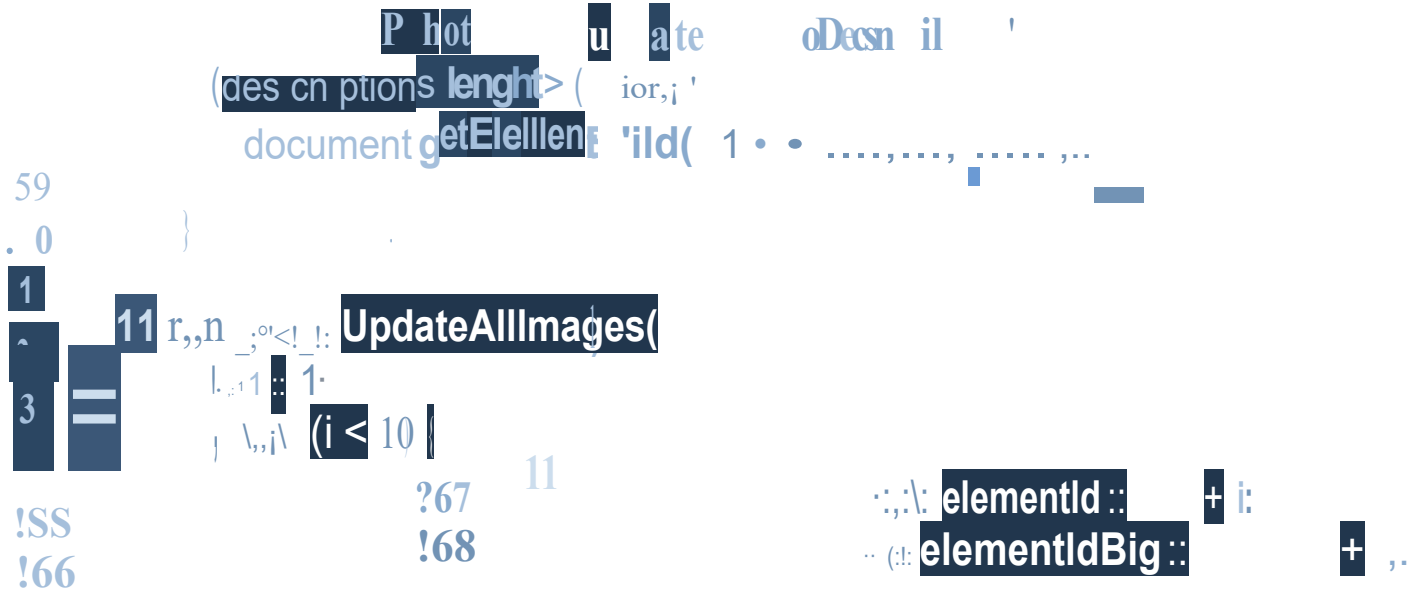
Módulo 2

Sistemas operativos monopuesto



Módulo
Sistemas
monopu

UF 1: Introducción a los sistemas operativos

1. Caracterización de los sistemas operativos, tipos y aplicaciones

1.1. El sistema informático: componentes físicos y lógicos

Según la RAE, un sistema es “un conjunto de cosas que relacionadas entre sí ordenadamente contribuyen a un determinado objeto”.

Un sistema informático es aquel que nos permite **almacenar información**.

SISTEMA INFORMÁTICO. Se compone de tres partes:

HARDWARE **SOFTWARE** **USUARIOS**

- ***Hardware***: son los componentes electrónicos (también denominada parte física). Engloba todas las piezas que componen nuestro ordenador, como, por ejemplo, el procesador o las memorias.
- ***Software***: es la parte virtual. Se considera *software* tanto al sistema operativo como a las aplicaciones que utilizamos.
- **Usuarios**: se engloba tanto a los usuarios del sistema como a los que lo crean y mantienen.

LOS SISTEMAS INFORMÁTICOS PUEDEN **CLASIFICARSE EN BASE A**:

USO | **Funcion. Procesador** | **TIPO DE COMPUTADORA**

USO

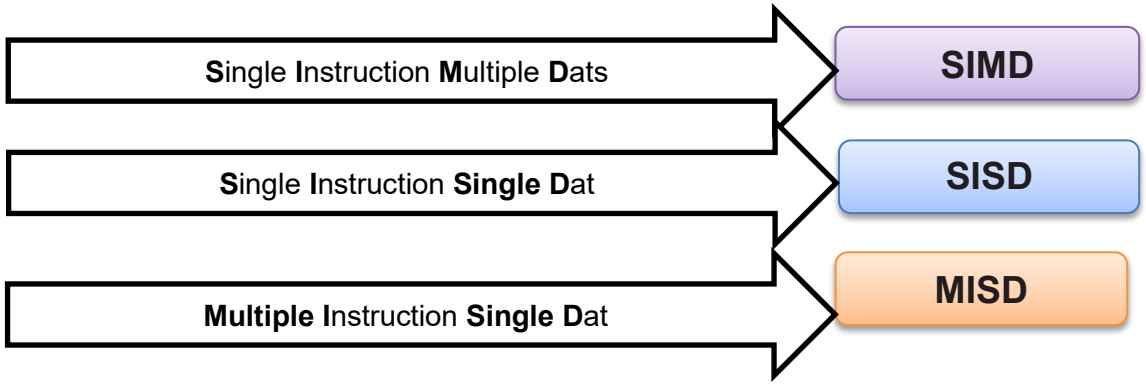
- Sistemas de uso específico
- Sistemas de uso general

Funcion. Procesador

MIMD: significa “múltiples instrucciones, múltiples datos”. Hay varias unidades de control interpretando instrucciones simultáneamente. Los diferentes procesadores funcionan de forma independiente y asíncrona

TIPO DE COMPUTADORA

- Estaciones de trabajo
- Macrocomputadoras
- Minicomputadoras
- Supercomputadoras
- Terminales ligeros



- **SIMD**: significa “una instrucción, múltiples datos”. Computadores que aplican una misma operación a diferentes conjuntos de datos.
- **SISD**: significa “una instrucción, un dato”. La máquina tiene un único procesador con un único flujo de instrucciones.
- **MISD**: sigifica “múltiples instrucciones, un dato”. Existen muchas unidades funcionales que realizan distintas operaciones sobre un único dato.

HARDWARE – (Componentes físicos)

➢ HARDWARE

COMPONENTES **PERIFÉRICOS**

El ***hardware*** es el conjunto de componentes que integran la parte material de una computadora, es decir, los componentes tangibles del sistema.

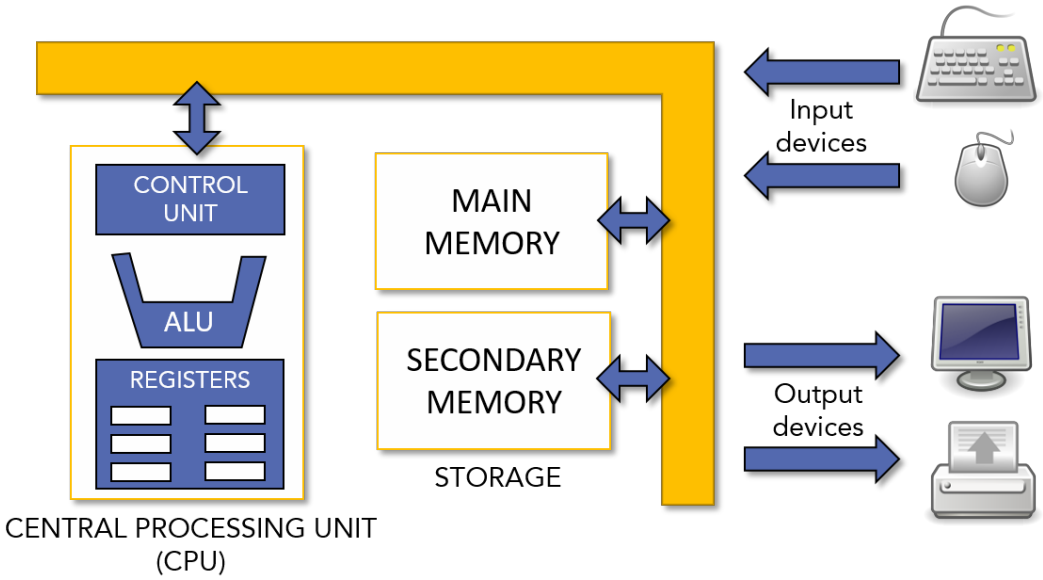
Podemos dividirlo en **componentes** y **periféricos**.

- Los **componentes** son todos aquellos elementos que forman parte del computador y que se encuentran en la torre de los ordenadores de sobremesa. Entre ellos encontramos el disco duro, la placa base, la tarjeta gráfica, etcétera.

Los **periféricos** son los elementos ajenos al sistema que ofrecen funcionalidad al mismo, como el teclado, el ratón o la impresora.

Es necesario que tanto los componentes como los periféricos se comuniquen para que todo funcione correctamente. En las computadoras actuales, se sigue la llamada **arquitectura de Von Neumann**.

ARQUITECTURA DE VON NEUMANN.



Como se puede observar en la imagen, el ordenador consta de dos elementos principales: la CPU y la memoria.

La CPU es el cerebro del ordenador, su función es realizar las operaciones con los diferentes datos. Se compone de dos partes:

➢ CPU

unidad de control (UC) **unidad aritmético-lógica (ALU)**

- La **unidad de control (UC)**: se encarga de buscar las instrucciones en la memoria principal, interpretarlas y ejecutarlas.
- La **unidad aritmético-lógica (ALU)**: se encarga de ejecutar las diferentes operaciones entre los datos almacenados en los registros de la CPU.

Como se puede ver en la imagen, también hay **dos tipos de memoria** denominadas memoria principal y secundaria.

Las características más importantes que las definen son su **capacidad**, **velocidad** y **coste por bit**. Se establece una jerarquía de memorias en función de ellas.

En esta imagen se puede observar que las memorias más cercanas a la CPU tienen menos capacidad, pero son más rápidas. Eso se debe a que la velocidad de acceso disminuye cuanto mayor es la capacidad de la memoria.

Por otro lado, se puede separar esta pirámide en dos grandes escalones: el primero estaría formado por los tres niveles superiores (memoria interna) y los otros dos niveles formarían la memoria externa.

REGISTROS DEL PROCESADOR

Los registros son aquellos tipos de **memoria con poca capacidad que tienen una alta velocidad de acceso**. Esto se debe a que continuamente están almacenando los datos de las instrucciones que el procesador tiene que ejecutar. Por ello, es una memoria volátil.

Una **memoria volátil** es aquella que pierde la información cuando se corta la corriente.

MEMORIA CACHÉ

La **memoria caché** se encarga de **almacenar los datos más usados** por el procesador. Es muy útil para reducir el tiempo de acceso y agilizar el procesamiento de la CPU.

Cuando el equipo requiere un dato, este se almacena en la memoria caché, que es más rápida que la memoria principal. Es por esto que, en los accesos siguientes, la CPU comprueba primero si se encuentra disponible en ella.

La **memoria caché** se divide en niveles, los cuales se denominan L1, L2 y L3 (aunque este último no se encuentra en todos los tipos de procesadores). Los datos se almacenan en uno de estos niveles dependiendo de la frecuencia que tengan.

L1 Datos
L1 Instrucciones
L2 Datos
L2 Instrucciones
L3

MEMORIA RAM

La **memoria principal** o memoria **RAM** es aquella que almacena todas las instrucciones para que la CPU las ejecute.

Al igual que los registros, este tipo de memorias también son volátiles. Además, se consideran una extensión de la memoria caché. Cuando el procesador busca un dato que no encuentra en la memoria caché, comprueba si se encuentra en esta memoria, ya que son de acceso más rápido que el disco.

Cuando la memoria RAM se llena, el disco duro destina una parte de su capacidad a cumplir con sus funciones. En este caso, el usuario detecta que el ordenador se ha ralentizado porque los programas tardan en responder.

Además, **las memorias RAM pueden ser estáticas o dinámicas**, y su principal diferencia es la necesidad de refresco: mientras que las memorias estáticas mantienen los datos siempre que estén alimentadas, las dinámicas pueden perder la información en cualquier momento. Esto tiene un inconviente, puesto que no se puede acceder a la información mientras se está actualizando.

➢ INTERCONEXIÓN DE LA CPU

➢ LOS BUSES

BUS DATOS | **BUS DIRECCIONES** | **BUS DE CONTROL**

La comunicación de la CPU con el resto de unidades funcionales se realiza a través de los **buses**.

Un **bus** es un **canal de comunicación físico** por el que se transmite la información.

Existen tres tipos:

- **Bus de datos**: canal de comunicación por el que se intercambian los datos entre la CPU y el resto de los componentes del sistema.
- **Bus de direcciones**: canal que se encarga de transportar las direcciones de memoria desde la CPU a la memoria principal.
- **Bus de control**: canal por el que se transportan las órdenes de la CPU.

➢ COMPONENTES LÓGICOS

Según la RAE, el *software* es el "conjunto de programas, instrucciones y reglas informáticas para ejecutar ciertas tareas en una computadora".

Se clasifica según su función:

- ***Software* de aplicación**: aquellos programas orientados al usuario final como, por ejemplo, las aplicaciones ofimáticas o los videojuegos.
- ***Software* de programación**: aquellos programas que ayudan a crear nuevo *software*. Son los editores de lenguajes de programación, los compiladores, los intérpretes o los entornos de desarrollo integrado (IDE).
- ***Software* de sistema**: programas que se encargan de ocultar la complejidad del *hardware* tanto al programador como al usuario final. Son los sistemas operativos y las diferentes herramientas de optimización y diagnóstico.

SOFTARE DE SISTEMA

Un *firmware* es un *software* que maneja físicamente al *hardware*.

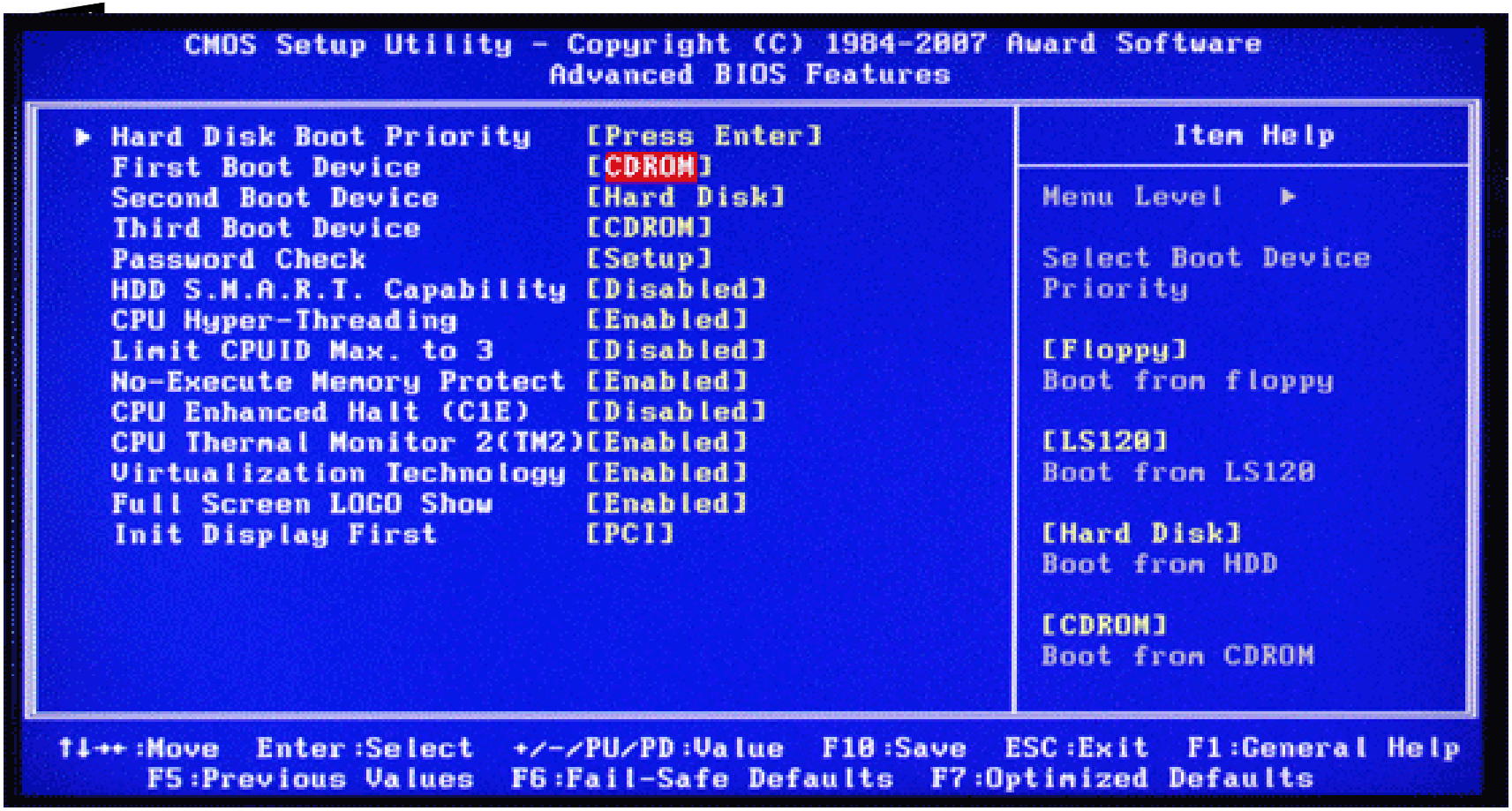
El *software* base o *software* de sistema es aquel que sirve para controlar e interactuar con el sistema operativo.

A diferencia del *software* de aplicación, el ***software* de sistema** proporciona control sobre el *hardware*.

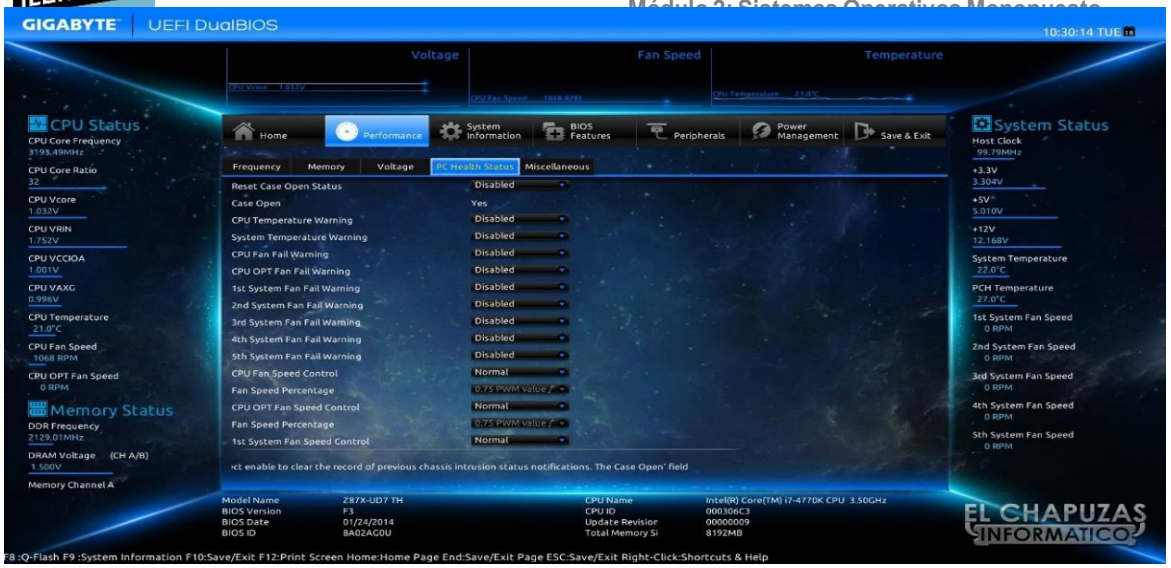
Entre los diferentes tipos de este *software*, se debe destacar la **BIOS**.

Se trata de un ***firmware*** que se encuentra almacenado en un circuito integrado de la placa base. Cuando encendemos el PC, es el primer programa que se ejecuta. Además, al estar integrada en la placa base, tanto la interfaz gráfica como su menú son diferentes para cada una. También cambia la tecla con la que podemos acceder cuando encendemos el ordenador.

En las placas antiguas, la BIOS carece de interfaz gráfica. Cuando accedemos a ella, encontramos una pantalla con texto y las opciones a modificar, las cuales pueden seleccionarse únicamente con el teclado.



No obstante, hoy en día las BIOS se han modernizado y han añadido una interfaz gráfica que permite utilizar tanto el teclado como el ratón. De esta forma, se ha aumentado la usabilidad de este *firmware*.



- **UEFIBIOS**.Este tipo de BIOS se denomina **UEFIBIOS**.

La diferencia principal es que este tipo de BIOS es **independiente del sistema operativo**, mientras que las BIOS tradicionales pueden tener características asociadas a la ejecución de dicho sistema. Además, estas últimas solo

soportan sistemas operativos de 16 bits (como puede ser MSDOS o Windows 95), mientras que las actuales soportan tanto los de 32 como los de 64 bits.

Otra de las características que añade la UEFI frente a la BIOS es la **posibilidad de configurar el *Secure Boot***. Esto permite la ejecución de programas certificados desde un *pendrive* antes de arrancar el sistema operativo, mientras que las BIOS originales permiten cualquiera de estos programas. Por otro lado, las UEFIBIOS **agilizan el arraque del sistema operativo**.

➢ FUNCIONES DE LA BIOS

➢

Las **funciones** de la BIOS son:

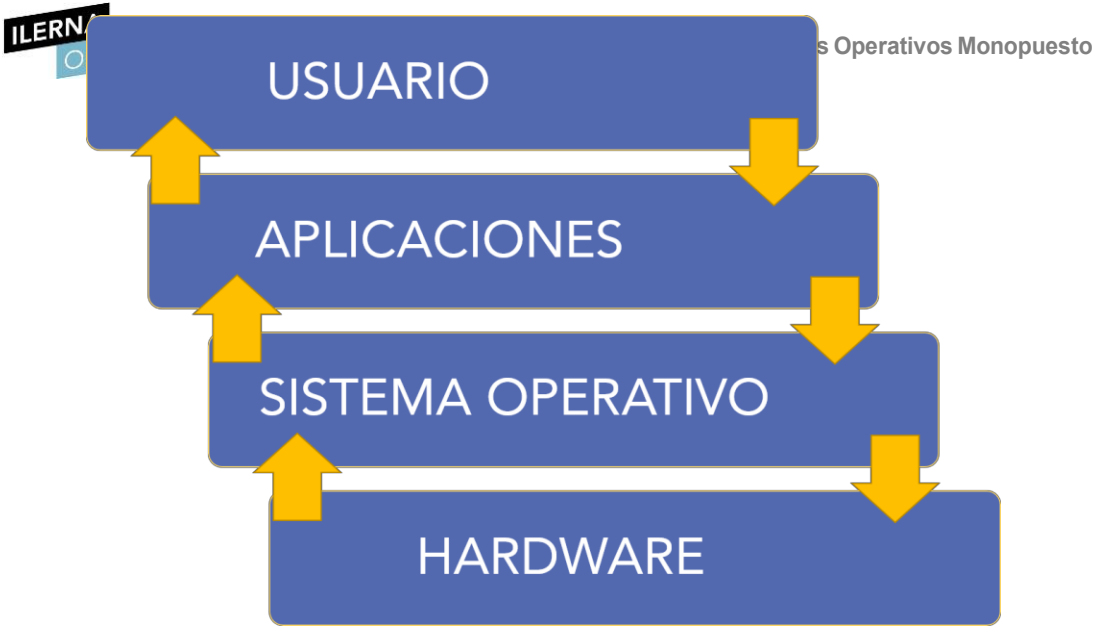
- Iniciar y configurar el ordenador.
- Comprobar el estado de todas las piezas *hardware*.
- Cargar el gestor de arranque.

Hay que tener en cuenta que el sistema solo arrancará si todas las comprobaciones anteriores son positivas. Además, el *software* de sistema más importante es el sistema operativo, pues sin él no podríamos utilizar el ordenador.

➢ 1.2 El Sistema Operativo

El *software* que se encarga de que podamos instalar un programa, administrar el *hardware* y usar las aplicaciones, entre otras cosas, recibe el nombre de sistema operativo.

> El sistema operativo es el conjunto de órdenes y programas que controlan los procesos básicos de una computadora y permiten el funcionamiento de otros programas.

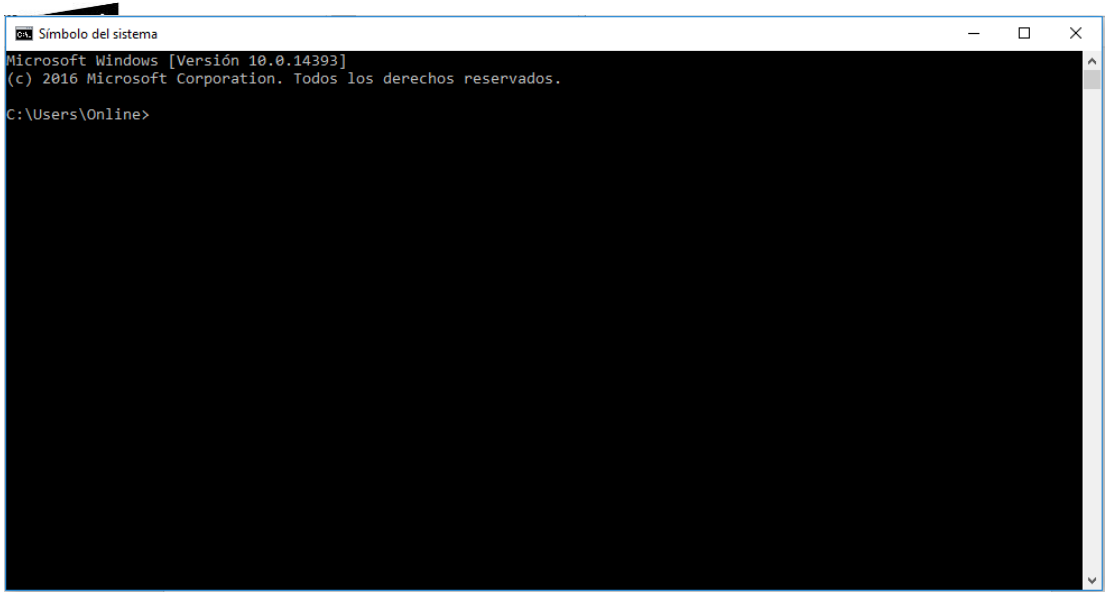


La parte más importante de un sistema operativo es el núcleo, también denominado *kernel*. Su función principal es facilitar a los programas un acceso seguro al *hardware* de la computadora y gestionar recursos.

KERNEL | INTERPRETE DE COMANDOS | SISTEMA DE ARCHIVOS

El sistema operativo se compone de diferentes elementos que interactúan entre sí para facilitar la comunicación del *hardware*.

- **Núcleo o *kernel***: anteriormente se han comentado las principales funciones que tiene el núcleo de un sistema operativo. Por ello, se diferencian dos partes: el **control de procesos** y el **control de la memoria**.
- **Intérprete de comandos**: un intérprete de comandos, también denominado *Shell*, es el programa informático que traduce las órdenes que introducen los usuarios para propiciar la comunicación entre ellos y el sistema operativo.
- **Sistema de archivos**: es el encargado de almacenar la información y establecer una jerarquía entre los distintos datos.



Los **discos duros** se organizan en bloques, y cada uno de ellos puede estar libre o lleno. El sistema de archivos del sistema operativo se encarga de controlar el estado de los mismos y, si están ocupados, conocer a qué fichero pertenecen.

Existen distintos tipos de sistemas de archivos (como pueden ser *FAT*, *NTFS* o *ext4*, entre otros) que se estudiarán en los temas posteriores, junto con las particularidades de cada uno de ellos.

1.3. Funciones del sistema operativo. Recursos

Gestionar la GPU | **Gestionar la RAM** | **Gestionar la e/s**

Gestionar la CPU

Reparte la cantidad de CPU dedicada a cada uno de los procesos que se encuentran en ejecución.

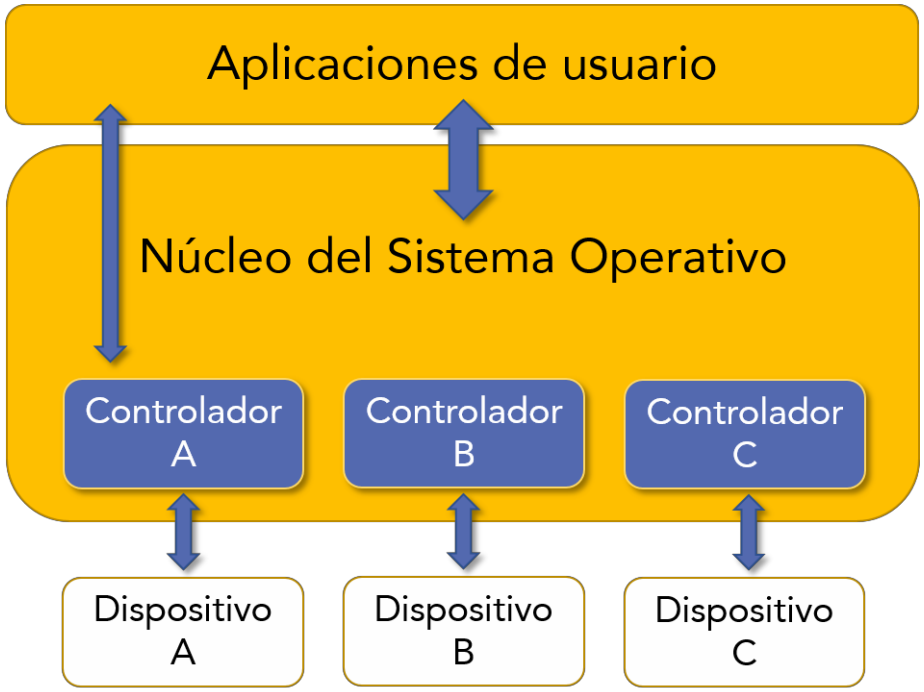
Gestionar la RAM

Asigna el espacio de memoria a cada aplicación. Además, es el encargado de crear la memoria virtual en el disco duro para adaptar sus funciones, como si fuera una RAM.

Gestionar la entrada/salida

Mediante los ***drivers***, controla el acceso de los programas a los componentes *hardware* del sistema.

Un **driver**, también denominado **controlador de dispositivo**, es el programa que permite la interacción entre el sistema operativo y un periférico. Su función principal es abstraer el *hardware* para proporcionar una interfaz que permita utilizar el dispositivo.



Gestionar los procesos

Un **proceso** es el **conjunto de instrucciones** que se van a ejecutar.

Es el responsable de organizar los procesos que se encuentran en el ordenador; puede crearlos, ejecutarlos, suspenderlos, reanudarlos y matarlos. Además, también se encarga de asignar los recursos disponibles para que puedan realizar su tarea.

Gestionar los permisos

Todos los elementos que se encuentran en un ordenador tienen una serie de permisos (lectura, escritura y ejecución). El sistema operativo se encarga de garantizar que los usuarios solo puedan acceder a los que les corresponden.

Gestionar los archivos

Esta función está relacionada con la anterior. Los archivos que se encuentran en un ordenador forman un sistema de archivos y tanto este como los ficheros tienen una serie de permisos. Lo que hace el sistema operativo es gestionar las operaciones que los usuarios pueden realizar en dicho sistema de archivos.

Gestionar información

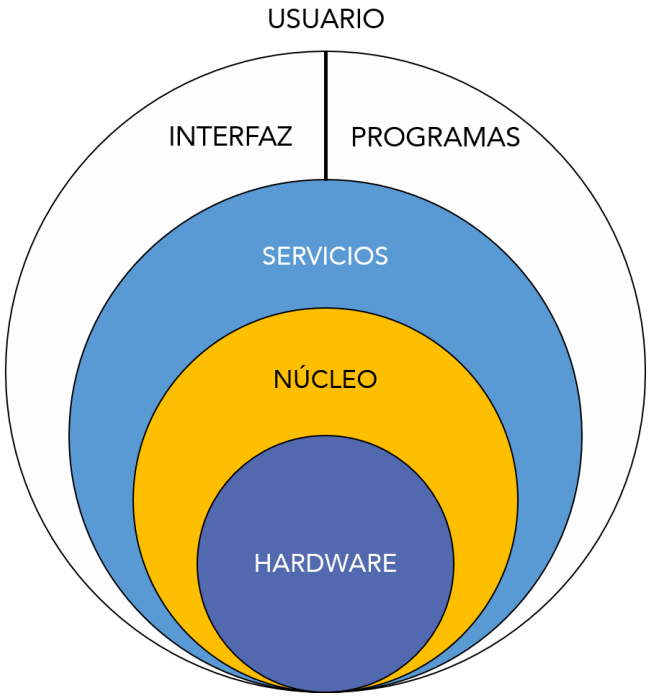
Proporciona toda la información necesaria para un buen funcionamiento del ordenador.

1.4. Arquitectura del sistema operativo

Todos los sistemas operativos se estructuran en **tres capas diferenciadas**:

el ***kernel***, la **capa intermedia** y la **interfaz**.

Kernel | **Capa Intermedia** | **Interfaz**



En esta capa intermedia se encuentran muchas de las funciones del sistema operativo, como la comunicación con el *hardware*.

También se puede observar que los programas se encuentran en el nivel superior; cuando es necesario utilizar recursos del *hardware*, se comunican con él a través de esta capa intermedia.

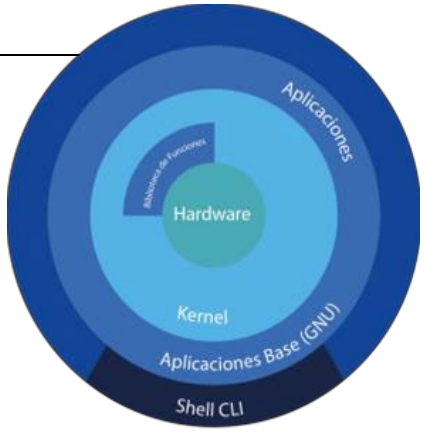
1.4. ARQUITECTURA DEL SISTEMA OPERATIVO

Monolítico | **Micronúcleo** | **Híbrido** | **Exonúcleo**

La arquitectura del sistema operativo depende del **tipo de núcleo** que tenga. Existen **cuatro tipos**:

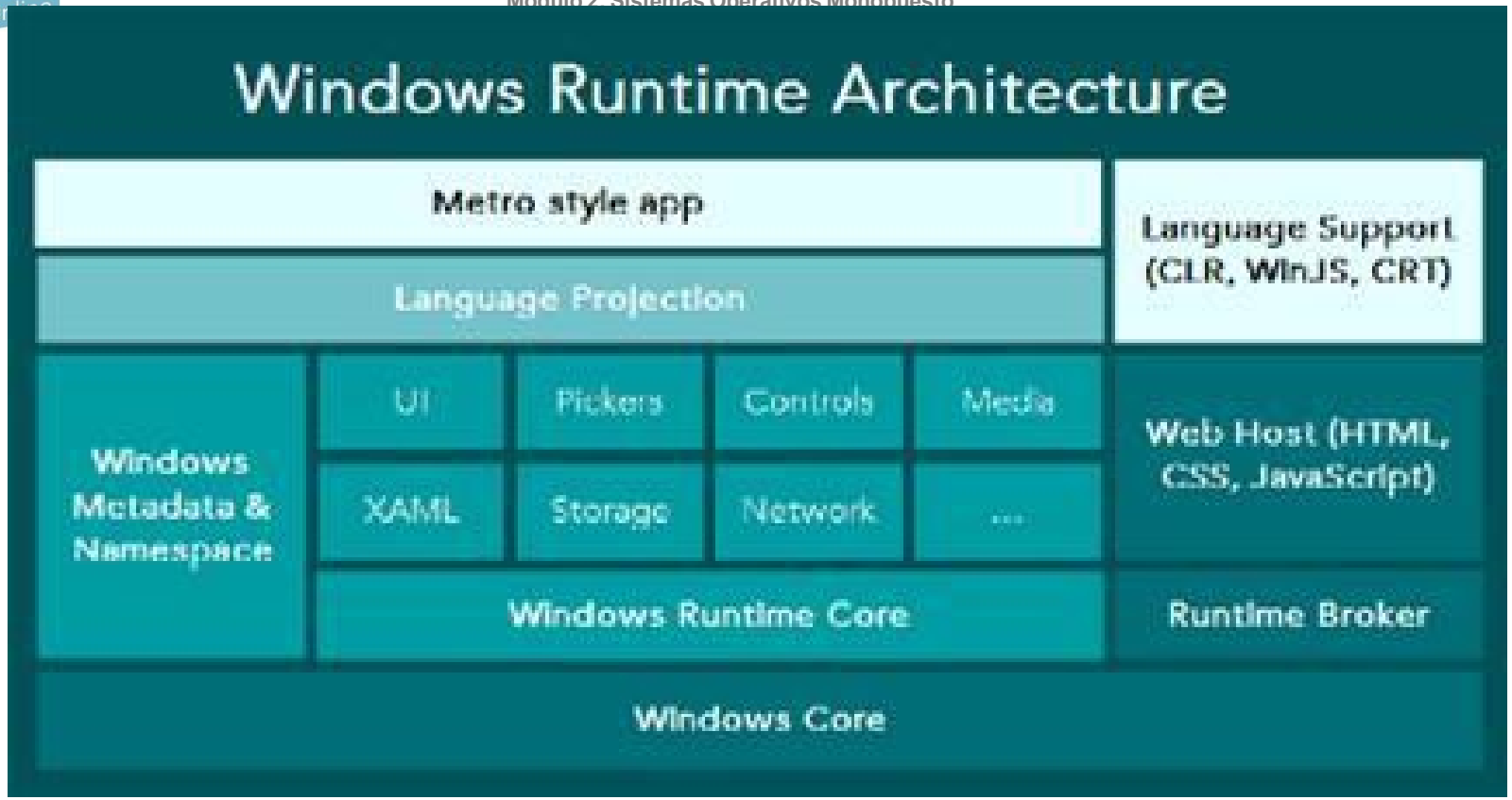
- **Monolítico**: los sistemas operativos tienen un único núcleo, el cual es el encargado de recoger todos los servicios del sistema. Todas las peticiones se concentran en un mismo programa, lo que implica que tenga un tamaño considerable. Es uno de los núcleos más utilizados actualmente y puede encontrarse en Linux, Unix, BSD, Solaris y MS- DOS.

- **Micronúcleo**: este núcleo se encarga de recoger las funciones más básicas del sistema operativo. Si se desea añadir funcionalidades, se debe hacer de forma modular. Sus principales **ventajas** son la seguridad y la portabilidad, mientras que sus **desventajas** refieren un mayor tiempo en la respuesta de las llamadas y una peor comunicación entre el *hardware* y sus controladores.



- **Híbrido**: se basa en la combinación de los anteriores, es decir, permite una mejor comunicación *drivers-hardware* y la gestión de las llamadas al sistema. La mayor parte de los sistemas operativos modernos tienen este tipo de núcleo, como Windows y Mac OS X.

- **Exonúcleo**: es el más moderno. El núcleo contiene una parte básica de gestión de recursos y es el desarrollador quien, mediante librerías, añade nuevos módulos. Esto libera de carga de memoria de procesamiento al núcleo y mejora la comunicación con el *software*.


Windows Runtime Architecture
Metro style app
Language Projection
Language Support (CLR, WinJS, CRT)
Windows Metadata & Namespace
UI
Pickers
Controls
Media
XAML
Storage
Network
...
Web Host (HTML, CSS, JavaScript)
Windows Runtime Core
Runtime Broker
Windows Core

1.3. Evolución histórica. Sistemas operativos actuales

Aunque la primera generación de ordenadores tiene su origen en la década

QDOS es un sistema operativo producido por Tim Paterson que después fue adquirido por Bill Gates.

de 1940, no es hasta 1970 cuando se empiezan a desarrollar sistemas operativos más complejos. A finales de la década anterior aparece Multics como uno de los primeros sistemas operativos escritos en lenguaje de alto nivel, el cual era, además, multiusario-multitarea.

Pero fue durante la década de los 80 cuando comenzó el auge de los ordenadores personales. En aquella época se buscaba una mayor usabilidad del sistema, por lo que aparecieron los entornos gráficos.

MS-DOS

Mac OS

Sun OS

Exonúcleo

Este sistema operativo apareció en 1982 y fue desarrollado por **Microsoft**. La empresa compró QDOS y, después de algunas modificaciones, lo publicó. Su núcleo es monolítico y una de sus principales características es que no tiene interfaz gráfica; su uso se basa en una línea de comandos.

Mac OS

Fue lanzado en 1984 y desarrollado por **Apple Inc**. Revolucionó la historia de la compañía, puesto que fue el primer sistema operativo con una interfaz gráfica de usuario que se creó con éxito. Otra de sus innovaciones fue la incorporación de un ratón, para evitar el uso de la línea de comandos. La línea de computadoras que utilizaban este sistema operativo fue denominada Macintosh.

Sistema operativo desarrollado por **Sun Microsystems**, basado en **Unix**. Este tipo de *software* podía encontrarse en las estaciones de trabajo y los servidores en la década de los 90.

En aquella época, la venta de ordenadores personales siguió en aumento, por lo que fue necesario que los sistemas operativos se actualizasen. Muchos de ellos se basaban en Unix (como son GNU/Linux, Solaris y FreeBSD, entre otros). Por otro lado, compañías como Microsoft dividieron su línea de productos, creando Windows NT para estaciones de trabajo y servidores y Windows para los ordenadores personales.
Por su parte, Windows 95 fue el primer sistema operativo de esta gama que incorporaba un entorno gráfico.

Pero a partir de la década del 2000 empezaron a aparecer tanto sistemas operativos móviles como de escritorio.

Cabe destacar que el sistema operativo libre por excelencia es GNU/Linux, en el que se basan ambos tipos de distribuciones.

GNU es un sistema operativo UNIX que se caracteriza por ser ***software* libre**.

Es la **combinación del sistema operativo GNU con un núcleo Linux**. Fue desarrollado por Linus Torvalds.

La diferencia entre ambos términos es que el primero (GNU) hace referencia al sistema operativo, mientras que el segundo (Linux) hace referencia al tipo de núcleo.
Está **escrito en C**, un lenguaje de nivel medio orientado a la implementación de los sistemas operativos y, además, ofrece la posibilidad al usuario de trabajar en modo consola mediante una interfaz gráfica.

Sistemas operativos de escritorio

perativos Monopuesto

Windows | **Mac OS X** | **Distribuciones GNU/Linux**

- **Windows**: en 2001 apareció Windows XP. Se trata de uno de los sistemas operativos de la compañía Microsoft más comercializados y más estables que han estado en el mercado. Ha seguido teniendo soporte hasta el año 2014.

- **Mac OS X**: el sistema operativo Mac OS fue evolucionando en el tiempo y dio lugar al sistema operativo Mac OS X en el año 2002. Está basado en Linux, pero, a diferencia de GNU/Linux, su licencia no es gratuita.

  Por otra parte, a lo largo de los años ha tenido múltiples versiones: las primeras llevaban nombres de felinos y las más conocidas han sido Leopard (versión 10.5) y Mountain Lion (versión 10.8).

- **Distintas distribuciones GNU/Linux**: los sistemas operativos Ubuntu, Debian y Fedora tienen en común que utilizan el núcleo Linux.

Sistemas operativos Móviles

Windows Phone | **iOS** | **Android**

- **Windows Phone**: el primer sistema operativo desarrollado por Microsoft para los teléfonos inteligentes y otros dispositivos móviles se denominó Windows Mobile, pero en 2014 cambió a Windows Phone.

  Este sistema operativo no triunfó porque el mercado fue abarcado por Android e iOS. Así pues, durante 2015 se anunció la retirada de este sistema operativo, que fue sustituido por Windows 8 y utilizado tanto en los dispositivos de escritorio como en los móviles.

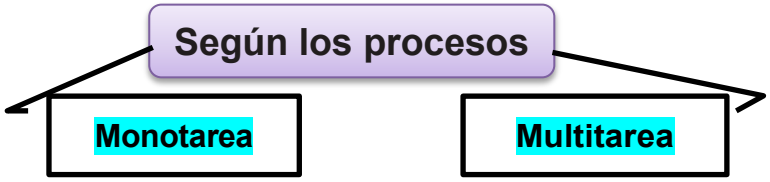
- **iOS**: desarrollado por Apple y lanzado en 2009. Este sistema operativo ha aumentado las ventas de la compañía. Originalmente, se desarrolló

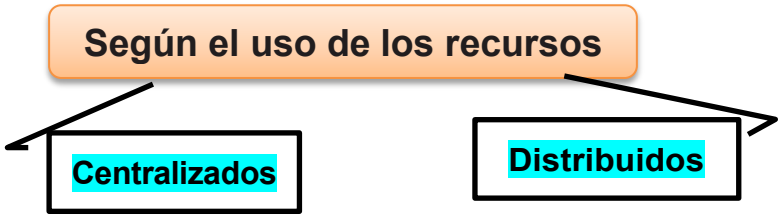
para los iPhone (teléfonos inteligentes), aunque ha terminado formando parte de todos los dispositivos móviles creados por Apple.

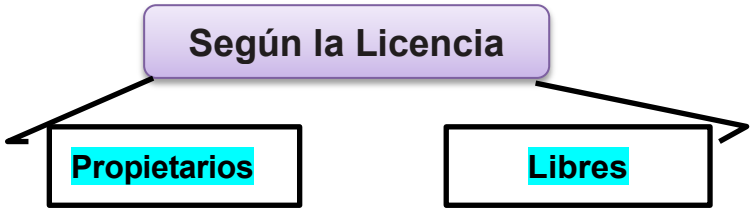
- **Android**: es un sistema operativo con núcleo Linux, lanzado en 2008. Fue desarrollado por Android Inc., aunque en 2005 fue adquirido por Google. Es el sistema operativo móvil más vendido a nivel mundial porque se utiliza en teléfonos inteligentes, *tablets*, relojes inteligentes (*SmartWatches*) y televisores (*Smart TV*).

1.6. Clasificación de los sistemas operativos

Según los procesos | Según los Usuarios | Según el uso de los recursos







Sistemas operativos ***según los procesos*** que pueden ejecutar al mismo tiempo…….

- **Monotarea**: solo permite ejecutar un **único proceso**. Únicamente podrá empezar su ejecución otro cuando el primero finalice o sea interrumpido. Un ejemplo de este tipo de sistemas operativos es el MS-DOS.

- **Multitarea**: permite ejecutar varios procesos porque se encarga de gestionar los recursos disponibles entre todos los que se encuentran en ejecución. Actualmente, todos los sistemas operativos son multitarea.

Sistemas operativos ***según los usuarios*** que pueden ejecutar programas……:

- **Monousuario**: no es posible que haya más de un usuario a la vez. Un ejemplo de este tipo de sistemas operativos son las versiones domésticas de Windows.

- **Multiusuario**: los usuarios pueden ejecutar varios programas simúltaneamente. Los sistemas operativos en red son multiusuario, ya que varias personas pueden estar trabajando en el mismo sistema y compartir sus recursos (CPU, memoria RAM, almacenamiento, programas y periféricos), abstraídos del resto de usuarios.

Sistemas operativos ***según el uso de los recursos***……..::

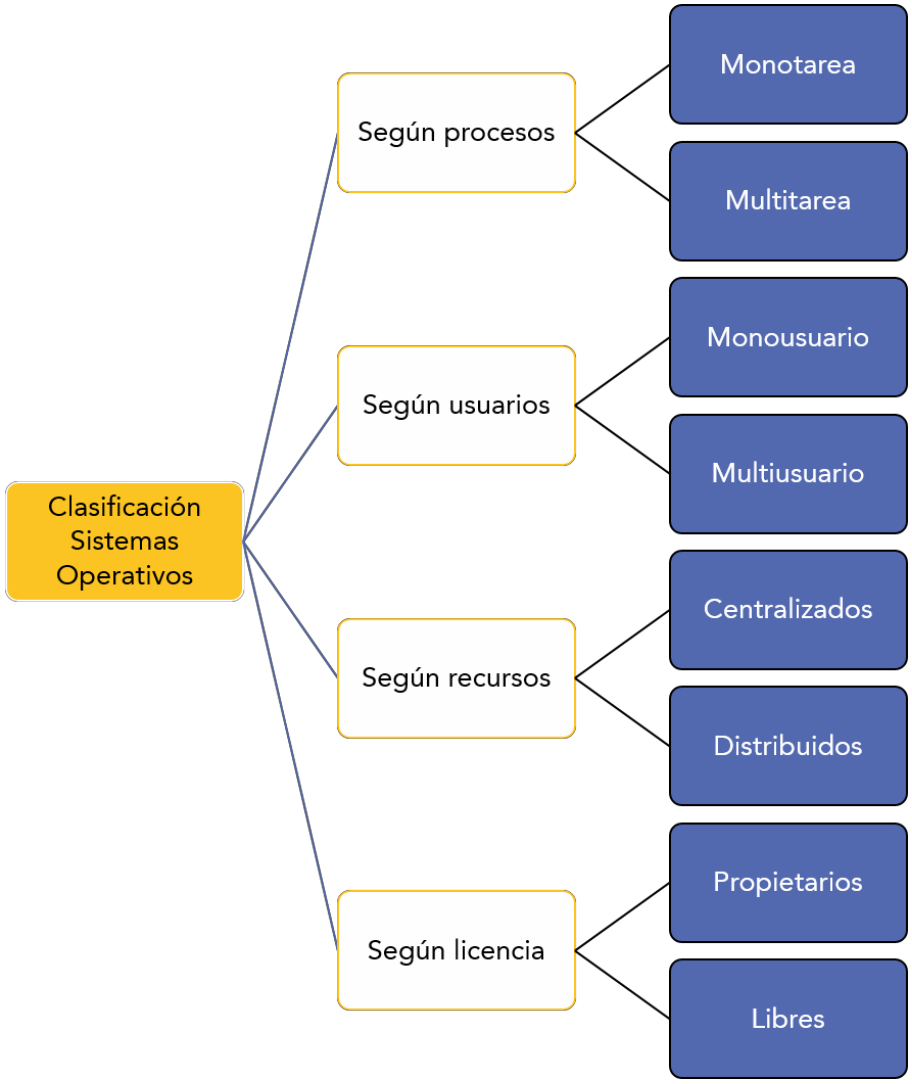
- **Centralizados**: los recursos del sistema operativo se encuentran en una única computadora. Son aquellos que se encuentran en los PCs, Windows, Linux, Mac, etcétera.

- **Distribuidos**: los recursos utilizados pueden pertenecer a distintas computadoras que están conectadas por red.

Sistemas operativos ***según el tipo de licencia***……..::

- **Propietarios**: son aquellos que tienen limitaciones en el uso (no permiten que se modifique ni se redistribuya).

- **Libres**: son aquellos que permiten su modificación para mejorar el producto o crear otro parecido.



1.7. Sistemas transaccionales

Un sistema transaccional es un tipo de sistema de información cuya función es **recolectar, almacenar, modificar y recuperar la información** generada por las transacciones de una organización.

Una **transacción** es un conjunto de operaciones que generan o modifican información que se encuentra almacenada en el sistema.

Sus **funciones** son:

- Mantener la seguridad y la consistencia de los datos.
- Deshacer operaciones para evitar errores.
- Controlar y administrar múltiples transacciones en un mismo momento.

Estos sistemas los encontramos, por ejemplo, en los bancos. Cuando realizamos operaciones con el dinero, se tienen que realizar una serie de operaciones de forma atómica. Esto quiere decir que hay dos opciones: o se realizan todas las operaciones o no se realiza ninguna.

Para que un sistema se considere transaccional debe pasar el **test ACID**, el cual mide las cuatro propiedades básicas de este tipo de sistemas:

TEST ACID — **Atomicidad** — **Consistencia** — **Aislamiento** — **Durabilidad**

- **Atomicidad**: la transacción no puede quedarse a medias (o se realizan todas las operaciones o no se realiza ninguna).
- **Consistencia**: las acciones a realizar deben cumplir las normas necesarias para que se rompa la integridad de una base de datos.
- **Aislamiento**: las transacciones no interfieren unas con otras.
- **Durabilidad**: no es vulnerable si se producen fallos durante las transacciones.

Este tipo de sistemas son los implementados en aquellos *softwares* o páginas web que se encargan de la administración de entradas para eventos, viajes y ventas, entre otros.

➢ Sistemas por LOTES

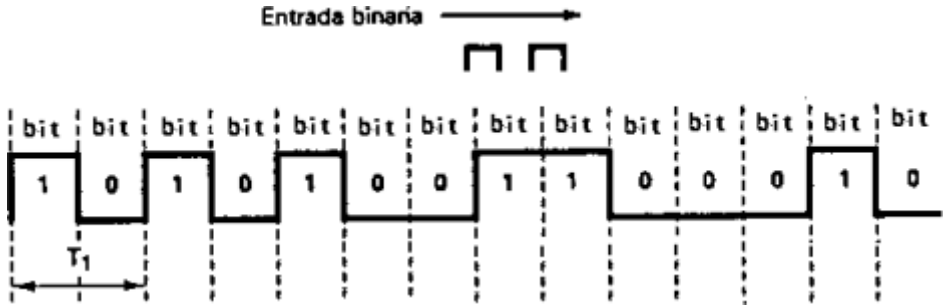
El sistema cuya funcionalidad es contraria a esta se denomina **sistema por lotes**. En él, las operaciones son realizadas una a una: si una instrucción falla, el programa finaliza, pero los cambios realizados hasta ese momento quedan operativos. Este tipo de sistemas son utilizados en los *scripts*.

Un **script** es un programa de texto plano con instrucciones para realizar en el sistema, las cuales serán ejecutadas mediante el procesamiento por lotes con la línea de comandos. En temas posteriores, los veremos tanto en sistemas operativos Windows como Linux.

2. Codificación de la información en diferentes sistemas de representación

➢ 2.1. Sistemas de representación

Toda información que maneja el ordenador se representa mediante dos símbolos (0 y 1), los cuales corresponden a dos estados eléctricos, es decir, a los dos niveles de tensión que pueden llegar a tomar.



En definitiva, la información se mantiene utilizando dos valores de una magnitud física representable mediante ceros y unos.

Un **bit** es la unidad mínima de información. Puede tener dos valores, que serán sus estados. Puede estar *apagado* (su valor es 0) o *encendido* (su valor es 1).

A partir de estos símbolos básicos, el ordenador es capaz de construir, almacenar y representar distintos tipos de información, pues puede codificarla. Además, hay distintos tipos de representaciones:

- Representación de **textos**
- Representación de **valores numéricos**
- Representación de **instrucciones**
- Representación de **sonidos**
- Representación de **imágenes y vídeos**

Para almacenar estos tipos de documentos, es necesario codificarlos, lo cual se hace mediante la transformación de la información al **sistema binario**.

En este tema, se van a ver los sistemas de codificación existentes y cómo se puede trabajar con ellos.

La **codificación** es la operación que permite convertir los datos de un sistema de información a otro.

2.2. Medidas de información

Palabra **Byte**

Además de los bits, en la información digital existen diferentes tipos de unidades de información:

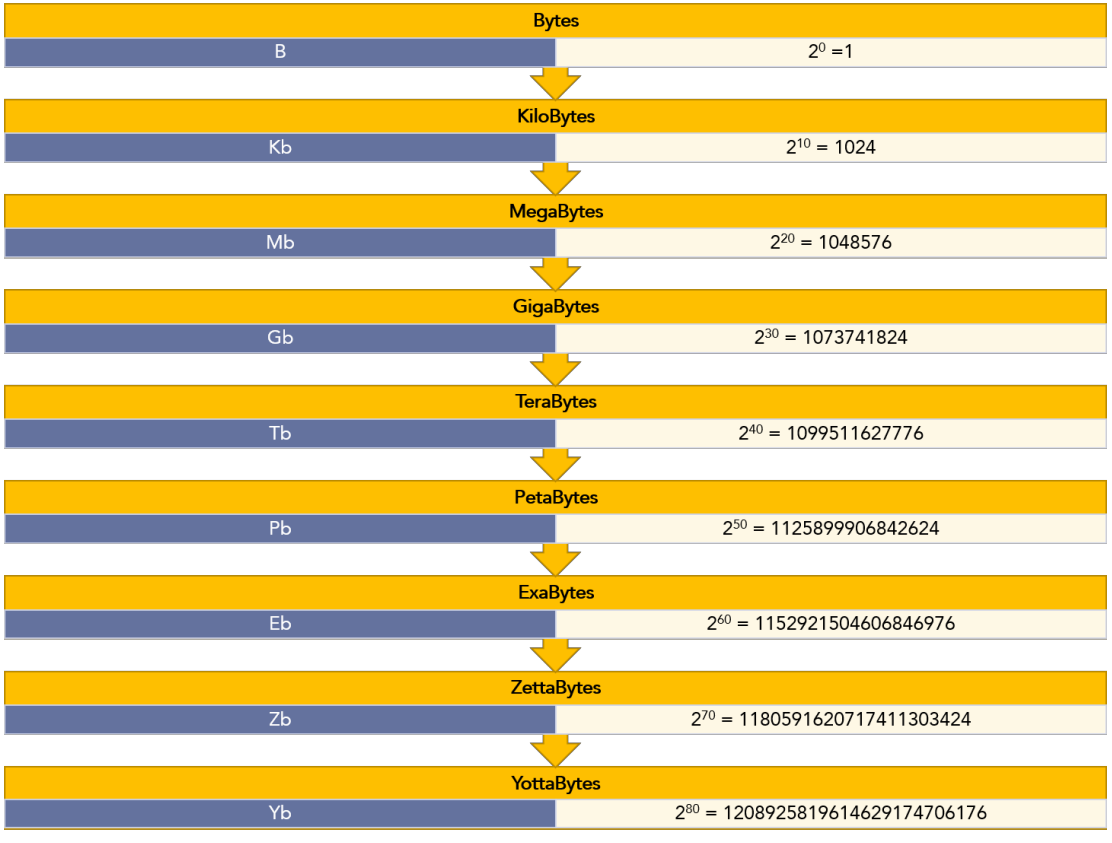
- **Palabra**: conjunto de *n* bits. La longitud de una palabra hace referencia al número de bits contenidos en ella. Además, su tamaño puede variar, pero en los ordenadores modernos suelen tener una longitud de **16, 32** o **64 bits**.
- **Byte**: conjunto de 8 bits.

Al almacenar la información
tenemos que tener en cuenta **dos factores**:

Unidad de medida **Capacidad de almacenamiento**

- **La unidad de medida**: los datos pueden ser de gran tamaño, por lo que, para simplificar su valor, se ha establecido una escala con diferentes unidades de medida.

- **La capacidad de almacenamiento**: se refiere a la cantidad de datos que pueden almacenarse en un dispositivo. Dependiendo del dispositivo, las unidades de medida más utilizadas pueden ser *megabytes* o *gigabytes*.



Dependiendo del tipo de información que vayamos a almacenar, podemos utilizar una u otra unidad. Si queremos medir el tamaño de un fichero de texto usaremos los kilobytes*, mientras que, si queremos medir el tamaño de una canción en formato MP3, utilizaremos los* megabytes*.*

2.3. Sistemas de codificación alfanumérica

ASCII

EBCDIC (Unicode)

Para la representación de los datos no numéricos o alfanuméricos se emplean códigos como el **ASCII**, el **EBCDIC** o el **Unicode**.

ASCII

El **código ASCII** es el más usado entre los sistemas informáticos actuales.

El **ASCII se utiliza para representar caracteres**. Se trata de un código estándar, independiente del lenguaje que usemos y de la computadora utilizada. Además, está formado por **8 bits**, de manera que cada carácter se expresa por un número comprendido entre 0 y 255.

Por otra parte, cabe mencionar que la información se guarda en 7 bits y el octavo se reserva para comprobar la paridad y prevenir errores.

En este sistema podemos distinguir **dos grupos**: los primeros 128 caracteres se denominan código **ASCII estándar** y representan los caracteres que aparecen en una máquina de escribir convencional. De estos, los primeros 32 son caracteres de control. Este tipo se refiere a aquellos códigos que no representan información imprimible. Por otro lado, los 128 restantes se denominan código **ASCII ampliado**, que son

asignados a un número de caracteres que no aparecen en la máquina de escribir y que son muy utilizados en la computadora, como pueden ser operadores matemáticos o caracteres gráficos.

Caracteres ASCII de control		
00	NULL	(carácter nulo)
01	SOH	(inicio encabezado)
02	STX	(inicio texto)
03	ETX	(fin de texto)
04	EOT	(fin transmisión)
05	ENQ	(consulta)
06	ACK	(reconocimiento)
07	BEL	(timbre)
08	BS	(retroceso)
09	HT	(tab horizontal)
10	LF	(nueva línea)
11	VT	(tab vertical)
12	FF	(nueva página)
13	CR	(retorno de carro)
14	SO	(desplaza afuera)
15	SI	(desplaza adentro)
16	DLE	(esc.vínculo datos)
17	DC1	(control disp. 1)
18	DC2	(control disp. 2)
19	DC3	(control disp. 3)
20	DC4	(control disp. 4)
21	NAK	(conf. negativa)
22	SYN	(inactividad sinc)
23	ETB	(fin bloque trans)
24	CAN	(cancelar)
25	EM	(fin del medio)
26	SUB	(sustitución)
27	ESC	(escape)
28	FS	(sep. archivos)
29	GS	(sep. grupos)
30	RS	(sep. registros)
31	US	(sep. unidades)
127	DEL	(suprimir)

Caracteres ASCII imprimibles					
32	espacio	64	@	96	`
33	!	65	A	97	a
34	"	66	B	98	b
35	#	67	C	99	c
36	$	68	D	100	d
37	%	69	E	101	e
38	&	70	F	102	f
39	'	71	G	103	g
40	(	72	H	104	h
41	)	73	I	105	i
42	*	74	J	106	j
43	+	75	K	107	k
44	,	76	L	108	l
45	-	77	M	109	m
46	.	78	N	110	n
47	/	79	O	111	o
48	0	80	P	112	p
49	1	81	Q	113	q
50	2	82	R	114	r
51	3	83	S	115	s
52	4	84	T	116	t
53	5	85	U	117	u
54	6	86	V	118	v
55	7	87	W	119	w
56	8	88	X	120	x
57	9	89	Y	121	y
58	:	90	Z	122	z
59	;	91	[	123	{
60	<	92	\	124	|
61	=	93	]	125	}
62	>	94	^	126	~
63	?	95	_		

ASCII extendido							
128	Ç	160	á	192	└	224	Ó
129	ü	161	í	193	┴	225	ß
130	é	162	ó	194	┬	226	Ô
131	â	163	ú	195	├	227	Ò
132	ä	164	ñ	196	─	228	õ
133	à	165	Ñ	197	┼	229	Õ
134	å	166	ª	198	ã	230	µ
135	ç	167	º	199	Ã	231	þ
136	ê	168	¿	200	╚	232	Þ
137	ë	169	®	201	╔	233	Ú
138	è	170	¬	202	╩	234	Û
139	ï	171	½	203	╦	235	Ù
140	î	172	¼	204	╠	236	ý
141	ì	173	¡	205	═	237	Ý
142	Ä	174	«	206	╬	238	¯
143	Å	175	»	207	¤	239	´
144	É	176	░	208	ð	240	≡
145	æ	177	▒	209	Ð	241	±
146	Æ	178	▓	210	Ê	242	‗
147	ô	179	│	211	Ë	243	¾
148	ö	180	┤	212	È	244	¶
149	ò	181	Á	213	ı	245	§
150	û	182	Â	214	Í	246	÷
151	ù	183	À	215	Î	247	¸
152	ÿ	184	©	216	Ï	248	°
153	Ö	185	╣	217	┘	249	¨
154	Ü	186	║	218	┌	250	·
155	ø	187	╗	219	█	251	¹
156	£	188	╝	220	▄	252	³
157	Ø	189	¢	221	¦	253	²
158	×	190	¥	222	Ì	254	■
159	ƒ	191	┐	223	▀	255	nbsp

EBCDIC
(Unicode)

El **sistema de codificación Unicode** es un sistema de 16 bits que se utiliza en otros sistemas de escritura como, por ejemplo, el árabe, el griego o el japonés, entre otros. Esto se debe a su mayor capacidad con respecto a otros sistemas.

➢ 2.4. Sistemas de codificación numérica

DECIMAL BINARIO HEXADECIMAL OCTAL

Un sistema de numeración es el **conjunto de símbolos y reglas** que se utilizan para representar datos numéricos.

Los sistemas más comunes son:

DECIMAL

- **Decimal**: se compone de 10 dígitos, por lo que decimos que tiene **base 10**. Su conjunto de dígitos va desde el 0 hasta el 9. D = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}
  Es el sistema que utilizamos en la vida cotidiana.

BINARIO

- **Binario**: se compone únicamente de 2 dígitos, por lo que su base es 2.
- Los dígitos que la componen son 0 y 1.
  D = {0, 1}
  Como se ha comentado anteriormente, es el sistema que utilizan internamente los ordenadores.

- **HEXADECIMAL**
- **Hexadecimal**: se compone de 16 dígitos, donde se combinan números y letras. **Su base es 16**.
-

  D = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F}

  Es el sistema de numeración que suelen utilizar las CPU.

OCTAL

- **Octal**: se compone de 8 dígitos, cuya numeración va del 0 al 7. **Su base es 8**.
-

  D = {0, 1, 2, 3, 4, 5, 6, 7}

  En algunos casos, se utiliza el sistema octal en vez del hexadecimal. Esto se debe a la ventaja de no tener que codificar letras. Para poder utilizar este sistema en vez del hexadecimal, es necesario añadir delante el prefijo 0x.

A(16 = 0x12(8

➢ 2.5. Conversión entre diferentes sistemas de numeración

En este apartamos, se van a explicar los diferentes métodos que tenemos para la conversión de los datos.

PASO DE DECIMAL A:
• Paso de decimal a cualquier otra base
• Paso de decimal a binario
• Paso de decimal a octal
• Paso de decimal a Hexadecimal

PASO de:
• Paso cualquier base a decimal
• Paso de binario a octal o hexadecimal
• Paso de octal a hexadecimal y viceversa
• Paso de hexadecimal a octal

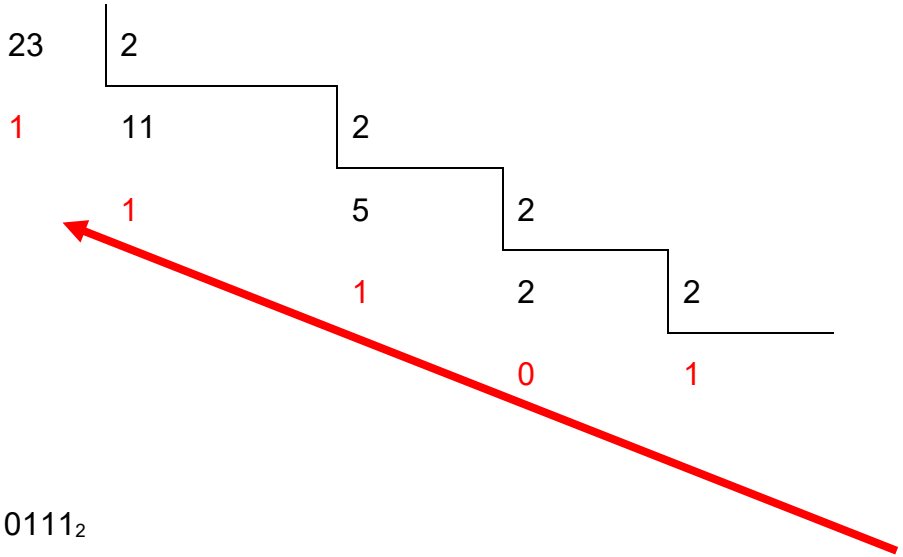
• Paso de decimal a cualquier otra base

Para realizar este tipo de conversiones, es necesario separar la parte decimal de la parte entera del número, pues la operación se lleva a cabo de distinta forma.

- **Parte entera**: dividimos el número original (en decimal) entre la base del sistema de numeración de destino, hasta obtener un resto menor que el divisor, es decir, sin decimales. Después, formaremos el número en binario con el último cociente y los restos obtenidos.

- **Parte decimal**: al contrario que en la parte entera, aquí multiplicamos los dígitos. El procedimiento consiste en multiplicar la parte decimal por la base a la que se va a transformar el número hasta que no haya parte decimal. Después, se escogen las cifras de la parte entera que resultan de cada operación.

- **Parte entera**: ejemplo: $23_{10} = ?_2$



$23_{10} = 10111_2$

- **Parte decimal**: ejemplo: $0{,}375_{10} = ?_2$

$0{,}375 \cdot 2 = 0{,}75$
$0{,}75 \cdot 2 = 1{,}5$
$0{,}5 \cdot 2 = 1{,}0$
$0{,}0 \cdot 2 = 0{,}0$

$0{,}375_{10} = 0{,}011_2$

2.1.2. Paso de decimal a octal

- **Parte entera**: ejemplo: $23_{10} = ?_8$

$23_{10} = 27_8$

- **Parte decimal**: ejemplo: $0{,}625_{10} = ?_8$

$0{,}625 \cdot 8 = 5{,}0$

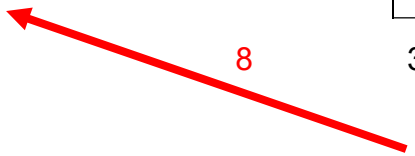
$0{,}675_{10} = 0{,}5_8$

23	8
7	2

2.1.3. Paso de decimal a hexadecimal

- **Parte entera**: ejemplo: $897_{10} = ?_{16}$

897	16	
1	56	16
	8	3



$897_{10} = 381_{16}$

- **Parte decimal**: ejemplo: $0{,}625_{10} = ?_{16}$

$0{,}625 \cdot 16 = 10{,}0$

$0{,}675_{10} = 0{,}A_{16}$

2.1.4. Paso de cualquier base a decimal

Para realizar la conversión a decimal se debe multiplicar cada dígito por su base elevada a la posición en la que se encuentra. Para obtener la posición, se debe diferenciar la parte entera de la decimal:

- La **parte entera** comienza en la posición 0 en el dígito al lado de la coma.
- La **parte decimal** comienza en la posición -1 en el dígito al lado de la coma.

Paso de binario a decimal

- **Parte entera**: ejemplo: $10101010_{12} = ?_{10}\ 1\cdot2^8+0\cdot2^7+1\cdot2^6+0\cdot2^5+1\cdot2^4+0\cdot2^3+1\cdot2^2+0\cdot2^1+1\cdot2^0$

  $1\cdot256+0\cdot128+1\cdot64+0\cdot32+1\cdot16+0\cdot8+1\cdot4+0\cdot2+1\cdot1$

  $101010101_2 = 341_{10}$

- **Parte decimal**: ejemplo: $0{,}101_2 = ?_{10}\ 1\cdot2^{-1}+0\cdot2^{-2}+1\cdot2^{-3}$

  $1\cdot0{,}5+0\cdot0{,}25+1\cdot0{,}125$

  $0.101_2 = 0{,}625_{10}$

Paso de octal a decimal

- **Parte entera**: ejemplo: $123_8 = ?_{10}\ 1 \cdot 8^2 + 2 \cdot 8^1 + 3 \cdot 8^0$

  $1 \cdot 64 + 2 \cdot 8 + 3 \cdot 1$

  $123_8 = 83_{10}$

- **Parte decimal**: ejemplo: $0{,}45_8 = ?_{10}$

  $4 \cdot 8^{-1} + 5 \cdot 8^{-2}$

  $4 \cdot 0{,}125 + 5 \cdot 0{,}015625$

  $0{,}45_8 = 0{,}578125_{10}$

Paso de hexadecimal a decimal

- **Parte entera**: ejemplo: $A001_{16} = ?_{10}\ A \cdot 16^3 + 0 \cdot 16^2 + 0 \cdot 16^1 + 1 \cdot 16^0$

  $10 \cdot 4096 + 0 * 256 + 0 \cdot 16 + 1 * 1$

  $A001_{16} = 40961_{10}$

- **Parte decimal**: ejemplo: $0{,}07_{16} = ?_{10}\ 0 \cdot 16^{-1} + 7 \cdot 16^{-2}$

  $0 \cdot 0{,}0625 + 7 \cdot 0{,}00390625$

$0{,}07_{16} = 0{,}02734375_{10}$

2.1.5. Paso de binario a octal o hexadecimal

Paso de binario a octal

Para realizar la conversión de binario a octal, debemos realizar grupos de 3, teniendo como referencia la coma. Cada uno de estos grupos de dígitos es un dígito en octal, por lo que se debe realizar dicha conversión.

Binario	Octal
001	1
010	2
011	3
100	4
101	5
110	6
111	7

Ejemplo: $01011100,110012 = ?8$

001 011 10 , 110 01
0 0
1 3 4 , 6 2

$01011100,110012 = 134,628$

Paso de binario a hexadecimal

Para realizar la conversión de binario a hexadecimal, debemos realizar grupos de 4, teniendo como referencia la coma. Cada uno de estos grupos de dígitos es un dígito en hexadecimal, por lo que se debe realizar dicha conversión.

Binario	Hexadecimal
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7
1000	8
1001	9
1010	A
1011	B
1100	C
1101	D
1110	E
1111	F

Ejemplo: $11101100010101,11001_2 = ?_{16}$

0011 1011 0001 0101 , 1100 1000
3 B 1 5 , C 8

$01011100,11001_2 = 3B15,C4_{16}$

Paso de octal a binario

Hay que realizar la conversión de todos los dígitos a binario. Cada uno de ellos se transformará en 3 dígitos binarios.

Ejemplo: $234{,}628 = ?2$

2	3	4	,	6	2
01	01	10	,	11	01
0	1	0		0	0

$234{,}628 = 010011100{,}110010_2$

Paso de hexadecimal a binario

Hay que realizar la conversión de todos los dígitos a binario. Cada uno de ellos se transformará en 4 dígitos binarios.

Ejemplo: $ABD30{,}C1_{16} = ?2$

A	B	D	3	0	,	C	1
1010	1011	1101	0011	0000	,	1100	0001

$ABD30{,}C1 = 10101011110100110000{,}11000001_2$

2.1.7. Paso de octal a hexadecimal y viceversa

Paso de octal a hexadecimal

Para realizar la conversión de octal a hexadecimal se debe hacer un paso intermedio, es decir, primero hay que transformar el número a binario para después convertirlo a hexadecimal.

Ejemplo: $3C6_{16} = ?_8$

1. **Pasar** a binario:

3	C	6
0011	1100	0110

2. **Agrupar** de tres en tres:

00	11	00	11
1	1	0	0
1	7	0	6

$3C6_{16} = 17068$

2.1.8. Paso de hexadecimal a octal

Para realizar la conversión de hexadecimal a octal se debe realizar un paso intermedio, es decir, primero se transforma el número a binario y después se realiza la conversión a octal.

Ejemplo: $17068 = ?16$

1. **Pasar** a binario:

1	7	0	6
00	11	000	110
1	1		

2. **Agrupar** de cuatro en cuatro:

0011 1100 0110
3 C 6

$17068 = 3C616$

2.2. Aritmética binaria

Adicción + | **Sustración -**

Multiplicación * | **División /**

La aritmética binaria es el **conjunto de operaciones aritméticas y lógicas** que se realizan con variables representadas en el sistema binario.

Operaciones aritméticas

- **Adición (+)**:

A	B	A+B
0	0	0
0	1	1
1	0	1
1	1	0*

* Además, colocamos un 1 en la posición inmediata superior (me llevo una); es el digito de arrastres.

Ejemplo: suma los números binarios 11111,100 y 1001000,011

```
        1  1
           1  1  1  1  1  ,  1  0  0
+  1  0  0  1  0  0  0  ,  0  1  1
   1  1  0  0  1  1  1  ,  1  1  1
```

- **Sustracción (-)**:

A	B	A-B
0	0	0
0	1	1*
1	0	1
1	1	0

* Además, colocamos un -1 en la posición inmediata superior (y debo una); es el digito de arrastres.

Ejemplo: resta los números binarios 11111,100 y 1001000,011

-1	-1	-1	-1	-1	-1	-1				
1	0	0	1	0	0	0	,	0	1	1
		1	1	1	1	1	,	1	0	0
0	1	0	1	0	0	0	,	1	1	1

- **Multiplicación (*)**:

A	B	A·B
0	0	0
0	1	0
1	0	0
1	1	1

Ejemplo: multiplica los números binarios 11111,100 y 1001000,011

				1	1	1	1	1	,	1	
				X		1	0	1	,	0	1
					1	1	1		1	1	1
				0	0	0	0		0	0	
			1	1	1	1	1		1		
		0	0	0	0	0	0				
	1	1	1	1	1	1					
1	0	1	0	0	1	0	1	,	0	1	1

- **División (/)**:

A	B	A/B
0	0	Indeterminado
0	1	0
1	0	Infinito
1	1	1

Ejemplo: divide los números binarios 1001 y 11

```
1001 | 11
 11  | 11
----
0011
  11
  --
  00
```

➢ 2.7. Métodos para representar números enteros

Signo y magnitud | **Complemento a 1** | **Complemento a 2**

Exceso a 2n-1

Los ordenadores tienen la necesidad de almacenar números y, al trabajar en binario, deben usar algún método para representar los números enteros, los positivos y los negativos.

En la actualidad, existen diferentes técnicas:

- **Signo y magnitud**: el bit que se encuentra más a la izquierda representa el signo. Si su valor es 0, entonces el número es positivo, mientras que si su valor es 1, entonces el número es negativo.

  El resto de bits representa el módulo del número.

  Ejemplo: representar el 12 y -12 en una palabra de 8 bits en signo y módulo:

  ```
  12   0   0001100
   -   1   0001100
  12
  ```

  Una de las desventajas de este método es la doble representación del 0(10). Tendríamos el conjunto 1000000(2) (-0(10)) y el conjunto 00000000(2)
  (+0(10).

- **Complemento a 1**: se realiza una diferenciación entre los números positivos y los números negativos. En los números positivos, el bit que se encuentra más a la izquierda representa el signo y el resto de los bits corresponden al módulo del número. No obstante, en los números negativos se parte del número positivo, pero después se debe cambiar cada uno de los dígitos. De esta forma, todos los ceros pasan a ser unos y viceversa.

  Ejemplo: representar 12 y -12 en una palabra de 8 bits en complemento a1:

  ```
   12   0   0001100
  -12   1   1110011
  ```

  La desventaja es la misma que en la técnica de signo y magnitud, es decir, tendríamos doble representación del $0_{(10)}$. Tendríamos el conjunto $00000000_{(2)}$ ($+0_{(10)}$) y $11111111_{(2)}$ ($-0_{(10)}$).

Complemento a 2: al igual que en el complemento a 1, se realiza una diferenciación entre los números positivos y los números negativos. En los números positivos, el bit que se encuentra más a la izquierda representa el signo y el resto de los bits corresponden al módulo del número. No obstante, en los números negativos, primero se debe realizar la transformación al complemento a 1 y, después, sumar 1 a dicho resultado. Si el ultimo dígito de la suma tiene acarreo, se desprecia.

Ejemplo: representar 12 y -12 en una palabra de 8 bits en complemento a2:

Positivo:	12	0	0001100
Negativo		1	1110011 Primer paso
			+ 1 Segundo paso
	-12	1	1110100

En esta técnica ya no se puede representar $0_{(10)}$ de formas diferentes. Su única representación será $00000000_{(2)}$.

Ejemplo: representar 0 y -0 en una palabra de 8 bits en complemento a2:

Positivo:	0	0	0000000
Negativo		1	1111111 Primer paso
			+ 1 Segundo paso
	-0	1|0	0000000

Se descarta el acarreo (el 1).

Exceso a 2n-1: si se usa este método, no habrá ningún bit para el signo, sino que todos los bits que componen el número tendrán peso en el valor total del mismo.

Consiste en representar el cero como un valor intermedio, que para *n bits* está representado por 2^{n-1}, y colocar los números negativos antes de ese valor y los positivos después de él. De aquí proviene el nombre.

Ej.: representar 12 y -12 en una palabra de 8 bits en exceso a 2^{n-1}:

Para 8 bits el exceso es $2^{8-1} = 2^7 = 128$. Por tanto, para representar los números pedidos tendremos que sumarle esa cantidad:

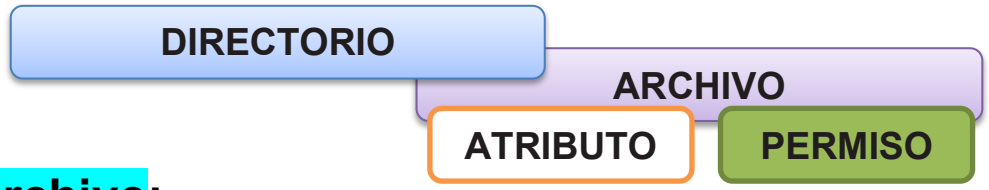
$$12 \rightarrow 12+128=140 \quad 10001100$$
$$-12 \rightarrow -12+128=116 \quad 01110100$$

El mayor inconveniente de esta técnica es su complejidad respecto a las otras, puesto que requiere operaciones intermedias. En la actualidad, los ordenadores utilizan la técnica del complemento a 2 para representar los números enteros.

3.1. Gestión de los archivos

Como se ha comentado anteriormente, la gestión de los archivos es una de las funciones del sistema operativo. Toda la información que se encuentra en el ordendor está almacenada en ficheros y es necesario que el sistema operativo controle su distribución.

Sistemas de archivos



Archivo:

Nombre Archivo.(Extensión) Tamaño

Es un conjunto de bits almacenado que es tratado como una única unidad. Siempre están identificados por un nombre, pero también tienen una extensión y un tamaño, normalmente expresado en bytes (como se ha visto en el capítulo anterior). La extensión es el indicador del formato que tiene un archivo, y este es un estándar que define cómo se codifica la información de dicho archivo. Todos los archivos se guardan con el *nombre.extensión* (ejemplo: *foto1234.jpg*).

Por otro lado, un documento de texto se puede guardar con diferentes formatos

(.***doc***, .***odt*** o .***pdf***),

pero si tenemos un archivo de vídeo, los formatos más utilizados serán

(.***mp4***, .***avi***, .***mpeg***, .***wmv)***, entre otros.

Todos los archivos permiten tres operaciones básicas sobre ellos, que son la **creación**, la **apertura** y el **cierre**.

ARCHIVO		
CREAR	**ABRIR**	**CERRAR**

DIRECTORIO:

Directorio: es un archivo que almacena otro archivos y subdirectorios, por lo que se le denomina contenedor virtual. Además, guarda la ruta y los atributos de dichos archivos.

En los sistemas con entorno gráfico, los directorios forman la estructura de jerarquía entre ellos.

En los temas posteriores, veremos la estructura de los sistemas operativos tanto propietarios como libres.

Atributo:

Los atributos de un archivo son las características del mismo (como *archivo oculto*, que indica si el archivo está visible o no). Los atributos de los archivos y directorios son los mismos para todos los usuarios y grupos que hay en el sistema. No obstante, aquellos que existen entre directorios y archivos son diferentes, pero también cambian en función del tipo de sistema operativo que gestione el sistema.

Permisos:

Son una serie de reglas de acceso para las operaciones de lectura, escritura y ejecución que garantizan la seguridad en el sistema de archivos. Al contrario que los atributos, estos se establecen para cada tipo de usuario.

Un usuario puede crear un fichero que solo pueda utilizar él o crear uno que puedan utilizar todos. Además, como los permisos se establecen para cada una de las operaciones, el usuario puede crear un fichero sobre el que pueda realizar todas las operaciones y el resto de usuario solo leerlo.

Más adelante, se verán los permisos sobre los archivos y directorios de forma específica, tanto en los sistemas operativos propietarios como libres, y también como se pueden modificar estos permisos.

3.2. Gestión de la memoria

En los sistemas operativos multiproceso, la memoria RAM no suele tener capacidad suficiente para abarcar todos los procesos que se encuentran en ejecución en un ordenador. Por ello, el sistema operativo debe distribuir la memoria entre varios.

A veces, la cantidad de memoria necesaria para la ejecución de un programa es mayor que la disponible dentro de nuestro sistema, lo que puede provocar situaciones indeseables. Para poder hacer frente a este problema se diseñó un mecanismo llamado *overlay*, es decir, **solapamiento**.

Esta técnica permite **dividir el programa vitualmente en procesos** para que estos se ejecuten en diferentes partes de la memoria RAM. De esta forma,

una parte del programa reside en el disco duro, mientras que otra está en ejecución en la memoria.

A pesar de que el problema se solucionó, este método tenía un problema relacionado con la programación. Durante el desarrollo, el programador era el encargado de ejecutar las llamadas al sistema dentro de la aplicación para realizar estas divisiones. Al final, esto fue inviable puesto que cada sistema se compone de unas características diferentes (por ejemplo, no todos tienen la misma cantidad de memoria RAM) y necesitaba una programación diferente para todas las aplicaciones.

Resultó inútil para los sistemas multiusuario pues, además de dividir los programas, era necesario reservar memoria para los distintos usuarios que se encontraran usando el sistema. Se llegó entones a la conclusión de que era necesario **gestionar la memoria dinámicamente**.

Para llevar esto a cabo, el primer paso consiste en asegurarse de que un proceso se ejecuta en una parte libre de la memoria principal. Además, es necesario controlar el acceso a los recursos compartidos. Los datos se comparten entre todos los procesos del sistema y hay que tener en cuenta que, si un proceso está accediendo a uno de ellos, ya sea para leer o escribir, no es posible que otro proceso acceda al mismo dato. Por último, también es necesario que el gestor de memoria resuelva las “colas” de ejecución, es decir, que un proceso no se quede esperando para conseguir una posición en la memoria cuando hay otra libre.

GESTIÓN DE LA MEMORIA EN LA MULTIPROGRAMACIÓN

(Técnicas)

Particiones fijas | **Particiones variables** | **Memoria virtual**

En la multiprogramación, la gestión de la memoria se puede realizar a través de diferentes técnicas, entre las que encontramos **particiones fijas**, **variables** y la memoria virtual.

ESQUEMA

Particiones fijas

• Gestión de memoria mediante particiones fijas

- Fragmentación interna
- Fragmentación externa

La memoria se divide en particiones que no podrán modificarse después. Los procesos se van ejecutando en estas partes de la memoria, pero hay que tener en cuenta que en una partición solo se ejecuta un único proceso.

Para **asignar un proceso a una partición**, el gestor de la memoria tiene dos estrategias:

- Cola única
- Cola por cada partición

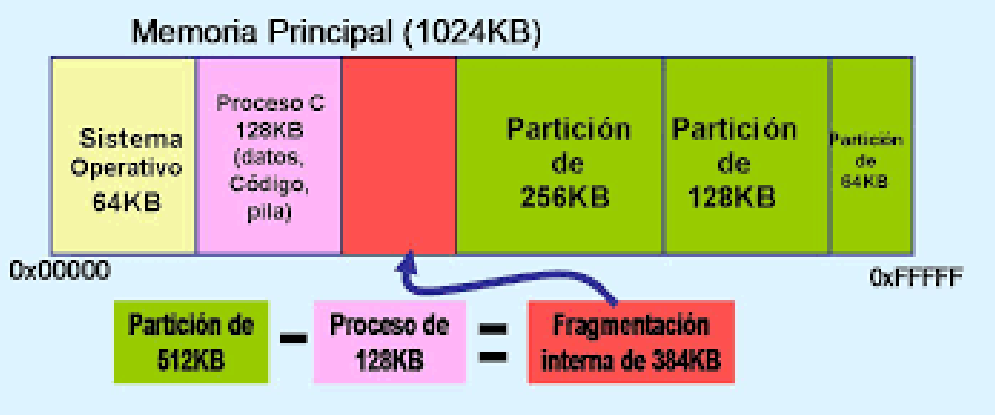
Se puede realizar la **asignación de memoria** de diferentes maneras:

- Asignar el **primer proceso de la cola a un espacio según quede libre**.
- Asignar el **primer proceso de la cola que quepa en el espacio que ha quedado libre**.
- Asignar el **proceso más grande de la cola que quepa en el espacio que ha quedado libre**.

EXPLICACIÓN

Por lo tanto, esto tiene dos grandes problemas: la **fragmentación interna** y la **fragmentación externa**. Ambos son consecuencia de que parte de la memoria principal se desaprovecha.

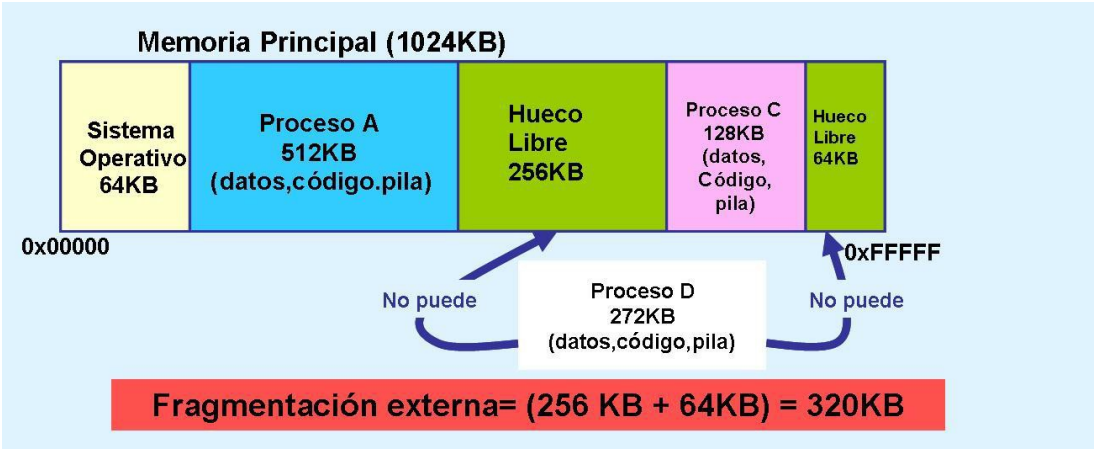
- **Fragmentación interna**: hay procesos que ocupan un espacio menor que el que tienen asignado



en la memoria principal.

Fuente: Blog el mundo informático

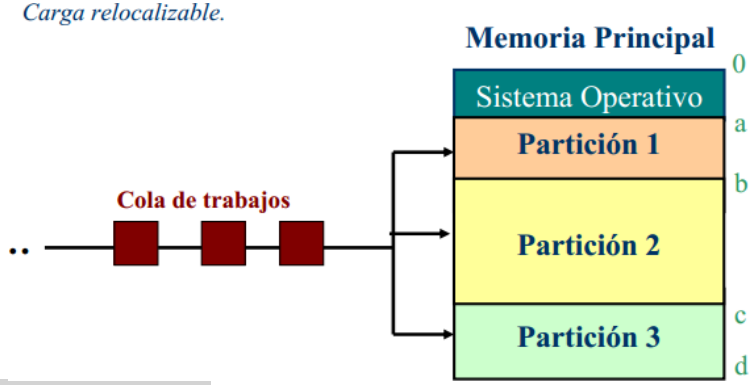
- **Fragmentación externa**: hay procesos que ocupan más que la partición de la memoria principal, por lo que esa partición queda libre.



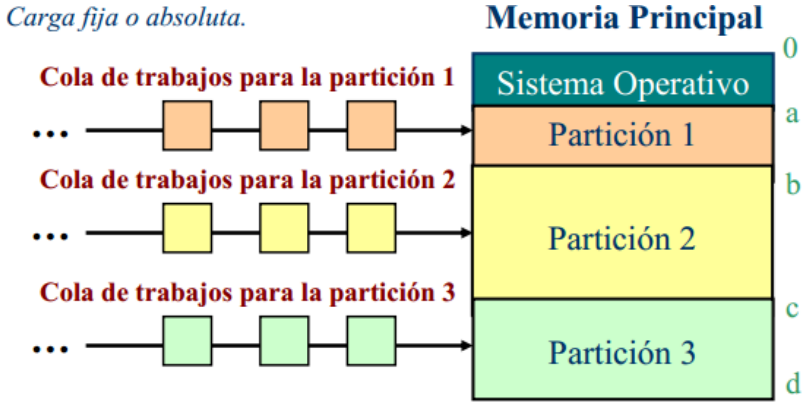
Para **asignar un proceso a una partición**, el gestor de la memoria tiene dos estrategias:

- Cola única
- Cola por cada partición

- **Cola única**: *Carga relocalizable.*



- **Cola por cada partición**: el usuario es el encargado de establecer estas particiones. No es necesario que todas tengan el mismo espacio, pues el gestor de memoria es el encargado de controlar la ejecución de estos procesos, es decir, indica en qué partición se colocará cada uno de ellos.

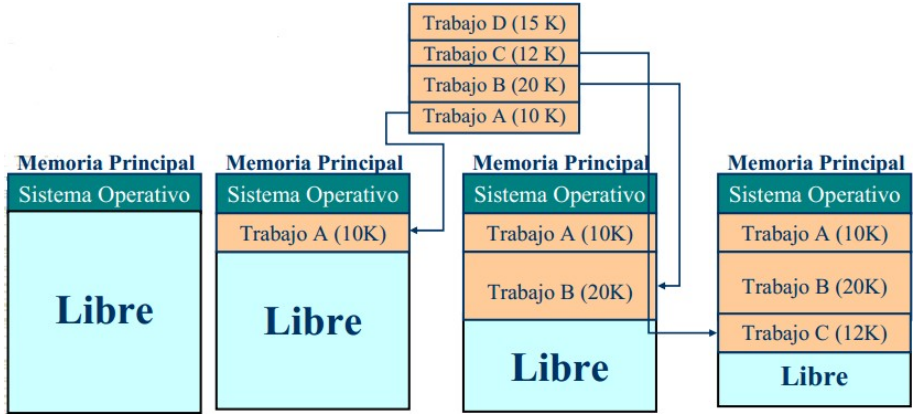


Se puede realizar la **asignación de memoria** de diferentes maneras:

- Asignar el **primer proceso de la cola a un espacio según quede libre**. Si el proceso es mayor que el hueco, no se ejecuta.
- Asignar el **primer proceso de la cola que quepa en el espacio que ha quedado libre**.
- Asignar el **proceso más grande de la cola que quepa en el espacio que ha quedado libre**. En este algoritmo se excluye a los procesos más cortos, puesto que se prioriza los largos para aprovechar al máximo la memoria. Esto se podría resolver estableciendo una partición de tamaño menor para este tipo de procesos.

Gestión de memoria mediante particiones variables

Mediante este método, **la memoria se particiona según la ejecución de los procesos**, es decir, un proceso solo ocupa en memoria el espacio que necesita. El número de particiones, tamaño y posición cambian según se va utilizando esta.



Se puede utilizar de **dos formas**:

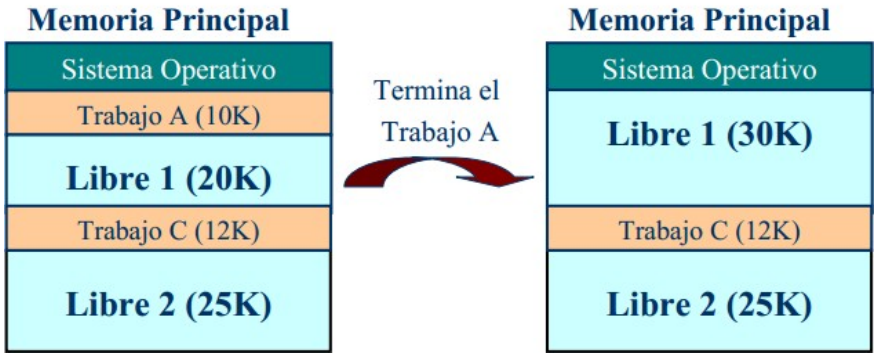
- **Cuando un proceso termina, se combina el hueco que deja libre con el que hay disponible** al lado.
- **Cuando un proceso termina, se compactan los espacios ocupados de la memoria**

Estrategias: (De 1er ajuste)… (De 2o ajuste)… (De 3er ajuste)… (De 4o ajuste)…

Se puede utilizar de **dos formas**:

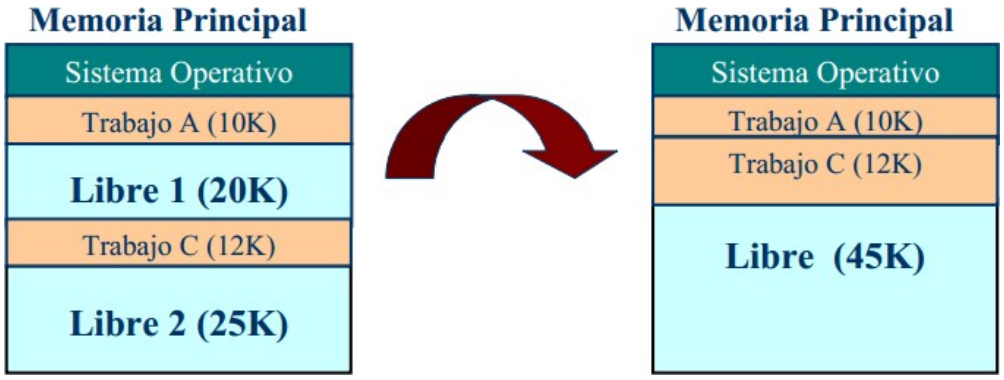
- **Cuando un proceso termina, se combina el hueco que deja libre con el que hay disponible** al lado.

Combinación de huecos:

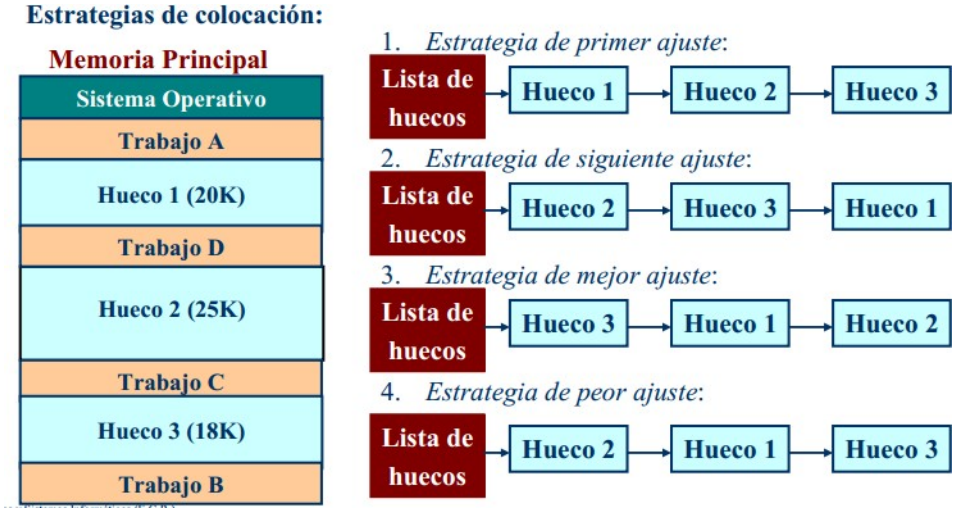


Cuando un proceso termina, se compactan los espacios ocupados de la memoria, por lo que al principio está toda la memoria ocupada y después toda la memoria libre.

Compactación de memoria:



- Esta técnica tiene diferentes estrategias para gestionar los procesos:



- **De primer ajuste**: se asigna al primer proceso de la cola el primer hueco que le sirva.
- **De siguiente ajuste**: se asignan los procesos por orden de cola.
- **De mejor ajuste**: se asigna el hueco más pequeño al proceso que mejor se adapte al espacio.
- **De peor ajuste**: se asigna el hueco más grande al primer proceso de la cola.

Gestión de la memoria mediante memoria virtual

El gestor de memoria utiliza el disco duro o un espacio secundario de almacenamiento como si fuera parte de la **memoria principal del sistema**. De esta forma, trabaja con una memoria RAM de mayor almacenamiento que la que el sistema tiene físicamente.

PROGRAMA

Un programa se divide en **capas activas** y **capas inactivas**.

Las primeras se forman con aquellos procesos que se encuentran en ejecución dentro de la memoria principal, mientras que las segundas están formadas por aquellos que se encuentran en la memoria secundaria.

Mediante esta técnica, el usuario piensa que el programa se localiza en la memoria RAM, pero en ella solo está realmente la parte que se está ejecutando. El resto del programa se encuentra en la memoria virtual, esperando a que sea necesaria su ejecución.

CAMBIAR LOS PROCESOS DE MEMORIA MEDIANTE ESTAS TÉCNICAS

➢ *Swapping*

En este caso, se pueden cambiar dichos procesos de memoria mediante la técnica denominada ***swapping***.

ESQUEMA

Existen **dos técnicas principales** para usar la memoria secundaria como memoria virtual:

- **Paginación**:
- **Segmentación**:

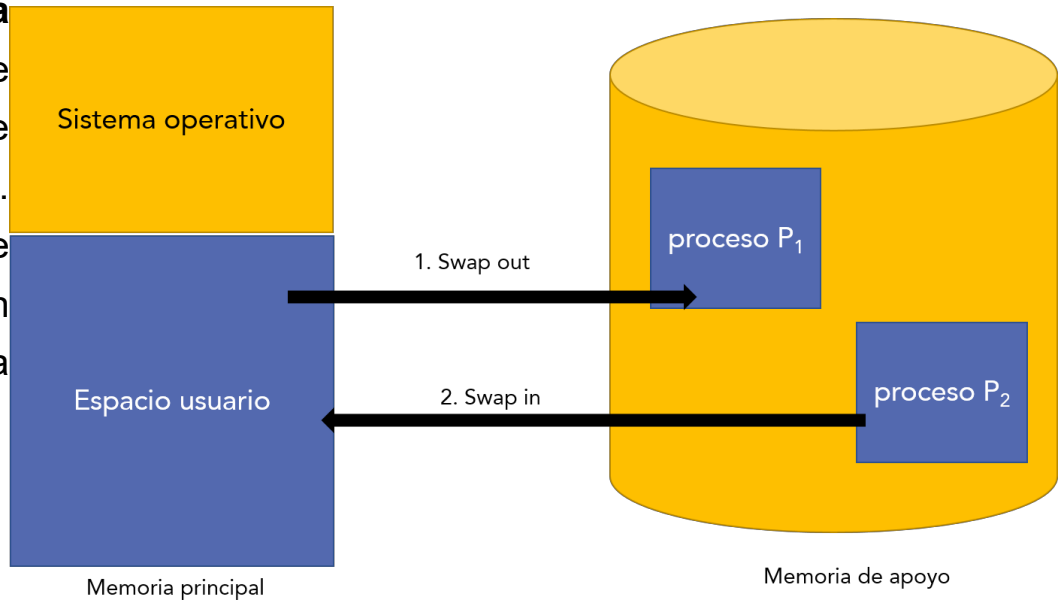
EXPLICACIÓN

➢ *Swapping*

Esta técnica consiste en mover un proceso que se encuentra en la memoria principal al disco duro, y después, devolverlo a la memoria principal.

DESVENTAJA

El principal problema de esta técnica reside en la ralentización del sistema, puesto que constantemente se produce un intercambio de información entre la memoria RAM y el disco duro. Además, si un proceso hace referencia a otro que aún no se encuentra en la memoria, se produce un fallo de página que obliga al gestor de memoria a recuperar ese proceso de la memoria virtual.



Existen **dos técnicas principales** para usar la memoria secundaria como memoria virtual:

- **Paginación**: tanto la propia memoria como los procesos se dividen en unidades más pequeñas.

  Los programas están distribuidos en segmentos dentro de la memoria principal, denominadas **unidades lógicas** o **páginas**, mientras que la memoria lo hace en diferentes secciones o partes de igual tamaño que las páginas, conocidas como **marcos de página**.

  Esta técnica minimiza la fragmentación interna y evita la externa, puesto que, cuando la memoria y los segmentos de los procesos tienen el mismo tamaño, no se desperdicia memoria RAM por cada partición, sino solamente en la última página de un programa.

Para conocer cómo el sistema operativo relaciona las **direcciones de memoria** física con las lógicas mediante esta técnica, se puede visitar este enlace

(https://es.wikipedia.org/wiki/*Paginaci*%C3%B3n_de_memoria)

- **Segmentación**: los programas también se dividen en segmentos, pero, al contrario que en la paginación, estos son de diferente tamaño.

  Esta técnica permite, entre otras cosas, la modularidad de programas, es decir, que cada rutina dentro de un programa permita cambios que no afecten al resto del programa. También admite la compilación de módulos por separado, lo que hace que la modificación de cada uno de ellos sea más fácil.

3.3. Gestión de los procesos

Como se ha comentado anteriormente, un **proceso** es un conjunto de instrucciones que se ejecutan dentro de la CPU.

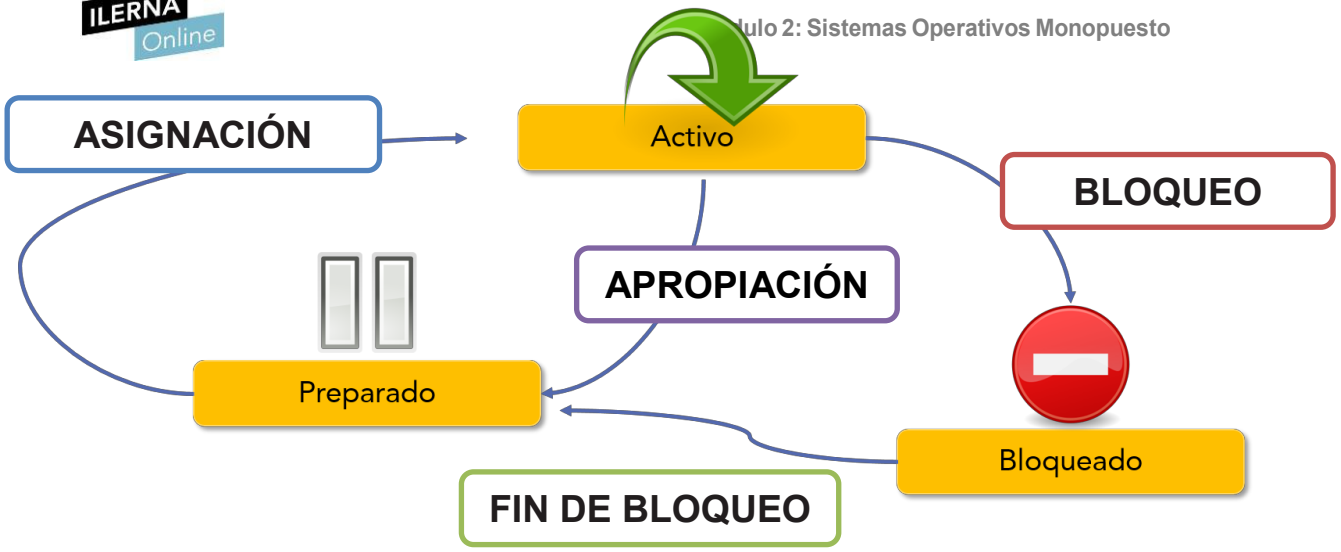
Estado de los procesos

Todos los procesos tienen un **indicador** que define la situación en que se encuentran con respecto a su funcionamiento. Existen, como mínimo, tres estados diferentes:

ACTIVO o en Ejecución | **Bloqueado** | **Preparado**

- **Activo o en ejecución**: proceso que está asignado para ejecutarse en el procesador.
- **Bloqueado**: proceso que ha interrumpido su ejecución y que se encuentra a la espera de que termine la operación que le ha dejado bloqueado.
- **Preparado**: proceso que se encuentra disponible para ejecutarse.

Durante la vida de un proceso se producen cambios entre sus distintos estados. En el siguiente diagrama se muestran las transiciones permitidas entre ellos.


ASIGNACIÓN
Activo
BLOQUEO
APROPIACIÓN
Preparado
Bloqueado
FIN DE BLOQUEO

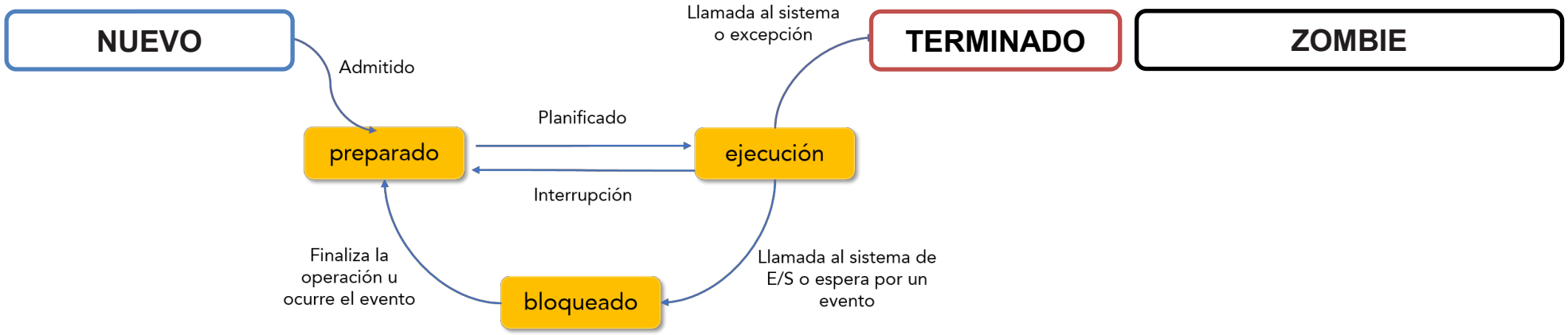
- **Bloqueo**: cuando un proceso se está ejecutando y produce una llamada al sistema, debe bloquearse para evitar consumir CPU. Es lo que ocurre, por ejemplo, cuando requiere información y pide que se lean datos; en este caso, debe esperar a que se complete esa operación.

- **Apropiación**: cuando un proceso se encuentra en ejecución y el gestor de procesos indica que debe detenerse, tiene que salir de la CPU hasta que pueda volver a estar activo.

- **Asignación**: cuando un proceso entra a ejecutarse en la CPU.

- **Fin de bloqueo**: cuando un proceso está esperando a que acabe la operación por la cual ha pasado a estar en el estado de bloqueo para continuar con su ejecución.

En Unix existen más estados para los procesos que los que se han explicado anteriormente. En este diagrama se pueden observar las transiciones entre ellos.



Además, cabe destacar que tenemos varios estados nuevos:

- **Nuevo**: proceso que aún no ha sido elegido para iniciar su procesamiento.
- **Terminado**: proceso que ha finalizado su ejecución.
- **Zombie**: proceso que ha finalizado su ejecución pero que no ha liberado los recursos que ha utilizado.

El sistema operativo debe agrupar la información de todos los procesos del sistema, la cual se refiere al identificador del proceso, estado, prioridad, recursos y permisos asignados, etcétera. Además, se encuentra en el bloque de control del proceso (BCP), que se crea a la vez que el proceso. Cuando este es eliminado, se borra también toda la información.

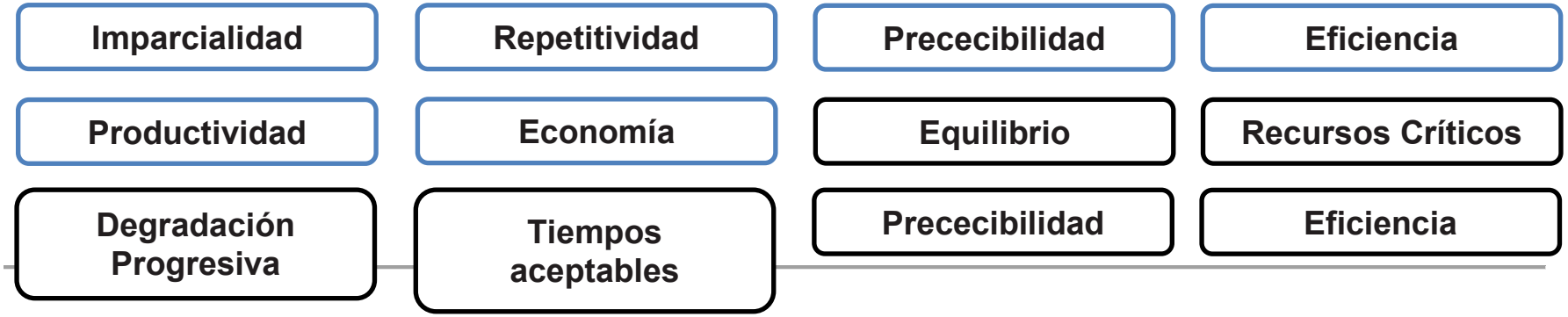
➢ PRIORIDAD Y PLANIFICACIÓN

En un sistema complejo, un buen **criterio de planificación** debe tener en cuenta muchos aspectos, algunos de ellos contrarios. Por ejemplo, sería deseable que todos los procesos obtuvieran el procesador en cuanto lo necesitaran, pero el tiempo del procesador es limitado y todo criterio que favorezca a un tipo de proceso perjudicará a otros.

De la misma forma, es necesario que siempre haya trabajo preparado para cada dispositivo de cara a aumentar el **aprovechamiento** del equipo. Esto obliga a aumentar el número de procesos en espera, lo que incrementa el tiempo que los procesos están inactivos por falta de algún recurso.

Necesitamos encontrar un equilibrio entre esperas y recursos, el cual varía según la finalidad del equipo. Por ejemplo, en los sistemas de tiempo real es necesario que los procesos obtengan los recursos rápidamente, para lo que hay que tener varias unidades de cada uno; así será más probable que haya alguno disponible, aunque estarán más tiempo sin usarse y se aprovecharán menos. Por otro lado, en otros tipos de sistemas tienen preferencia la economía y el aprovechamiento de los recursos.

Los **ASPECTOS PRINCIPALES** PARA TENER EN CUENTA PARA **DISEÑAR UN MÉTODO DE PLANIFICACIÓN** son:



- **Imparcialidad**: el planificador debe asegurar que cada proceso tenga la fracción de tiempo de procesador que le corresponde.

- **Repetitividad**: es de esperar que, con cargas de trabajo similares, se presenten comportamientos similares.

- **Predecibilidad**: el tiempo de procesamiento de un trabajo y el coste de ejecutarlo serán iguales, más o menos, con cualquier carga de trabajo del equipo.

- **Eficiencia**: el planificador debe procurar que el procesador y los demás recursos del equipo estén trabajando el mayor tiempo posible.

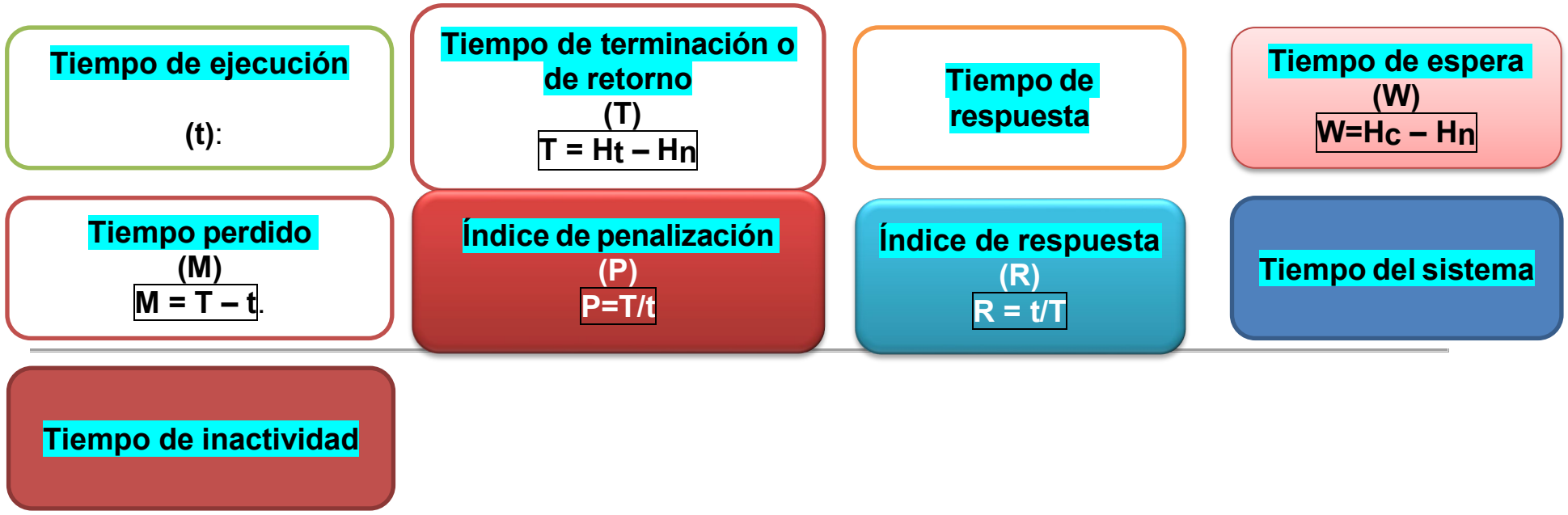
- **Productividad**: se expresa mediante la cantidad de trabajo que se realiza por unidad de tiempo.

- **Economía**: el objetivo es reducir los gastos añadidos al mínimo.
- **Equilibrio**: se procura equilibrar el aprovechamiento de los recursos manteniendo ocupados todos los componentes del equipo.
- **Recursos críticos**: se debe dar preferencia a aquellos procesos que están ocupando recursos críticos, para que terminen lo antes posible y los liberen.
- **Degradación progresiva**: la respuesta del sistema debe presentar una degradación lo más uniforme posible al incrementarse la carga de trabajo.
- **Tiempos aceptables**: el grado de satisfacción que tienen los usuarios respecto al sistema depende del tiempo que deben esperar.

Asimismo, NECESITAMOS REFERENCIAS NUMÉRICAS PARA COMPARAR OBJETIVAMENTE LOS MÉTODOS DE PLANIFICACIÓN. ESTO LLEVA AL ESTABLECIMIENTO DE UNA SERIE DE

PARÁMETROS E ÍNDICES

QUE SE BASAN EN CARACTERÍSTICAS DE TIEMPO:



(H_n) = Tiempo de llegada (Aparece en la tabla)

(t) = Tiempo de ejecución (Aparece en la tabla del ejercicio práctico)

Son las ejecuciones (nº de veces) en la Tabla

- **Tiempo de ejecución (t)**: es el tiempo de servicio que necesita un proceso.

- **Tiempo de terminación o de retorno (T)**: es el tiempo que transcurre entre la hora de llegada del trabajo al ordenador y su hora de finalización. Mide el tiempo que un proceso está presente en el equipo y la fórmula para calcularlo es: **T = Ht – Hn**.

- **Tiempo de respuesta**: es el tiempo que transcurre desde que se solicita algo hasta que se obtiene.

- **Tiempo de espera (w)**: es el tiempo que transcurre entre la hora de llegada del proceso y la hora en que empieza a ejecutarse, es decir, es el tiempo que el proceso debe esperar hasta que pasa por primera vez al estado de preparado. La fórmula para calcularlo es: **W=Hc – Hn**.

- **Tiempo perdido (M)**: es la diferencia del tiempo de ejecución al tiempo de finalización. La fórmula para calcularlo es: **M = T – t**.

- **Índice de penalización (P)**: es el cociente entre el tiempo de finalización y el tiempo de ejecución. La fórmula para calcularlo es: **P=T/t**.

- **Índice de respuesta (R)**: es el inverso al anterior. La fórmula para calcularo es: **R = t/T**.

- **Tiempo del sistema**: es el tiempo que consume el sistema operativo en ejecutar los métodos de planificación establecidos, los cuales incluyen la comunicación de un proceso a otro.

- **Tiempo de inactividad**: es el tiempo que el procesador permanece desocupado cuando no hay procesos preparados para ejecutar.

Los métodos de planificación de procesos se **clasifican** como:

No apropiativos

Apropiativos

- **No apropiativos**: si, una vez asignado el procesador a un proceso, ya no se le puede quitar. Los inconvenientes son el coste en pérdidas de tiempo al cambiar de proceso y la coordinación del acceso a datos compartidos. Además, hay que evitar que las estructuras de datos del núcleo puedan quedar inconsistentes por los cambios de contexto.

- **Apropiativos**: si, una vez asignado el procesador a un proceso, se le puede retirar. Esto genera un problema de desaprovechamiento de la CPU.

Método:

No apropiativos

Métodos no apropiativos

FCFS (*First Come First Served*)

SJN (Shortested Job Next)

Metodo1. > FCFS (*First Come First Served*):

Se le asigna un recurso al primer proceso que llega. Es el procedimiento más sencillo y se emplea en las dos planificaciones:

- **De trabajos**: se ejecutan en el orden de llegada.
- **De procesos**: se añade al final de la cola y se ejecutan según el orden de incorporación.

Ventajas	Inconvenientes
Fácil de programar	Los índices de funcionamiento no son buenos
Necesita pocos recursos	
Consume muy poco tiempo de procesador	

Ejemplo. Imaginemos que tenemos varios trabajos, con los tiempos de ejecución que se indican en la siguiente tabla.

Tarea	H_n	t
A	0	3
B	1	5
C	3	2
D	9	5
E	12	5

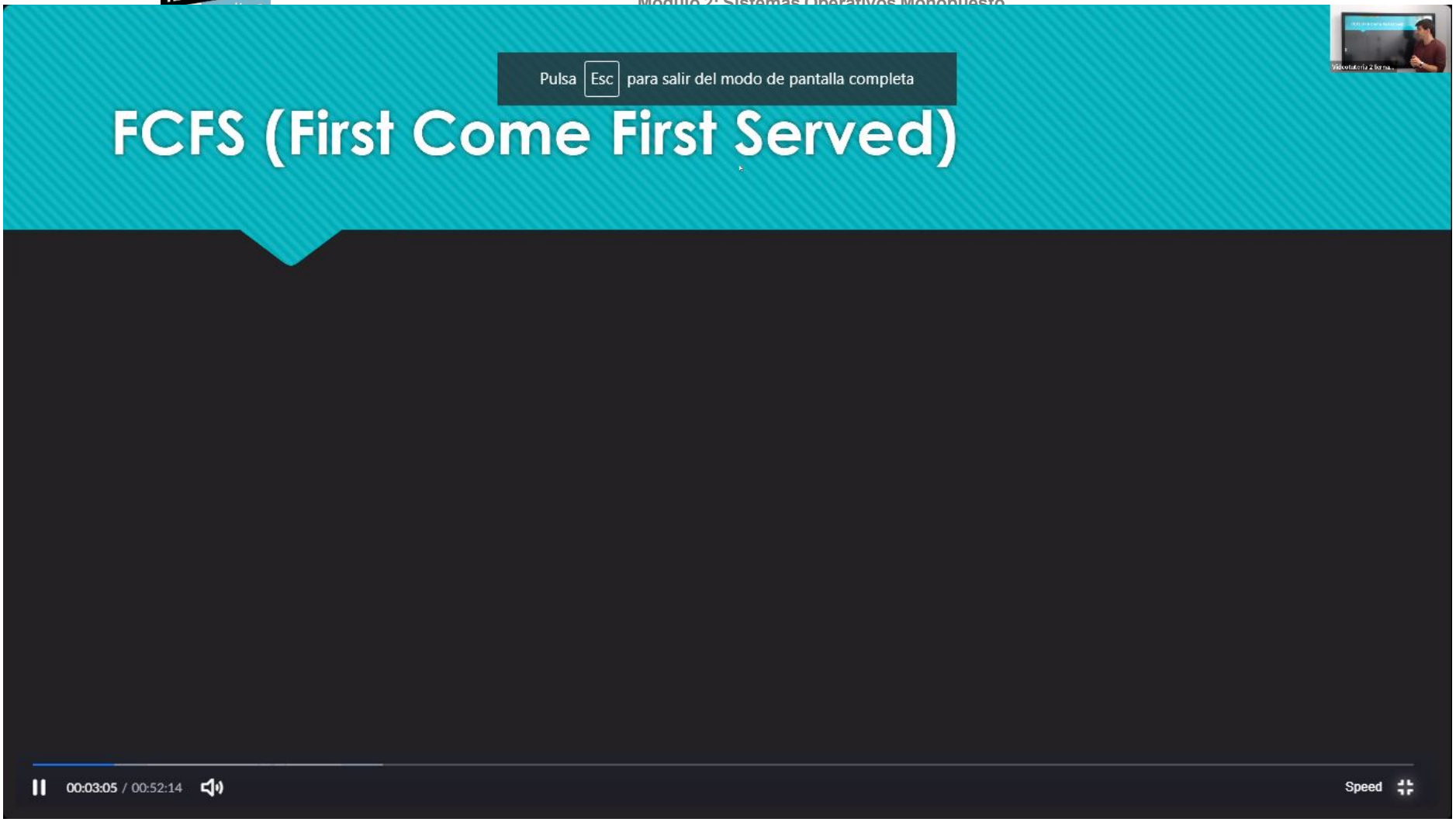
Si cada tarea se inicia cuando acaba la anterior, obtendríamos el siguiente resultado:

Tarea	H_n	t	H_c	H_t	T	M	P
A	0	3	0	3	3	0	1,0
B	1	5	3	8	7	2	1,4
C	3	2	8	10	7	5	3,5
D	9	5	10	15	6	1	1,2
E	12	5	15	20	8	3	1,6
Valores medios					6,2	2,2	1,74

Los valores medios pueden parecer aceptables, pero, si miramos con detenimiento, el índice de penalización del proceso C resulta exagerado. Eso sucede porque todo trabajo corto que llega poco después de uno largo tiene un índice de penalización grande.

Este método se utiliza poco, pero es frecuente encontrarlo combinado con otros.


Pulsa Esc para salir del modo de pantalla completa
GESTIÓN DE PROCESOS
MÉTODOS NO APROPIATIVOS
Y
MÉTODOS APROPIATIVOS
00:00:55 / 00:52:14
Speed

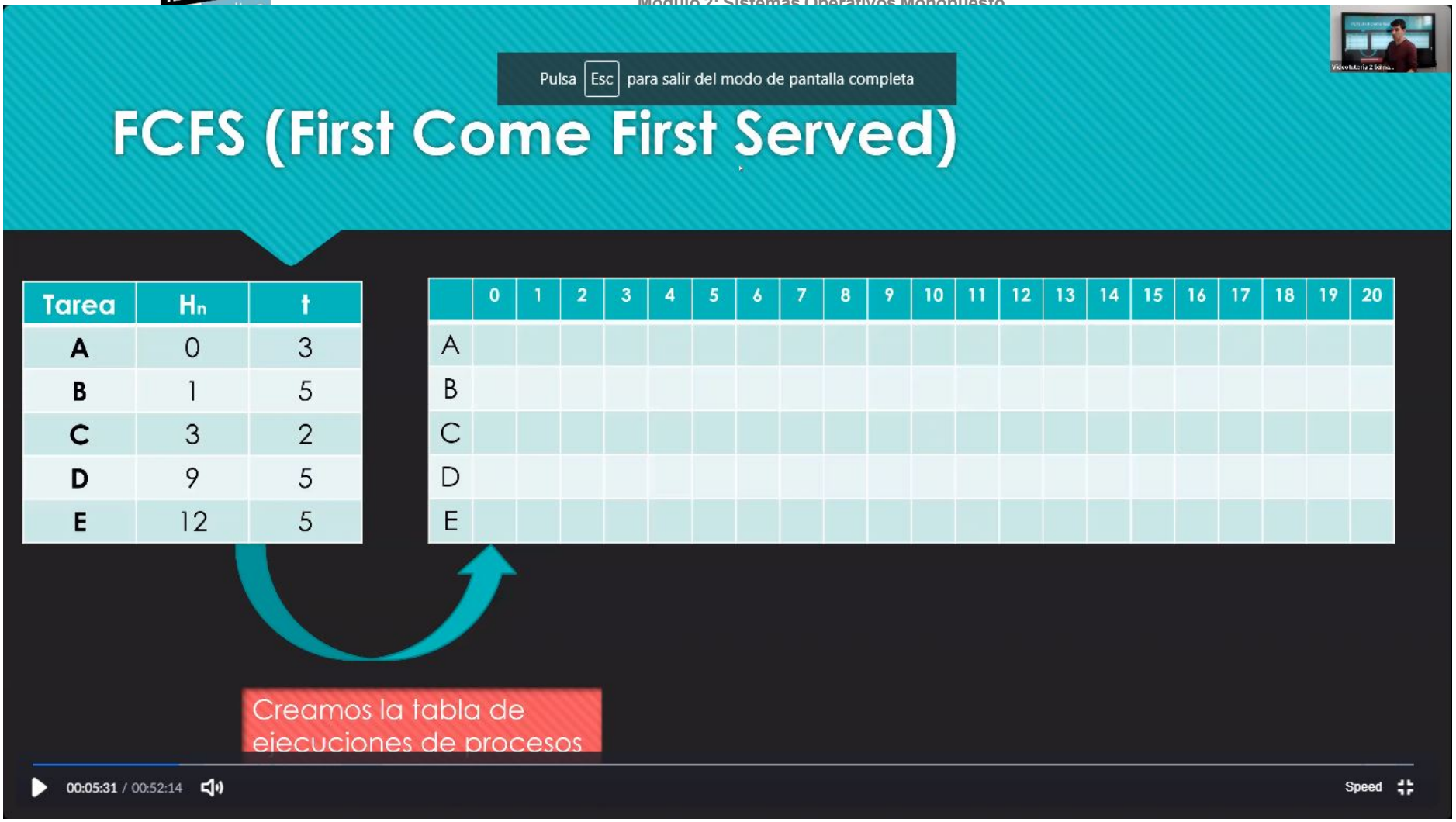

Pulsa Esc para salir del modo de pantalla completa
FCFS (First Come First Served)
00:03:05 / 00:52:14
Speed

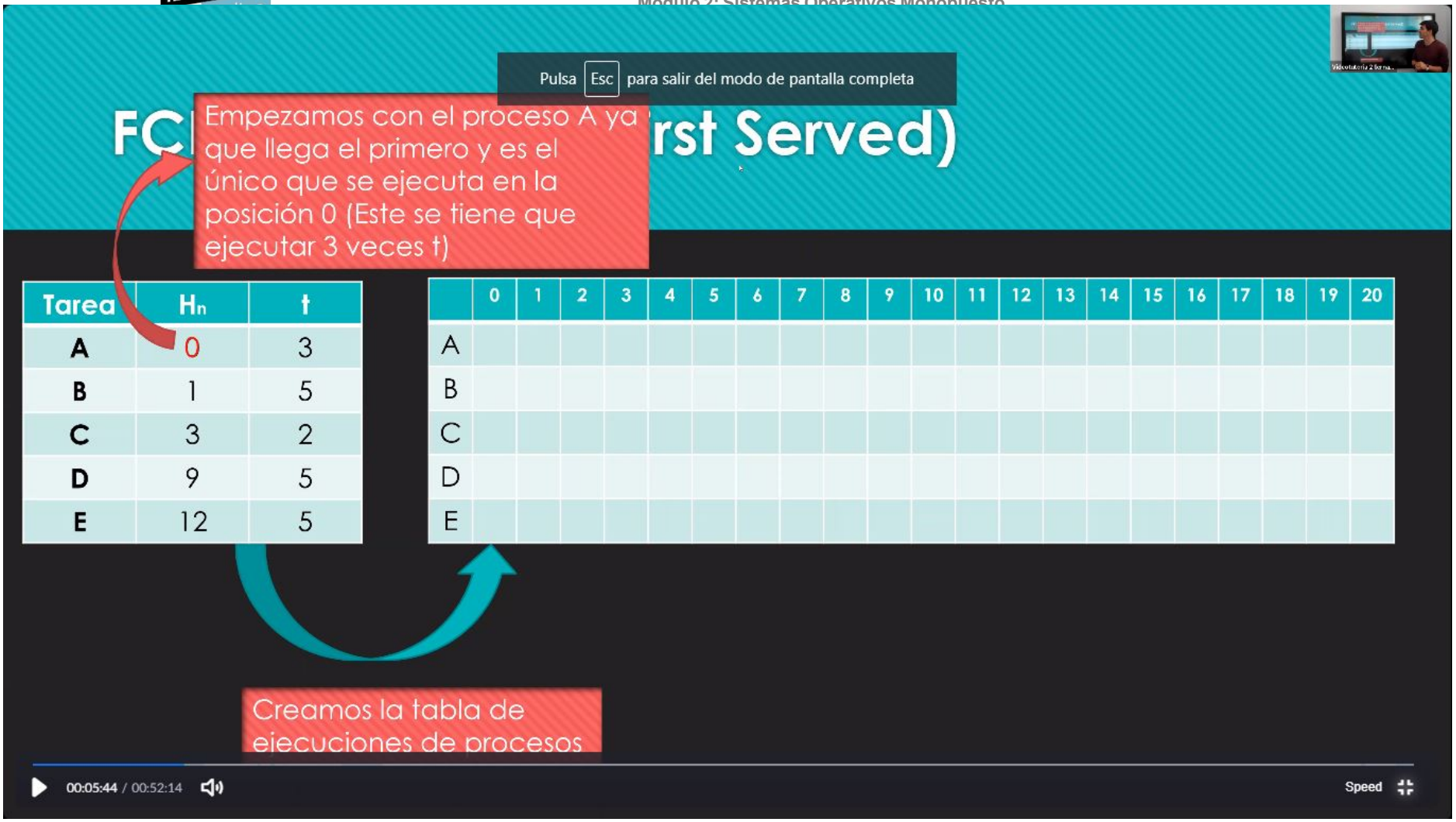
Pulsa Esc para salir del modo de pantalla completa

FCFS (First Come First Served)

Tarea	H_n	t
A	0	3
B	1	5
C	3	2
D	9	5
E	12	5

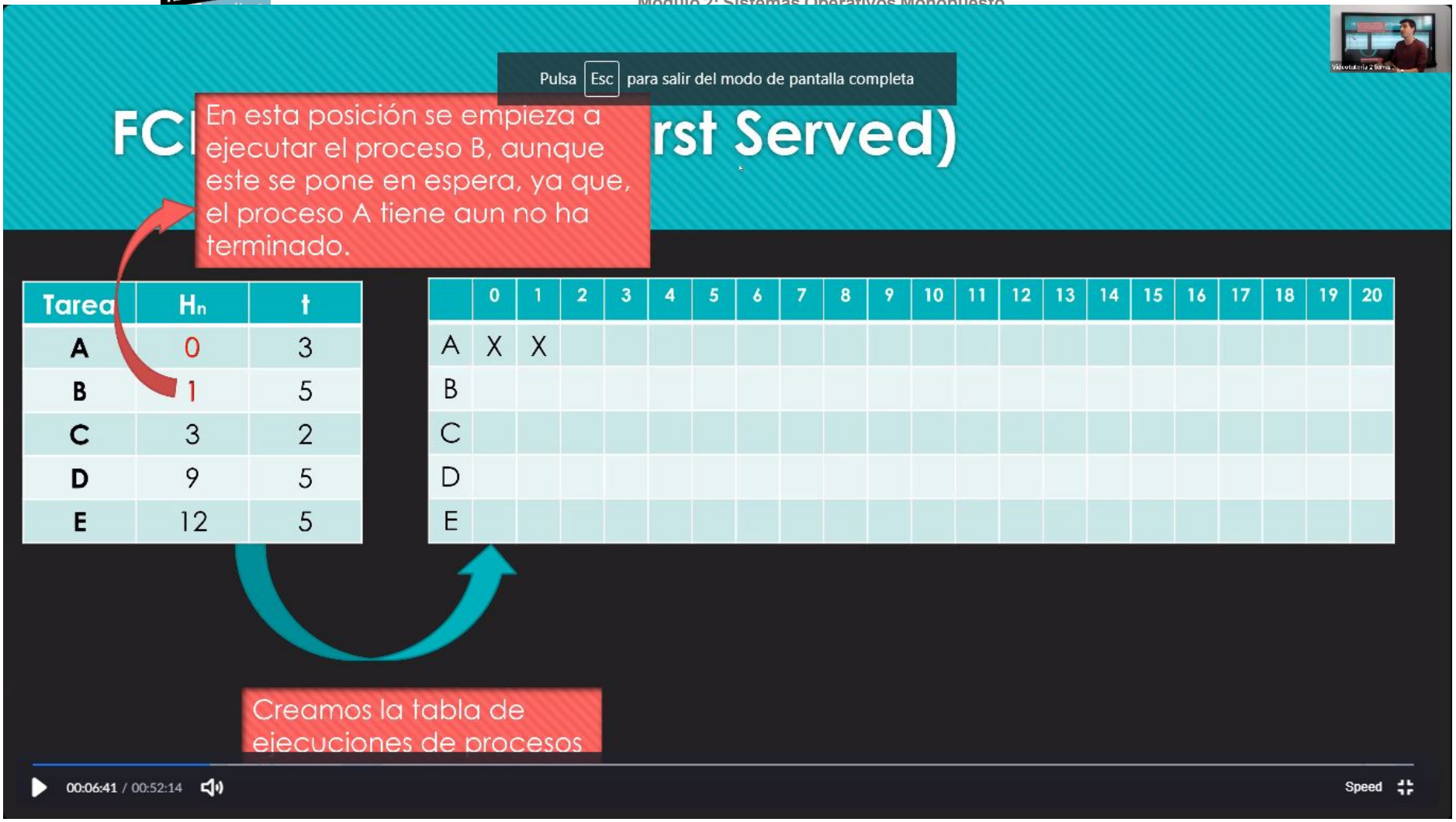
00:05:02 / 00:52:14 Speed


Pulsa Esc para salir del modo de pantalla completa
FCFS (First Come First Served)
Tarea | Hn | t
A | 0 | 3
B | 1 | 5
C | 3 | 2
D | 9 | 5
E | 12 | 5
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
A
B
C
D
E
Creamos la tabla de ejecuciones de procesos
00:05:31 / 00:52:14
Speed



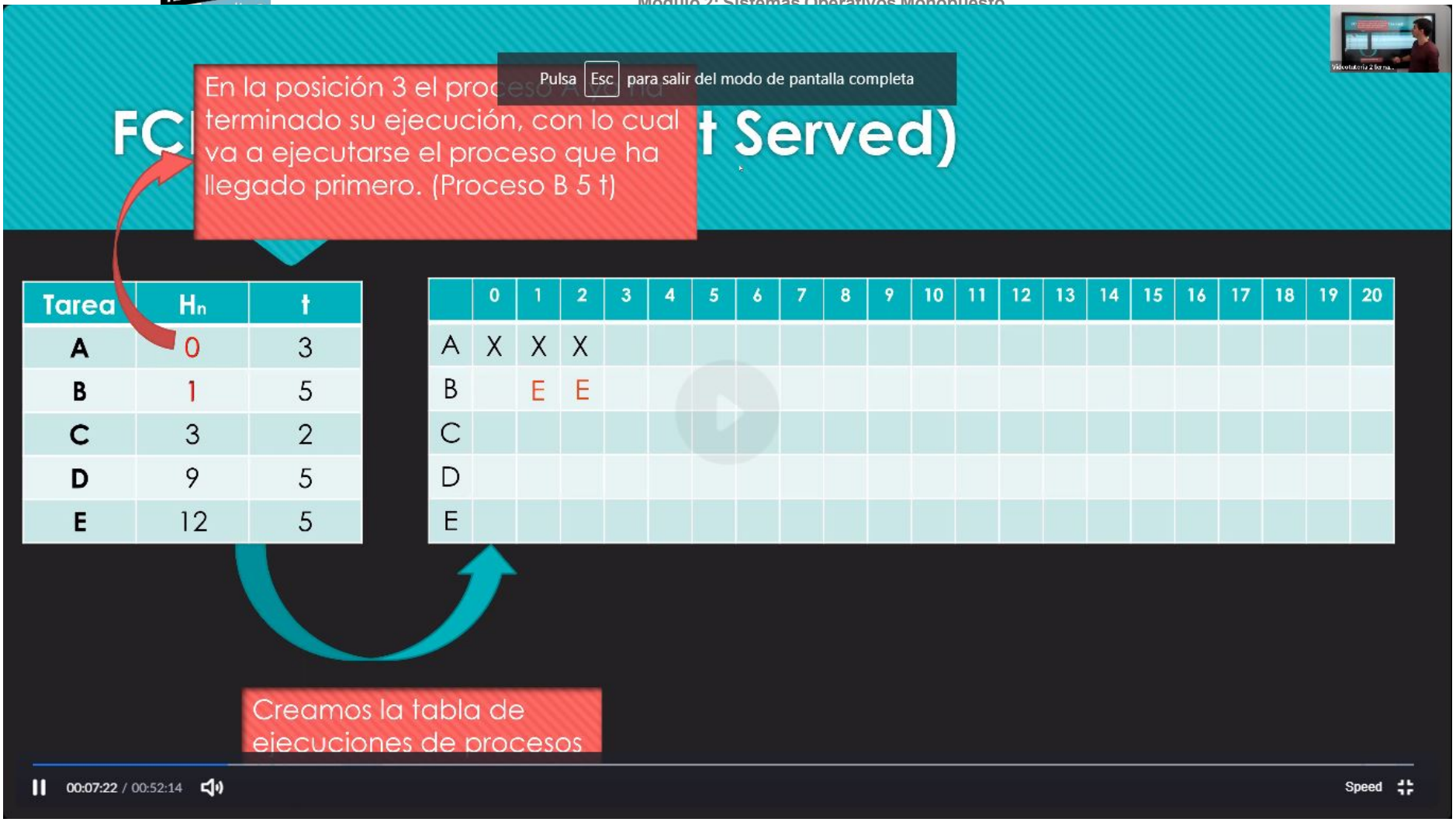
Tarea	H_n	t
A	0	3
B	1	5
C	3	2
D	9	5
E	12	5

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
A																					
B																					
C																					
D																					
E																					



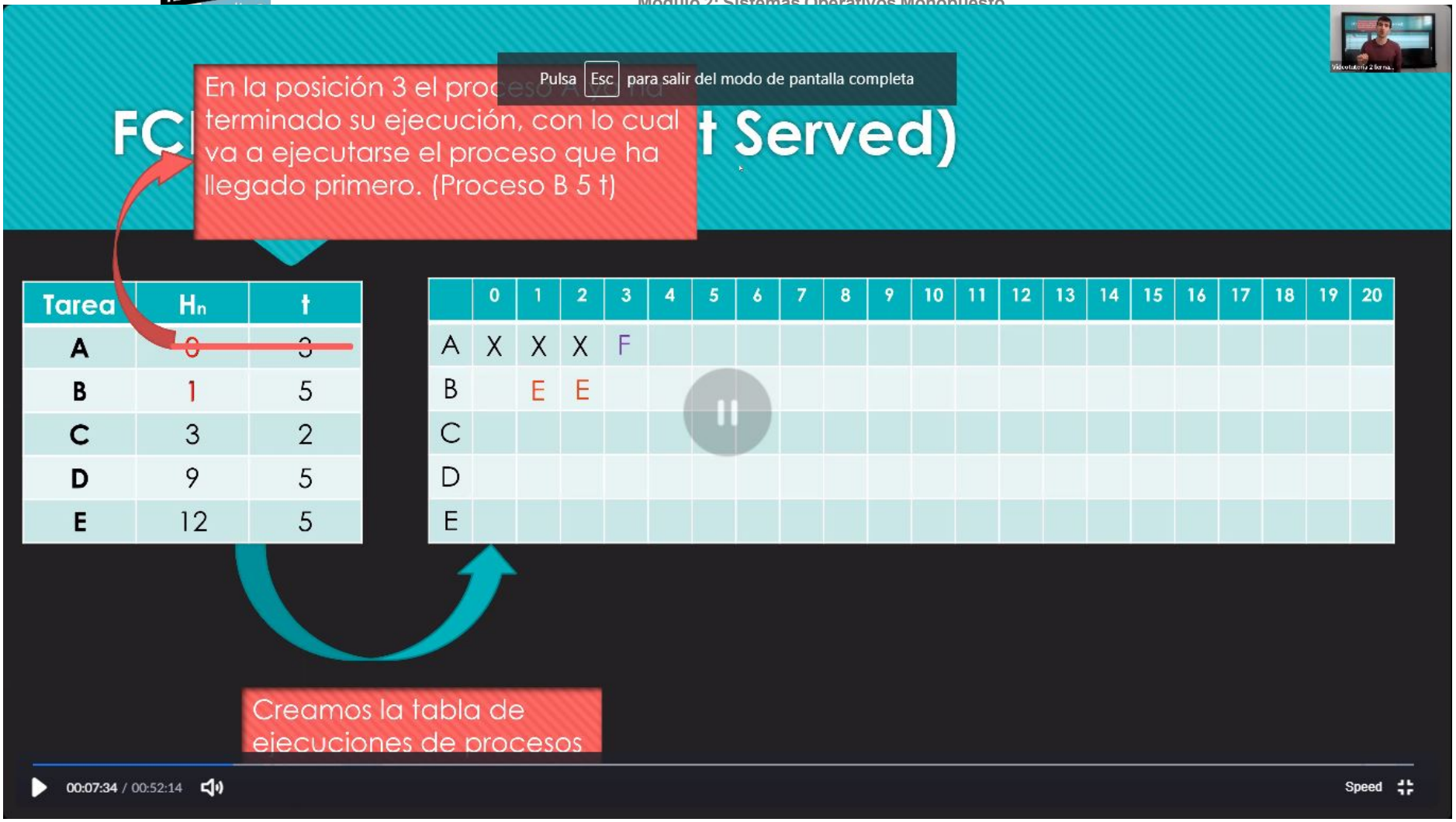
Tarea	H_n	t
A	0	3
B	1	5
C	3	2
D	9	5
E	12	5

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
A	X	X																			
B																					
C																					
D																					
E																					



Tarea	H_n	t
A	0	3
B	1	5
C	3	2
D	9	5
E	12	5

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
A	X	X	X																		
B		E	E																		
C																					
D																					
E																					


Pulsa Esc para salir del modo de pantalla completa
FC t Served)
En la posición 3 el proceso A ha terminado su ejecución, con lo cual va a ejecutarse el proceso que ha llegado primero. (Proceso B 5 t)
Tarea | Hn | t
A | 0 | 3
B | 1 | 5
C | 3 | 2
D | 9 | 5
E | 12 | 5
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
A X X X F
B E E
C
D
E
Creamos la tabla de ejecuciones de procesos
00:07:34 / 00:52:14
Speed

FC[illegible] Served)

En esta posición llega el proceso C aunque se pone en espera, ya que, el proceso B a un no ha terminado.

Tarea	H_n	t
A	~~0~~	~~3~~
B	1	5
C	3	2
D	9	5
E	12	5

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
A	X	X	X	F																	
B		E	E	X																	
C																					
D																					
E																					

Creamos la tabla de ejecuciones de procesos

FCI rst Served)

El proceso B a llegado al final de su ejecución y empezara a ejecutarse el siguiente proceso en llegar (Proceso C 2t).

Tarea	H_n	t
A	~~0~~	~~3~~
B	1	5
C	3	2
D	9	5
E	12	5

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
A	X	X	X	F																	
B		E	E	X	X	X	X	X													
C				E	E	E	E	E													
D																					
E																					

Creamos la tabla de ejecuciones de procesos

FCI rst Served)

El proceso B a llegado al final de su ejecución y empezara a ejecutarse el siguiente proceso en llegar (Proceso C 2t).

Tarea	H_n	t
A	~~0~~	~~3~~
B	~~1~~	~~5~~
C	3	2
D	9	5
E	12	5

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
A	X	X	X	F																	
B		E	E	X	X	X	X	X	F												
C				E	E	E	E	E	X												
D																					
E																					

Creamos la tabla de ejecuciones de procesos

FC rst Served)

El proceso D empieza a ejecutarse, aunque este se pondrá en espera, ya que el proceso C aun no a terminado su ejecución.

Tarea	H_n	t
A	~~0~~	~~3~~
B	~~1~~	~~5~~
C	3	2
D	9	5
E	12	5

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
A	X	X	X	F																	
B		E	E	X	X	X	X	X	F												
C				E	E	E	E	E	X	X											
D																					
E																					

Creamos la tabla de ejecuciones de procesos

FC st Served)

El proceso C a llegado al final de su ejecución y empezara a ejecutarse el siguiente proceso en llegar (Proceso D 5t).

Tarea	H_n	t
A	~~0~~	~~3~~
B	~~1~~	~~5~~
C	3	2
D	9	5
E	12	5

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
A	X	X	X	F																	
B		E	E	X	X	X	X	X	F												
C				E	E	E	E	E	X	X											
D										E											
E																					

Creamos la tabla de ejecuciones de procesos

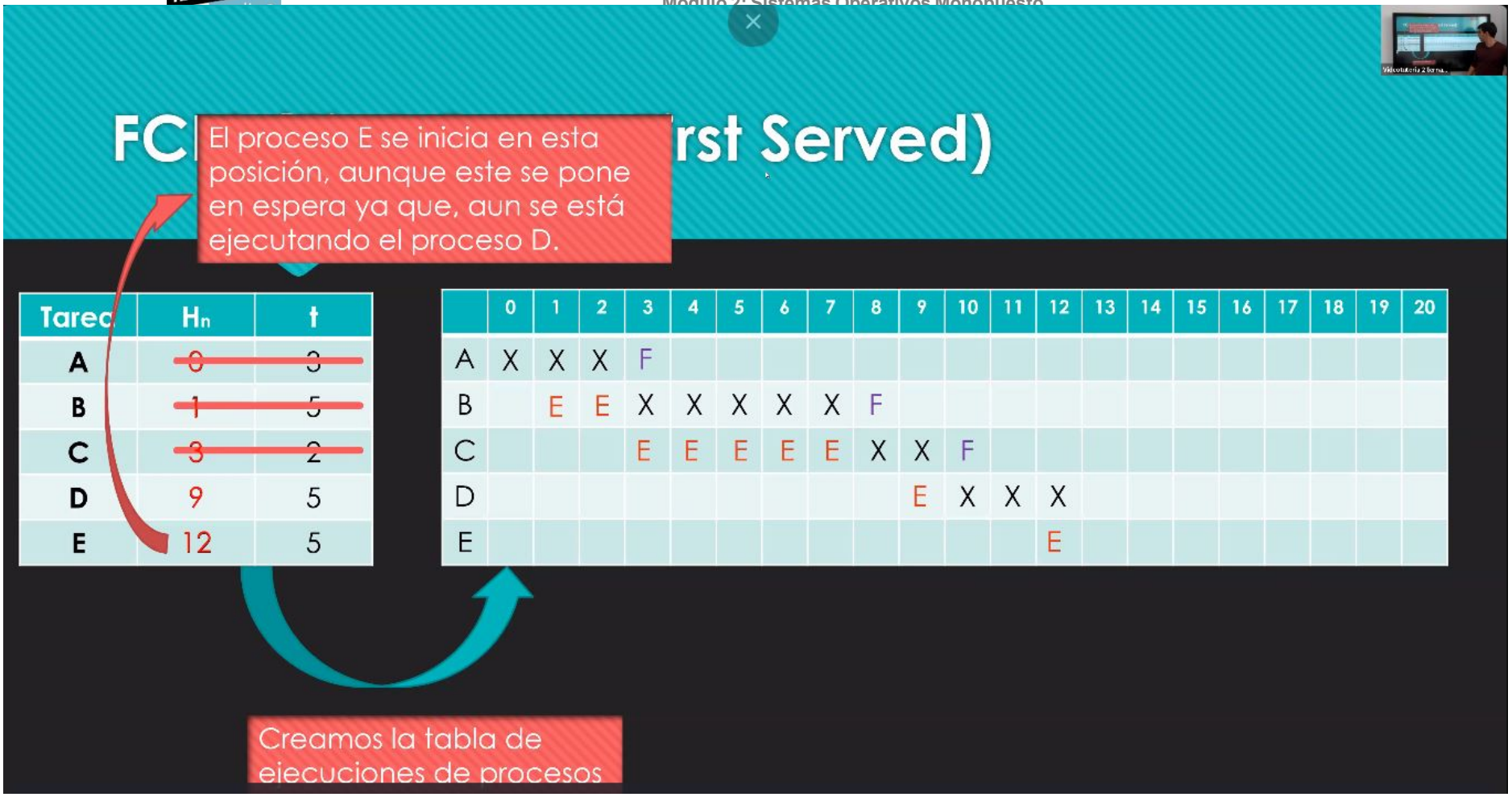
FC st Served)

El proceso C a llegado al final de su ejecución y empezara a ejecutarse el siguiente proceso en llegar (Proceso D 5t).

Tarea	H_n	t
A	~~0~~	~~3~~
B	~~1~~	~~5~~
C	~~3~~	~~2~~
D	9	5
E	12	5

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
A	X	X	X	F																	
B		E	E	X	X	X	X	X	F												
C				E	E	E	E	E	X	X	F										
D										E	X										
E																					

Creamos la tabla de ejecuciones de procesos



Tarea	H_n	t
A	~~0~~	~~3~~
B	~~1~~	~~5~~
C	~~3~~	~~2~~
D	9	5
E	12	5

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
A	X	X	X	F																	
B		E	E	X	X	X	X	X	F												
C				E	E	E	E	E	X	X	F										
D										E	X	X	X								
E													E								

FCFS (First Come First Served)

En este punto el proceso D llega a su fin y se empieza a ejecutar el proceso E y último.

Tarea	H_n	t
A	~~0~~	~~3~~
B	~~1~~	~~5~~
C	~~3~~	~~2~~
D	~~9~~	~~5~~
E	12	5

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
A	X	X	X	F																	
B		E	E	X	X	X	X	X	F												
C				E	E	E	E	E	X	X	F										
D										E	X	X	X	X	X	F					
E													E	E	E						

Creamos la tabla de ejecuciones de procesos

FCFS (First Come First Served)

Finalizamos el ultimo proceso que estaba en ejecución.

Tarea	H_n	t
A	~~0~~	~~3~~
B	~~1~~	~~5~~
C	~~3~~	~~2~~
D	~~9~~	~~5~~
E	~~12~~	~~5~~

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
A	X	X	X	F																	
B		E	E	X	X	X	X	X	F												
C				E	E	E	E	E	X	X	F										
D										E	X	X	X	X	X	F					
E													E	E	E	X	X	X	X	X	F

Creamos la tabla de ejecuciones de procesos

FCFS (First Come First Served)

Tarea	Hn	t	Hc	Ht	T	M	P
A	0	3					
B	1	5					
C	3	2					
D	9	5					
E	12	5					

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
A	X	X	X	F																	
B		E	E	X	X	X	X	X	F												
C				E	E	E	E	E	X	X	F										
D										E	X	X	X	X	X	F					

FCFS (First Come First Served)

Esta columna indica el tiempo en el cual se empieza a ejecutar el proceso.

Tarea	H_n	t	H_c	H_t	T	M	P
A	0	3					
B	1	5					
C	3	2					
D	9	5					
E	12	5					

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
A	X	X	X	F																	
B		E	E	X	X	X	X	X	F												
C				E	E	E	E	E	X	X	F										
D										E	X	X	X	X	X	F					

FCFS (First Come First Served)

Esta columna indica el tiempo en el cual se empieza a ejecutar el proceso.

Tarea	Hn	t	Hc	Ht	T	M	P
A	0	3	0				
B	1	5	3				
C	3	2	8				
D	9	5	10				
E	12	5	15				

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
A	X	X	X	F																	
B		E	E	X	X	X	X	X	F												
C				E	E	E	E	E	X	X	F										
D										E	X	X	X	X	X	F					

FCFS (First Come First Served)

Esta columna indica el tiempo en el cual se ha terminado de ejecutar el proceso (El primer tiempo que no está activo)

Tarea	Hn	t	Hc	Ht	T	M	P
A	0	3	0				
B	1	5	3				
C	3	2	8				
D	9	5	10				
E	12	5	15				

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
A	X	X	X	F																	
B		E	E	X	X	X	X	X	F												
C				E	E	E	E	E	X	X	F										
D										E	X	X	X	X	X	F					

FCFS (First Come First Served)

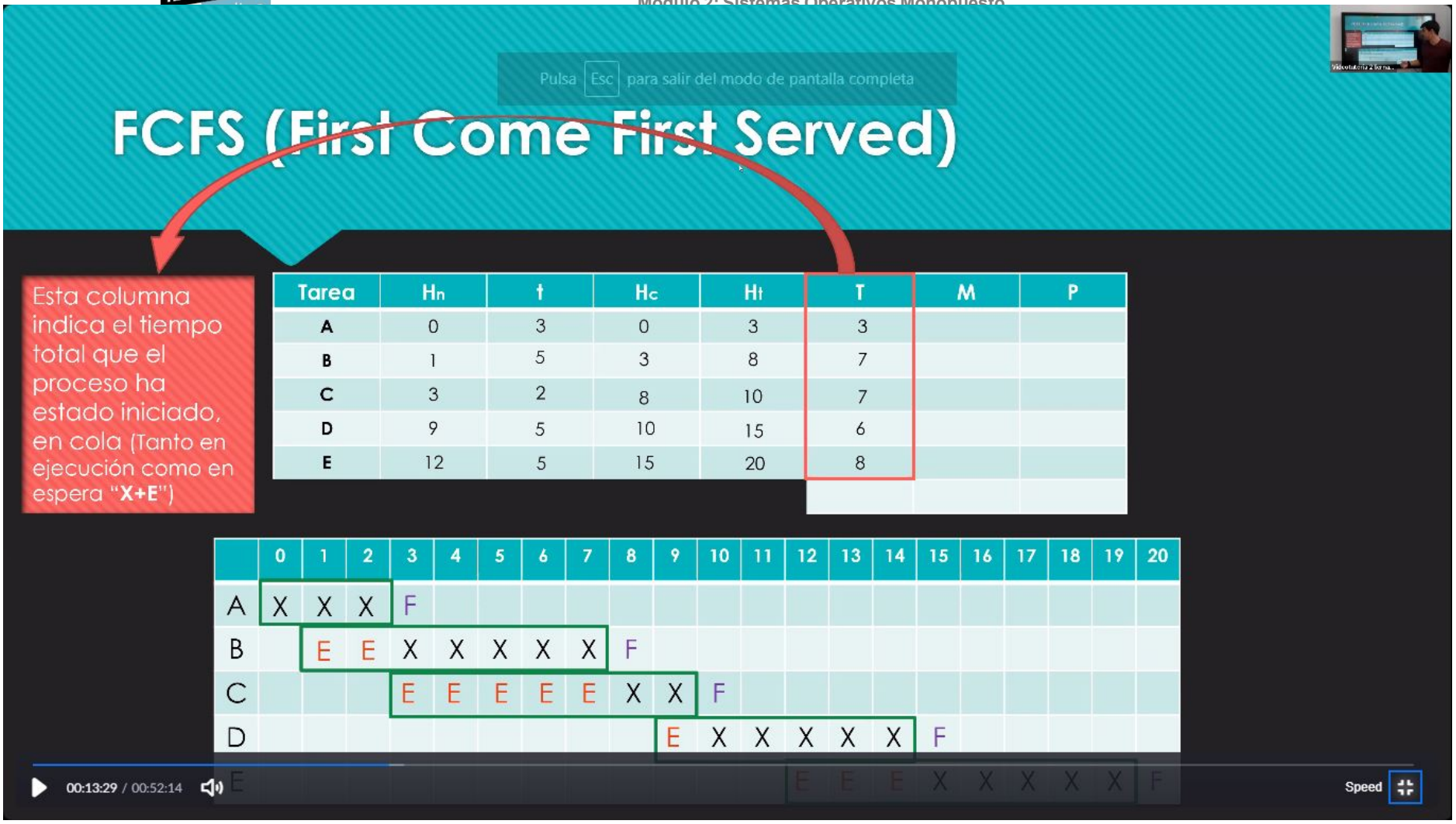
Esta columna indica el tiempo en el cual se ha terminado de ejecutar el proceso (El primer tiempo que no está activo)

Tarea	Hn	t	Hc	Ht	T	M	P
A	0	3	0	3			
B	1	5	3	8			
C	3	2	8	10			
D	9	5	10	15			
E	12	5	15	20			

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
A	X	X	X	F																	
B		E	E	X	X	X	X	X	F												
C				E	E	E	E	E	X	X	F										
D										E	X	X	X	X	X	F					
E													E	E	E	X	X	X	X	X	F

00:12:30 / 00:52:14

Speed


Pulsa Esc para salir del modo de pantalla completa
FCFS (First Come First Served)
Esta columna indica el tiempo total que el proceso ha estado iniciado, en cola (Tanto en ejecución como en espera "X+E")
Tarea | Hn | t | Hc | Ht | T | M | P
A | 0 | 3 | 0 | 3 | 3
B | 1 | 5 | 3 | 8 | 7
C | 3 | 2 | 8 | 10 | 7
D | 9 | 5 | 10 | 15 | 6
E | 12 | 5 | 15 | 20 | 8
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
A X X X F
B E E X X X X X F
C E E E E E X X F
D E X X X X X F
E E E E X X X X X F
00:13:29 / 00:52:14
Speed

FCFS (First Come First Served)

Esta columna indica el numero de esperas de cada tarea.

Tarea	Hn	t	Hc	Ht	T	M	P
A	0	3	0	3	3	0	
B	1	5	3	8	7	2	
C	3	2	8	10	7	5	
D	9	5	10	15	6	1	
E	12	5	15	20	8	3	

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
A	X	X	X	F																	
B		E	E	X	X	X	X	X	F												
C				E	E	E	E	E	X	X	F										
D										E	X	X	X	X	X	F					
E													E	E	E	X	X	X	X	X	F

00:14:10 / 00:52:14

Speed

FCFS (First Come First Served)

Esta columna indica el índice de penalización que ha tenido cada tarea, este se calcula mediante el siguiente cálculo **(T / t)**

Tarea	Hn	t	Hc	Ht	T	M	P
A	0	3	0	3	3	0	1
B	1	5	3	8	7	2	1,4
C	3	2	8	10	7	5	3,5
D	9	5	10	15	6	1	1,2
E	12	5	15	20	8	3	1,6

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
A	X	X	X	F																	
B		E	E	X	X	X	X	X	F												
C				E	E	E	E	E	X	X	F										
D										E	X	X	X	X	X	F					
E													E	E	E	X	X	X	X	X	F

00:15:07 / 00:52:14

Speed

FCFS (First Come First Served)

Estos valores serán las medias de cada columna a la cual hacen referencia.

Tarea	Hn	t	Hc	Ht	T	M	P
A	0	3	0	3	3	0	1
B	1	5	3	8	7	2	1,4
C	3	2	8	10	7	5	3,5
D	9	5	10	15	6	1	1,2
E	12	5	15	20	8	3	1,6

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
A	X	X	X	F																	
B		E	E	X	X	X	X	X	F												
C				E	E	E	E	E	X	X	F										
D										E	X	X	X	X	X	F					

FCFS (First Come First Served)

Tarea	Hn	t	Hc	Ht	T	M	P
A	0	3	0	3	3	0	1
B	1	5	3	8	7	2	1,4
C	3	2	8	10	7	5	3,5
D	9	5	10	15	6	1	1,2
E	12	5	15	20	8	3	1,6
					6,2		

Para realizar el cálculo de la media realizaremos la siguiente operación: **(3+7+7+6+8)/5**

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
A	X	X	X	F																	
B		E	E	X	X	X	X	X	F												
C				E	E	E	E	E	X	X	F										
D										E	X	X	X	X	X	F					

TAREA	Hn	t
A	0	3
B	1	5
C	3	2
D	9	5

Columna1	0	1	2	3	4	5
A	X	X	X	F		
B		E	E	X	X	X
C				E	E	E
D						
E						

TAREA	Hn	t	Hc	Ht	T	M	P	P2 con decimales
A	0	3	0	3	3	0	1	1
B	1	5	3	8	7	2	1	1,4
C	3	2	8	10	7	5	3	3,5
D	9	5	10	15	6	1	1	1,2
E	12	5	15	20	8	3	1	1,6

Valor PROMEDIO de cada columna	6,2	2,2	1,4	1,74

Hc	Tiempo en el cual se empieza a ejecutar el proceso

(Posición de la tabla en la cual se va a empezar a ejecutar el Proceso)

Ht	Tiempo en el cual se ha terminado de ejecutar el proceso

(Primer tiempo que no está activo). a la F (Fin de proceso) de cada PROCESO

T	Tiempo TOTAL en el que el proceso ha estado iniciado, en cola

(Tanto en ejecución como en espera

M	Esta columna indica el número de esperas de cada tarea

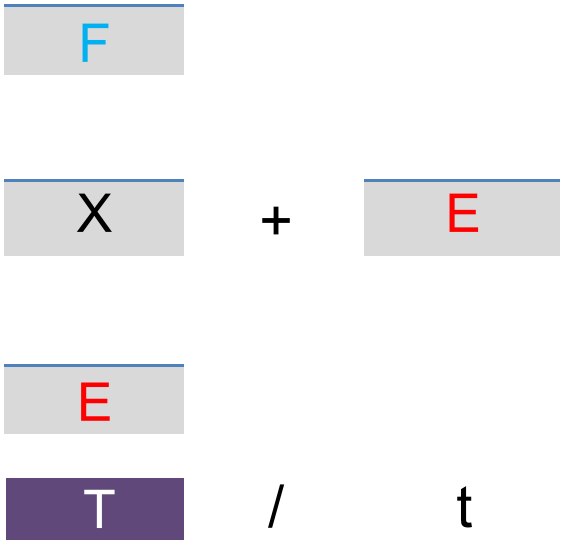
Son el equivalente a las E en la tabla que hemos rellenado arriba

P	Índice de penalización que ha tenido cada tarea

Se calcula con el siguiente cálculo

Valor PROMEDIO de cada columna	Sumamos los valores de cada columna / cantidad de valores

6	7	8	9	10	11	12	13	14	15	16
X	X	F								
E	E	X	X	F						
			E	X	X	X	X	X	F	
						E	E	E	X	X


F
X
+
E
E
T
/
t

17	18	19	20
X	X	X	F

Metodo2. > SJN (*Shortested Job Next*).

Para su ejecución, selecciona el trabajo que necesita menos tiempo de entre todos los que estén listos par a ejecutarse.

Ejemplo. Imaginemos que tenemos varios trabajos, con los tiempos de ejecución que se indican en la siguiente tabla.

Tarea	H_n	t
P	0	100
Q	0	10
R	0	1

Si la siguiente tarea a ejecutar es la más corta, tendríamos:

Tarea	H_n	t	H_c	H_t	T	M	P

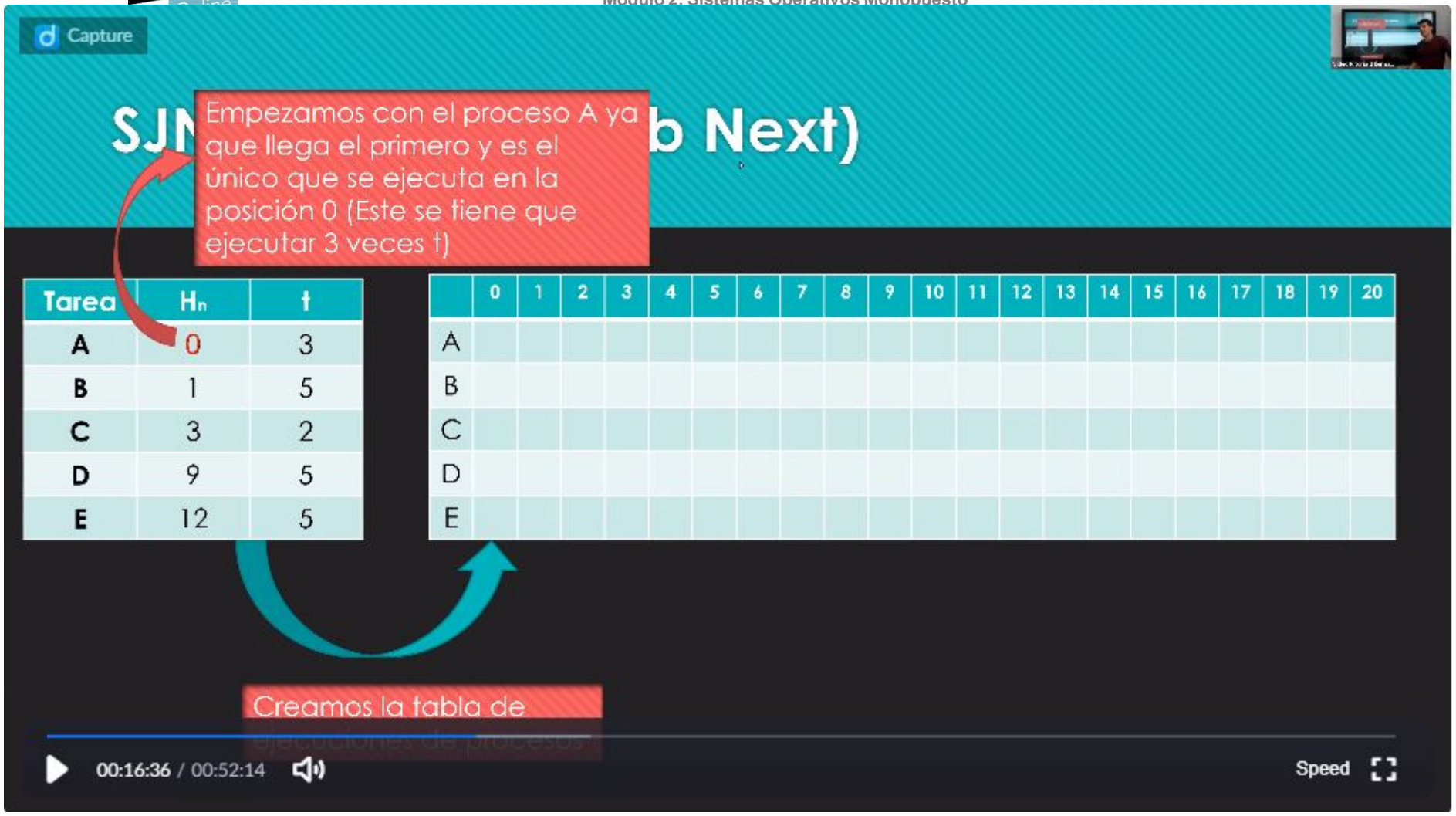
P	0	100	11	111	111	11	1,11
Q	0	10	1	11	11	1	1,10
R	0	1	0	1	1	0	1
Valores medios					41,0	4,0	1,07

Como se aprecia, el valor medio del tiempo de terminación mejora considerablemente al adelantar los procesos cortos, aunque en perjuicio de los largos. Además, se reducen los valores medios, aunque el propietario del trabajo largo no esté de acuerdo.

El problema principal de este procedimiento es que necesita conocer con antelación el tiempo de ejecución de cada proceso, lo que no es posible en muchas ocasiones.

Existen distintas **soluciones** a las desventajas que presentan estos métodos. El usuario debe incluir una **estimación del tiempo** de ejecución. Pero surge otro problema, pues también tendrán que poner como tiempo estimado el menor valor que se permita para que sus trabajos salgan favorecidos.

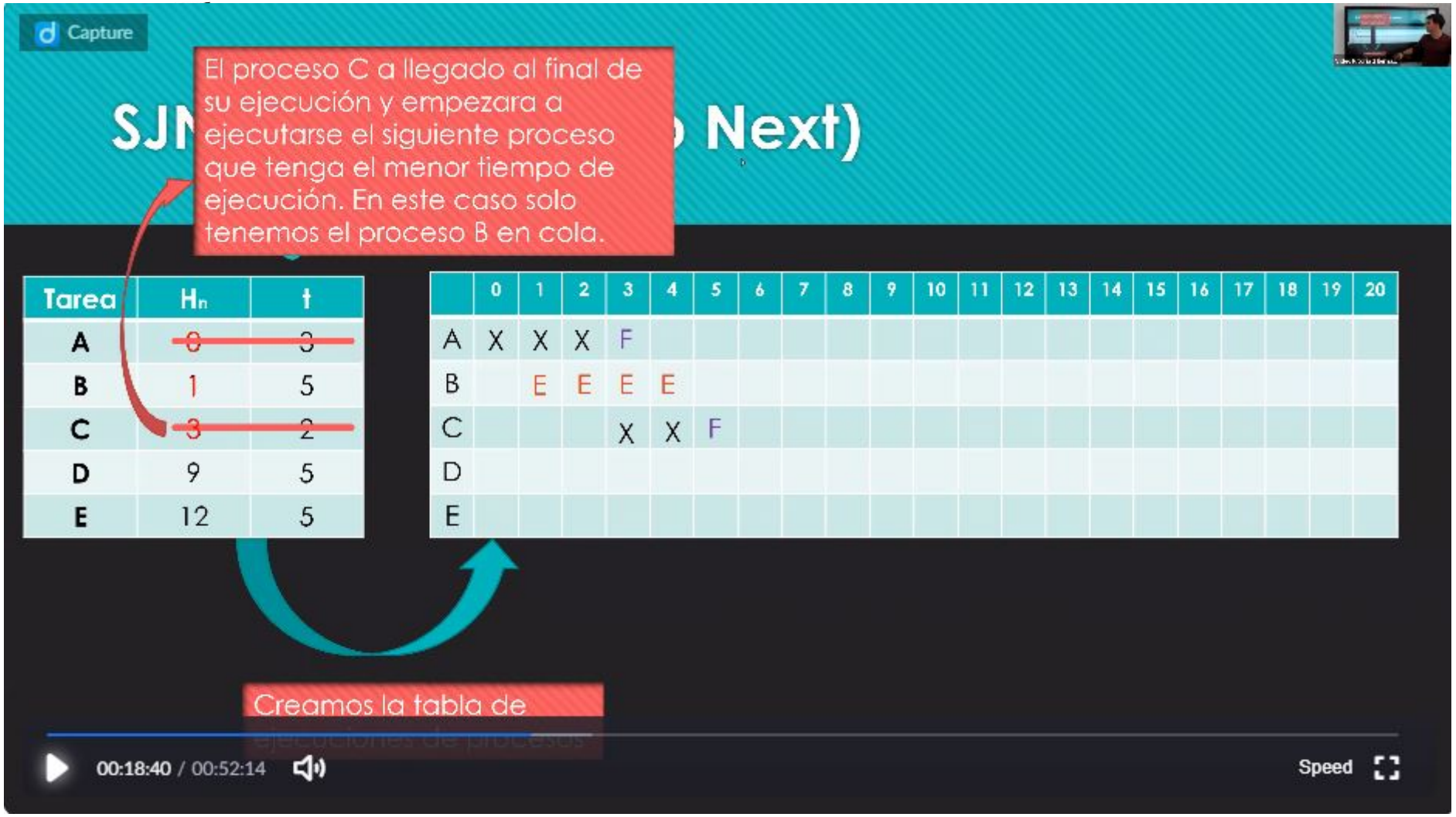
El sistema operativo calcula las estimaciones mediante aproximaciones sucesivas en las diversas ejecuciones. Se empieza con una estimación inicial del tiempo y se ejecuta el trabajo, lo que permite obtener una evaluación certera del tiempo.

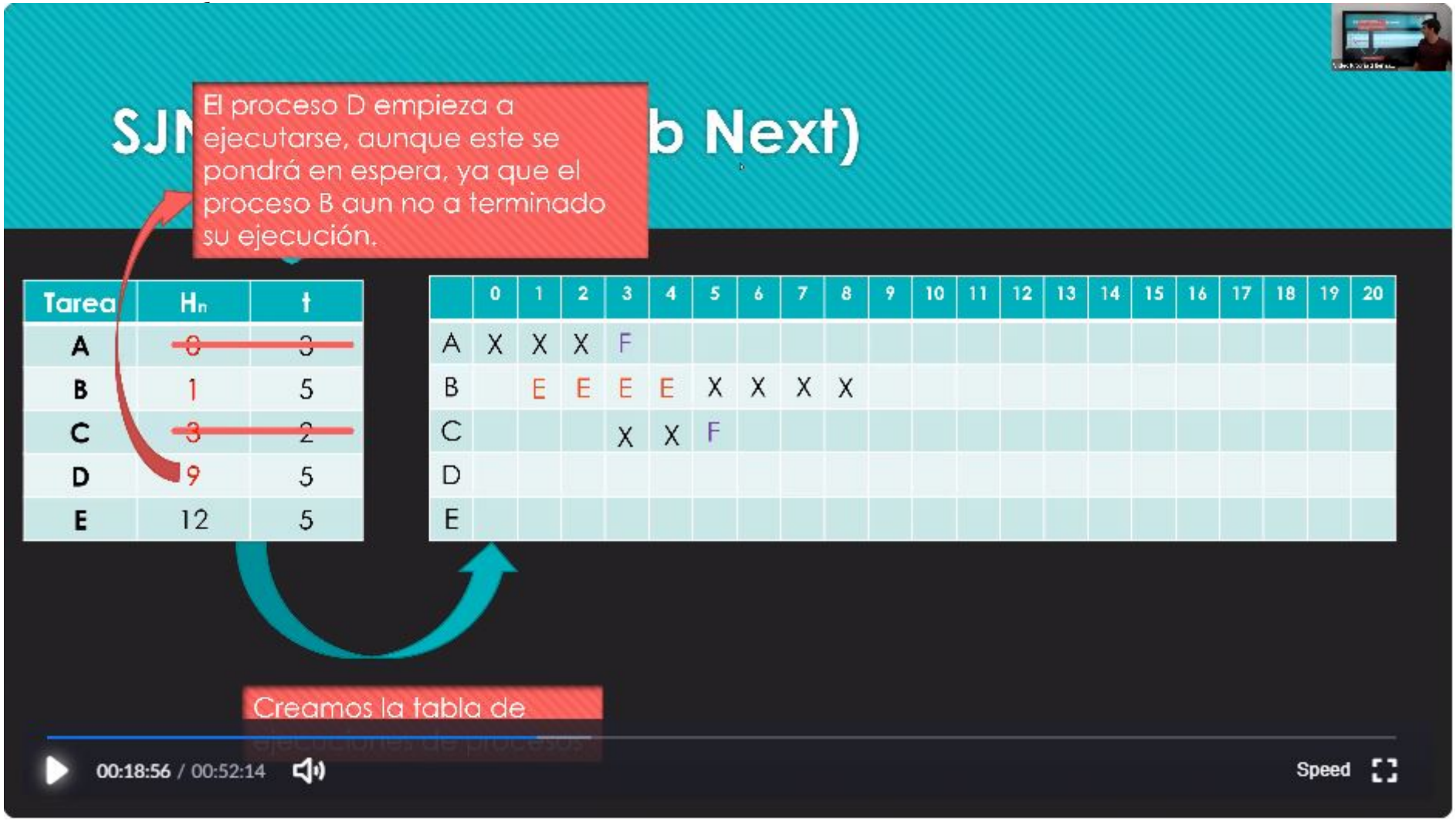


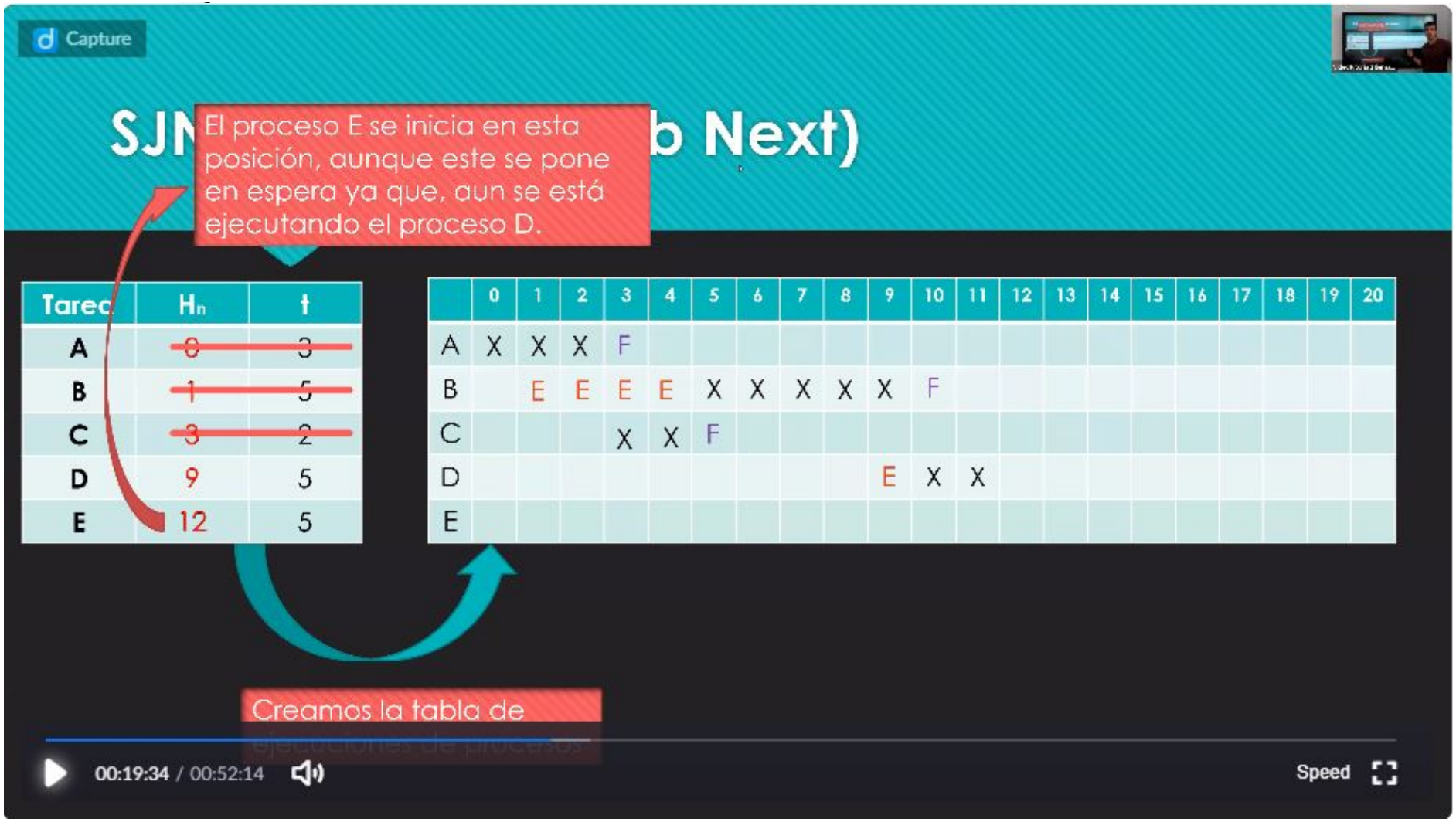
Tarea	H_n	t
A	0	3
B	1	5
C	3	2
D	9	5
E	12	5

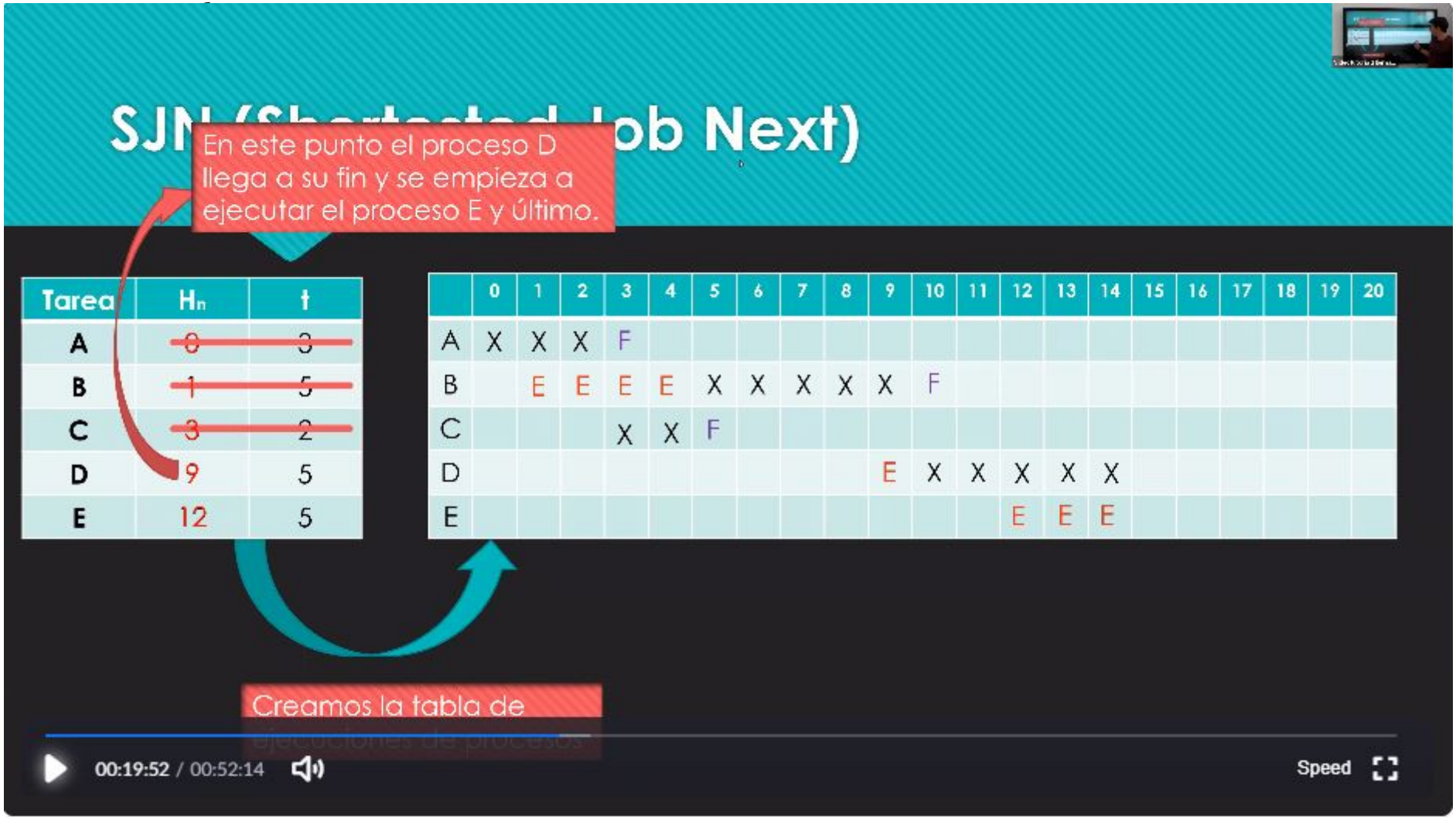
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
A																					
B																					
C																					
D																					
E																					


Capture
En esta posición llega el proceso C con 2 tiempos ejecución y también tenemos el proceso B en espera con 5 tiempos de ejecución en este caso se ejecutara el proceso C ya que su tiempo es menor y el proceso B seguirá en espera.
SJN
Next)
Tarea
Hn
t
A
8
3
B
1
5
C
3
2
D
9
5
E
12
5
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
A X X X F
B E E
C
D
E
Creamos la tabla de
00:17:22 / 00:52:14
Speed


Capture
SJN
Next)
El proceso C a llegado al final de su ejecución y empezara a ejecutarse el siguiente proceso que tenga el menor tiempo de ejecución. En este caso solo tenemos el proceso B en cola.
Tarea | Hn | t
A | 0 | 3
B | 1 | 5
C | 3 | 2
D | 9 | 5
E | 12 | 5
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
A X X X F
B E E E E
C X X F
D
E
Creamos la tabla de
00:18:40 / 00:52:14
Speed


SJN
b Next)
El proceso D empieza a ejecutarse, aunque este se pondrá en espera, ya que el proceso B aun no a terminado su ejecución.
Tarea | Hn | t
A | 8 | 3
B | 1 | 5
C | 3 | 2
D | 9 | 5
E | 12 | 5
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
A X X X F
B E E E E X X X X
C X X F
D
E
Creamos la tabla de
00:18:56 / 00:52:14
Speed

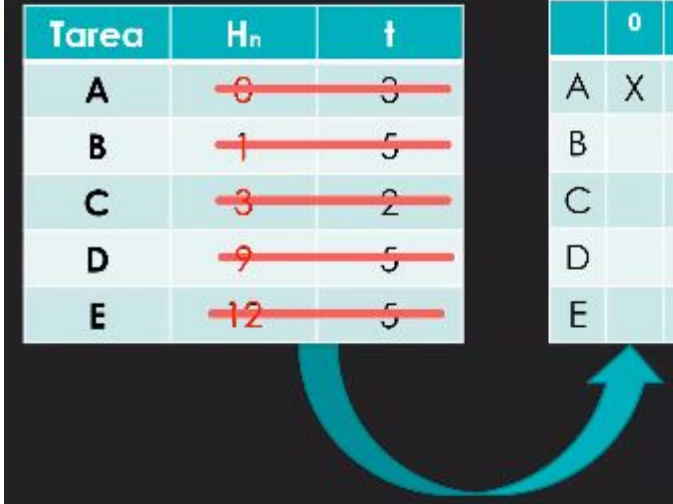

Capture
SJN
b Next)
El proceso E se inicia en esta posición, aunque este se pone en espera ya que, aun se está ejecutando el proceso D.
Tarea
Hn
t
A 8 3
B 1 5
C 3 2
D 9 5
E 12 5
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
A X X X F
B E E E E X X X X X F
C X X F
D E X X
E
Creamos la tabla de
00:19:34 / 00:52:14
Speed


SJN (Shortest Job Next)
En este punto el proceso D llega a su fin y se empieza a ejecutar el proceso E y último.
Tarea | Hn | t
A | 0 | 3
B | 1 | 5
C | 3 | 2
D | 9 | 5
E | 12 | 5
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
A X X X F
B E E E E X X X X X F
C X X F
D E X X X X X
E E E E
Creamos la tabla de
00:19:52 / 00:52:14
Speed

SJN (Shortested Job Next)

Tarea	H_n	t
A	~~0~~	~~3~~
B	~~1~~	~~5~~
C	~~3~~	~~2~~
D	~~9~~	~~5~~
E	~~12~~	~~5~~

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
A	X	X	X	F																	
B		E	E	E	E	X	X	X	X	X	F										
C				X	X	F															
D										E	X	X	X	X	X	F					
E													E	E	E	X	X	X	X	X	F



00:20:02 / 00:52:14

Speed

Método:

Apropiativos

Métodos **Apropiativos**

SRT

Metodo3. > **SRT:** **combina las ventajas de los procedimientos apropiativos con las del SJN. El procesador se adjunta al proceso más corto en cada momento, pero como el tiempo que falta para terminar solo disminuye para el proceso activo, solo puede ocurrir una apropiación si llega un proceso nuevo que necesite menos tiempo del que le falta al actual.**

Este método reduce las conmutaciones improductivas, las que no se deben a lectura/escritura. Además, tiende a mantener la cola de procesos preparados lo más corta posible, por lo que reduce el valor medio del tiempo de espera.

Ejemplo. Imaginemos que tenemos varios trabajos, con los tiempos de ejecución que se indican en la siguiente tabla.

Tarea	H_n	t
H	0	6
I	1	1
J	2	3

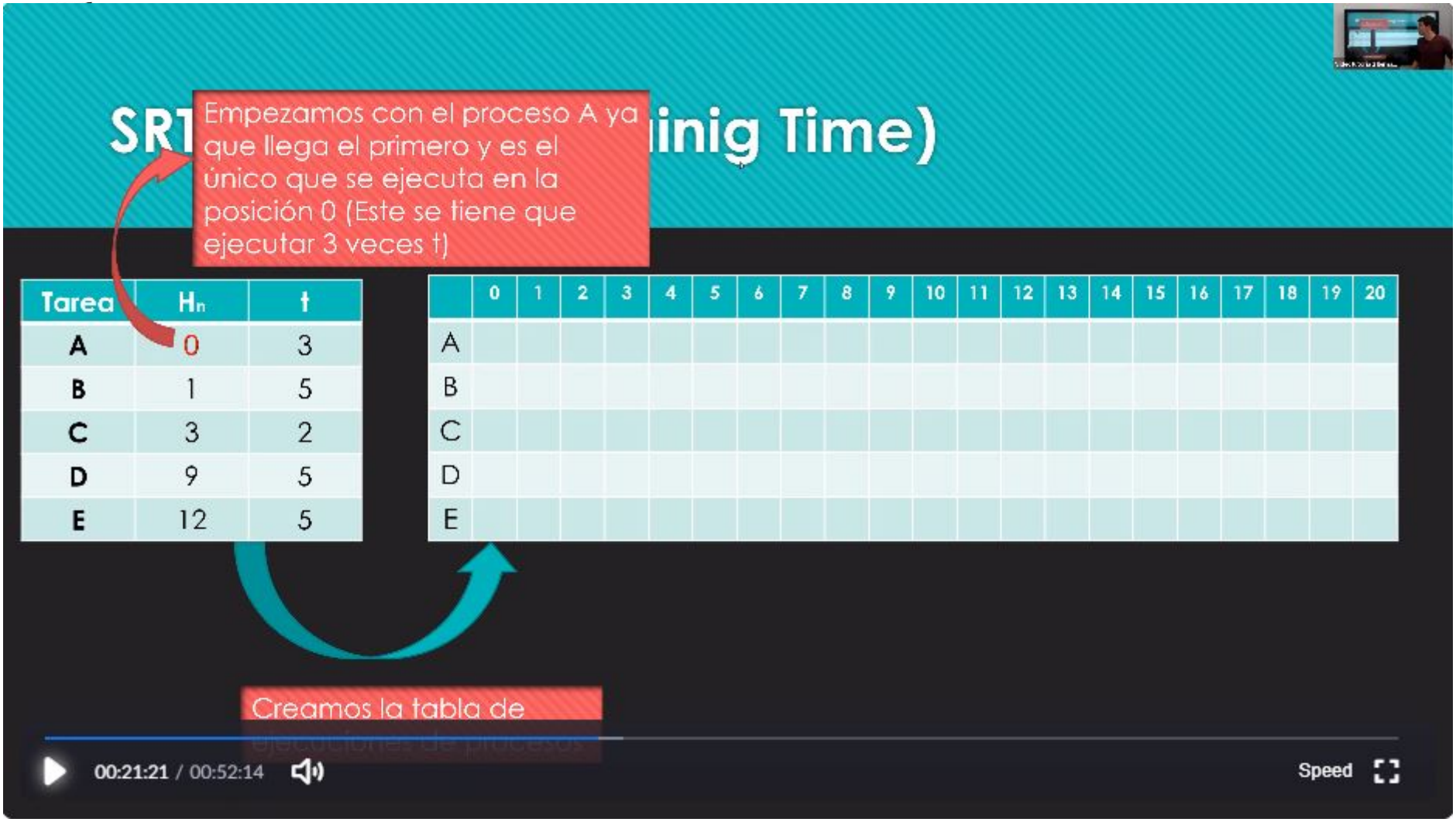
Si la siguiente tarea a ejecutar es la más corta con apropiación, tendríamos:

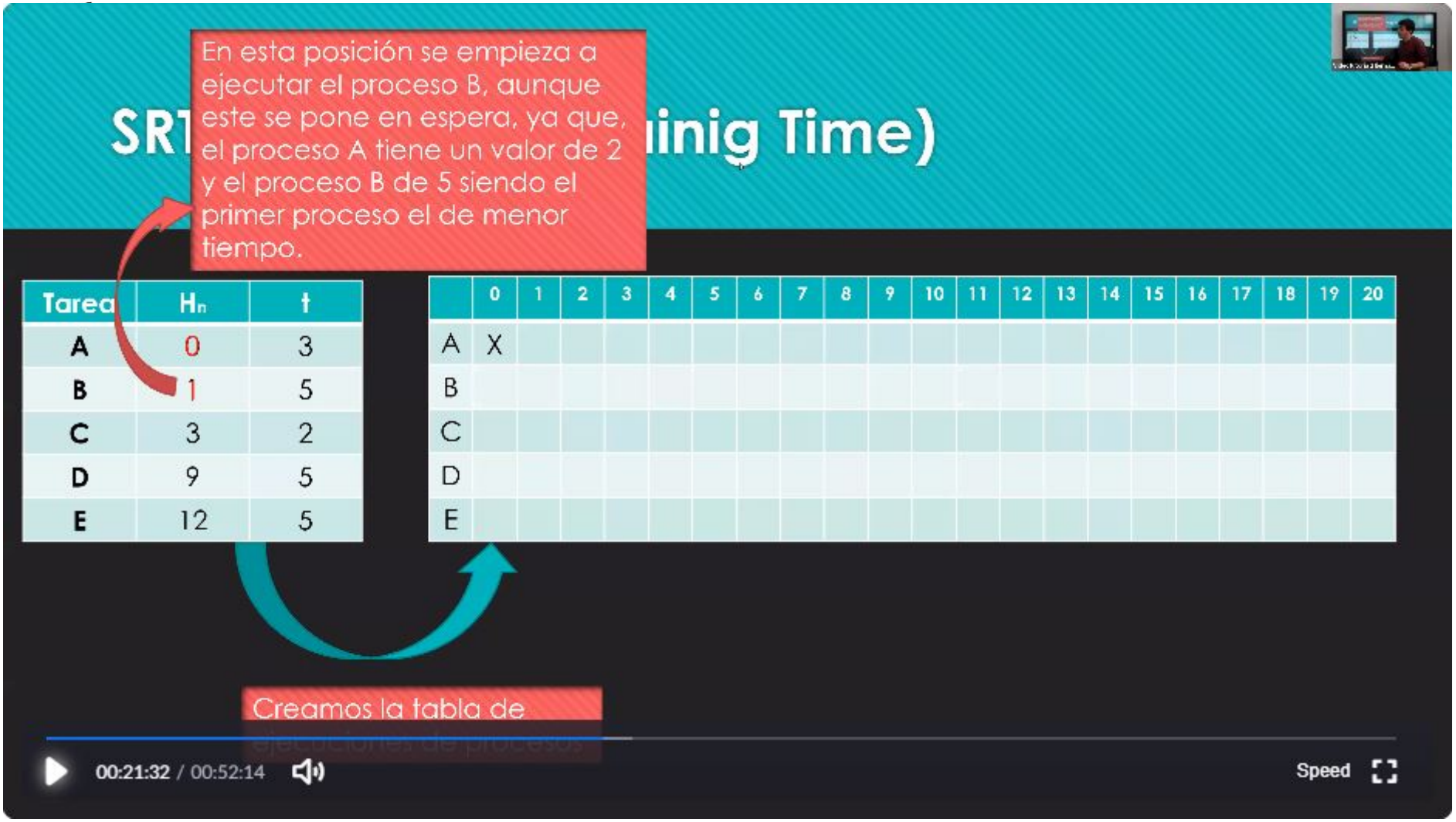
Tarea	H_n	t	H_c	H_t	T	M	P
H	0 5	6	0 5	0 10	10	4	1,67
I	1	1	1	2	1	0	1,0
J	2	3	2	5	3	0	1,0
Valores medios					4,67	1,33	1,22

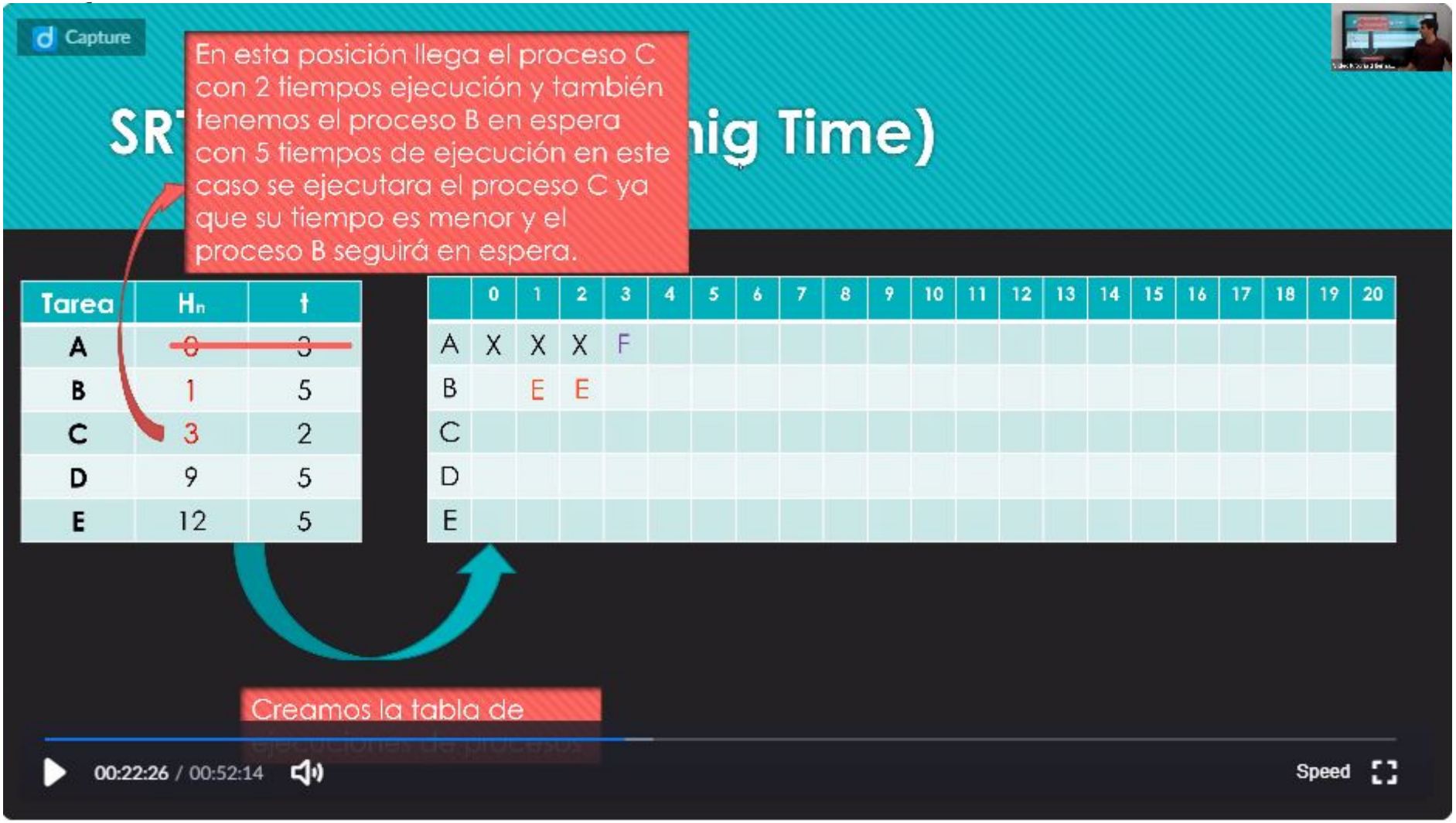
El índice de penalización y el tiempo perdido son menores que en el SJN, excepto para los procesos más largos, que se ven perjudicados.


Capture
SRT (Shortest Remainig Time)
00:21:00 / 00:52:14
Speed

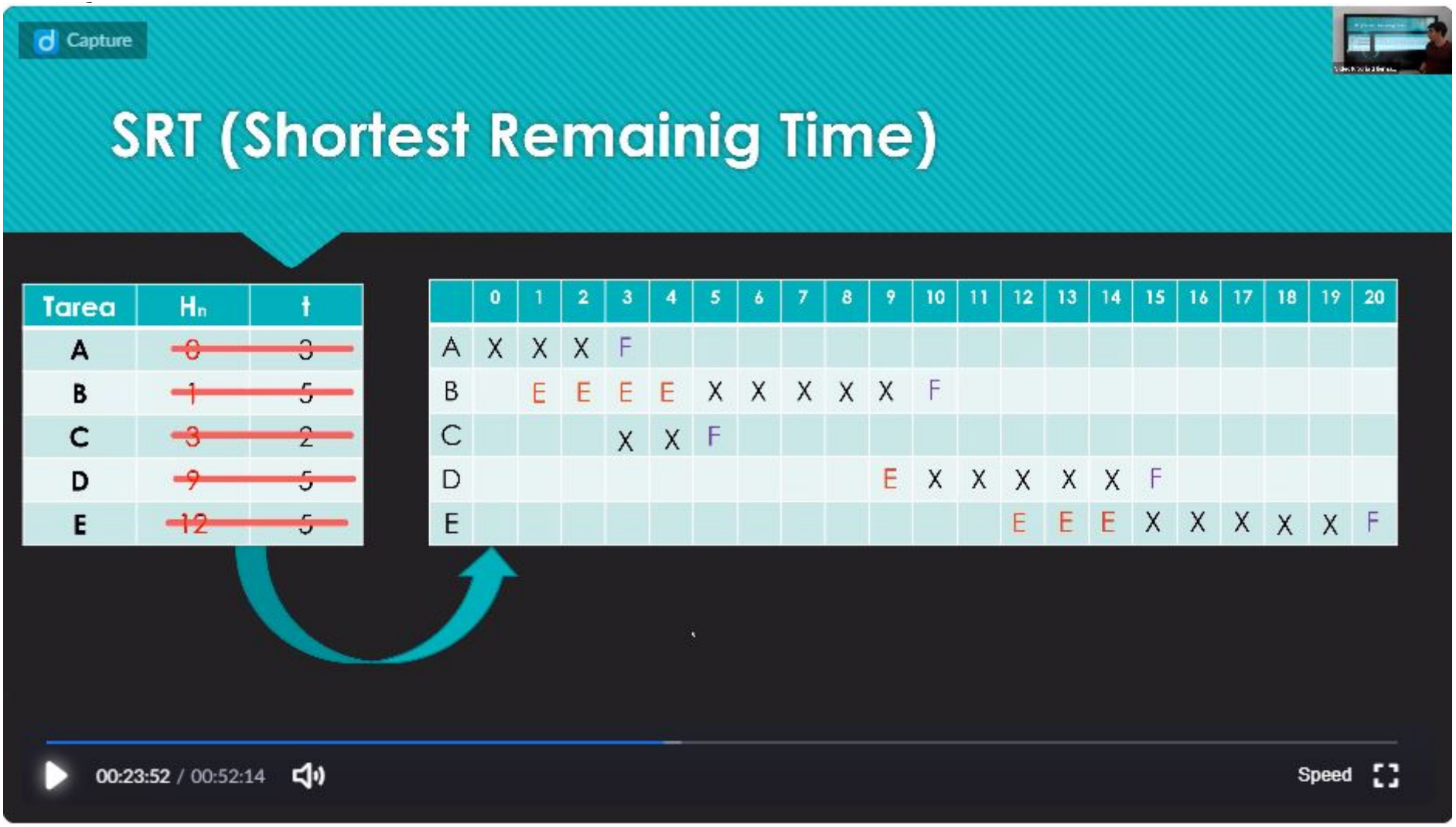

Capture
SRT (Shortest Remainig Time)
00:21:00 / 00:52:14
Speed


SRT
inig Time)
Empezamos con el proceso A ya que llega el primero y es el único que se ejecuta en la posición 0 (Este se tiene que ejecutar 3 veces t)
Tarea
Hn
t
A
0
3
B
1
5
C
3
2
D
9
5
E
12
5
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
A
B
C
D
E
Creamos la tabla de
00:21:21 / 00:52:14
Speed


SR
iinig Time)
En esta posición se empieza a ejecutar el proceso B, aunque este se pone en espera, ya que, el proceso A tiene un valor de 2 y el proceso B de 5 siendo el primer proceso el de menor tiempo.
Tarea | Hn | t
A | 0 | 3
B | 1 | 5
C | 3 | 2
D | 9 | 5
E | 12 | 5
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
A X
B
C
D
E
Creamos la tabla de
00:21:32 / 00:52:14
Speed


Capture
En esta posición llega el proceso C con 2 tiempos ejecución y también tenemos el proceso B en espera con 5 tiempos de ejecución en este caso se ejecutara el proceso C ya que su tiempo es menor y el proceso B seguirá en espera.
SR
ig Time)
Tarea
Hn
t
A
3
B
1
5
C
3
2
D
9
5
E
12
5
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
A X X X F
B E E
C
D
E
Creamos la tabla de
00:22:26 / 00:52:14
Speed


Capture
SR
ainig Time)
El proceso D empieza a ejecutarse, aunque este se pondrá en espera, ya que el valor del proceso B en este punto es 1 y el valor del proceso D es 5.
Tarea | Hn | t
A | 0 | 3
B | 1 | 5
C | 3 | 2
D | 9 | 5
E | 12 | 5
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
A X X X F
B E E E E X X X X
C X X F
D E
E
Creamos la tabla de
00:23:03 / 00:52:14
Speed


Capture
SRT (Shortest Remainig Time)
Tarea
Hn
t
A
8
3
B
1
5
C
3
2
D
9
5
E
12
5
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
A X X X F
B E E E E X X X X X F
C X X F
D E X X X X X F
E E E E X X X X X F
00:23:52 / 00:52:14
Speed

3.1. Gestión de entrada/salida

Como se ha comentado anteriormente, una de las funciones del sistema operativo es **controlar la comunicación y los recursos** asociados a los dispositivos de entrada y salida con la memoria principal. Se puede realizar de **tres formas** diferentes:

- **Por sondeo**: el gestor del dispositivo de entrada/salida realiza comprobaciones periódicas del estado del dispositivo.
- **Por interrupciones**: una interrupción es una señal que proviene del dispositivo de entrada/salida y que notifica al procesador que requiere atención.
- **Híbrida**: es una combinación de las anteriores. En general, se trata la gestión de entrada/salida mediante interrupciones, pero, en momentos de carga alta, se atienden en bloques cada cierto tiempo para evitar que un dispositivo sature al procesador. Se utiliza en los sistemas modernos.

4.1. Máquina real y máquina virtual

La **máquina real** es el sistema informático que tenemos físicamente, el cual tiene instalado un sistema operativo sobre el que trabaja el usuario. No obstante, a veces hay operaciones que no se deben realizar en él, ya sea porque la configuración no lo permite o porque estas puedan interferir con el resto de las aplicaciones del sistema. Por otro lado, una **máquina virtual** es el *software* que permite instalar nuevos sistemas operativos, como si se tratara de una nueva máquina real.

Se considera **anfitrión** al sistema operativo de la máquina real y **huésped** al de la máquina virtual. Además, esta última permite ejecutar programas que son independientes del sistema anfitrión.

Tipos de máquinas virtuales

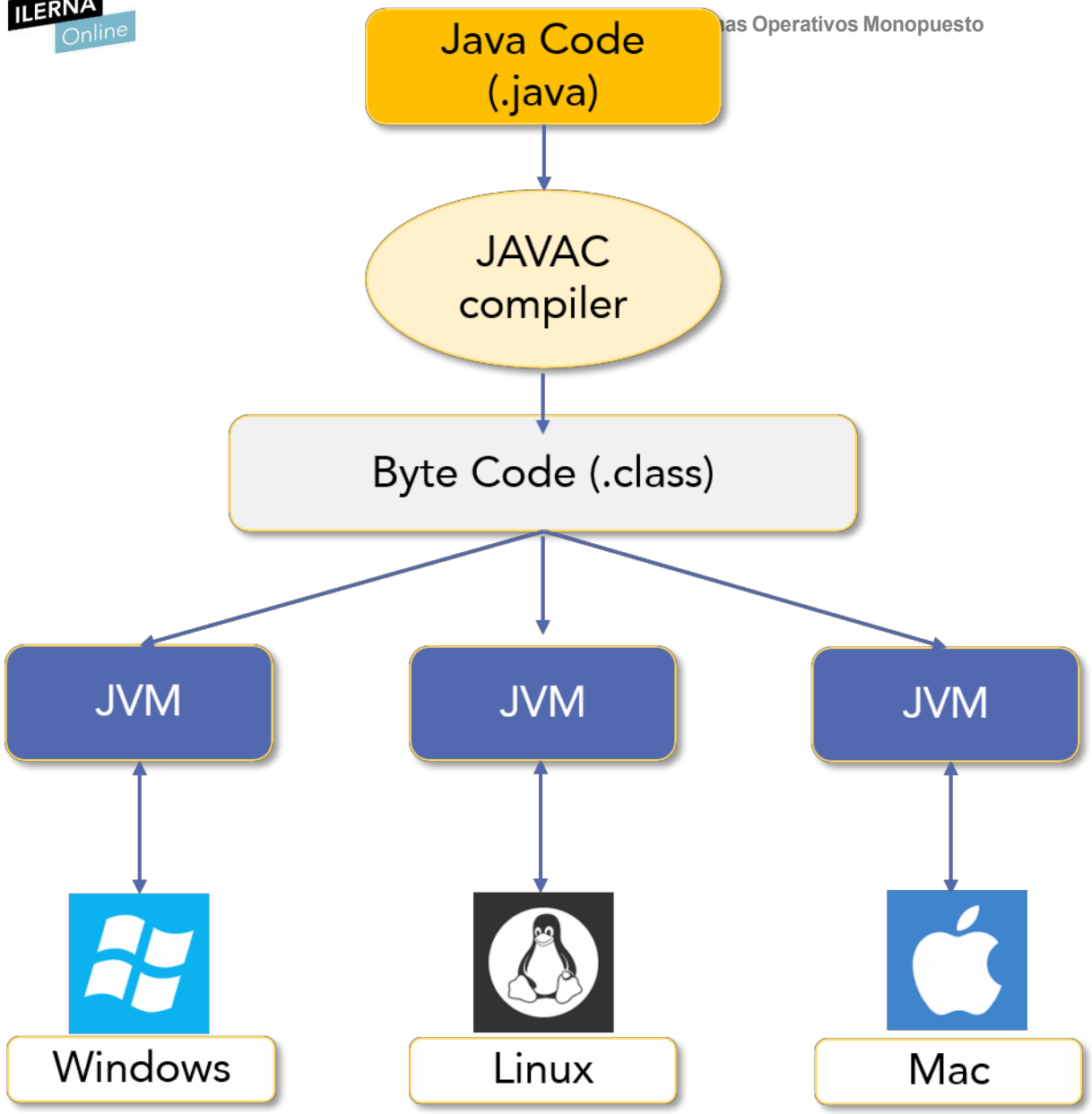
- **De sistema**: son herramientas que permiten instalar más de un sistema operativo en el mismo dispositivo físico sin crear distintas particiones físicas y ejecutarlos todos a la vez.

  A este tipo de máquinas virtuales pertenecen los *softwares* como VMWare o VirtualBox.

- **De proceso**: son herramientas que solo permiten virtualizar un único proceso. No se instala ningún sistema operativo, simplemente se aisla su aplicación. Este tipo de máquinas virtuales se crean cuando se lanza la aplicación y se cierran cuando finalizan. Durante su ejecución sí que requiere de los recursos de la máquina física para poder funcionar.

Este tipo de máquinas virtuales se utiliza para crear aplicaciones independientes del sistema operativo, como es el caso de JVM, la máquina virtual de Java.

Java es un **lenguaje de alto nivel** que se utiliza para crear aplicaciones multiplataforma.


Java Code
(.java)
JAVAC
compiler
Byte Code (.class)
JVM
JVM
JVM
Windows
Linux
Mac

4.2. Virtualización y máquina virtual

Sus ventajas son:

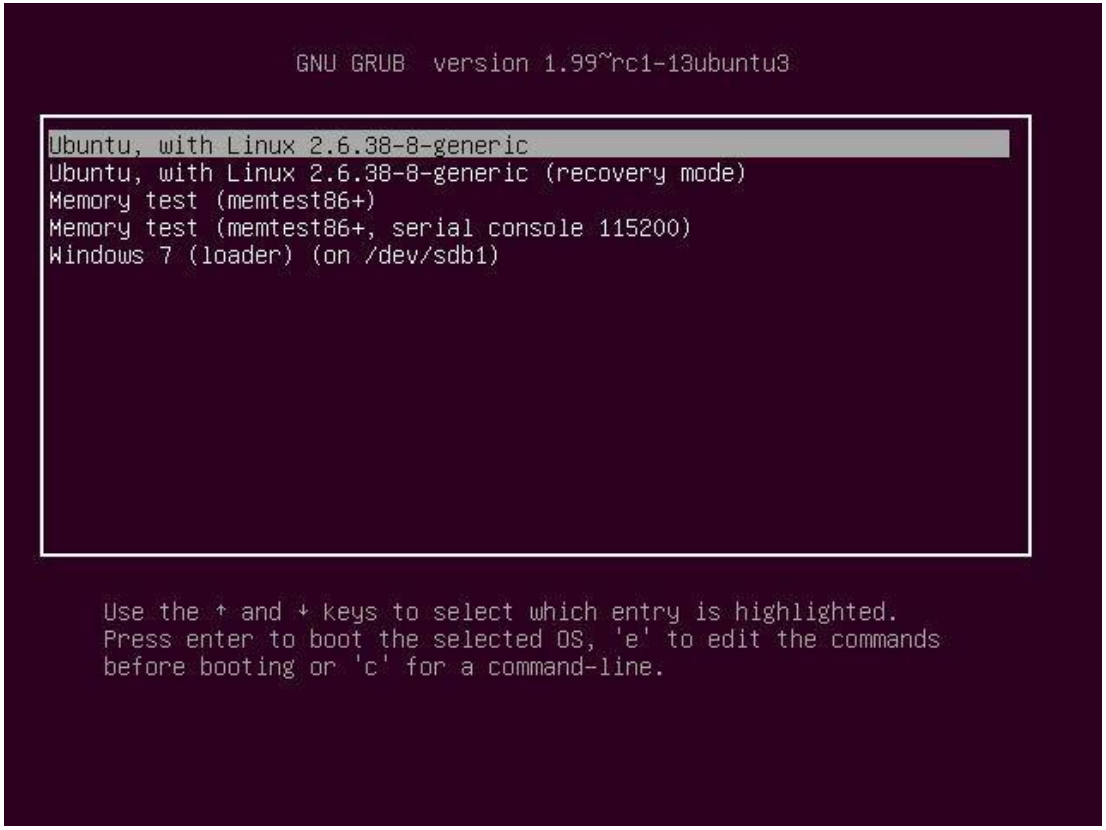
- **Aislamiento**: en una misma máquina real podemos tener varias máquinas virtuales independientes entre sí, al igual que lo son del sistema anfitrión. Esto significa que un fallo en una aplicación de una máquina virtual solo la afecta a ella.
- **Seguridad**: si entra un virus en la máquina virtual, solo se daña esta máquina, es decir, la máquina real no sufre daños.
- **Portabilidad**: la creación de una máquina virtual genera una carpeta en la máquina física que puede ser copiada a cualquier otro ordenador.
- Requieren **menos recursos *hardware*** porque los comparten con la máquina real.
- Los *softwares* virtualizadores permiten al usuario la opción ***Guardar estado***, por lo que encontramos la máquina virtual como la dejamos.
- Permite **crear sistemas con una cantidad de recursos fijos**.
- Podemos **compartir archivos y directorios** entre diferentes sistemas operativos.
- Sirven como **entornos de prueba**.

Los inconvenientes son:

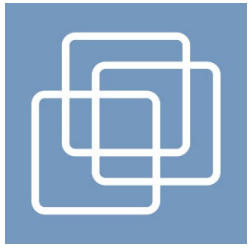
- **Ralentiza la ejecución de una aplicación**: es notable el tiempo de respuesta en la ejecución de la misma aplicación en una máquina virtual que en una real.
- **Ralentiza el sistema**: iniciar la máquina virtual significa que la memoria RAM asociada queda ocupada, aunque no la necesite en ese momento.

Existe otra manera de tener más de un sistema operativo en el mismo dispositivo físico, sin necesidad de crear una máquina virtual, pero presenta más inconvenientes. Este mecanismo se denomina **arranque dual**: cuando el usuario enciende el ordenador, el *boot* de arranque le permite elegir el sistema operativo en el que quiere trabajar. El arranque dual obliga a realizar

una partición real en el disco duro, por lo que se debe destinar un espacio físico a cada sistema operativo.



4.3. Software para la creación de máquinas virtuales

Software para la creación de máquinas virtuales					
				 Microsoft Hyper-V	
Virtual Box	**VMWare**	**Virtual PC**	**Parallels**	**Hyper-V**	**QEMU**

Software para la creación de máquinas virtuales

Virtual Box		VMWare		Virtual PC		Parallels		Hyper-V		QEMU	
	Windows		Windows		Windows	OS X	Mac OS	Windows Server	Windows Server 2008		
OS X	MAC OS X		GNU/Linux						Windows 8 a 10		
	GNU/Linux										
	OS/2										
solaris	Solaris/ OpenSolaris										

Software para la creación de máquinas virtuales

Virtual Box		VMWare		Virtual PC		Parallels		Hyper-V		QEMU	
MS-DOS		Windows		MS-DOS		Windows 7 a 10		Windows Server 2008		DOS	
Windows		GNU/Linux		Windows		Mac OS		Windows 7 a 10		Windows	
GNU/Linux		Versiones Linux				Linux				GNU/Linux	
	FreeBSD		Mint								BSD
	OPenBSD		CentOS								Solaris

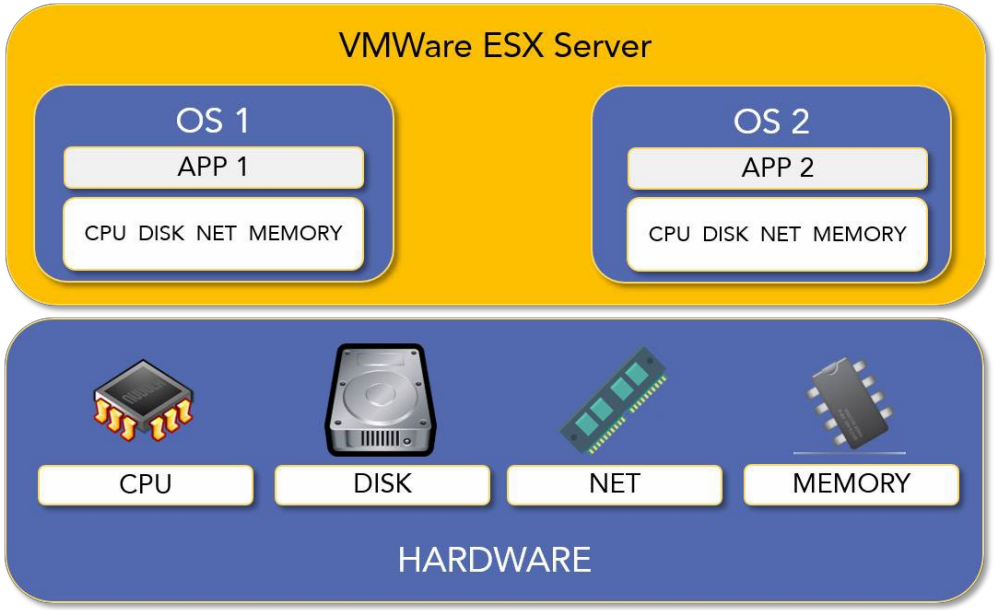
➢ Virtual Box

Pertenece a **Oracle**. Su versión actual es la 5.1, una de las más conocidas y de uso más extendido globalmente. Cuenta con licencias gratuitas (para uso personal) y privativas (Oracle VM Virtual Box).
Una de sus grandes ventajas es que ofrece una gran
cantidad de sistemas operativos sobre los que podemos instalar este *software*: GNU/Linux, MAC OS X, OS/2, Solaris/OpenSolaris y Windows. Una vez instalado, podemos **virtualizar**: GNU/Linux, FreeBSD, OPenBSD, diferentes vesiones de Windows y MS-DOS, entre otros.

➢ VMWare

Pertenece a **Dell**. Está disponible para arquitecturas de procesador x86, tanto de 32 como de 64 bits. Además, permite la virtualización de sistemas operativos cliente y de servidor. Junto con Virtual Box, es uno de los *softwares* de virtualización más usados.

Es posible instalarlo sobre Linux y Windows, y permite virtualizar Windows, Ubuntu, Red Hat Enterprise, Suse, Debian, Fedora, Mint, CentOS y Oracle Linux, entre otros. Al igual que Virtual Box, ofrece licencias de *software* tanto privativas como gratuitas, como es el caso de VMWare Workstation Player.



➢ Virtual PC

Es propiedad de **Microsoft**. Emula el *hardware* de un equipo en sistemas operativos Windows pero que no virtualiza el procesador en la máquina virtual.

Permite emular la totalidad de todas las distribuciones de los sistemas operativos Windows y MS-DOS, y su principal diferencia con el resto de *software* vistos es que no es recomendable si vamos a utilizar Linux, porque permite su instalación, pero la emulación es muy lenta.

➢ Parallels

Es un *software* de virtualización para el sistema operativo **Mac OS** con procesadores intel, propiedad de **Parallels**.

Puesto que Mac ya permite la instalación de Windows dentro de una partición del disco físico, es necesario el uso de BootCamp para elegir el sistema operativo al iniciar el ordenador.

Nos permite utilizar las aplicaciones de Windows en Mac OS, sin necesidad de tener que hacer uso del BootCamp. De esta forma, conseguimos virtualizar Windows dentro del Mac reutilizando toda la configuración de esa partición y creando una máquina virtual. Además, nos permite virtualizar otros sistemas operativos como: Mac OS, Windows (las versiones 7, 8.1 y 10) y Linux, entre otros.

Es un *software* de pago, pero se puede realizar una prueba de forma gratuita.

➢ Hyper-V

Es un *software* de virtualización de **Microsoft** para los procesadores de 64 bits. Se utiliza en sistemas operativos de los servidores como Windows Server 2008, lo que permite virtualizar hasta tres máquinas, una dentro de otra. Actualmente, también está disponible en sistemas operativos cliente (como, por ejemplo, en las versiones 8, 8.1 y 10).

Es gratuito porque está incluido dentro del sistema operativo de Microsoft.

➢ QEMU

Es un ***software* libre** con el que se pueden virtualizar la gran mayoría de sistemas operativos y que permite instalaciones tanto de 32 como de 64 bits. Además, es capaz de emular una máquina de forma completa, incluyendo el procesador y los periféricos. Carece de interfaz gráfica, aunque es posible utilizar un *software* que nos permite comunicarnos con QEMU si lo hacemos desde un sistema operativo Windows.

Los sistemas operativos que podemos virtualizar son Linux, Solaris, Microsoft Windows, DOS y BSD.

Instalación de VMWare

4.2. Instalación de VMWare.

4.3. Creación de Máquinas Virtuales para S.O. Libres y S.O. Propietarios.

4.4. Configuración y utilización de Máquinas Virtuales.

4.5. Tools.

4.6. Relación con el sistema operativo anfitrión.

4.7. Realización de pruebas de rendimiento del sistema.

4.8. La monitorización.

4.9. Comprobación del funcionamiento correcto de las instalaciones y configuraciones realizadas

Errores más comunes que se pueden encontrar son:

4.10. Interpretación Documentación Técnica.

4.2.Instalación de VMWare

Una vez se ha realizado la descarga, hay que ejecutar el archivo. Nos pedirá aceptar la licencia de uso para proceder a la instalación

En las siguientes pantallas encontraremos la configuración del *software*, como, por ejemplo, la comprobación automática de las actualizaciones.

4.3. Creación de Máquinas Virtuales para S.O. Libres y S.O. Propietarios

Una vez se ha instalado el *software*, se puede comenzar a crear máquinas virtuales. Para ello, hay que pulsar la opción

***Create a new virtual machine*.**

Existen diferentes opciones de creación, es decir, se puede crear la máquina virtual sin instalar el sistema operativo o indicando el sistema operativo que utilizaremos.
Aunque se indique que el sistema operativo se instalará más tarde,

Sí es necesario indicar de qué tipo será.

Por último, se realizará la configuración de la máquina virtual indicando características como, por ejemplo, la cantidad de RAM dedicada para ella o el espacio de almacenamiento del disco duro.

4.4.Configuración y utilización de Máquinas Virtuales

Acceder a la configuración de la máquina virtual que se ha creado

(esta opción se encuentra en *Edit virtual machine settings*).

Se pueden **editar los parámetros configurados en el apartado anterior en esta ventana**.

Si se ha instalado una máquina virtual para Windows 7 de 32 bits a la que hemos dedicado 1 GB de RAM, se podrá ejecutar y trabajar con ella, aunque si en algún momento se necesita aumentar el rendimiento, es aquí donde se debe configurar.

Además, en el apartado CD/DVD (SATA) hay que **indicar cómo se instalará el sistema operativo.**

Lo más cómodo es descargar una imagen (ISO) del mismo e indicar la ruta en la que se encuentra.
Hace algunos años, solo se podía instalar un sistema operativo desde un dispositivo óptico (como un CD o DVD). Actualmente, estos dispositivos se pueden copiar en un solo archivo con extensión .iso que realiza la instalación.

En la siguiente pestaña (*Options*) se puede modificar el nombre de la máquina y el directorio donde se guardará.

Esta carpeta es la que se puede copiar en otro ordenador

Si se busca este directorio en la máquina física, se pueden ver todos los ficheros que forman la máquina virtual.

Existe un archivo llamado igual que ella, con extensión *.vmx* (por ejemplo, *ubuntu64.vmx*), que es su acceso directo.

No obstante, si se abre con un editor de texto plano, como, por ejemplo, el Bloc de notas, es posible conocer su configuración.

Tools

Después de la instalación del sistema operativo, es recomendable instalar las herramientas que VMWare pone a nuestra disposición a través de la aplicación Tools.

Se instalan dentro de la máquina virtual, seleccionando con el botón derecho del ratón la opción *Install/Upgrade VMWare Tools*.

Estas herramientas tienen dos funciones básicas:

- Instalar los drivers necesarios para el uso de la máquina virtual, como,por ejemplo, los del ratón o los de red.

Mejorar la gestión de los recursos de la máquina virtual para que hayamás fluidez.

4.6. Relación con el sistema operativo anfitrión

A veces, puede resultar útil compartir información entre el sistema anfitrión y el invitado para acceder a determinados recursos (como, por ejemplo, memorias USB o discos físicos, entre otros).

Para poder hacer uso de esto, es necesario configurarlo previamente en el *software* de virtualización. Además, hay que tener en cuenta que no todos los programas permiten compartir recursos de forma directa (VMWare y Virtual Box sí lo permiten).

Una vez se inicia el *software* de máquina virtual, y antes de entrar en el sistema operativo, hay que realizar los siguientes pasos:

1. **Crear o seleccionar un directorio/carpeta a compartir.**
2. **Agregar la ruta de esta carpeta en la configuración del *software*.**
3. **Habilitar el modo** *Compartir carpetas*.

En Virtual Box es necesario un paso más, que es la instalación de *Guest additions*, **el cual implementa la configuración necesaria para poder utilizar cualquier recurso compartido de forma autómatica.**

La mayor ventaja que presenta es que se puede arrastrar un archivo entre los diferentes sistemas operativos para transferirlo entre el sistema operativo anfitrión y el invitado. Por tanto, se agiliza y facilita enormemente tanto la transferencia de archivos como el acceso a recursos del sistema anfitrión.

Por otro lado, como se ha indicado anteriormente, no solo se pueden compartir archivos de una máquina a otra, sino que también es posible compartir cualquier tipo de periféricos (como memorias USB, ratón, teclado, CD, DVD, etcétera). De esta forma, se podrá utilizar el mismo dispositivo indistintamente en cada uno de los sistemas operativos.

La instalación de *Guest Additions*, **en el caso de Virtual Box, provee además de una interfaz que facilita la compartición de**

4.7. Realización de pruebas de rendimiento del sistema

Una vez iniciada una máquina virtual, es importante saber si los recursos físicos que se han destinado a ella son adecuados o no, es decir, comprobar que su funcionamiento es correcto y que no perjudica al rendimiento del equipo.

La mejor forma de realizar estas comprobaciones es monitorizando **los recursos del sistema operativo anfitrión.**

Existen muchos programas dedicados a esta tarea, los cuales ofrecen los recursos utilizados por cada una de las máquinas virtuales en tiempo real. Algunos de estos recursos son la carga de la CPU, la memoria RAM, el disco duro o la GPU.

La monitorización se puede realizar haciendo uso de los *softwares* que ya están instalados en el sistema operativo. En Windows 7 se encuentra siguiendo la ruta:

Inicio ⬬ *Todos los programas* ⬬ *Accesorios* ⬬ *Herramientas del sistema* ⬬

Monitor de recursos

Para realizar una correcta asignación de recursos es recomendable verificar antes cuáles son los requisitos mínimos de cada uno de los sistemas operativos que se van a instalar. Si esta asignación no es la correcta, es posible que se pierdan datos del disco, se disminuya el rendimiento del equipo o, en el caso de la instalación de servidores, se bloqueen algunos de los servicios.

Algunos de los indicios que indican que la configuración de una máquina virtual no ha sido la adecuada son:

- **El inicio del sistema operativo invitado** tarda en arrancar**.**
- **Los programas abiertos en la máquina virtual** tardan en responder**.**
- **Los programas** tardan **mucho tiempo** en ejecutarse**.**

4.8. Comprobación del funcionamiento correcto de las instalaciones y configuraciones realizadas

El primer paso después de haber establecido la configuración de recursos asignados a una máquina virtual es "encenderla". Para ello, se escoge y selecciona la máquina a iniciar. Entonces, aparecerá una pantalla resumen con la configuración *hardware* de dicha máquina y, en la parte inferior, la opción de iniciarla (normalmente aparece un icono triangular de Play**).**

Tras realizar esta acción, se ejecutará la máquina y se iniciará el proceso normal de arranque del sistema operativo, el cual incluye la aparición de errores durante este proceso de inicio.

Los errores más comunes que se pueden encontrar son:

- ***Operating System not found*:** informa de que se ha producido un error al intentar cargar el sistema operativo. Esto significa que no se ha indicado correctamente la ruta de acceso al archivo con la imagen del mismo y que no se ha podido instalar.

- ***Unable to open kernel device*:** para solucionar este fallo, es necesario revisar el archivo de configuración de la máquina virtual que se encuentra dentro de la ruta de instalación. Se ha de comprobar que dentro de este archivo está habilitada la opción de: **vmci0.present = "TRUE".**

- ***This kernel requires an x86-64 CPU*:** este error puede deberse a dos cosas:

  • El sistema operativo es de 64 bits, mientras que el procesador es de32.

  En este caso, se debe buscar una imagen del sistema operativo de 32 bits

.

Si el error perdura, es necesario conocer si el procesador permite virtualizar.

Para evitar posibles fallos en la máquina virtual, es recomendable apagarla de forma correcta y segura, tal como se hace con el equipo con el sistema anfitrión.

Esto se realiza desde el menú de opciones dentro de la máquina virtual, donde se despliega el menú superior y se selecciona Apagar o *Power off*.

Por otro lado, hay veces en las que dentro de la máquina virtual no se detectan el teclado, el ratón u otros periféricos.

Esto se debe a que es necesario instalar las *herramientas tools* dentro del *software* de virtualización, como ya se ha explicado en un apartado anterior.

4.9. Documentación del proceso de instalación y de las incidencias aparecidas con sus soluciones

Cuando se trabaja con máquinas virtuales, es necesario realizar un documento técnico sobre el trabajo realizado**.**

Este debe tener dos partes distintas:

- **La primera se compone de las** especificaciones de la máquina real**, tanto de *hardware* como de *software*.**

- **La segunda contiene el mismo tipo de información, pero sobre la** máquina virtual**.**

En cuanto al *hardware* de la máquina real**, es necesario especificar las características principales, como pueden ser de los componentes que están relacionados con la máquina virtual, es decir, los componentes de la máquina física que se comparten con el huésped (tipo de procesador, memoria RAM y disco duro).**

Los apartados que definen las características del sistema operativo anfitrión **son:**

- **Nombre**
- **Versión**
- **Arquitectura**
- ***Software* de virtualización instalado**

En lo referente al *hardware* de la máquina virtual**, es necesario que quede constancia de los parámetros escogidos para cada uno de los componentes *hardware* (RAM y espacio de almacenamiento), junto con la fecha y hora, puesto que también tiene que reflejar cuándo se han registrado los cambios y sus nuevos valores.**

- **Los apartados que definen las características del sistema operativo huésped son:**

- **Nombre**
- **Versión**
- **Arquitectura**
- **Clave del producto**
- **Fecha y hora de instalación**
- **Usuario administrador**
- **Contraseña**
- **Licencias instaladas**
- **Observaciones**

Hay que tener en cuenta que, en función del tipo de sistema operativo que se instale, hay

parámetros que no se podrán reflejar (como, por ejemplo, la clave del producto en un sistema operativo libre).

4.10. Interpretación de la documentación técnica

Antes de realizar la instalación de una máquina virtual hay que comprobar si el equipo cuenta con el *hardware* necesario para ello.

El componente que permite realizar virtualizaciones es el procesador, por lo que es necesario que el nuestro tenga integrada esta tecnología. Es por ello que las páginas web de los fabricantes de procesadores ofrecen un apartado de características que permite acceder de forma rápida esta información.

Por otro lado, el nombre de las tecnologías asociadas a la virtualización también depende del tipo de procesador. En este caso, Intel cuenta con **Intel® V (VT- x)**, mientras que AMD tiene **AMD-V**.

En la actualidad, la mayoría de los procesadores cuentan con esta tecnología.

En el caso de Intel, los requisitos recomendados son tener un Intel core i5 o superior; en caso contrario, la virtualización puede ralentizar el rendimiento de nuestro equipo.

Para AMD son *Socket AM3*, *Socket AM2*, *Socket S1* y *Socket F*.

Para consultar las **características** de los distintos **procesadores** de **Intel**:
https://www.intel.es/content/www/es/es/products/processors.html

Para consultar las **características** de los distintos **procesadores** de **AMD**:
http://www.amd.com/es-xl/products/processors

Una vez se han comprobado los requisitos *hardware* para la virtualización, suele ser necesario habilitar esta característica en la BIOS. Para llevar esto a cabo, se entra dentro de ella y se navega por el menú hasta encontrar y seleccionar la opción ***Virtualización*** (en ***Seguridad*** o en ***Configuración***). Entonces se activa, se guardan los cambios y se reinicia del equipo, para que en el nuevo arranque del sistema operativo se pueda empezar a instalar y configurar la máquina virtual.