



UT5.GESTIÓN DE LA INFORMACIÓN

Sistemas Informáticos

Profesorado:
Diego J. García
Rosa María Zapata Calle

Introducción a las redes.

Índice

1. Sistemas de archivos (FS).
2. Gestión de FS mediante comandos y entornos gráficos.
3. Estructura de directorios de SSOO libres y propietarios.
4. Búsqueda de información del SO mediante comandos y herramientas gráficas.
5. Identificación del Sw instalado mediante comandos y herramientas gráficas.
6. Herramientas de administración de discos. Particiones y volúmenes. Desfragmentación y chequeo.
7. Montaje y desmontaje de dispositivos en sistemas operativos.
8. Tareas automáticas.
9. Tolerancia a fallos.
10. Copias de seguridad.
11. Recuperación del sistema.



1. SISTEMAS DE ARCHIVOS

- 1. Definiciones
- 2. Tipos

1.Sistemas de archivos. Definiciones

Un **sistema de ficheros** es un conjunto de algoritmos y estructuras de datos encargados de facilitar, al sistema operativo, la tarea de acceder a los datos en los diferentes dispositivos de almacenamiento al mismo tiempo que para el usuario le resulta en una manera sencilla y transparente.

Se va a encargar de representar la información de los diferentes ficheros, tanto datos como metadatos (nombre, fechas, permisos, tamaño, inodo, localización en disco, ...)

Un **bloque o clúster** es la unidad de almacenamiento del sistema de ficheros. Es el tamaño lógico con el que trabaja. No se pueden realizar lecturas/escrituras menores de este tamaño. Bloques grandes mejoran el rendimiento para ficheros grandes pero desperdician espacio para pequeños

1.Sistemas de archivos. Definiciones

Un **inodo** es una estructura de datos encargada de almacenar los metadatos de un objeto del sistema de ficheros que puede ser (un fichero “normal”, un enlace, un directorio, un dispositivo, ...)

Cada sistema operativo almacena la información de forma diferente en los inodos aunque el propósito es el mismo

La **FAT** es una tabla con información de los inodos.

El **superbloque** es una estructura de datos que contiene información global de todo el sistema de ficheros.

Ciertos SSOO contienen un **journaling**, sistema de control que permite revertir operaciones incompletas y evitar incoherencias al desconectar dispositivos incorrectamente

1.Sistemas de archivos. Tipos

Existen multitud de tipos de sistemas de ficheros. De hecho, cada sistema operativo puede manejar varios. El uso de uno u otro será transparente para el usuario pero en caso de problemas o situaciones límite habrá algunos más adecuados que otros, según la necesidad.

Los sistemas de ficheros **basados en bases de datos** almacenan los metadatos en BBDD en lugar de estructuras jerárquicas, pudiendo acceder a un fichero a través de consultas SQL independientemente de su localización.

Identifican los ficheros por sus características como son el tipo, el asunto, al autor, ... Ejemplos de ellos son BFS y WinFS

1.Sistemas de archivos. Tipos

Los sistemas de ficheros **transaccionales** almacenan los eventos o transacciones del fichero. Aseguran la integridad del sistema comprobando que toda operación se acabe.

Si pierde el acceso al sistema de ficheros y una operación queda a medias, se refleja en el *journal* y en el siguiente arranque se deshacen las operaciones incompletas.

Ejemplos (ReiserFS, ext3, ext4, NTFS solo en metadatos)

Sistemas de ficheros **en red** son los usados para permitir el acceso remoto a los ficheros como (NFS, SMB, CIFS)

Sistemas de ficheros **de propósito general** son aquellos que no son ni de disco ni de red y se usan para sistemas de archivos volátiles y temporales (ProcFS, DevFS)



2. Gestión de sistemas de archivos mediante comandos y entornos gráficos

1. Navegación por el sistema
2. Listado de ficheros y directorios
3. Creación de ficheros y directorios
4. Copiando ficheros y directorios
5. Moviendo y renombrando ficheros y directorios
6. Eliminando ficheros y directorios

2. Gestión de sistemas de archivos.

Navegación por el sistema

- Los sistemas de ficheros de los diferentes sistemas operativos han evolucionado (y siguen haciéndolo) a lo largo de las diferentes versiones.
- En MS-DOS 1, por ejemplo, todos los ficheros del sistema estaban en un mismo sitio
- En UNIX, a lo largo de su desarrollo han cambiado ciertos lugares de configuración del sistema.
- En Windows, dependiendo de la versión, y si es servidor o no, se ha cambiado la estructura de directorios.
- Además, dependiendo de la configuración de cada usuario (independientemente del sistema operativo) se pueden cambiar las localizaciones de los ficheros.

2. Gestión de sistemas de archivos.

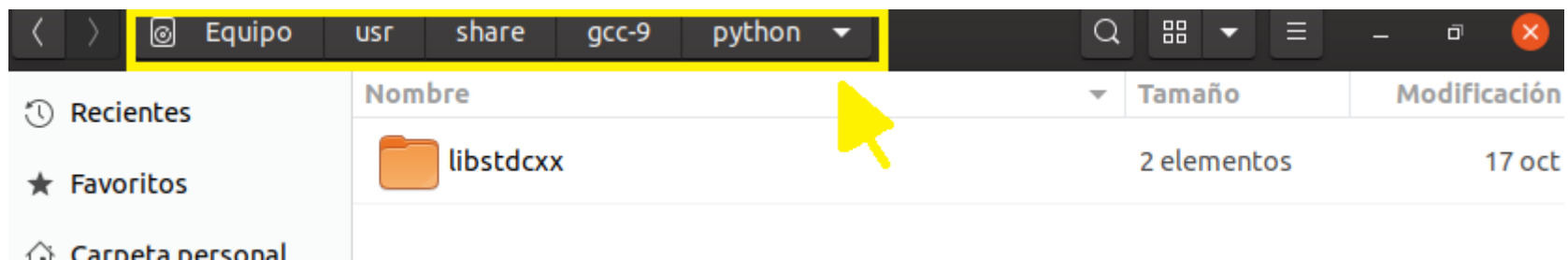
Navegación por el sistema

- Además, cada usuario puede organizar la información dependiendo de sus preferencias personales.
- Por estas, y otras muchas razones, es necesario saber navegar por los diferentes directorios del sistema.
- Para poder movernos (o realizar otros tipos de tareas) por el sistema usaremos dos tipos de rutas que son equivalentes. Las rutas **relativas** y las rutas **absolutas**.
- Las rutas **absolutas** son aquellas que muestran **todas** las partes de la ruta empezando siempre desde la raíz.
- Las rutas **relativas** son las que se crean desde el lugar donde nos encontramos actualmente.
- Aunque se pueden usar indistintamente, es común usar las relativas siempre que se pueda.

2. Gestión de sistemas de archivos.

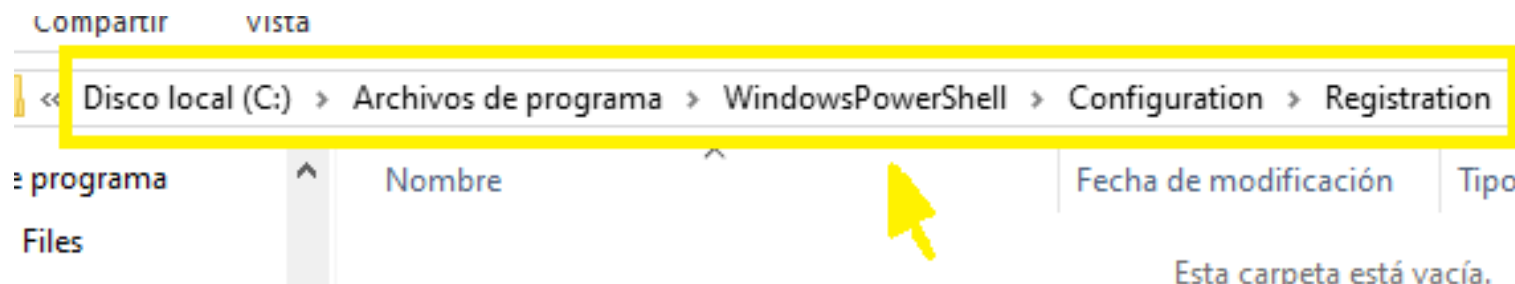
Navegación por el sistema

- En ocasiones es necesario saber el directorio donde nos encontramos. Para verlo, dependiendo de si es entorno gráfico o por comando lo veremos de una forma u otra.
- En Linux Desktop, si navegamos por los directorios, por ejemplo, a la ruta absoluta `/usr/share/gcc-9/python` usando el explorador de archivos podremos ver la ruta en la parte superior, teniendo en cuenta que equipo es “/”
- La visualización dependerá del explorador de archivos.



2. Gestión de sistemas de archivos. Navegación por el sistema

- En Windows 10, en un explorador de archivos, podemos ver que en la parte superior aparece el directorio donde nos encontramos actualmente.
- Por ejemplo, si entramos en la ruta *C:\Archivos de Programa\WindowsPowerShell\Configuration\Registration*



- En la primera parte se puede ver “Disco local (C:)” pero la ruta real sería “C:\”
- También se debe observar que las barras usadas en la ruta de Windows “\” es diferente de la usada en Linux “/”

2. Gestión de sistemas de archivos.

Navegación por el sistema

- No se puede llegar a administrar un sistema operativo sin saber usar la **Shell**, también llamada **CLI (Command Line Interface)**, línea de comandos o terminal.
- Para usarla en Windows ejecutaremos el programa **cmd**
- En Linux podemos usar una terminal virtual pulsando (en Ubuntu Desktop) **Ctrl + Alt + t**
- También podemos abrir las aplicaciones y buscar “**term**” y nos aparecerá el icono con el enlace a la terminal virtual.
- Los programas anteriores son emuladores de terminal. Además, se pueden usar terminales puras. Normalmente se suelen lanzar 6 y una/dos gráficas. Para abrir las consolas en modo texto pulsamos **Ctrl + Alt + Fn** siendo Fn la teclas de función.

2. Gestión de sistemas de archivos.

Navegación por el sistema

- En la consola es normal ver la ruta en la que estamos simplemente mirando en el texto que nos aparece nada más cargarla y que nos indica que podemos empezar a escribir comandos. Este mensaje se llama **prompt**.
- En el cmd de Windows aparecería algo como:

```
C:\Program Files\GIMP 2\bin>_
```

- En el caso de Linux lo que veríamos sería:

```
usuario@usuario:/usr/share/fonts$
```

- Pero este prompt es configurable y depende mucho de la versión y distribución del sistema operativo en cuestión.
- De hecho, en muchos casos solamente vemos un \$ (usuario sin privilegios) o un # (root) como prompt del CLI

2. Gestión de sistemas de archivos.

Navegación por el sistema

- Si queremos conocer la ruta donde nos encontramos, independientemente de lo que aparezca en el prompt, podemos usar el comando **pwd**
- Este comando no está disponible para **cmd** de Windows pero sí que aparece en Linux y en la nueva terminal de Windows, mucho más completa llamada **PowerShell**, en este caso es un alias del cmdlet (nombre que se les da a los comandos de powershell) **get-location**
- Si necesitásemos saber donde nos encontramos en la consola de windows podemos usar el comando **cd** pero habría que tener en cuenta que este comportamiento es diferente en PowerShell y también es distinto en Linux

2. Gestión de sistemas de archivos.

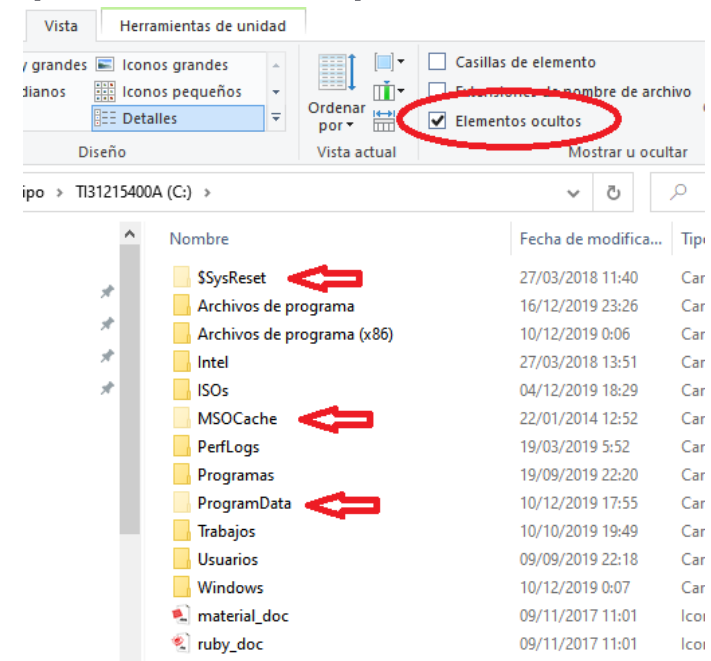
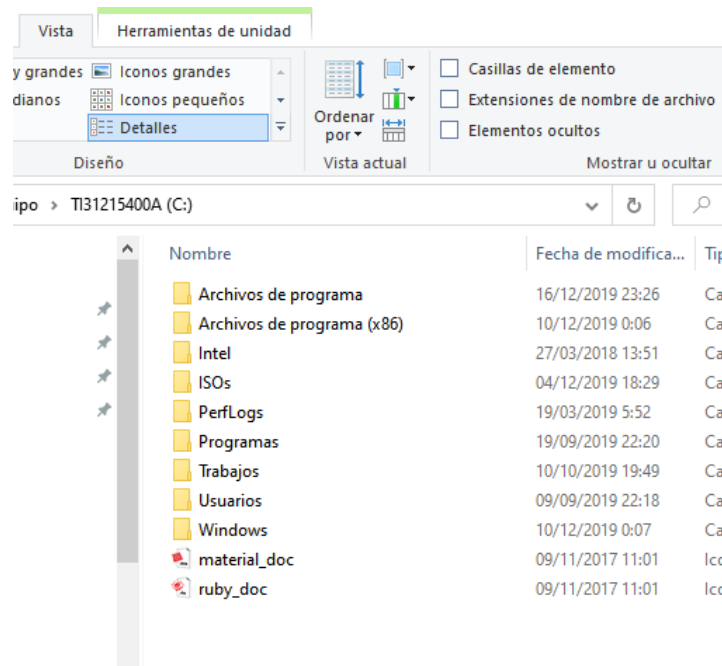
Navegación por el sistema

- Para movernos de un directorio a otro, en los entornos de escritorio, simplemente vamos pulsando, con el ratón, en las carpetas o en los árboles/listas del explorador.
- En la consola, en Linux, cmd o powershell usamos **cd**
- La sintaxis básica es **cd [ruta del directorio]**
- Si no pasamos argumentos el comportamiento en cmd es mostrar el path actual, en Linux nos lleva a nuestro **\$HOME** (directorio personal) y en **PS** (PowerShell) nada.
- Se pueden usar rutas relativas o absolutas teniendo en cuenta que hay dos directorios especiales:
- El directorio “..” que es el directorio padre
- El directorio “.” que es el directorio actual

2. Gestión de sistemas de archivos.

Listado de ficheros y directorios

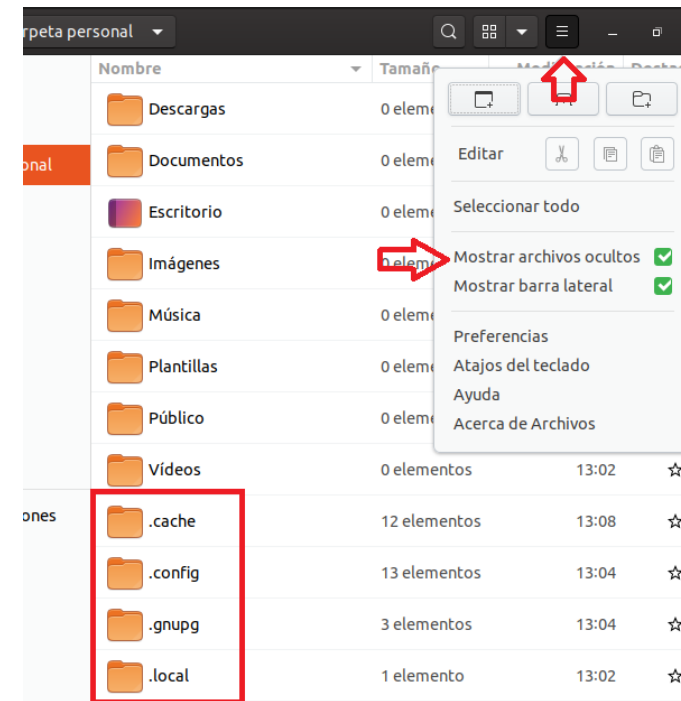
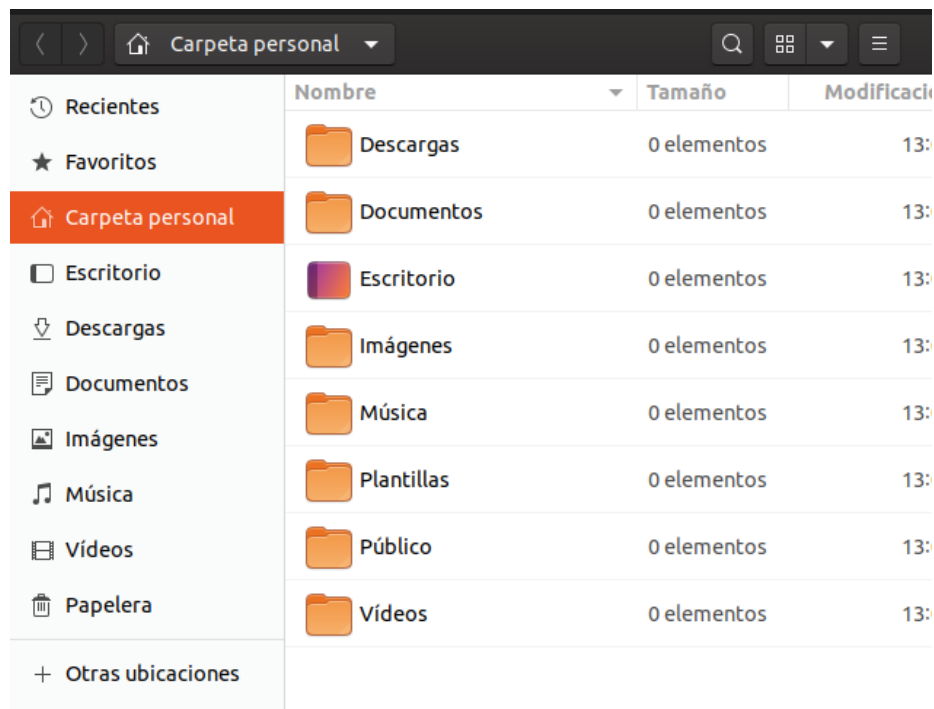
- Para mostrar los objetos (directorios, ficheros, enlaces,...) que tiene dentro un directorio en modo gráfico debemos usar el explorador de archivos y acceder al directorio.
- Puede ser que haya elementos ocultos. Para verlos habrá que activar las opciones de carpeta correspondientes.



2. Gestión de sistemas de archivos.

Listado de ficheros y directorios

- En Linux, los ficheros ocultos son aquellos que empiezan por un punto. Independientemente del nombre
- Igualmente, se deben marcar las opciones adecuadas para mostrar todo el contenido del directorio.



2. Gestión de sistemas de archivos.

Listado de ficheros y directorios

- En la terminal de comandos usaremos principalmente el comando **ls**. Existe la posibilidad de usar el comando **echo**, cuyo uso principal es mostrar mensajes por pantalla, para poder listar los ficheros de un directorio.
- El comando **ls** se puede usar tanto en **Bash** (la shell más extendida de Linux) como en **PS** para windows.
- Sin embargo, no se puede usar en **cmd** ya que no está implementada. En caso de necesitarlo se usa **DIR**
- La sintaxis básica es **ls [opciones] [rutas a ficheros]**
- Las opciones de un comando son modificadores que cambian la forma de actuar del comando. Normalmente van precedidas de uno o dos guiones

2. Gestión de sistemas de archivos.

Listado de ficheros y directorios

- En Linux, las opciones más comunes del comando **ls**, que, además, es probablemente, el comando más usado:
- -a Muestra todos los archivos, incluidos los ocultos
- -l Muestra la lista de archivos en formato extendido, es decir, con más detalles de los metadatos como el tipo de fichero, los permisos, enlaces al fichero, propietario, grupo propietario, fecha de modificación y nombre
- -lh Muestra el tamaño del archivo en formato “humano” y no en bytes. Es obligatorio usarlo en formato extendido
- -li Muestra el inodo donde se alojan los metadatos.
- -r Muestra los archivos en orden inverso
- -R Muestra la información de forma recursiva

2. Gestión de sistemas de archivos.

Creación de ficheros y directorios

- Para crear nuevos ficheros en un **entorno gráfico** podemos pulsar en una zona “vacía” del explorador de archivos con el botón derecho y elegir la opción “nuevo” y después el tipo de archivo que queremos.
- En este caso solamente nos ofrecerá un listado muy reducido de los tipos de ficheros más habituales.
- Al crear estos tipos de ficheros debemos tener en cuenta que, en la mayoría de los casos, solamente crea un fichero con un nombre completamente vacío, es decir, da igual el nombre y **extensión** que le pongamos.
- En otros casos, si ocupa algo de espacio, le inserta las estructuras básicas del documento para almacenar datos.

2. Gestión de sistemas de archivos.

Creación de ficheros y directorios

- En la consola existe un comando, cuyo propósito inicial es **cambiar la fecha de acceso** a un fichero cualquiera.
- Si el fichero no existía, lo creaba, así que con el tiempo se ha usado más para crear nuevos archivos que para aquello que estaba diseñado. El comando es **touch**
- La sintaxis básica es **touch <lista de ficheros>**, permite usar comodines (si los ficheros existen) y expresiones regulares (si existen o los queremos crear)
- Su equivalente (para creación de ficheros) en PS es **New-Item** pero la creación con expresiones regulares necesita de comandos más avanzados (como bucles) que no es el propósito del curso actual.

2. Gestión de sistemas de archivos.

Creación de ficheros y directorios

- Para crear directorios, en entorno gráfico, usamos “botón derecho” - “nuevo” - “carpeta” sobre un espacio vacío
- En la consola se usa el comando **mkdir** que nos creará un nuevo directorio.
- La sintaxis básica **mkdir <lista de directorios a crear>**
- Existe la opción **-p** que nos permite crear una ruta de directorios completa en caso de que no existan los padres lo que nos evitaría tener que crear cada uno de los directorios de la ruta y poder hacerlo todo a la vez
- Para el uso en sistemas windows se puede usar **md** o, si se usa en PS, el comando **New-Item**, que, con el modificador **-ItemType “Directory|File”** crearía uno u otro

2. Gestión de sistemas de archivos.

Copiando ficheros y directorios

- Para copiar objetos (ya sean ficheros o directorios) en entorno gráfico, se seleccionan los objetos, se pulsa el botón derecho sobre ellos, y, en el menú desplegable se elige la opción “**copiar**” (la alternativa es la combinación de teclas Ctrl+c) y posteriormente, en el directorio destino se pulsa con el botón derecho y, en el menú desplegable se selección la opción “**pegar**” (la alternativa por teclado es la combinación de teclas Ctrl-v)
- Por consola, el comando que se usaría es **cp** (en la consola cmd sería **copy** y en PS **Copy-Item**)
- Su sintaxis básica es **cp <ficheros origen> <ruta de destino>**

2. Gestión de sistemas de archivos. Copiando ficheros y directorios

- Algunas opciones interesantes para la copia son:
- -r copia de forma recursiva (tanto ficheros con directorios)
- -i pide confirmación antes de sobrescribir ficheros
- -u solo copia el fichero si la fecha del origen es posterior a la fecha que tiene en el destino (si se ha modificado)
- Además es muy común usar comodines, especialmente:
- * Sustituye **cualquier número** de caracteres (**0 o más**) sin importar qué caracteres son
- ? Sustituye a **un solo** carácter sea cual sea. Es importante tener en cuenta que es uno y solo uno. Si queremos sustituir tres caracteres, debemos usar tres signos de interrogación

2. Gestión de sistemas de archivos.

Moviendo ficheros y directorios

- Para mover objetos en entorno gráfico, se seleccionan los objetos, se pulsa el botón derecho sobre ellos, y, en el desplegable se elige la opción “**cortar**” (la alternativa es la combinación de teclas Ctrl+c) y posteriormente, en el directorio destino se pulsa con el botón derecho y, en el menú desplegable se selecciona la opción “**pegar**” (la alternativa por teclado es la combinación de teclas Ctrl-v)
- Por consola, el comando que se usaría es **mv** (en la consola cmd sería **move** y en PS **Move-Item**)
- Sintaxis: **mv <ficheros origen> <ruta de destino>**
- Igual que con cp, se pueden usar opciones (sobre todo **-r**) y también es muy común el uso de **comodines** y **regexp**

2. Gestión de sistemas de archivos. Eliminando ficheros y directorios

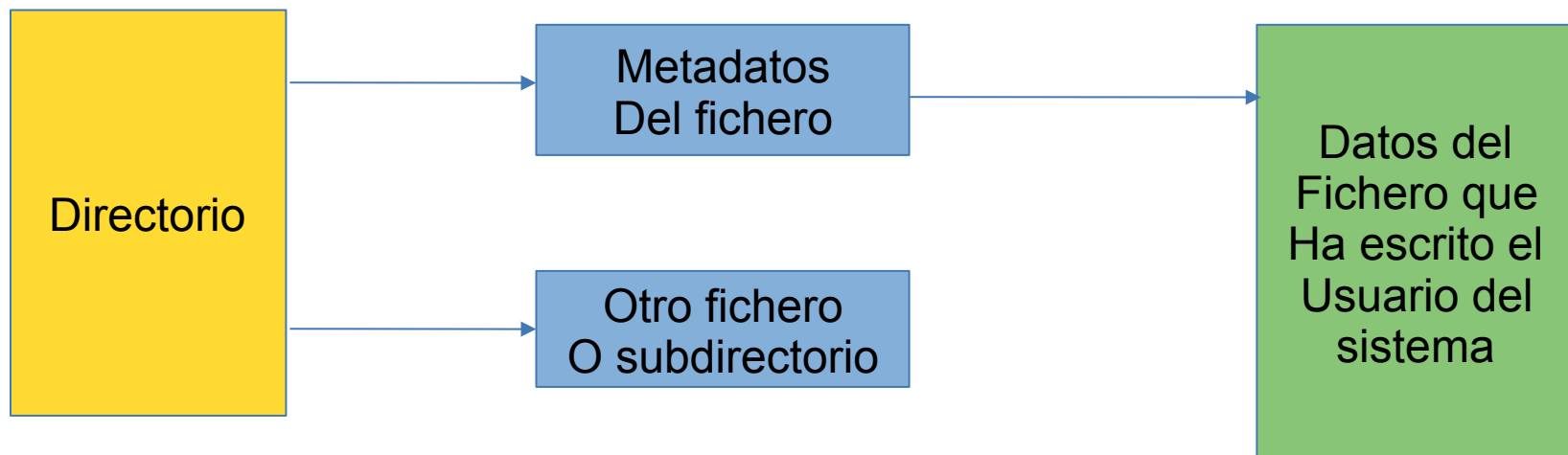
- Para eliminar objetos en entorno gráfico, se seleccionan los objetos, se pulsa el botón derecho sobre ellos, y, en el desplegable se elige la opción “**eliminar**” (la alternativa por teclado es **supr**)
- Esto hará que el fichero se mueva a la **papelera de reciclaje**, lo que lo hace recuperable
- Para que ¿no sea recuperable? se pulsa la tecla **shift** en el momento de pulsar “**eliminar**” o “**suprimir**”
- Por consola, el comando que se usaría es **rm** (en la consola cmd sería **del** y en PS **Remove-Item**)
- Sintaxis: **rm <ficheros origen> <ruta de destino>**
- Igual que con **cp**, se pueden usar opciones (sobre todo **-r**) y también es muy común el uso de **comodines** y **regex**


2. Gestión de sistemas de archivos. Eliminando ficheros y directorios

- Realmente, **rm**, es un comando para eliminar **solamente** ficheros. Para borrar directorios se usa **rmdir** pero tiene una condición y es que los directorios deben estar completamente vacíos, no pueden contener ficheros (incluidos los ocultos) ni tampoco directorios.
- Para evitar este problema se usa la opción **-r** (recursiva) especialmente cuando hay un **árbol** de directorios con muchos subdirectorios, pero **NO es adecuado** usarlo para borrar directorios que están vacíos
- Una buena pregunta sería ¿Qué pasa con los ficheros borrados? ¿Como se borra internamente un fichero?

2. Gestión de sistemas de archivos. Eliminando ficheros y directorios

- El contenido de un directorio son enlaces a ficheros. Cada fichero se divide en datos y metadatos
- Al borrar un fichero se borra el enlace y se marca el espacio como libre. Por eso, con ciertos programas se puede recuperar la información buscando aquellos sitios del disco duro que tienen datos que no se han sobrescrito





3. Estructura de directorios de SSOO libres y propietarios

1. Estructura de directorios Linux
2. Estructura de directorios Windows 10

3. Estructura de directorios de SSOO FHS Linux

- En un sistema operativo se conoce como **FHS** (File Hierarchy Standard) a la estructura de directorios base que se suele mantener en la mayoría de las versiones y **sabores** (distribuciones) a lo largo del tiempo.
- En esta jerarquía estándar se definen los directorios, sus rutas y el contenido que deben tener sin indicar si deben estar en una partición u otra pero sí la función que cumplen
- Gracias a esta jerarquía nos resultará más cómodo localizar los diferentes ficheros, programas, librerías del sistema
- Esta jerarquía deriva en gran medida de UNIX y es aplicable también a los sistemas BSD y por tanto Apple

3. Estructura de directorios de SSOO FHS Linux

- / Es el directorio raiz o **root** el que “cuelga” todo el sistema. Desde él nacen todos los directorios aunque estén almacenados en diferentes discos o particiones. Sabremos que una ruta es absoluta si empezar por /
- **/bin** es un directorio donde se almacenan los ficheros “**binarios**” (programas ejecutables) usados para las tareas más comunes de **todos** los usuarios. En la subestructura de directorios es muy común utilizar un subdirectorio de una aplicación llamado **bin** para indicar donde se encuentran los programas ejecutables
- **/sbin** es un directorio de binarios pero usados para tareas administrativas del sistema que ejecutará el **root**

3. Estructura de directorios de SSOO FHS Linux

- **/boot** Directorio que contiene los archivos necesarios para el arranque del sistema. Son los ejecutables usados antes de lanzar el kernel. Además contiene el GRUB
- **/dev** Directorio que contiene todos los dispositivos del sistema. Sus ficheros son **representaciones lógicas** del hardware del sistema. Contiene algunos dispositivos muy curiosos como “**null**” que es un fichero al que se le puede enviar cualquier cosa que la absorberá pero nunca se podrá extraer nada de él, o también “**random**” que es un generador de números aleatorios del sistema, aparte de los dispositivos instalados como discos, teclado, ratón,...
- **/etc** Almacena los ficheros de configuración tanto del sistema como de programas instalados posteriormente

3. Estructura de directorios de SSOO FHS Linux

- **/home** Directorio donde se almacenan (salvo que se configure de otra forma) los directorios y ficheros de los usuarios del sistema. Cada directorio incluido será de un usuario que tendrá control sobre ellos. Es común crear una partición solo para este directorio de forma que, si se borra/formatea/reinstala el sistema, los ficheros de los usuarios permanezcan intactos.
- **/lib** Directorio con las bibliotecas necesarias para poder ejecutar los binarios de /bin o /sbin y del kernel. Con el auge de los sistemas de 64 bits aparece /lib64 que es un directorio con la versión de estas librerías pero para aplicaciones compiladas para arquitecturas de 64 bits.

3. Estructura de directorios de SSOO FHS Linux

- **/media** Es el punto de montaje de los volúmenes lógicos que se montan temporalmente, como unidades USB, discos duros externos, CD/DVD, otras particiones,... Dentro de este directorio se suele crear un subdirectorio para cada usuario donde montará sus dispositivos. Antiguamente se usaba el directorio **/mnt** pero a día de hoy, la mayoría de las distribuciones lo están cambiando.
- **/opt** es un directorio “opcional” en el que se incluyen los archivos de “solo lectura” de programas que no siguen los estándares de almacenar sus ficheros en los distintos subdirectorios que inicialmente sería lo recomendable
- **/root** Es el directorio “HOME” del superusuario del sistema. Por defecto, está separado del resto de usuarios

3. Estructura de directorios de SSOO FHS Linux

- **/proc** Es un directorio en un sistema de ficheros “virtual” pues no se almacena en disco, solamente en memoria. Contiene información de los procesos y del estado y configuración del sistema operativo en un momento **puntual**, en concreto, justo cuando alguien lo consulta.
- **/sys** Contiene archivos virtuales que nos dan información del **kernel** relativa a eventos del sistema operativo.
- **/tmp** Usado para almacenar ficheros temporales. Se borra automáticamente al reiniciar el sistema. Si por alguna razón se quisieran conservar estos ficheros hay otro directorio para ello situado en **/var/tmp** además de otros posibles directorios que se pueden usar para ello

3. Estructura de directorios de SSOO FHS Linux

- **/usr** Su nombre significa “User System Resources” y se suele usar para almacenar los archivos de solo lectura relativos a las diferentes aplicaciones de cada usuario. Usualmente software instalado por gestores de paquetes
- Contiene una gran variedad de directorios como:
- **/usr/bin /usr/include /usr/lib /usr/sbin /usr/src**
- Cabe destacar **/usr/share** que almacena los ficheros compartidos por todos los usuarios como las **fuentes**, listas de palabras, imágenes, ...
- **/var** Contiene información del sistema como archivos de **logs**, emails, bases de datos, u otra información variable. Cabe destacar el directorio **/var/www** para servicio web

3. Estructura de directorios de SSOO

Árbol de directorios de Windows

- La estructura de directorios de Windows tiene un objeto que engloba a los diferentes discos y particiones llamado “Equipo” (dependiendo de la versión puede cambiar a “Mi PC”, “Mi equipo”, “Este equipo”, o un nombre similar)
- En este objeto se incluye, entre otros el disco donde se han instalado Windows, que salvo que se cambie en la instalación suele ser el volumen “C:\” y es a partir de este sitio del que (salvo que se configuren las localizaciones) dependerá el resto del sistema.
- En la raíz del disco nos vamos a encontrar con los directorios principales del sistema. En ocasiones ser denominan también como variables especiales para no ser dependientes del lugar de instalación

3. Estructura de directorios de SSOO

Árbol de directorios de Windows

- **%windir%** normalmente situado en C:\Windows es el directorio que contiene la instalación del sistema junto con todos los ficheros y utilidades del mismo.
- **%programfiles%** normalmente situado en “**C:\Windows\Archivos de Programa**” o también en “**C:\Windows\Program Files**” es el directorio donde se almacenan los directorios de instalación de los programas de usuario. En muchos casos aparece también un directorio llamado “**C:\Windows\Archivos de Programa (x86)**” que tiene los programas que están compilados para arquitecturas de 32 bits. Ambos contienen un subdirectorio llamado “**Common Files**” con datos comunes a varios programas

3. Estructura de directorios de SSOO

Árbol de directorios de Windows

- **C:\users** también llamado **C:\usuarios** es el directorio donde se almacenan los directorios personales de cada uno de los usuarios de Windows. Cada vez que se crea un usuario se le creará automáticamente un directorio con los siguientes subdirectorios:

- **C:\usuarios\Escritorio**
- **C:\usuarios\Música**
- **C:\usuarios\Videos**
- **C:\usuarios\Búsquedas**
- **C:\usuarios\Descargas**
- **C:\usuarios\Juegos guardados**

3. Estructura de directorios de SSOO

Árbol de directorios de Windows

- **C:\PerfLogs** es un directorio del sistema que contiene archivos de registro (**logs**) con la actividad del usuario.
- **C:\ProgramData** es un directorio que almacena información específica de diferentes aplicaciones. Suelen ser datos genéricos de la aplicación que se usarán para los diferentes usuarios.
- Directorios principales de C:\Windows:
- **System32** contiene diferentes utilidades del sistema, así como muchas librerías DLL compartidas. Contiene diferentes directorios con ficheros y configuraciones necesarias para el correcto funcionamiento de Windows
- **Fonts** contiene los tipos de letra del sistema

3. Estructura de directorios de SSOO

Árbol de directorios de Windows

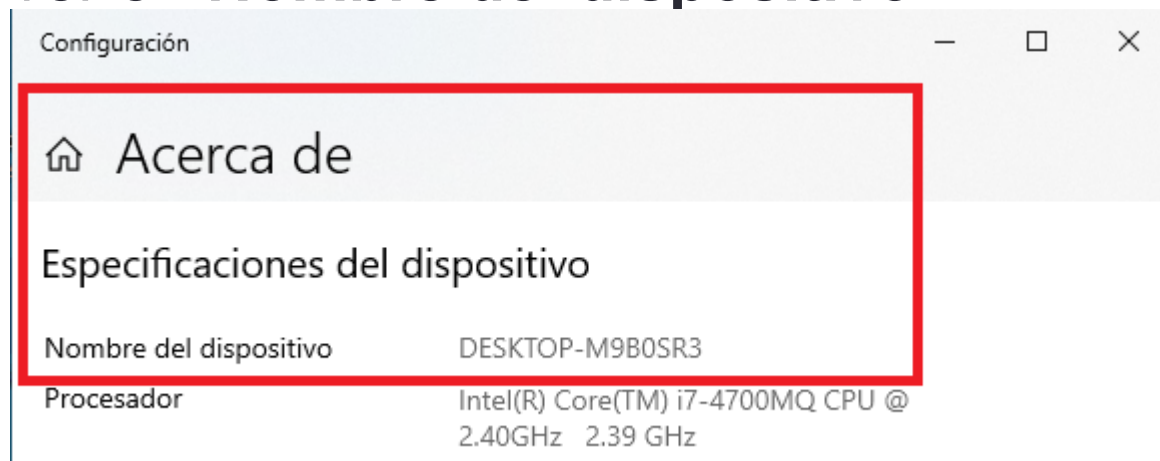
- Dentro del directorio **C:\Windows\System32** podemos destacar los siguientes directorios:
- **Drivers** que contiene los ficheros con extensión **sys** cuya función es configurar correctamente los drivers de los dispositivos periféricos instalados en el sistema. Dentro de esta carpeta hay otra llamada **etc** en la que se pueden ver distintos ficheros de configuración de red como **hosts** que permite crear un “DNS” local fácilmente.
- **Config** que contiene los datos del registro de Windows, entre los ficheros podemos ver el fichero **SAM** que almacena los datos de usuarios y sus claves “hasheadas”

4. Búsqueda de información del sistema mediante comandos y herramientas gráficas

1. Nombre del equipo
2. Versión del Sistema Operativo
3. Información de la CPU
4. Información de la RAM
5. Información de los dispositivos PCI
6. Información de los dispositivos USB

4. Búsqueda de información del sistema. Nombre del equipo

- En Windows, para ver información del equipo hay varias formas, una de ellas es pulsar con el botón derecho del ratón en el botón “**Inicio**” del escritorio. En el menú que aparece se debe elegir la opción “**Sistema**” y en la pantalla que se abre llamada “**Acerca de**” se puede ver la información tanto de este apartado como algunos otros.
- En la parte de **Especificaciones del dispositivo** se puede ver el “**Nombre del dispositivo**”

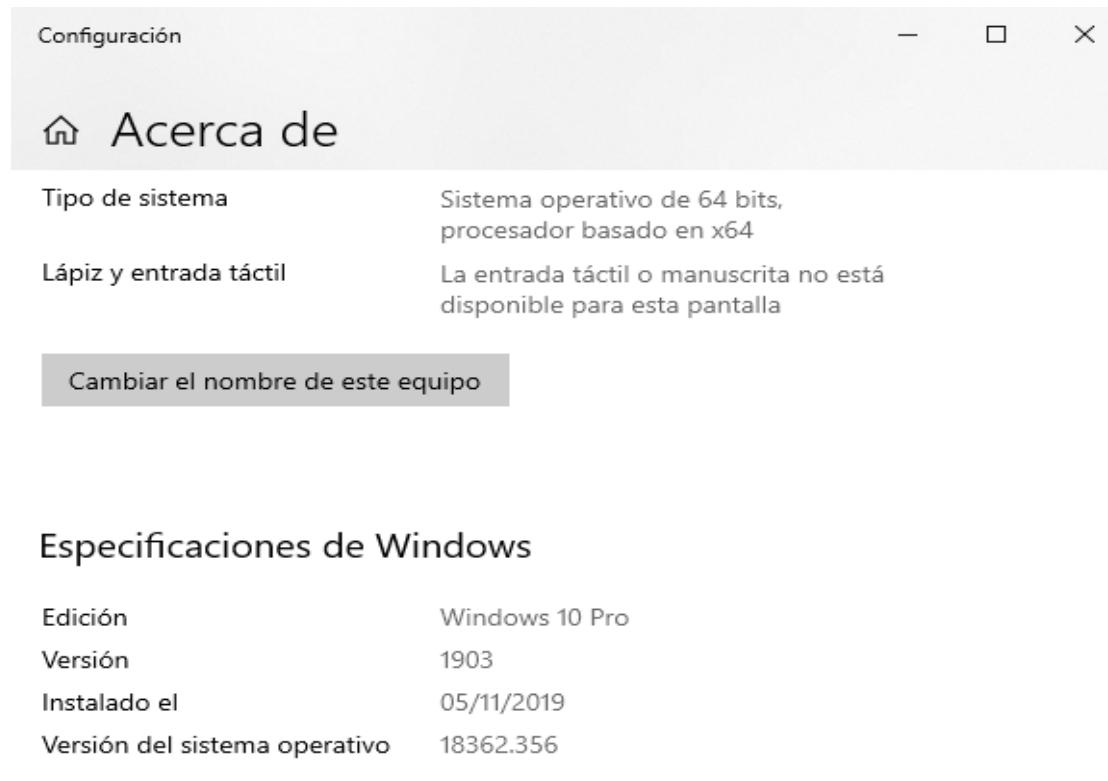


4. Búsqueda de información del sistema. Nombre del equipo

- En Linux, es común ver el nombre en la consola. El comando para verlo es **hostname** Este nombre se almacena en el fichero **/etc/hostname** con lo cual también podemos verlo con el comando “**cat**” que sirve para mostrar el contenido de los ficheros. La sintaxis sería: **cat /etc/hostname**
- El comando tanto en **cmd** como en **Powershell** también es **hostname**
- Con el cmdlet de Powershell **Get-ComputerInfo** podemos obtener mucha información del sistema. Para extraer solamente el nombre del equipo podemos usar:
- **Get-ComputerInfo -Property csname**

4. Búsqueda de información del sistema. Versión del SO

- Para ver la versión del sistema operativo en windows accedemos a la pantalla anterior y nos indica la arquitectura, la distribución de Windows, su versión y su compilación

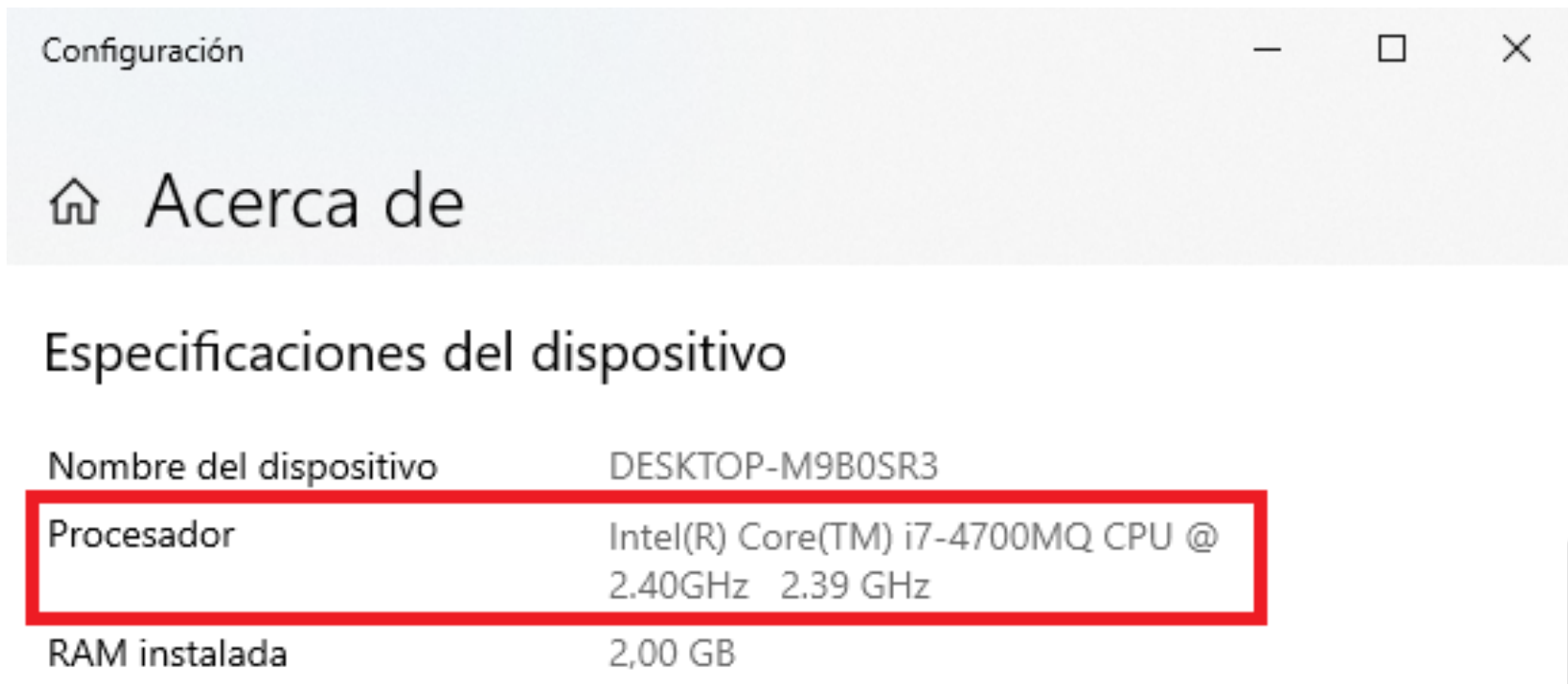


4. Búsqueda de información del sistema. Versión del SO

- Para verla en Ubuntu, igualmente accederemos a la consola y ejecutaremos el comando **uname -a** que nos mostrará la versión del kernel que se está ejecutando y la arquitectura para la que se ejecuta entre otros datos.
- Si queremos ver los datos de la distribución podemos mostrar el fichero “**/etc/lsb-release**” con cat
- Para obtener estos datos en PowerShell utilizamos el cmdlet anterior pero le añadimos una lista de propiedades que queremos mostrar (podemos ampliarla cuanto queramos si es necesario). La sintaxis sería:
- **Get-ComputerInfo -Property osname, osversion, windowsversion, osarchitecture, windowsbuildlabex**

4. Búsqueda de información del sistema. CPU

- La CPU, Windows se ve en la misma pantalla anterior, en este caso hay que buscarla en las especificaciones del dispositivo, en el apartado “**Procesador**”



The screenshot shows a Windows window titled 'Configuración' (Settings) with standard window controls. Below the title bar is a navigation bar with a home icon and the text 'Acerca de' (About). The main content area is titled 'Especificaciones del dispositivo' (Device specifications). It contains a table of system information. The 'Procesador' (Processor) row is highlighted with a red rectangular border. The table lists the device name, processor details, and installed RAM.

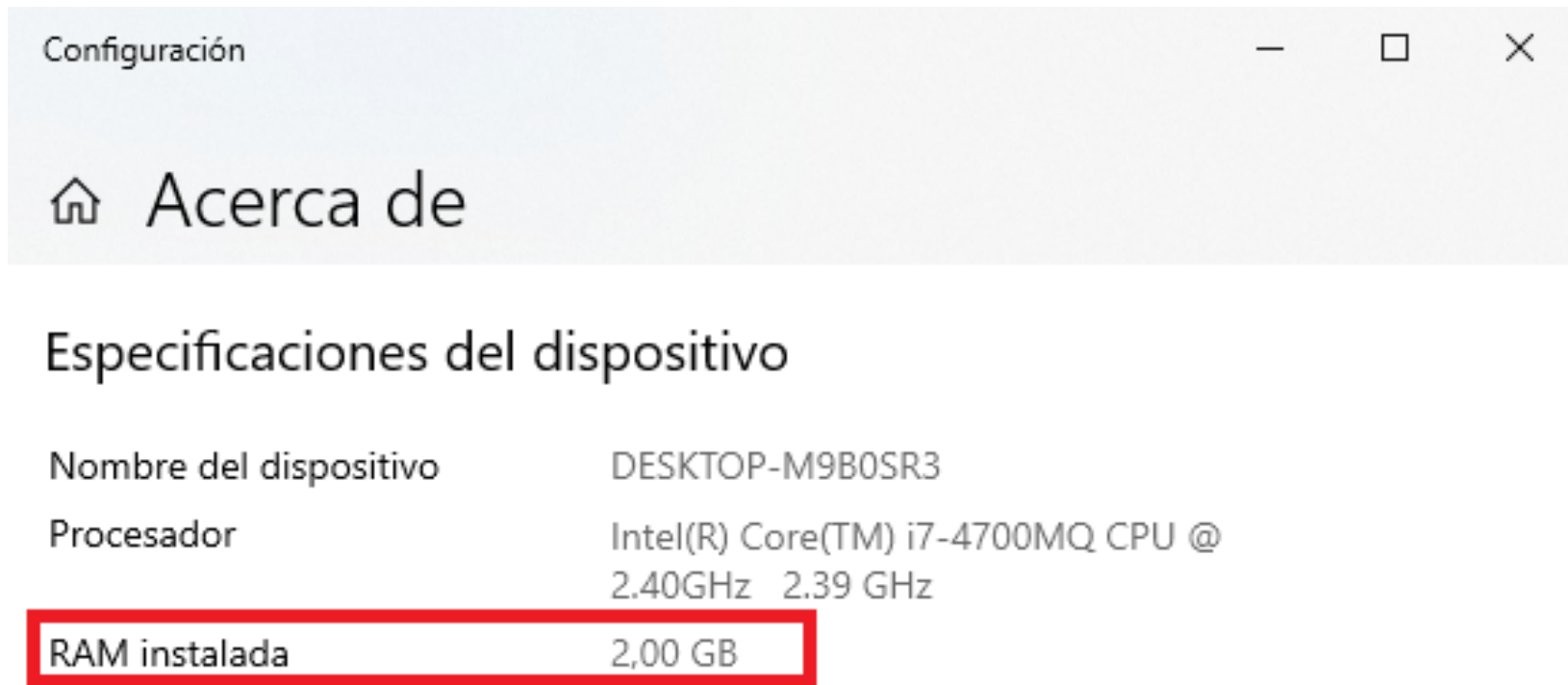
Nombre del dispositivo	DESKTOP-M9B0SR3
Procesador	Intel(R) Core(TM) i7-4700MQ CPU @ 2.40GHz 2.39 GHz
RAM instalada	2,00 GB

4. Búsqueda de información del sistema. CPU

- Para mostrar los datos del procesador en Linux debemos usar el comando “**lscpu**” que nos muestra por pantalla información detallada del procesador.
- Otra forma muy común de ver información de la CPU es visualizando el fichero virtual del directorio **/proc**, su sintaxis sería: **cat /proc/cpuinfo**
- Para ver la información de la CPU en PowerShell volvemos a usar el comando anterior pero mostrando las siguientes propiedades:
- **Get-ComputerInfo -Property CsProcessors, CsNumberOfProcessors, CsNumberOfLogicalProcessors,**

4. Búsqueda de información del sistema. RAM

- La memoria RAM instalada en el sistema Windows se puede ver en la misma pantalla que hemos utilizado en los casos anteriores:

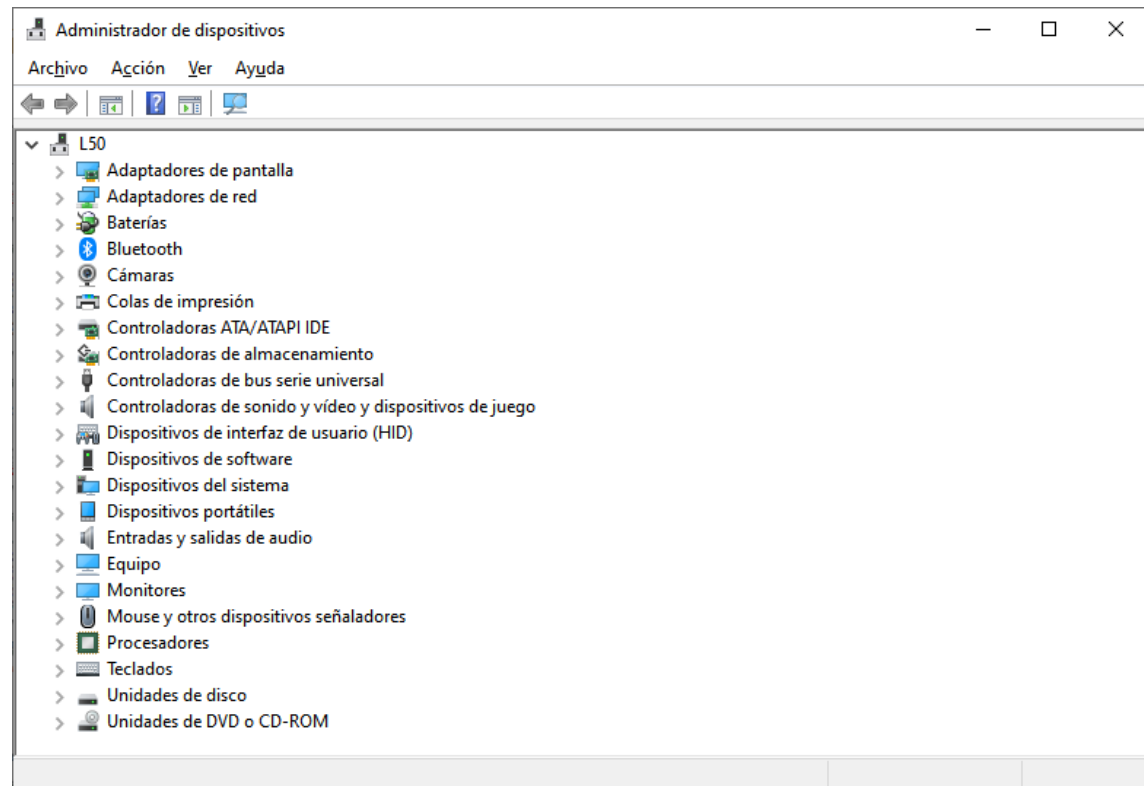


4. Búsqueda de información del sistema. RAM

- En Linux, hay varias formas de ver la RAM que tenemos en el sistema
- El comando más común es **free** que nos muestra información de la memoria total, la usada, la libre y swap
- Otra forma con la que se vería un estudio mucho más detallado sería viendo el fichero virtual **/proc/meminfo**
- Para ver la memoria en PowerShell hay varios métodos. Uno de ellos es usando el mismo cmdlet que en ocasiones anteriores. Podemos mostrar varias propiedades, entre ellas:
- **Get-ComputerInfo -property OsTotalVisibleMemorySize**

4. Búsqueda de información del sistema. Dispositivos PCI

- Para ver los dispositivos conectados en el sistemas pulsamos con el botón derecho del ratón en el botón inicio del sistema. En el menú elegimos “**Administrador de dispositivos**”
- En la pantalla que aparece
- podremos verlos:



4. Búsqueda de información del sistema. Dispositivos PCI

- En Linux se pueden mostrar los dispositivos conectados a nuestro equipo con el comando **lspci**
- Para ver los datos con más detalles se puede usar la opción **-v** o mucho más detallada con **-vv**
- Para verlos en PowerShell utilizaremos el cmdlet:

Get-PnpDevice

- Este cmdlet nos muestra todos los dispositivos conectados a la placa base, incluyendo procesador, impresoras que se han detectado por el sistema, BIOS, y también los USB reconocidos por Windows entre muchos otros tipos de periféricos

4. Búsqueda de información del sistema. Dispositivos USB

- La forma de mostrar los dispositivos USB conectados a nuestro Linux es con el comando **lsusb**
- Igualmente se pueden mostrar más detalles con la opción **-v** de forma similar a como se hacía con **lspci**.
- Con el comando **dmidecode** podemos ver información de la BIOS de nuestro sistema.
- Hay que tener en cuenta que, a través de la línea de comandos podemos obtener una gran cantidad de información de los dispositivos de forma más extensa, explícita y sobre todo mucho más rápida.
- Con el cmdlet de PowerShell “**Get-WmiObject <objeto>**” podemos extraer datos de la mayoría de dispositivos



5. Identificación del software instalado mediante comandos y herramientas gráficas

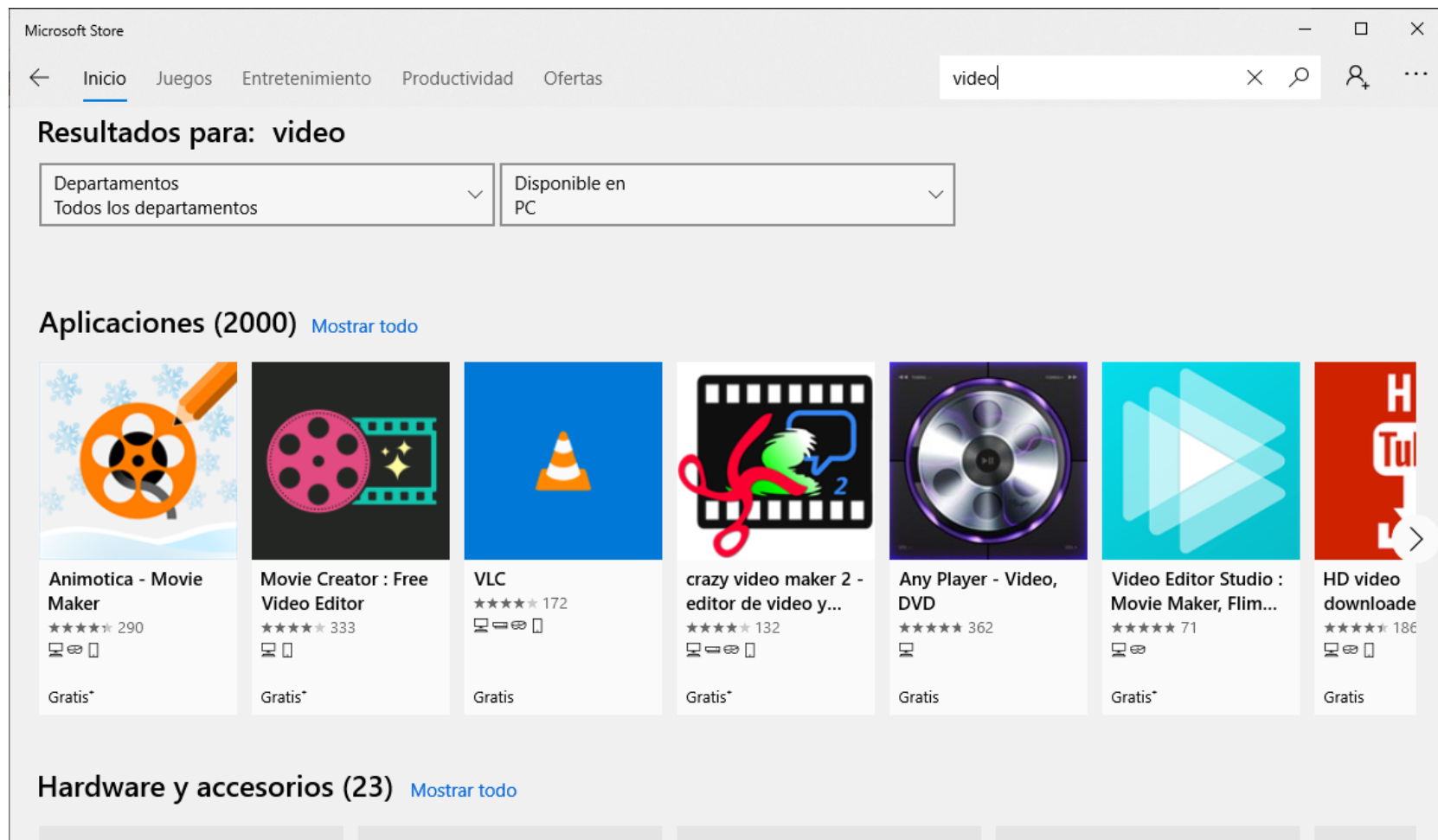
1. Instalación de software
2. Identificación del Software instalado
3. Desinstalación de software

5. Identificación del Sw instalado.

Instalación

- Para instalar software en Windows se pueden usar “**programas instaladores**” que se encargan, por sí mismos, de instalar el software, sus dependencias (bibliotecas que necesita para funcionar correctamente), así como de configurarlo en el registro de Windows.
- Existen un tipo de paquete con extensión **.msi** que funcionan de la misma forma. Necesitan un instalador.
- Otra forma muy común es usar la tienda de Windows (**Microsoft Store**) que ofrece gran cantidad de software categorizado. Permite realizar búsquedas de programas, comprarlos, descargarlos e instalarlos directamente desde la misma tienda.

5. Identificación del Sw instalado. Instalación

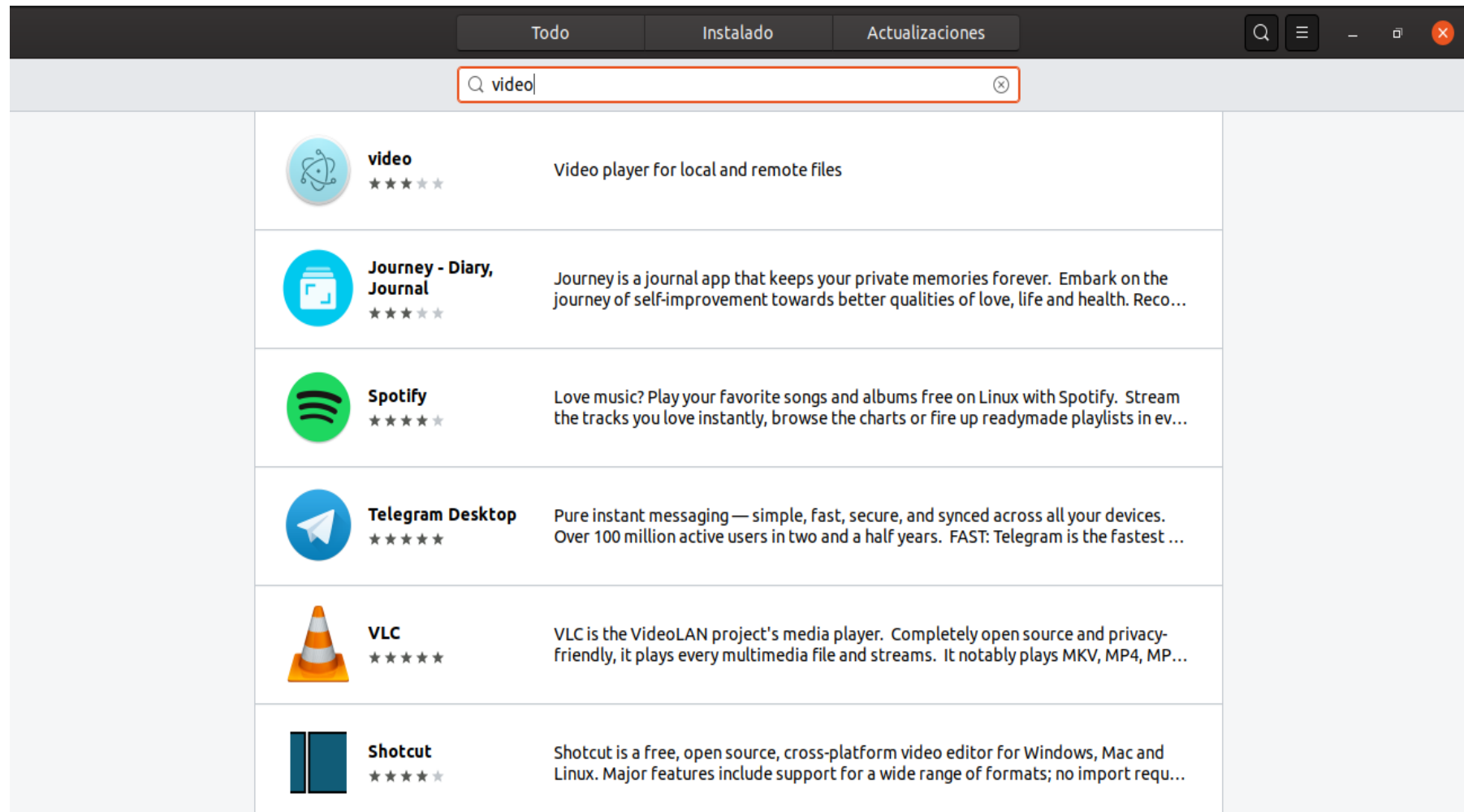


5. Identificación del Sw instalado.

Instalación

- Para instalar software en Linux Desktop se suele usar un programa gestor de paquetes gráfico. Depende de la distribución y el “**sabor**” de esta que estemos utilizando puede variar el programa y o la distribución.
- Uno de los gestores más famosos es Synaptic pero, con el tiempo ha perdido popularidad en favor de los gestores específicos de cada sabor de las distribuciones.
- Normalmente se instala por defecto en aquellas versiones que no llevan un gestor de paquetes preinstalado.
- Permite buscar, descargar, instalar y desinstalar software directamente desde el mismo gestor de paquetes. Lo que facilita la tarea para usuarios sin conocimientos de Linux

5. Identificación del Sw instalado. Instalación



5. Identificación del Sw instalado.

Instalación

- Para instalar programas desde la consola de Linux debemos usar un gestor de paquetes o un instalador. Hay varios tipos de gestores dependiendo de la rama de la que dependa la distribución que se está usando.
- Los más conocidos son los gestores basados en paquetes **deb** usados en las distribuciones que derivan de **Debian**, los basados en paquetes **rpm** usados en las derivadas de **Red Hat** y los basados en **pkgbuilds** usados por el gestor **pacman** en las derivadas de **Arch**
- De los tres, los más conocidos son los dos primeros aunque todos ellos funcionan de forma similar. Se pide el programa y el gestor lo busca, lo descarga y lo instala

5. Identificación del Sw instalado.

Instalación

- Lo primero que hay que hacer **siempre** es actualizar la lista de paquetes que hay en los repositorios para poder descargar la última versión:
- **sudo apt update**
- A partir de este comando tendremos una lista completa con la última versión de los paquetes del repositorio
- Para buscar un paquete por palabra clave usamos:
- **sudo apt search <palabras clave>**
- Si queremos instalar o actualizar algún paquete:
- **sudo apt install <nombre del paquete>**
- Si queremos actualizarlos todos:
- **sudo apt upgrade**

5. Identificación del Sw instalado.

Instalación

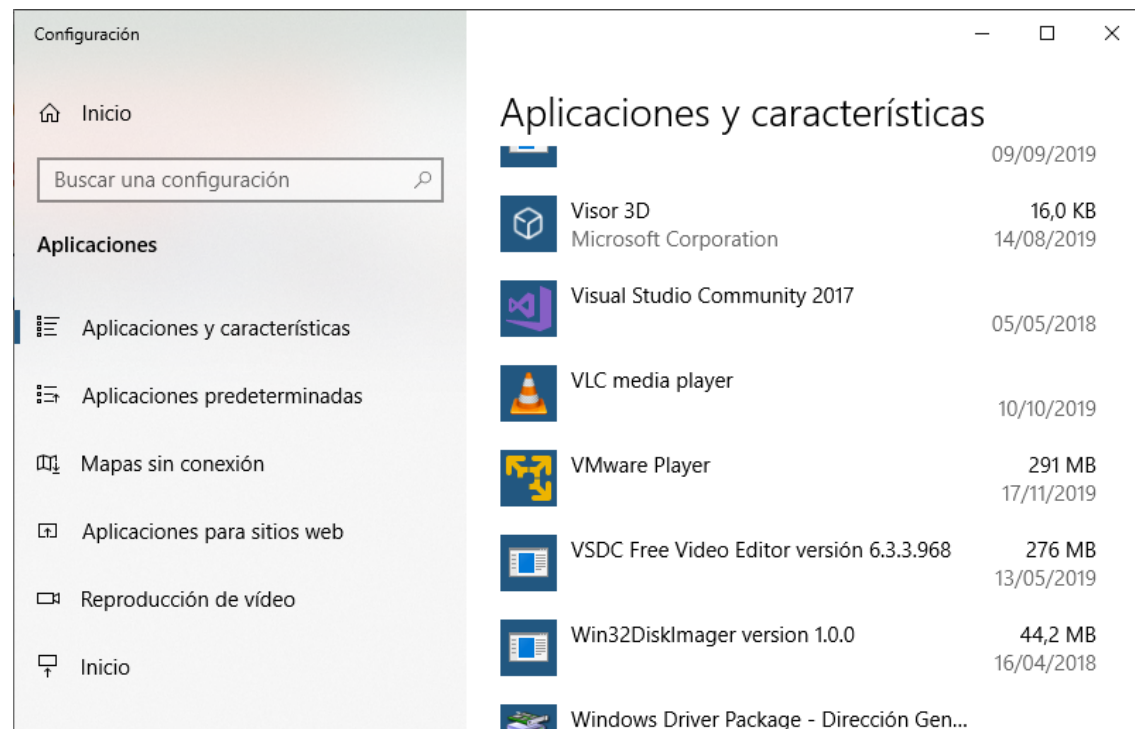
- Puede ser que, en ocasiones, no tengamos un repositorio o que el paquete haya que instalarlo “manualmente”
- Para añadir un repositorio, (suele ser una cadena de texto con el formato “**deb <servidor> <version_distribucion> <seccion>**”) añadimos esa cadena al fichero: **/etc/apt/sources.list**
- También podemos descargar el fichero .deb e instalarlo manualmente utilizando: **sudo dpkg -i paquete.deb**
- Una forma todavía más rudimentaria todavía es descargarnos las fuentes, compilarlo e instalarlo. Nos descargamos el fichero. Lo descomprimos. En este directorio ejecutamos **./configure** Lo compilamos con **make** y, por último, lo instalamos con **sudo make install**

5. Identificación del Sw instalado.

Identificación del software

- En Windows 10, para ver los programas instalados pulsamos con el botón derecho del ratón en el botón inicio de Windows y elegimos “**Aplicaciones y características**”

- En la parte de la derecha podremos ver las aplicaciones instaladas tanto algunas propias de Windows como las nuestras



5. Identificación del Sw instalado.

Identificación del software

- En Linux, la forma más cómoda de ver el listado de paquetes es a través de la consola
- Utilizaremos el comando (con l minúscula):
- **dpkg -l**
- Si usamos el mismo comando (con L mayúscula) seguido de un paquete en concreto, nos muestra todos los ficheros que se instalaron con la aplicación:
- **dpkg -L <nombre del paquete>**
- Si utilizamos la opción **-S** seguida de un texto o nombre de paquete o fichero nos mostrará las aplicaciones que están usando ese fichero. Es muy útil para conocer los programas que tienen una dependencia concreta

5. Identificación del Sw instalado.

Desinstalación

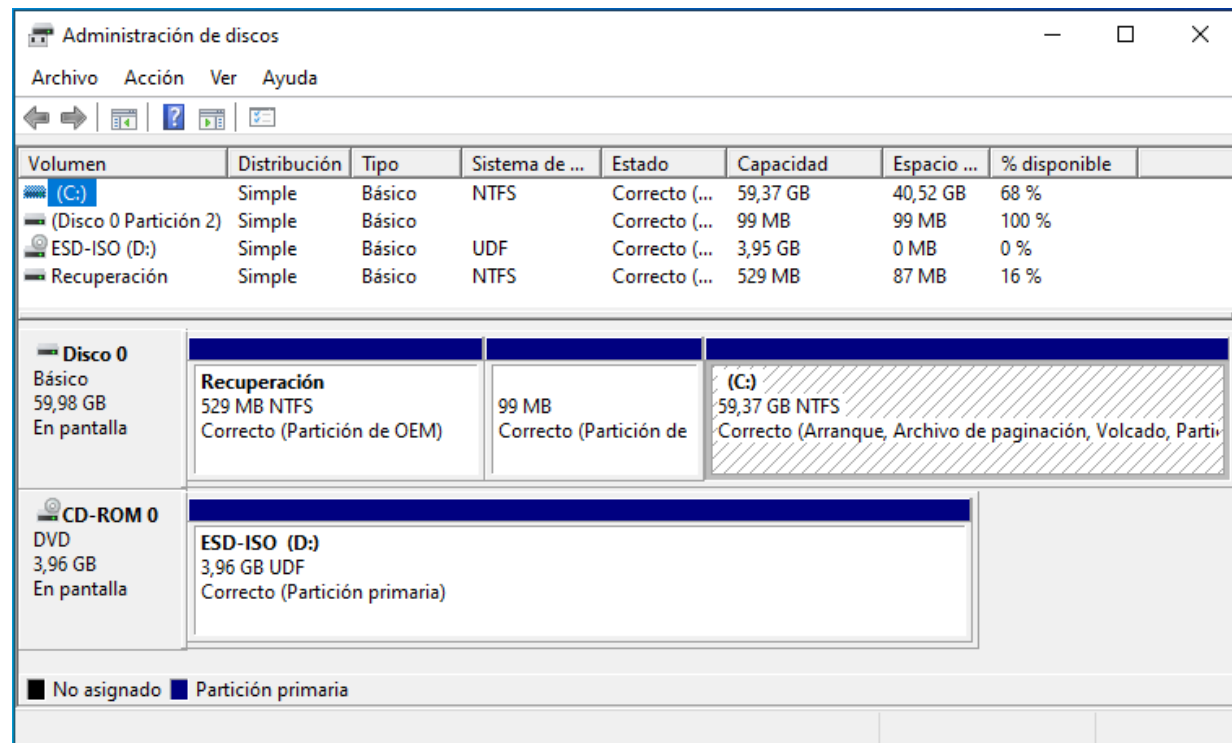
- Tanto en Windows como en Linux con entorno gráfico, la forma más sencilla de desinstalar una aplicación es accediendo al gestor de aplicaciones/paquetes, seleccionar el programa que queremos **desinstalar** y pulsar la opción que nos ofrece realizar esta acción.
- En la consola, por comandos utilizaremos:
- **sudo apt remove <paquete a desinstalar>**
- En ocasiones, los programas dejan “restos”, por ejemplo dependencias que necesitaban o ficheros de configuración. Para eliminar gran parte de estos restos, sobre todo los ficheros de configuración:
- **sudo apt remove --purge <paquete a desinstalar>**

6. HERRAMIENTAS DE ADMINISTRACIÓN DE DISCOS. PARTICIONES Y VOLÚMENES. DESFRAGMENTACIÓN Y CHEQUEO

1. Herramientas de administración de discos.
2. Particiones y volúmenes.
3. Desfragmentación
4. Chequeo.

6. Herramientas de admin. de discos

- En Windows, pulsando con el botón derecho del ratón sobre el botón inicio del sistema, desplegamos un menú y elegimos “Administración de discos” mostramos la pantalla donde se muestran los discos instalados así como sus particiones y, si es el caso, el espacio libre.

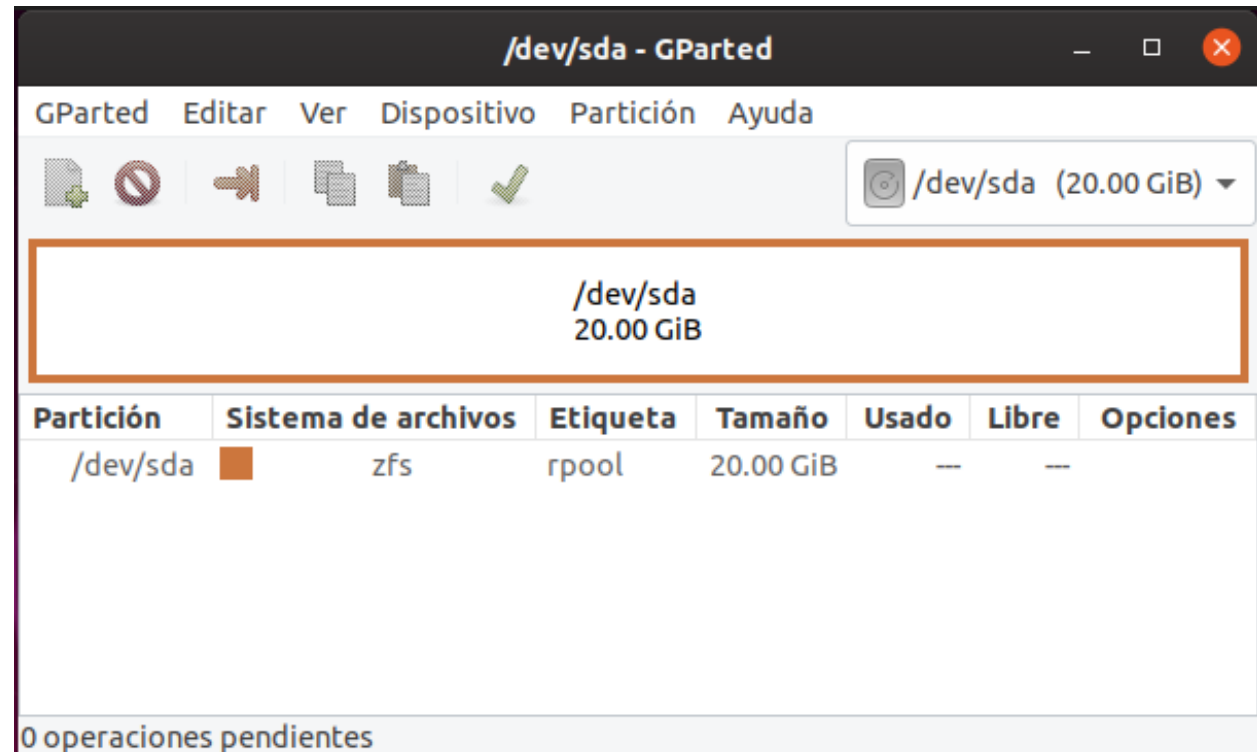


6. Herramientas de admin. de discos

- En la versión de escritorio de Linux se suelen usar programas orientadas a la administración de discos.
- Suele ser necesario tener que instalarlos ya que no es un software que

se utilice muy a menudo.

Uno de los más famosos es **Gparted**



6. Herramientas de admin. de discos

- Si se quieren gestionar las particiones en consola, el comando más utilizado es **fdisk**, especialmente en Linux ya que en Windows 10 se ha reemplazado por una nueva aplicación llamada **diskpart**
- En el primer caso, para ver las particiones disponibles junto con los datos asociadas a ellas utilizamos:
- **sudo fdisk -l**
- En el segundo hay que lanzar el programa diskpart
- Una vez dentro del programa ejecutamos el comando:
- **list disk** para mostrar los discos
- **list volume** para mostrar los volúmenes y particiones disponibles en el sistema

6. Particiones y volúmenes

- Físicamente los usuarios solamente tienen discos (ya sean CD/DVD, discos duros de rotación, SSD, USB, ...)
- Pero “**lógicamente**”, para el sistema operativo y para los usuarios esta estructura es transparente ya que trabajan con particiones y/o volúmenes (depende del sistema) en las que se divide este disco duro físico.
- Una **partición** es una división de un disco. Hay varios tipos de particiones:
 - **Primaria**: Permite instalar y arrancar un sistema operativo
 - **Extendida**: Admite crear varias particiones lógicas dentro
 - **Lógica**: particiones para almacenar datos en general
 - **Activa**: Es la partición primaria que tenemos arrancada

6. Particiones y volúmenes

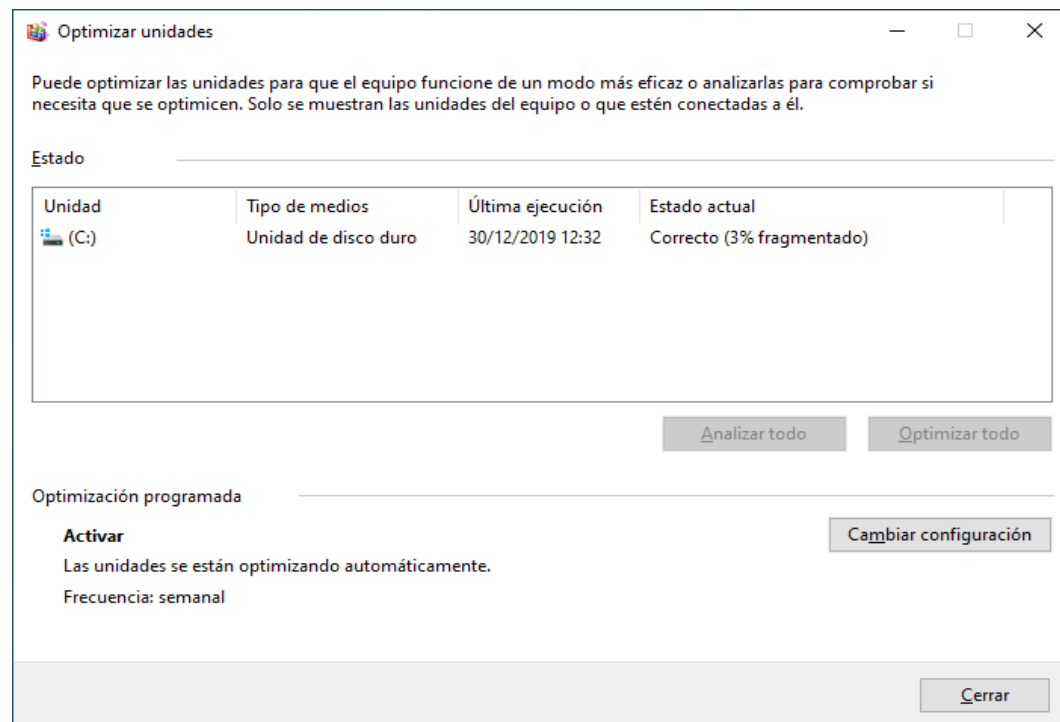
- Puede ser que tengamos particionado un disco pero no hayamos formateado todavía todas las particiones
- Un volumen, término utilizado especialmente en Windows, es una partición que ha sido formateada utilizando un sistema de ficheros determinado.
- Solamente se podrá acceder (salvo con programas especiales) a estos datos si la partición está formateada, es decir, si la hemos convertido en un volumen utilizable.
- En Windows, se puede observar que, en el **equipo** aparecen los **volúmenes** con los que podemos trabajar, sin embargo, si accedemos al **administrador de discos** la lista es más larga pues se incluyen particiones ocultas

6. Desfragmentación

- Un fichero cuyo tamaño sea mayor del tamaño de cluster, debe almacenarse en dos o más clústers. Siempre que se puede el sistema operativo usa espacios adyacentes. Con el uso (creación y borrado de ficheros) del disco se irán necesitando bloques que fueron usados y, en algún momento, quedaron disponibles de nuevo.
- Esta separación de las diferentes partes de un fichero en varios clústers se llama **fragmentación de fichero**. Los ficheros fragmentados provocan fragmentación de disco
- Esta fragmentación ralentiza el sistema si hay muchas operaciones de disco cuando los ficheros tienen partes muy separadas entre sí debido a los movimientos de giro del disco como de las agujas de dentro a afuera.

6. Desfragmentación

- En Windows, para desfragmentar el disco, pulsamos en el menú inicio (con el botón izquierdo) y, dentro del menú **“Herramientas administrativas de Windows”** podemos encontrar **“Desfragmentar y optimizar unidades”**
- El desfragmentador está configurado para ejecutarse automáticamente aunque podemos hacer un análisis de forma manual



6. Desfragmentación

- Existe una **falsa creencia** de que en Linux no aparece la fragmentación de discos. Esto es debido a que los sistemas de ficheros utilizados almacenan la información de forma que la fragmentación tarda más en producirse
- Existen diferentes programas para consola, como **e4defrag**, que permiten la desfragmentación de discos en Linux. En este caso habría que indicarles el dispositivo “**en bruto**” almacenado en el directorio **/dev**
- Es importante recordar que, si los discos duros son SSD, **NO** es conveniente **desfragmentar** porque genera operaciones de escritura innecesarias (que estropearían el disco con el tiempo) y no lo optimizarían porque ya tarda el mismo tiempo en acceder a todos los clústers.

6. Chequeo

- En ocasiones podemos tener problemas en algún disco por diferentes motivos, siendo los más comunes los ocasionados por apagar/reiniciar el ordenador sin un apagado seguro (desconectando la corriente) o por desconectar un dispositivo sin haberlo **desmontado** previamente ya que puede haber ficheros abiertos y su información quedaría corrompida.
- Para acceder al chequeo en Windows pulsamos con el botón derecho sobre el dispositivo que queremos chequear y seleccionamos **Propiedades**. Posteriormente elegimos la pestaña **Herramientas** y, ahí, el botón **Comprobar** y “**Analizar unidad**”

6. Chequeo

- En Linux, existen diferentes programas, especialmente para la consola. El más utilizado para chequear discos es **e2fsck** que permite tanto chequear el disco como intentar reparar los sectores que se hayan detectado defectuosos
- Es muy **importante** ver la ayuda, ya sea con la opción **-h**, con la opción **–help** o con el comando **man**, en este tipo de programas porque en muchos casos nos indica, no solo cómo hay que usar el comando sino también cómo hay que tener conectados los dispositivos. Hay muchos programas en los que es necesario que **NO** estén **montados** porque podría suponer estropearlo más e incluso llegar a perder los datos contenidos previamente



7. Montaje y desmontaje de dispositivos en ss00

1. Montaje y desmontaje de dispositivos en sistemas operativos

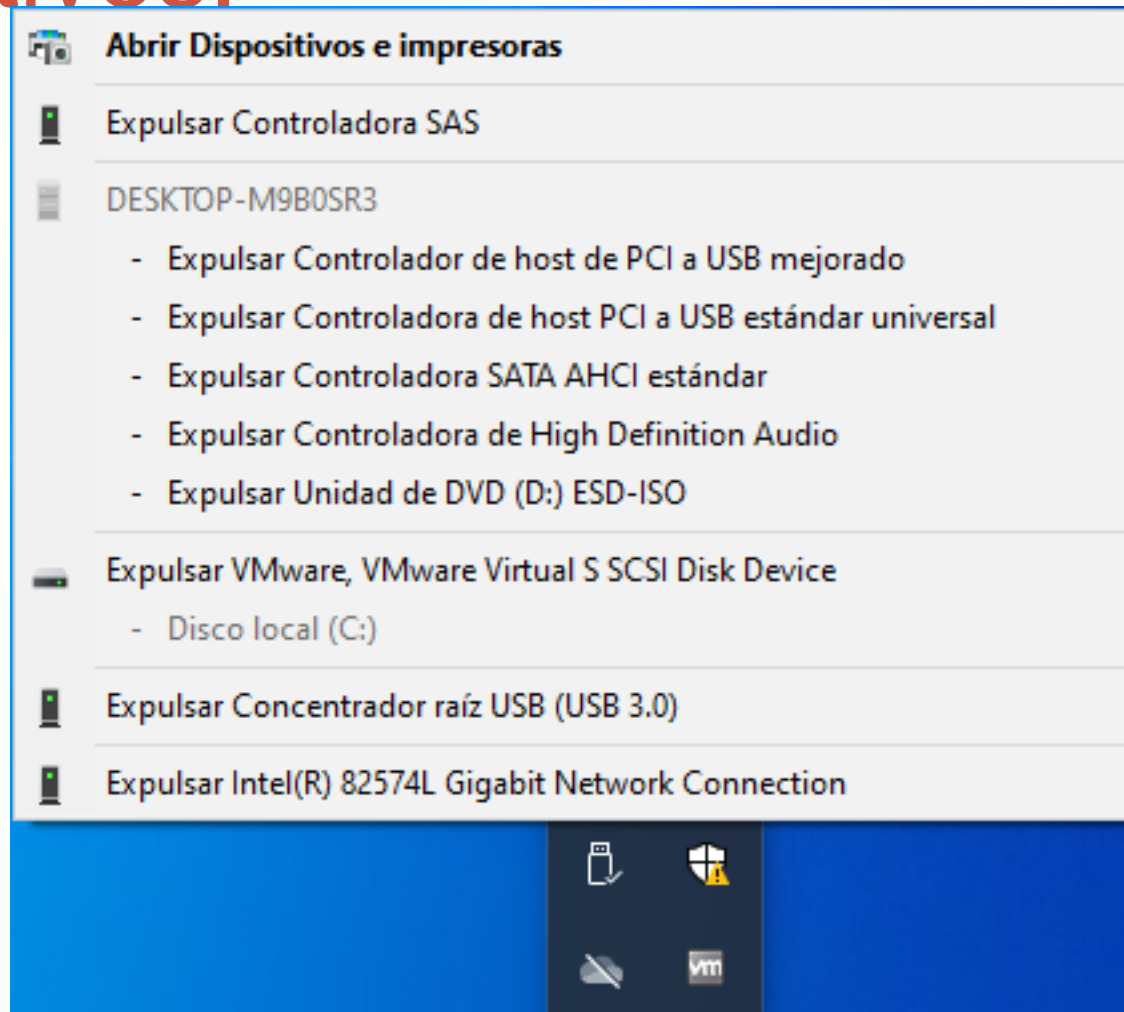
7 Montaje y desmontaje de dispositivos en sistemas operativos.

- Un volumen de un disco no es accesible por el sistema si no está montado previamente. Todos los sistemas operativos necesitan montar los volúmenes antes de poder acceder a los datos de sus sistemas de ficheros.
- El proceso de montaje del dispositivo incluye no solo la preparación del sistema de ficheros del volumen sino también la inclusión en el sistema para que sea accesible de forma transparente por parte de los usuarios.
- Por norma general, los sistemas operativos con interfaz gráfica montan de forma automática los dispositivos que tienen conectados y dispositivos externos que se puedan conectar durante el uso del sistema.

7 Montaje y desmontaje de dispositivos en sistemas operativos.

- En la consola se utilizará el comando **mount** para montar una partición en el sistema:
- **sudo mount /dev/<dispositivo> <punto de montaje>**
- En ocasiones se pedirá incluir el tipo de partición que se va a montar. En estos casos se utilizará la opción **-t** seguida del tipo sistema de ficheros que se va a montar.
- Para **desmontar** un dispositivo se usa el comando **umount** seguido del dispositivo que se va a desmontar.
- En Windows, como en otros sistemas operativos, con los discos externos, hay que realizar la extracción segura **siempre**, que no es otra cosa que el desmontaje del dispositivo que le indiquemos

7 Montaje y desmontaje de dispositivos en sistemas operativos.





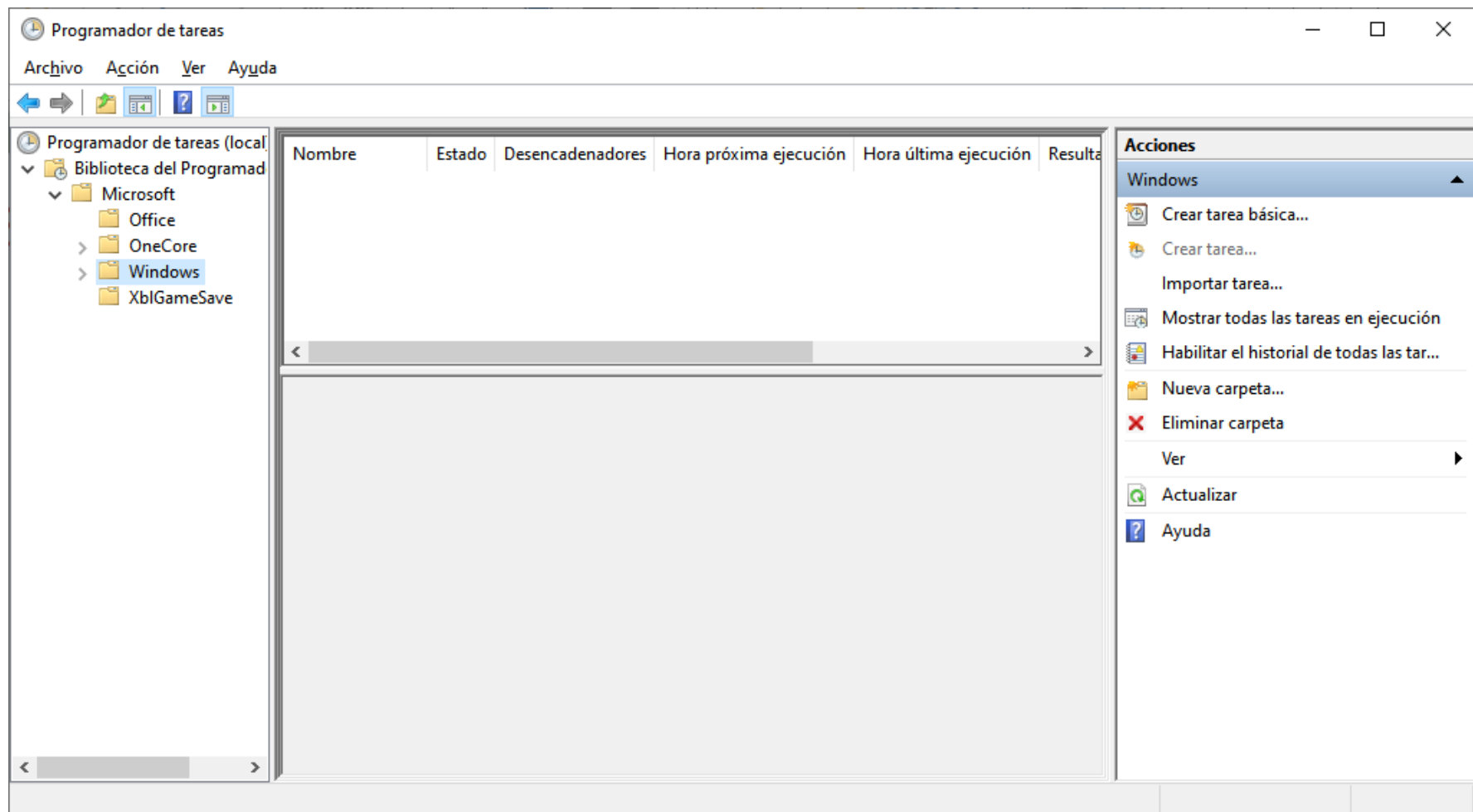
8. TAREAS AUTOMÁTICAS

1. Tareas automáticas

8. Tareas automáticas.

- Para automatizar el sistema se utilizan los programadores de tareas. En ellos se indican diferentes elementos como un evento que desencadenará el lanzamiento de la tarea o la hora en la que queremos programarla junto con el programa o **script** (lista de acciones) que se ejecutarán.
- En Windows se utiliza el programador de tareas al que se accede pulsando en el menú **inicio – herramientas administrativas de Windows – Programador de tareas**
- En el programa que aparece, a la izquierda, se puede ver un listado de las tareas que ya están programadas por diferentes aplicaciones.
- En la parte de la derecha se pueden crear más tareas

8. Tareas automáticas.



8. Tareas automáticas.

- Al crear una nueva tarea debemos especificar para qué usuarios va a ser posible ejecutarse, los privilegios que tendrá además de cómo y cuando se va a lanzar

The screenshot shows the 'Crear tarea' (Create Task) dialog box with the 'General' tab selected. The fields are as follows:

- Nombre:** (Empty text box)
- Ubicación:** \Microsoft\Windows
- Autor:** DESKTOP-BOSSDQC\Diego J. García
- Descripción:** (Empty text box)

Opciones de seguridad

- ☒ Al ejecutar la tarea, usar esta cuenta de usuario:
L50\Diego J. García Cambiar usuario o grupo...
- ☒ Ejecutar solo cuando el usuario haya iniciado sesión
- ☐ Ejecutar tanto si el usuario inició sesión como si no
 - ☐ No almacenar contraseña. La tarea solo tendrá acceso a los recursos del equipo local.
- ☐ Ejecutar con los privilegios más altos

☐ Oculta Configurar para: Windows 10

Aceptar Cancelar

8. Tareas automáticas.

- Los desencadenadores nos indicarán “**cuándo**” se va a realizar la tarea que estamos programando. Por defecto se indicará una fecha y una periodicidad por ser el desencadenador más común de las programaciones.

Nuevo desencadenador

Iniciar la tarea: **Según una programación**

Configuración

☒ Una vez

☐ Diariamente

☐ Semanal

☐ Mensual

Al iniciar la sesión

Al iniciar el sistema

Al estar inactivo

Al producirse un evento

Al crear o modificar tarea

Al conectarse a una sesión de usuario

Al desconectarse de una sesión de usuario

Al bloquearse la estación de trabajo

Al desbloquearse la estación de trabajo

☐ Sincronizar zonas

Configuración avanzada

☐ Retraso máx. (retraso aleatorio): 1 hora

☐ Repetir cada: 1 hora durante: 1 día

☐ Detener todas las tareas en ejecución al final de la duración de repetición

☐ Detener la tarea si se ejecuta durante más de: 3 días

☐ Expiración: 30/12/2020 18:51:42

☐ Sincronizar zonas horaria

☒ Habilitado

Aceptar Cancelar

Crear tarea

General Desencadenadores Acciones Condiciones Configuración

Quando se crea una tarea, se pueden especificar las condiciones que la activarán.

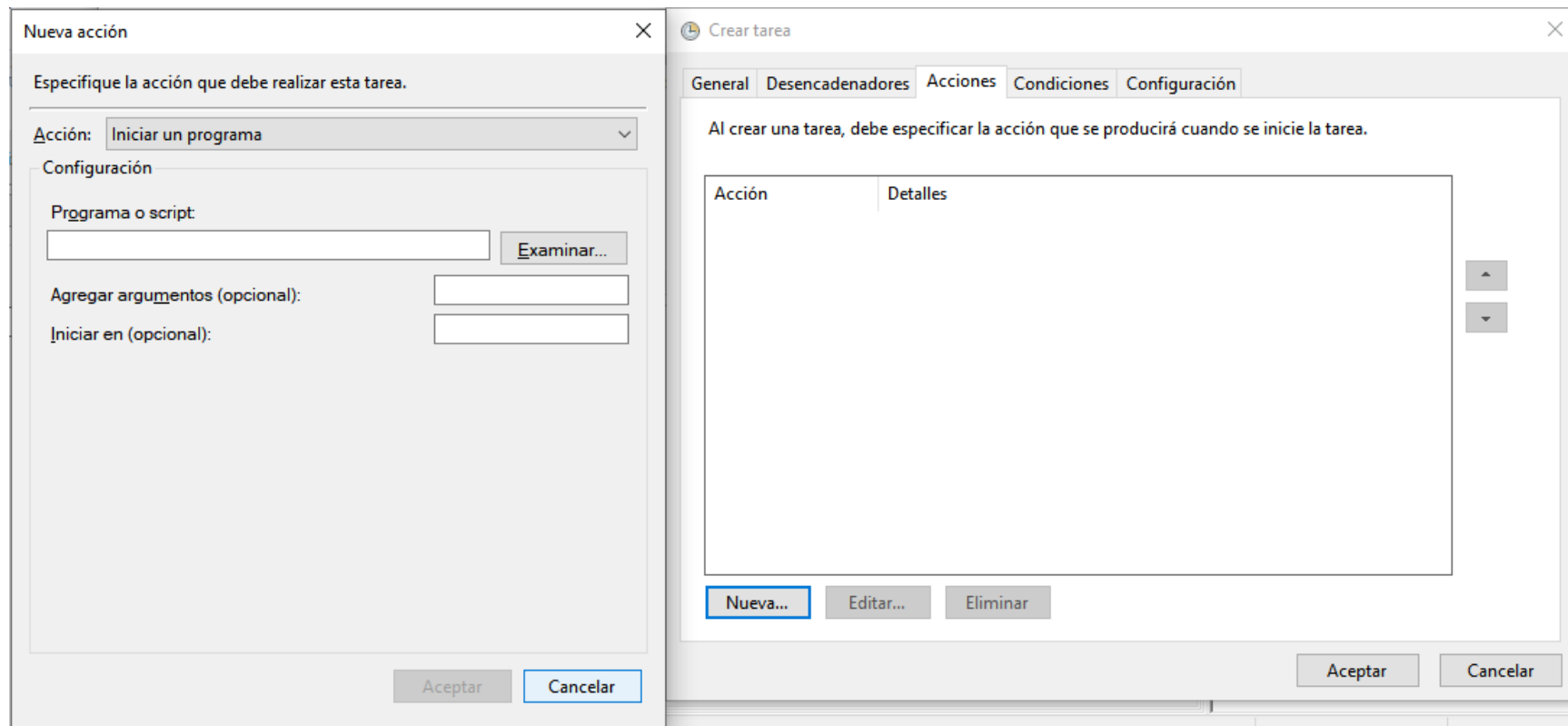
Desencadenador	Detalles	Estado
----------------	----------	--------

Nuevo... Editar... Eliminar

Aceptar Cancelar

8. Tareas automáticas.

- Las acciones son la lista de programas o scripts que vamos a ejecutar cuando se lance esta tarea



8. Tareas automáticas.

- Las condiciones y configuración son diferentes formas de modificar el comportamiento del lanzamiento de la tarea si se producen ciertas circunstancias en el sistema antes o mientras se ejecuta la tarea indicada.

The image displays two side-by-side screenshots of the 'Crear tarea' (Create task) dialog box in Windows Task Scheduler, illustrating the 'Condiciones' (Conditions) and 'Configuración' (Configuration) tabs.

Left Screenshot (Condiciones tab):

- General:** Desencadenadores, Acciones, **Condiciones**, Configuración.
- Text:** Especifique las condiciones que, junto a los desencadenadores, determinarán si se debe ejecutar la tarea. Si alguna de las condiciones especificadas no se cumple, no se ejecutará la tarea.
- Inactivo:**
 - ☐ Iniciar la tarea solo si el equipo está inactivo durante: 10 minutos
 - ☐ Esperar a que esté inactivo durante: 1 hora
 - ☒ Detener si el equipo deja de estar inactivo
 - ☐ Reiniciar si el estado de inactividad se reanuda
- Energía:**
 - ☒ Iniciar la tarea solo si el equipo está conectado a la corriente alterna
 - ☒ Detener si el equipo empieza a usar la batería
- Red:**
 - ☐ Activar el equipo para ejecutar esta tarea
 - ☐ Iniciar solo si la siguiente conexión de red está disponible: Cualquier conexión
- Buttons:** Aceptar, Cancelar.

Right Screenshot (Configuración tab):

- General:** Desencadenadores, Acciones, Condiciones, **Configuración**.
- Text:** Especifique la configuración adicional que afecta al comportamiento de la tarea.
- ☒ Permitir que la tarea se ejecute a petición
- ☐ Ejecutar tarea lo antes posible si no hubo inicio programado
- ☐ Si la tarea no se ejecuta, reiniciarla cada: 1 minuto
- ☐ Intentar el reinicio un máximo de: 3 veces
- ☒ Detener la tarea si se ejecuta durante más de: 3 días
- ☒ Detener tarea en ejecución si no finaliza cuando se solicite
- ☐ Eliminar tareas no reprogramadas después de: 30 días
- Text:** Aplicar la siguiente regla si la tarea ya está en ejecución:
- ☐ No iniciar una instancia nueva
- Buttons:** Aceptar, Cancelar.

8. Tareas automáticas.

- En Linux, se utiliza el **cron**. Un “**demonio**” que permite programar tareas en el sistema desde la consola.
- No hay que confundir el servicio cron con el **crontab** que es un fichero (una tabla) que programa al cron.
- Para ver las tareas programadas usamos **crontab -l**
- Para cambiar la lista de tareas (hay que tener en cuenta que la cambia completa) **crontab <lista_tareas.txt>**
- Para borrar la lista de tareas **crontab -d**
- El fichero con la lista de tareas es un fichero de texto que se puede editar con un editor de texto plano que tiene una tarea por línea y que se ejecutará con la periodicidad que le indiquemos según el siguiente formato:

8. Tareas automáticas.

- El fichero crontab tiene una tarea por cada línea que se programa con cinco “tiempos” diferentes. Si hay un asterisco significa que se ejecutará “siempre”.
- Si ponemos un valor en alguno de ellos se programará la tarea en ese momento.

```
*      *      *      *      *      usuario  ruta_al_script_o_programa  opciones argumentos
|      |      |      |      |
|      |      |      |      |      ---> Día de la semana (0 Domingo - 6 Sábado)
|      |      |      |      |      -----> Mes (1 Enero - 12 Diciembre)
|      |      |      |      |      -----> Día del mes (1 - 31)
|      |      |      |      |      -----> Hora (0 - 23)
|      |      |      |      |      -----> Minuto (0 - 59)
```

8. Tareas automáticas.

- Existen ciertos valores especiales que le podemos indicar al fichero para “ahorrar” escribir demasiados valores o líneas dentro de este fichero.
- El carácter / seguido de un valor indica “**cada valor**” por ejemplo /5 en el campo minutos sería cada 5 minutos
- Se pueden poner varios valores separados por comas indicando que se ejecutaría en cada uno de ellos
- También se pueden expresar rangos utilizando el guión de forma que se indicaría “**desde_inicio**”-“**hasta_final**”
- En algunos casos, incluso se pueden llegar a realizar combinaciones de los valores especiales anteriores.

8. Tareas automáticas.

- Es conveniente tener en cuenta la fecha y hora del sistema, especialmente cuando se van a programar tareas. En caso contrario se pueden lanzar tareas en momentos inadecuados o registrar eventos en momentos erróneos lo que puede tener consecuencias negativas.
- **date** nos muestra la fecha y hora del sistema. Si se usan “cadenas de formato” se pueden mostrar más o menos datos de los que se muestran por defecto.
- **cal** nos muestra un “calendario” del mes actual. Algunas opciones interesantes son la opción **-3** que muestra el mes pasado, el actual y el siguiente. **cal [[mes] año]** que nos muestra el calendario de un año y/o mes concreto. Por cierto, ¿Qué pasa con septiembre de 1752?

9. Tolerancia a fallos

1. RAID
2. RAID 0
3. RAID 1
4. RAID 5
5. RAID en Linux

9. Tolerancia a fallos. RAID

- Los sistemas operativos, con el tiempo y por diferentes motivos, pueden tener fallos que hagan que se puedan perder datos.
- Por ello se ofrecen diversas soluciones para aumentar la tolerancia a fallos que permita que, aunque se produzcan errores, sea transparente para el usuario y, si no lo es, se pueda recuperar el sistema rápidamente.
- La opción más socorrida es el uso de RAID (Redundant Array of Independent Disks) donde se permite que se almacenen los datos de forma que, si hay alguna pérdida se puedan recuperar.
- Hay varios niveles de configuración de RAID

9. Tolerancia a fallos. RAID 0

- Para crear los RAID en Windows se debe usar el administrador de discos del sistema.
- Un RAID 0 permite unir varios discos en un solo volumen. Permite sumar los espacios de los dos discos en un solo pero **no es tolerante a fallos** ya que si un disco falla se pierden **todos** los datos del volumen.
- Es necesario dos discos (o particiones de discos) de tamaños **exactamente** iguales.
- En el administrador elegimos uno de los discos y elegimos **Nuevo volumen seccionado...**
- En el asistente nos pedirá que elijamos el/los discos que queremos incorporar en el nuevo volumen RAID 0

9. Tolerancia a fallos. RAID 1

- Un RAID 1 permite hacer una copia exacta del contenido de un disco en otro disco. Este tipo de RAID se suele llamar **mirroring** y **disco espejo** al segundo disco.
- Solamente se utiliza el espacio de un disco para el almacenamiento del usuario ya que el otro es de respaldo
- Si un disco falla el sistema sigue funcionando hasta que se reemplace el disco y se reconstruya el RAID 1
- Igual que en el caso anterior los discos deben ser iguales.
- En el administrador elegimos uno de los discos y elegimos **Nuevo volumen reflejado...**
- En el asistente nos pedirá que elijamos el/los discos que queremos incorporar en el nuevo volumen RAID 1

9. Tolerancia a fallos. RAID 5

- El RAID 5 es una mezcla de los dos anteriores. Permite ampliar el espacio disponible perdiendo menos espacio.
- Se necesitan, al menos, 3 discos para la creación y el espacio disponible será el del tamaño de los discos multiplicado por el número de discos menos 1.
- Este tipo de RAID no se gestiona desde el administrador de discos sino desde **Espacios de almacenamiento** que se encuentran dentro del apartado **Sistema y seguridad** del **Panel de control**
- En el **Administrador de espacios** podemos crear un nuevo grupo en el que nos aparecen las unidades que no han sido formateadas todavía con las que se puede **crear un grupo** y definir el nivel de tolerancia a fallos

9. Tolerancia a fallos. RAID en Linux

- Para crear un RAID en Linux, usaremos el programa **mdadm** desde la terminal. Es necesario instalarlo.
- Una vez instalado podemos ver que ha aparecido un nuevo fichero llamado **/proc/mdstat** que, si miramos su contenido veremos que solo tiene 2 líneas sin contenido.
- Para crear un RAID 0 debemos indicar que hay que **crear** un nuevo dispositivo **/dev/mdx** donde x es el número de dispositivo RAID que estamos creando, será secuencial. El **número de dispositivos** que conforman el RAID, el **nivel** del RAID y la lista de dispositivos que lo conforman
- **mdadm --create /dev/md0 --raid-devices=2 --level=0 /dev/sdb /dev/sdc**
- Podemos comprobar que **/proc/mdstat** ha cambiado

9. Tolerancia a fallos. RAID en Linux

- El RAID creado está inicialmente **en bruto**, es decir, no se puede usar. Como en cualquier disco nuevo hay que formatearlo con un nuevo sistema de ficheros y montarlo.
- Para formatearlo usamos **mkfs -t ext4 /dev/md0**
- A continuación lo montamos **mount /dev/md0 /mnt/raid**
- Con el comando **df** podemos ver el espacio disponible y podremos comprobar que, al ser un RAID 0 el espacio es el de los dos discos que hemos usado juntos.
- Para eliminar un RAID debemos (desmontarlo si está montado previamente) pararlo y eliminarlo
- **mdadm --stop /dev/md0**
- **mdadm --remove /dev/md0**

9. Tolerancia a fallos. RAID en Linux

- Cuando el RAID que vamos a montar es de nivel 1 puede ser que nos aparezca un mensaje indicando que alguno de los discos forma parte de una RAID. Es conveniente comprobar que no estamos cometiendo errores ya que se corromperían los datos si así fuera.
- El único cambio del comando anterior es el nivel
- **mdadm --create /dev/md0 --raid-devices=2 --level=1 /dev/sdb /dev/sdc**
- Para crear un RAID 5 hay que tener en cuenta que el número de dispositivos es 3 (mínimo) y el nivel 5
- **mdadm --create /dev/md0 --raid-devices=3 --level=5 /dev/sdb /dev/sdc /dev/sdd**



10. COPIAS DE SEGURIDAD

1. Copias de seguridad

10. Copias de seguridad.

- Un .



11. Recuperación del sistema

1. Recuperación del sistema

11. Recuperación del sistema.

- Un .