunta **SÍ** molesta.
- Las siguientes cosas no son equivalentes (de a pares):
 - 1 Presenciar esta clase.
 - 2 Leer los apuntes.
 - 3 Presenciar las clases prácticas sobre el tema.
 - 4 Hacer los talleres.
 - 5 Hacer las prácticas.
- ¿Cómo sé si entendí los temas?
 - Los prácticos: si me salen los ejercicios (en un tiempo razonable).
 - Los teóricos: si soy capaz de explicarlos con mis propias palabras.

(4) Algunas aclaraciones preliminares (cont.)

- Bibliografía

(4) Algunas aclaraciones preliminares (cont.)

- Bibliografía
 - Silberschatz, A. and Galvin, P.B. and Gagne, G., *Operating system concepts*, Addison-Wesley.

(4) Algunas aclaraciones preliminares (cont.)

- Bibliografía

- Silberschatz, A. and Galvin, P.B. and Gagne, G., *Operating system concepts*, Addison-Wesley.
- Tanenbaum, A.S., *Modern operating systems*, Prentice Hall New Jersey.

(4) Algunas aclaraciones preliminares (cont.)

- Bibliografía
 - Silberschatz, A. and Galvin, P.B. and Gagne, G., *Operating system concepts*, Addison-Wesley.
 - Tanenbaum, A.S., *Modern operating systems*, Prentice Hall New Jersey.
- Temas de la materia

(4) Algunas aclaraciones preliminares (cont.)

- Bibliografía
 - Silberschatz, A. and Galvin, P.B. and Gagne, G., *Operating system concepts*, Addison-Wesley.
 - Tanenbaum, A.S., *Modern operating systems*, Prentice Hall New Jersey.
- Temas de la materia
 - *Diseño de sistemas operativos.*

(4) Algunas aclaraciones preliminares (cont.)

- Bibliografía
 - Silberschatz, A. and Galvin, P.B. and Gagne, G., *Operating system concepts*, Addison-Wesley.
 - Tanenbaum, A.S., *Modern operating systems*, Prentice Hall New Jersey.
- Temas de la materia
 - *Diseño* de sistemas operativos.
 - Y su *periferia*.

(4) Algunas aclaraciones preliminares (cont.)

- Bibliografía
 - Silberschatz, A. and Galvin, P.B. and Gagne, G., *Operating system concepts*, Addison-Wesley.
 - Tanenbaum, A.S., *Modern operating systems*, Prentice Hall New Jersey.
- Temas de la materia
 - *Diseño* de sistemas operativos.
 - Y su periferia.
 - No tanto porque nos interese formar diseñadores de SO...

(4) Algunas aclaraciones preliminares (cont.)

- Bibliografía
 - Silberschatz, A. and Galvin, P.B. and Gagne, G., *Operating system concepts*, Addison-Wesley.
 - Tanenbaum, A.S., *Modern operating systems*, Prentice Hall New Jersey.
- Temas de la materia
 - *Diseño* de sistemas operativos.
 - Y su periferia.
 - No tanto porque nos interese formar diseñadores de SO...
 - Además, los SO son un contexto natural para hablar de paralelismo.

(5) Qué es un SO y por qué dedicarle toda una materia

- Una forma de dividir a los sistemas informáticos:

(5) Qué es un SO y por qué dedicarle toda una materia

- Una forma de dividir a los sistemas informáticos:
 - Hardware

(5) Qué es un SO y por qué dedicarle toda una materia

- Una forma de dividir a los sistemas informáticos:
 - Hardware

(5) Qué es un SO y por qué dedicarle toda una materia

- Una forma de dividir a los sistemas informáticos:
 - Hardware (lo que se puede patear).

(5) Qué es un SO y por qué dedicarle toda una materia

- Una forma de dividir a los sistemas informáticos:
 - Hardware (lo que se puede patear).
 - Software específico


(5) Qué es un SO y por qué dedicarle toda una materia

- Una forma de dividir a los sistemas informáticos:
 - Hardware (lo que se puede patear).
 - Software específico


(5) Qué es un SO y por qué dedicarle toda una materia

- Una forma de dividir a los sistemas informáticos:
 - Hardware (lo que se puede patear).
 - Software específico (lo que sólo se puede putear).


(5) Qué es un SO y por qué dedicarle toda una materia

- Una forma de dividir a los sistemas informáticos:
 - Hardware (lo que se puede patear).
 - Software específico (lo que sólo se puede putear).
- Hace falta un intermediario entre ellos 


(5) Qué es un SO y por qué dedicarle toda una materia

- Una forma de dividir a los sistemas informáticos:
 - Hardware (lo que se puede patear).
 - Software específico (lo que sólo se puede putear).
- Hace falta un intermediario entre ellos 
 - ...para que el software específico no se tenga que preocupar con detalles de bajo nivel del HW (visión de usuario).


(5) Qué es un SO y por qué dedicarle toda una materia

- Una forma de dividir a los sistemas informáticos:
 - Hardware (lo que se puede patear).
 - Software específico (lo que sólo se puede putear).
- Hace falta un intermediario entre ellos 
 - ...para que el software específico no se tenga que preocupar con detalles de bajo nivel del HW (visión de usuario).
 - ...para que el usuario use correctamente el HW (visión del propietario del HW).


(5) Qué es un SO y por qué dedicarle toda una materia

- Una forma de dividir a los sistemas informáticos:
 - Hardware (lo que se puede patear).
 - Software específico (lo que sólo se puede putear).
- Hace falta un intermediario entre ellos 
 - ...para que el software específico no se tenga que preocupar con detalles de bajo nivel del HW (visión de usuario).
 - ...para que el usuario use correctamente el HW (visión del propietario del HW).
- Por qué toda una materia:


(5) Qué es un SO y por qué dedicarle toda una materia

- Una forma de dividir a los sistemas informáticos:
 - Hardware (lo que se puede patear).
 - Software específico (lo que sólo se puede putear).
- Hace falta un intermediario entre ellos 
 - ...para que el software específico no se tenga que preocupar con detalles de bajo nivel del HW (visión de usuario).
 - ...para que el usuario use correctamente el HW (visión del propietario del HW).
- Por qué toda una materia:
 - Porque esta capa es suficientemente específica e interesante como para estudiarla en detalle.

(5) Qué es un SO y por qué dedicarle toda una materia

- Una forma de dividir a los sistemas informáticos:
 - Hardware (lo que se puede patear).
 - Software específico (lo que sólo se puede putear).
- Hace falta un intermediario entre ellos 
 - ...para que el software específico no se tenga que preocupar con detalles de bajo nivel del HW (visión de usuario).
 - ...para que el usuario use correctamente el HW (visión del propietario del HW).
- Por qué toda una materia:
 - Porque esta capa es suficientemente específica e interesante como para estudiarla en detalle.
 - Porque aquí surgen problemas muy interesantes.

(5) Qué es un SO y por qué dedicarle toda una materia

- Una forma de dividir a los sistemas informáticos:
 - Hardware (lo que se puede patear).
 - Software específico (lo que sólo se puede putear).
- Hace falta un intermediario entre ellos 
 - ...para que el software específico no se tenga que preocupar con detalles de bajo nivel del HW (visión de usuario).
 - ...para que el usuario use correctamente el HW (visión del propietario del HW).
- Por qué toda una materia:
 - Porque esta capa es suficientemente específica e interesante como para estudiarla en detalle.
 - Porque aquí surgen problemas muy interesantes.
 - Y porque es el marco “natural” para estudiar algunos de esos problemas, que son de carácter más general.

(6) Un poco de Historia

(6) Un poco de Historia

- Década de 1950: aparecen las primeras computadoras comerciales. IBM 7090.



(6) Un poco de Historia

- Década de 1950: aparecen las primeras computadoras comerciales. IBM 7090.



- 1961: Clementina: primera computadora para fines científicos de Argentina.

(7) Máquinas de la época

- Las computadoras costaban millones de USD y estaban en ambientes dedicados.

(7) Máquinas de la época

- Las computadoras costaban millones de USD y estaban en ambientes dedicados.
- Miren http://www-03.ibm.com/ibm/history/exhibits/mainframe/mainframe_PP7090.html

(7) Máquinas de la época

- Las computadoras costaban millones de USD y estaban en ambientes dedicados.
- Miren http://www-03.ibm.com/ibm/history/exhibits/mainframe/mainframe_PP7090.html
- El usuario llevaba sus tarjetas perforadas en assembler, o típicamente en FORTRAN.

(7) Máquinas de la época

- Las computadoras costaban millones de USD y estaban en ambientes dedicados.
- Miren http://www-03.ibm.com/ibm/history/exhibits/mainframe/mainframe_PP7090.html
- El usuario llevaba sus tarjetas perforadas en assembler, o típicamente en FORTRAN.
- El operador las ponía en la entrada, las tarjetas se leían, el programa corría y luego imprimía el resultado, que el usuario pasaba a buscar después.



(7) Máquinas de la época

- Las computadoras costaban millones de USD y estaban en ambientes dedicados.
- Miren http://www-03.ibm.com/ibm/history/exhibits/mainframe/mainframe_PP7090.html
- El usuario llevaba sus tarjetas perforadas en assembler, o típicamente en FORTRAN.
- El operador las ponía en la entrada, las tarjetas se leían, el programa corría y luego imprimía el resultado, que el usuario pasaba a buscar después.



- Problema:

(7) Máquinas de la época

- Las computadoras costaban millones de USD y estaban en ambientes dedicados.
- Miren http://www-03.ibm.com/ibm/history/exhibits/mainframe/mainframe_PP7090.html
- El usuario llevaba sus tarjetas perforadas en assembler, o típicamente en FORTRAN.
- El operador las ponía en la entrada, las tarjetas se leían, el programa corría y luego imprimía el resultado, que el usuario pasaba a buscar después.



- Problema:

(7) Máquinas de la época

- Las computadoras costaban millones de USD y estaban en ambientes dedicados.
- Miren http://www-03.ibm.com/ibm/history/exhibits/mainframe/mainframe_PP7090.html
- El usuario llevaba sus tarjetas perforadas en assembler, o típicamente en FORTRAN.
- El operador las ponía en la entrada, las tarjetas se leían, el programa corría y luego imprimía el resultado, que el usuario pasaba a buscar después.



- Problema: mucho tiempo de procesamiento desperdiciado.

(7) Máquinas de la época

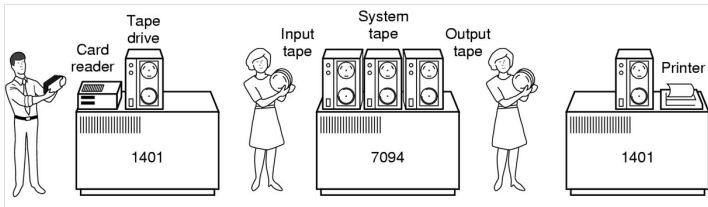
- Las computadoras costaban millones de USD y estaban en ambientes dedicados.
- Miren http://www-03.ibm.com/ibm/history/exhibits/mainframe/mainframe_PP7090.html
- El usuario llevaba sus tarjetas perforadas en assembler, o típicamente en FORTRAN.
- El operador las ponía en la entrada, las tarjetas se leían, el programa corría y luego imprimía el resultado, que el usuario pasaba a buscar después.



- Problema: mucho tiempo de procesamiento desperdiciado.

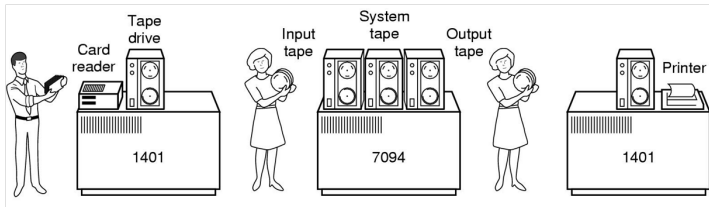
(8) Máquinas de la época (cont.)

- Solución: unas computadoras más baratas, que sólo costaban miles de dólares, eran adicionadas al sistema. Sabían hacer tarjeta → cinta, cinta → impresora. Los mainframes aprendieron a leer y escribir cintas magnéticas (mucho más rápido).



(8) Máquinas de la época (cont.)

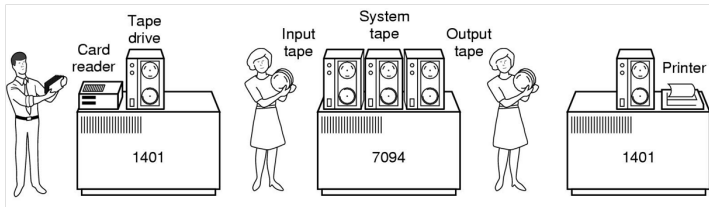
- Solución: unas computadoras más baratas, que sólo costaban miles de dólares, eran adicionadas al sistema. Sabían hacer tarjeta → cinta, cinta → impresora. Los mainframes aprendieron a leer y escribir cintas magnéticas (mucho más rápido).



- Estos sistemas se conocieron como *sistemas batch*, porque no se los manejaba interactivamente como hoy, sino que los programas se les daban en tandas (cada cinta).

(8) Máquinas de la época (cont.)

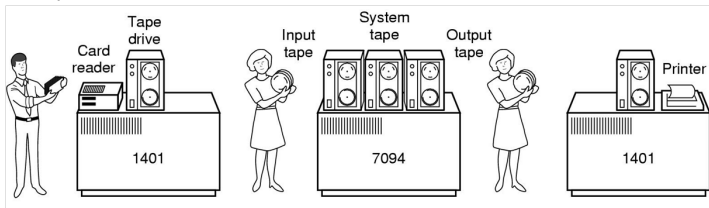
- Solución: unas computadoras más baratas, que sólo costaban miles de dólares, eran adicionadas al sistema. Sabían hacer tarjeta → cinta, cinta → impresora. Los mainframes aprendieron a leer y escribir cintas magnéticas (mucho más rápido).



- Estos sistemas se conocieron como *sistemas batch*, porque no se los manejaba interactivamente como hoy, sino que los programas se les daban en tandas (cada cinta).
- ¿Quién hacía la intermediación usuario/HW?

(8) Máquinas de la época (cont.)

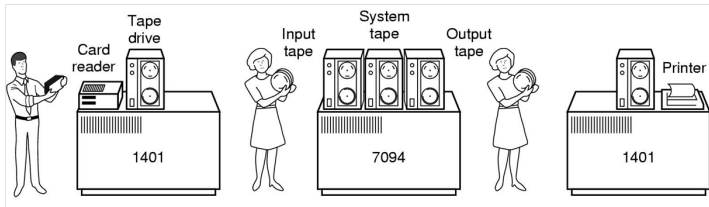
- Solución: unas computadoras más baratas, que sólo costaban miles de dólares, eran adicionadas al sistema. Sabían hacer tarjeta → cinta, cinta → impresora. Los mainframes aprendieron a leer y escribir cintas magnéticas (mucho más rápido).



- Estos sistemas se conocieron como *sistemas batch*, porque no se los manejaba interactivamente como hoy, sino que los programas se les daban en tandas (cada cinta).
- ¿Quién hacía la intermediación usuario/HW?

(8) Máquinas de la época (cont.)

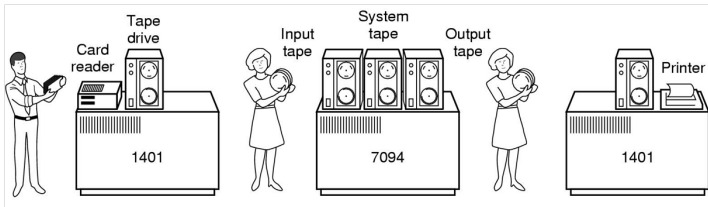
- Solución: unas computadoras más baratas, que sólo costaban miles de dólares, eran adicionadas al sistema. Sabían hacer tarjeta → cinta, cinta → impresora. Los mainframes aprendieron a leer y escribir cintas magnéticas (mucho más rápido).



- Estos sistemas se conocieron como *sistemas batch*, porque no se los manejaba interactivamente como hoy, sino que los programas se les daban en tandas (cada cinta).
- ¿Quién hacía la intermediación usuario/HW? El operador.

(8) Máquinas de la época (cont.)

- Solución: unas computadoras más baratas, que sólo costaban miles de dólares, eran adicionadas al sistema. Sabían hacer tarjeta → cinta, cinta → impresora. Los mainframes aprendieron a leer y escribir cintas magnéticas (mucho más rápido).



- Estos sistemas se conocieron como *sistemas batch*, porque no se los manejaba interactivamente como hoy, sino que los programas se les daban en tandas (cada cinta).
- ¿Quién hacía la intermediación usuario/HW? El operador.
- Queda planteada la **primer preocupación importante para los SO**: cada usuario sólo debe recibir sus impresiones.

(9) Unos años más tarde...

- La siguiente generación de computadoras, caracterizada por máquinas como la IBM/360, ya tenían un sistema operativo más formal (OS/360 en ese caso).

(9) Unos años más tarde...

- La siguiente generación de computadoras, caracterizada por máquinas como la IBM/360, ya tenían un sistema operativo más formal (OS/360 en ese caso).
- Pretendían solucionar un problema importante: mientras se leían las cintas en memoria, el procesador estaba ocioso. Ídem cuando se escribía el resultado.


(9) Unos años más tarde...

- La siguiente generación de computadoras, caracterizada por máquinas como la IBM/360, ya tenían un sistema operativo más formal (OS/360 en ese caso).
- Pretendían solucionar un problema importante: mientras se leían las cintas en memoria, el procesador estaba ocioso. Ídem cuando se escribía el resultado.
- Esto es muy indeseable, porque es caro.

(9) Unos años más tarde...

- La siguiente generación de computadoras, caracterizada por máquinas como la IBM/360, ya tenían un sistema operativo más formal (OS/360 en ese caso).
- Pretendían solucionar un problema importante: mientras se leían las cintas en memoria, el procesador estaba ocioso. Ídem cuando se escribía el resultado.
- Esto es muy indeseable, porque es caro.
- Idea: mientras se accede a los dispositivos, que el procesador procese otro trabajo, aunque sea un pedacito.

(10) Unos años más tarde... (cont.)

- Nace el concepto de *multiprogramación*. De esta manera, el *throughput* o *rendimiento* aumenta. El trabajo j_1 toma el mismo tiempo que antes, o incluso un poco más, pero $j_1 + j_2$ tarda menos. 

(10) Unos años más tarde... (cont.)

- Nace el concepto de *multiprogramación*. De esta manera, el *throughput* o *rendimiento* aumenta. El trabajo j_1 toma el mismo tiempo que antes, o incluso un poco más, pero $j_1 + j_2$ tarda menos. ⚠
- Con él, otro concepto fundamental: la *contención*. ⚠ Varios programas pueden querer acceder a un mismo recurso a la vez.

(11) Poco después...

- Usar los sistemas batch era un bajón, especialmente para programar y debuggear. Por eso surgió la idea de conectar muchas terminales a una misma computadora, y darles un poquito de tiempo de procesador a las que están siendo usadas.

(11) Poco después...

- Usar los sistemas batch era un bajón, especialmente para programar y debuggear. Por eso surgió la idea de conectar muchas terminales a una misma computadora, y darles un poquito de tiempo de procesador a las que están siendo usadas.
- Eso se llama *timesharing*, y es una variación de la multiprogramación.

(11) Poco después...

- Usar los sistemas batch era un bajón, especialmente para programar y debuggear. Por eso surgió la idea de conectar muchas terminales a una misma computadora, y darles un poquito de tiempo de procesador a las que están siendo usadas.
- Eso se llama *timesharing*, y es una variación de la multiprogramación.
- El pionero fue MULTICS, del que descende UNIX (Ken Thompson, Dennis Ritchie).

(11) Poco después...

- Usar los sistemas batch era un bajón, especialmente para programar y debuggear. Por eso surgió la idea de conectar muchas terminales a una misma computadora, y darles un poquito de tiempo de procesador a las que están siendo usadas.
- Eso se llama *timesharing*, y es una variación de la multiprogramación.
- El pionero fue MULTICS, del que descende UNIX (Ken Thompson, Dennis Ritchie).
- UNIX es fundamental en la historia de los SO, tanto por los conceptos que introdujo como por su vigencia.

(11) Poco después...

- Usar los sistemas batch era un bajón, especialmente para programar y debuggear. Por eso surgió la idea de conectar muchas terminales a una misma computadora, y darles un poquito de tiempo de procesador a las que están siendo usadas.
- Eso se llama *timesharing*, y es una variación de la multiprogramación.
- El pionero fue MULTICS, del que descende UNIX (Ken Thompson, Dennis Ritchie).
- UNIX es fundamental en la historia de los SO, tanto por los conceptos que introdujo como por su vigencia.
- Linux está inspirado en UNIX.

(11) Poco después...

- Usar los sistemas batch era un bajón, especialmente para programar y debuggear. Por eso surgió la idea de conectar muchas terminales a una misma computadora, y darles un poquito de tiempo de procesador a las que están siendo usadas.
- Eso se llama *timesharing*, y es una variación de la multiprogramación.
- El pionero fue MULTICS, del que descende UNIX (Ken Thompson, Dennis Ritchie).
- UNIX es fundamental en la historia de los SO, tanto por los conceptos que introdujo como por su vigencia.
- Linux está inspirado en UNIX.
- Si les interesa su historia (muy divertida):
<https://www.bell-labs.com/usr/dmr/www/hist.html>

(12) Veníamos diciendo...

- Recordemos:

(12) Veníamos diciendo...

- Recordemos:
 - ...para que el software específico no se tenga que preocupar con detalles de bajo nivel del HW (visión de usuario).

(12) Veníamos diciendo...

- Recordemos:
 - ...para que el software específico no se tenga que preocupar con detalles de bajo nivel del HW (visión de usuario).
 - ...para que el usuario use correctamente el HW (visión del propietario del HW).

(12) Veníamos diciendo...

- Recordemos:
 - ...para que el software específico no se tenga que preocupar con detalles de bajo nivel del HW (visión de usuario).
 - ...para que el usuario use correctamente el HW (visión del propietario del HW).
- Vimos un poco la segunda parte. Entendamos la primera.

(13) Leer un sector de un disquette...

- Forma de hablar directamente con el HW, para hacer una lectura de una disquettera:

(13) Leer un sector de un disquette...

- Forma de hablar directamente con el HW, para hacer una lectura de una disquettera:
- Empaquetar 13 parámetros en 9 bytes (dirección del bloque, sectores por track, modo de acceso físico, separación entre sectores, etc.).

(13) Leer un sector de un disquette...

- Forma de hablar directamente con el HW, para hacer una lectura de una disquettera:
- Empaquetar 13 parámetros en 9 bytes (dirección del bloque, sectores por track, modo de acceso físico, separación entre sectores, etc.).
- Esperar un rato (hay que saber cuánto).

(13) Leer un sector de un disquette...

- Forma de hablar directamente con el HW, para hacer una lectura de una disquettera:
- Empaquetar 13 parámetros en 9 bytes (dirección del bloque, sectores por track, modo de acceso físico, separación entre sectores, etc.).
- Esperar un rato (hay que saber cuánto).
- El controlador del floppy devuelve 23 campos de status y error empaquetados en 7 bytes.

(13) Leer un sector de un disquette...

- Forma de hablar directamente con el HW, para hacer una lectura de una disquettera:
- Empaquetar 13 parámetros en 9 bytes (dirección del bloque, sectores por track, modo de acceso físico, separación entre sectores, etc.).
- Esperar un rato (hay que saber cuánto).
- El controlador del floppy devuelve 23 campos de status y error empaquetados en 7 bytes.
- Además, hay que prender y apagar explícitamente el motor, porque si se lo deja prendido los disquettes se desgastan.

(13) Leer un sector de un disquette...

- Forma de hablar directamente con el HW, para hacer una lectura de una disquettera:
- Empaquetar 13 parámetros en 9 bytes (dirección del bloque, sectores por track, modo de acceso físico, separación entre sectores, etc.).
- Esperar un rato (hay que saber cuánto).
- El controlador del floppy devuelve 23 campos de status y error empaquetados en 7 bytes.
- Además, hay que prender y apagar explícitamente el motor, porque si se lo deja prendido los disquettes se desgastan.
- Y así.

(13) Leer un sector de un disquette...

- Forma de hablar directamente con el HW, para hacer una lectura de una disquettera:
- Empaquetar 13 parámetros en 9 bytes (dirección del bloque, sectores por track, modo de acceso físico, separación entre sectores, etc.).
- Esperar un rato (hay que saber cuánto).
- El controlador del floppy devuelve 23 campos de status y error empaquetados en 7 bytes.
- Además, hay que prender y apagar explícitamente el motor, porque si se lo deja prendido los disquettes se desgastan.
- Y así.
- El resto del HW no es más amigable.

(13) Leer un sector de un disquette...

- Forma de hablar directamente con el HW, para hacer una lectura de una disquettera:
- Empaquetar 13 parámetros en 9 bytes (dirección del bloque, sectores por track, modo de acceso físico, separación entre sectores, etc.).
- Esperar un rato (hay que saber cuánto).
- El controlador del floppy devuelve 23 campos de status y error empaquetados en 7 bytes.
- Además, hay que prender y apagar explícitamente el motor, porque si se lo deja prendido los disquettes se desgastan.
- Y así.
- El resto del HW no es más amigable.
- Tarea para el hogar: tomar una computadora sin SO y hacer un programa que lea nombres del disco, los ordene y los vuelva a escribir ordenados.

(13) Leer un sector de un disquette...

- Forma de hablar directamente con el HW, para hacer una lectura de una disquettera:
- Empaquetar 13 parámetros en 9 bytes (dirección del bloque, sectores por track, modo de acceso físico, separación entre sectores, etc.).
- Esperar un rato (hay que saber cuánto).
- El controlador del floppy devuelve 23 campos de status y error empaquetados en 7 bytes.
- Además, hay que prender y apagar explícitamente el motor, porque si se lo deja prendido los disquettes se desgastan.
- Y así.
- El resto del HW no es más amigable.
- Tarea para el hogar: tomar una computadora sin SO y hacer un programa que lea nombres del disco, los ordene y los vuelva a escribir ordenados.
- Con SO: 200 LOC.

(13) Leer un sector de un disquette...

- Forma de hablar directamente con el HW, para hacer una lectura de una disquettera:
- Empaquetar 13 parámetros en 9 bytes (dirección del bloque, sectores por track, modo de acceso físico, separación entre sectores, etc.).
- Esperar un rato (hay que saber cuánto).
- El controlador del floppy devuelve 23 campos de status y error empaquetados en 7 bytes.
- Además, hay que prender y apagar explícitamente el motor, porque si se lo deja prendido los disquettes se desgastan.
- Y así.
- El resto del HW no es más amigable.
- Tarea para el hogar: tomar una computadora sin SO y hacer un programa que lea nombres del disco, los ordene y los vuelva a escribir ordenados.
- Con SO: 200 LOC.
- Sin SO: para cortar... cualquier tipo de inspiración.

(14) Pasando en limpio...

- Un SO es una pieza de software que hace de intermediario entre el HW y los programas de usuario.

(14) Pasando en limpio...

- Un SO es una pieza de software que hace de intermediario entre el HW y los programas de usuario.
- Tiene que manejar la contención y la concurrencia de manera tal de lograr:


(14) Pasando en limpio...

- Un SO es una pieza de software que hace de intermediario entre el HW y los programas de usuario.
- Tiene que manejar la contención y la concurrencia de manera tal de lograr:
 - Hacerlo con buen rendimiento.


(14) Pasando en limpio...

- Un SO es una pieza de software que hace de intermediario entre el HW y los programas de usuario.
- Tiene que manejar la contención y la concurrencia de manera tal de lograr:
 - Hacerlo con buen rendimiento.
 - Hacerlo correctamente.

(14) Pasando en limpio...

- Un SO es una pieza de software que hace de intermediario entre el HW y los programas de usuario.
- Tiene que manejar la contención y la concurrencia de manera tal de lograr:
 - Hacerlo con buen rendimiento.
 - Hacerlo correctamente.
- Esto es un problema central. 

(14) Pasando en limpio...

- Un SO es una pieza de software que hace de intermediario entre el HW y los programas de usuario.
- Tiene que manejar la contención y la concurrencia de manera tal de lograr:
 - Hacerlo con buen rendimiento.
 - Hacerlo correctamente.
- Esto es un problema central. 
- Para lograr todo esto, corre en *nivel de privilegio 0*, es decir, *máximo privilegio*.

(15) Qué es y qué no es un SO

- Veamos cuánto ocupan algunos sistemas operativos:

(15) Qué es y qué no es un SO

- Veamos cuánto ocupan algunos sistemas operativos:
- Ubuntu 16.04.4 LTS: “25 GB of free hard drive space”

(15) Qué es y qué no es un SO

- Veamos cuánto ocupan algunos sistemas operativos:
- Ubuntu 16.04.4 LTS: “25 GB of free hard drive space”
- FreeBSD 7.2: 5 CDs de instalación.

(15) Qué es y qué no es un SO

- Veamos cuánto ocupan algunos sistemas operativos:
- Ubuntu 16.04.4 LTS: “25 GB of free hard drive space”
- FreeBSD 7.2: 5 CDs de instalación.
- Windows Seven: 16 GB para la edición “home basic”, 40 GB para las otras.

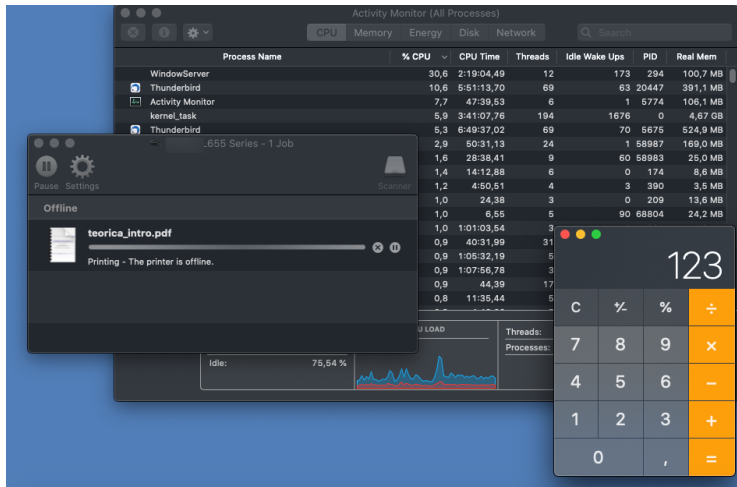
(15) Qué es y qué no es un SO

- Veamos cuánto ocupan algunos sistemas operativos:
- Ubuntu 16.04.4 LTS: “25 GB of free hard drive space”
- FreeBSD 7.2: 5 CDs de instalación.
- Windows Seven: 16 GB para la edición “home basic”, 40 GB para las otras.
- ¡¿Todo eso es necesario?!

(16) Qué es y qué no es un SO (cont.)



(17) Qué es y qué no es un SO (cont.)



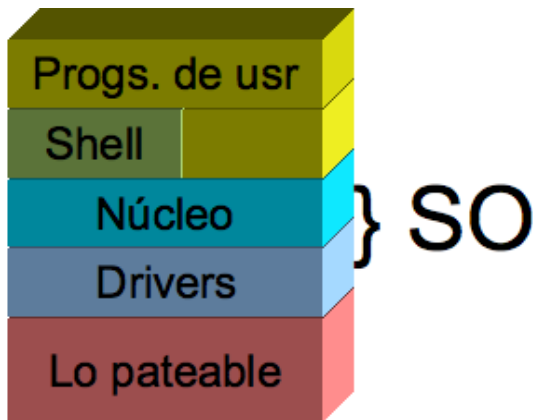
(18) Qué es y qué no es un SO (cont.)



(19) Qué es y qué no es un SO (cont.)

```
/usr/bin% ls -la | head -20
total 88488
drwxr-xr-x  1012 root   wheel   32384 Mar 18 12:15 .
drwxr-xr-x@   12 root   wheel    384 Aug 12 2021 ..
lrwxr-xr-x    1 root   wheel     74 Jun 11 2020 2to3- -> ../../System/Libra
ry/Frameworks/Python.framework/Versions/2.7/bin/2to3-2.7
lrwxr-xr-x    1 root   wheel     74 Jun 11 2020 2to3-2.7 -> ../../System/Li
brary/Frameworks/Python.framework/Versions/2.7/bin/2to3-2.7
-rwxr-xr-x    1 root   wheel   72400 Sep 21 2020 AssetCacheLocatorUtil
-rwxr-xr-x    1 root   wheel   74304 Sep 21 2020 AssetCacheManagerUtil
-rwxr-xr-x    1 root   wheel   58480 Sep 21 2020 AssetCacheTetheratorUtil
-rwxr-xr-x    1 root   wheel   31520 Sep 21 2020 BuildStrings
-rwxr-xr-x    1 root   wheel   31488 Sep 21 2020 CpMac
-rwxr-xr-x    1 root   wheel   31488 Sep 21 2020 DeRez
-rwxr-xr-x    1 root   wheel   31504 Sep 21 2020 GetFileInfo
-rwxr-xr-x    1 root   wheel   86864 Feb 25 06:06 IOAccelMemory
-rwxr-xr-x    1 root   wheel   31488 Sep 21 2020 MergePef
-rwxr-xr-x    1 root   wheel   31488 Sep 21 2020 MvMac
-rwxr-xr-x    1 root   wheel   31488 Sep 21 2020 ResMerger
-rwxr-xr-x    1 root   wheel   31488 Sep 21 2020 Rez
-rwxr-xr-x    1 root   wheel   31488 Sep 21 2020 RezDet
-rwxr-xr-x    1 root   wheel   31488 Sep 21 2020 RezWack
-rwxr-xr-x    1 root   wheel   45664 Sep 21 2020 SafeEjectGPU
/usr/bin%
```


(20) Qué es y qué no es un SO (cont.)



(21) Elementos básicos de un SO

- Drivers: programas que son parte del sistema operativo y manejan los detalles de bajo nivel relacionados con la operación de los distintos dispositivos.

(21) Elementos básicos de un SO

- Drivers: programas que son parte del sistema operativo y manejan los detalles de bajo nivel relacionados con la operación de los distintos dispositivos.
- Núcleo o Kernel: es el SO propiamente dicho, su parte central. Se encarga de las tareas fundamentales y contiene los diversos subsistemas que iremos viendo en la materia.

(21) Elementos básicos de un SO

- Drivers: programas que son parte del sistema operativo y manejan los detalles de bajo nivel relacionados con la operación de los distintos dispositivos.
- Núcleo o Kernel: es el SO propiamente dicho, su parte central. Se encarga de las tareas fundamentales y contiene los diversos subsistemas que iremos viendo en la materia.
- Intérprete de comandos o Shell: un programa más, que muchas veces es ejecutado automáticamente cuando comienza el SO, que le permite al usuario interactuar con el SO. Puede ser gráfico o de línea de comandos. Ejemplos en Unix: `sh`, `csh`, `ksh`, `bash`.

(21) Elementos básicos de un SO

- Drivers: programas que son parte del sistema operativo y manejan los detalles de bajo nivel relacionados con la operación de los distintos dispositivos.
- Núcleo o Kernel: es el SO propiamente dicho, su parte central. Se encarga de las tareas fundamentales y contiene los diversos subsistemas que iremos viendo en la materia.
- Intérprete de comandos o Shell: un programa más, que muchas veces es ejecutado automáticamente cuando comienza el SO, que le permite al usuario interactuar con el SO. Puede ser gráfico o de línea de comandos. Ejemplos en Unix: `sh`, `csh`, `ksh`, `bash`.
- Proceso: un programa en ejecución más su espacio de memoria asociado y otros atributos.

(22) Elementos básicos de un SO (cont.)

- Archivo: secuencia de bits con un nombre y una serie de atributos que indican permisos, etc.

(22) Elementos básicos de un SO (cont.)

- Archivo: secuencia de bits con un nombre y una serie de atributos que indican permisos, etc.
- Directorio: colección de archivos y directorios que contiene un nombre y se organiza jerárquicamente.

(22) Elementos básicos de un SO (cont.)

- Archivo: secuencia de bits con un nombre y una serie de atributos que indican permisos, etc.
- Directorio: colección de archivos y directorios que contiene un nombre y se organiza jerárquicamente.
- Dispositivo virtual: una abstracción de un dispositivo físico bajo la forma, en general, de un archivo, de manera tal que se pueda abrir, leer, escribir, etc.

(22) Elementos básicos de un SO (cont.)

- Archivo: secuencia de bits con un nombre y una serie de atributos que indican permisos, etc.
- Directorio: colección de archivos y directorios que contiene un nombre y se organiza jerárquicamente.
- Dispositivo virtual: una abstracción de un dispositivo físico bajo la forma, en general, de un archivo, de manera tal que se pueda abrir, leer, escribir, etc.
- Sistema de archivos: es la forma de organizar los datos en el disco para gestionar su acceso, permisos, etc.

(23) Elementos básicos de un SO (cont.)

- Directorios del sistema: son directorios donde el propio SO guarda archivos que necesita para su funcionamiento, como por ejemplo, /boot, /devices o C:\Windows\system32.

(23) Elementos básicos de un SO (cont.)

- Directorios del sistema: son directorios donde el propio SO guarda archivos que necesita para su funcionamiento, como por ejemplo, /boot, /devices o C:\Windows\system32.
- Binario del sistema: son archivos, que viven en los directorios del sistema. Si bien no forman parte del kernel, suelen llevar a cabo tareas muy importantes o proveer las utilidades básicas del sistema. Ejemplo:

(23) Elementos básicos de un SO (cont.)

- Directorios del sistema: son directorios donde el propio SO guarda archivos que necesita para su funcionamiento, como por ejemplo, /boot, /devices o C:\Windows\system32.
- Binario del sistema: son archivos, que viven en los directorios del sistema. Si bien no forman parte del kernel, suelen llevar a cabo tareas muy importantes o proveer las utilidades básicas del sistema. Ejemplo:
 - /usr/sbin/syslogd: es el encargado de guardar los eventos del sistema en un archivo.

(23) Elementos básicos de un SO (cont.)

- Directorios del sistema: son directorios donde el propio SO guarda archivos que necesita para su funcionamiento, como por ejemplo, /boot, /devices o C:\Windows\system32.
- Binario del sistema: son archivos, que viven en los directorios del sistema. Si bien no forman parte del kernel, suelen llevar a cabo tareas muy importantes o proveer las utilidades básicas del sistema. Ejemplo:
 - /usr/sbin/syslogd: es el encargado de guardar los eventos del sistema en un archivo.
 - /bin/sh: el Bourne Shell.

(23) Elementos básicos de un SO (cont.)

- Directorios del sistema: son directorios donde el propio SO guarda archivos que necesita para su funcionamiento, como por ejemplo, /boot, /devices o C:\Windows\system32.
- Binario del sistema: son archivos, que viven en los directorios del sistema. Si bien no forman parte del kernel, suelen llevar a cabo tareas muy importantes o proveer las utilidades básicas del sistema. Ejemplo:
 - /usr/sbin/syslogd: es el encargado de guardar los eventos del sistema en un archivo.
 - /bin/sh: el Bourne Shell.
 - /usr/bin/who: indica qué usuarios están sesionados en el sistema.

(23) Elementos básicos de un SO (cont.)

- Directorios del sistema: son directorios donde el propio SO guarda archivos que necesita para su funcionamiento, como por ejemplo, /boot, /devices o C:\Windows\system32.
- Binario del sistema: son archivos, que viven en los directorios del sistema. Si bien no forman parte del kernel, suelen llevar a cabo tareas muy importantes o proveer las utilidades básicas del sistema. Ejemplo:
 - /usr/sbin/syslogd: es el encargado de guardar los eventos del sistema en un archivo.
 - /bin/sh: el Bourne Shell.
 - /usr/bin/who: indica qué usuarios están sesionados en el sistema.
- Archivo de configuración: es un archivo más, excepto porque el sistema operativo saca de allí información que necesita para funcionar. Por ejemplo, /etc/passwd o C:\Windows\system32\user.dat.

(24) Elementos básicos de un SO (cont.)

- Usuario: la representación, dentro del propio SO, de las personas o entidades que pueden usarlo. Sirve principalmente como una forma de aislar información entre sí y de establecer limitaciones.

(24) Elementos básicos de un SO (cont.)

- Usuario: la representación, dentro del propio SO, de las personas o entidades que pueden usarlo. Sirve principalmente como una forma de aislar información entre sí y de establecer limitaciones.
- Grupo: una colección de usuarios.

(25) ¿Dónde estamos

- Vimos

(25) ● Dónde estamos

- Vimos
 - Qué es un SO.

(25) Dónde estamos

- Vimos
 - Qué es un SO.
 - Un administrador de recursos.

(25) Dónde estamos

- Vimos
 - Qué es un SO.
 - Un administrador de recursos.
 - Una interfaz de programación.

(25) Dónde estamos

- Vimos
 - Qué es un SO.
 - Un administrador de recursos.
 - Una interfaz de programación.
 - Un poco de su evolución histórica.

(25) Dónde estamos

- Vimos
 - Qué es un SO.
 - Un administrador de recursos.
 - Una interfaz de programación.
 - Un poco de su evolución histórica.
 - Su misión fundamental.

(25) Dónde estamos

- Vimos
 - Qué es un SO.
 - Un administrador de recursos.
 - Una interfaz de programación.
 - Un poco de su evolución histórica.
 - Su misión fundamental.
 - **SO batch e interactivos.**

(25) Dónde estamos

- Vimos
 - Qué es un SO.
 - Un administrador de recursos.
 - Una interfaz de programación.
 - Un poco de su evolución histórica.
 - Su misión fundamental.
 - SO batch e interactivos.
 - **Hablamos de multiprogramación.**

(25) Dónde estamos

- Vimos
 - Qué es un SO.
 - Un administrador de recursos.
 - Una interfaz de programación.
 - Un poco de su evolución histórica.
 - Su misión fundamental.
 - SO batch e interactivos.
 - Hablamos de multiprogramación.
 - Qué cosas son parte del SO y cuáles no.

(25) Dónde estamos

- Vimos
 - Qué es un SO.
 - Un administrador de recursos.
 - Una interfaz de programación.
 - Un poco de su evolución histórica.
 - Su misión fundamental.
 - SO batch e interactivos.
 - Hablamos de multiprogramación.
 - Qué cosas son parte del SO y cuáles no.
- La próxima clase:

(25) ● Dónde estamos

- Vimos
 - Qué es un SO.
 - Un administrador de recursos.
 - Una interfaz de programación.
 - Un poco de su evolución histórica.
 - Su misión fundamental.
 - SO batch e interactivos.
 - Hablamos de multiprogramación.
 - Qué cosas son parte del SO y cuáles no.
- La próxima clase:
 - Vamos a empezar a analizar al SO en tanto interfaz de programación y ver qué funcionalidades nos brinda.

(25) Dónde estamos

- Vimos
 - Qué es un SO.
 - Un administrador de recursos.
 - Una interfaz de programación.
 - Un poco de su evolución histórica.
 - Su misión fundamental.
 - SO batch e interactivos.
 - Hablamos de multiprogramación.
 - Qué cosas son parte del SO y cuáles no.
- La próxima clase:
 - Vamos a empezar a analizar al SO en tanto interfaz de programación y ver qué funcionalidades nos brinda.
 - **Vamos a analizar el concepto de proceso en detalle.**