Contenidos

- Introducción
- Preparación para la instalación de sistemas operativos
- Administración de software en sistemas Linux
  - Distribuciones Linux según administración del software
  - Administración de software en distribuciones Debian/Ubuntu
  - Administración de software en distribuciones RPM
- Instalación y administración de software en Windows
  - Utilidades de administración de software en Windows
  - Utilidades de administración de software en Windows
  - Características y Roles
  - Microsoft Store
  - Comparativa de opciones de instalación de software
  - Instalaciones desatendidas en Windows
    - Instalación de paquetes .msi con el comando msiexec
    - Gestor de software Chocolatey

Introducción

En esta unidad veremos como obtener, instalar y administrar el software en distintos sistemas operativos:

- Veremos que son los ficheros ISO, como los obtendremos y grabaremos en soportes físicos de almacenamiento para lanzar el programa de instalación del sistema operativo o simplemente para lanzar sistemas operativos Live CD. No entraremos en detalle del proceso de instalación de estos, al estar cubierto en el módulo de Implantación de Sistemas Operativos.
- Veremos las formas en las que se distribuye el software en sistemas Linux y las herramientas que estos nos proporcionan para administrarlo.
- Veremos como instalar software de forma manual en Windows y las herramientas que este nos proporciona para administrarlo.
- Para acabar veremos formas de instalar software en Windows de formas desatendida

Preparación para la instalación de sistemas operativos

Imágenes o Ficheros .ISO

Para instalar un sistema operativo necesitamos una imagen o fichero ISO, que es un tipo de archivo que se utiliza para almacenar **una copia exacta de un sistema de ficheros de una unidad óptica**.

Este formato es útil para poder distribuir por Internet copias de sistemas operativos, ya que al grabarlo podemos reproducir tal cual el disco original a partir del cual se generó, típicamente un DVD.

Estos discos a partir del cual se general el fichero ISO tienen **capacidad de arranque (bootable)**, lo cual implica que según definamos el orden de arranque en la BIOS al llegar a este disco según el caso arrancará:

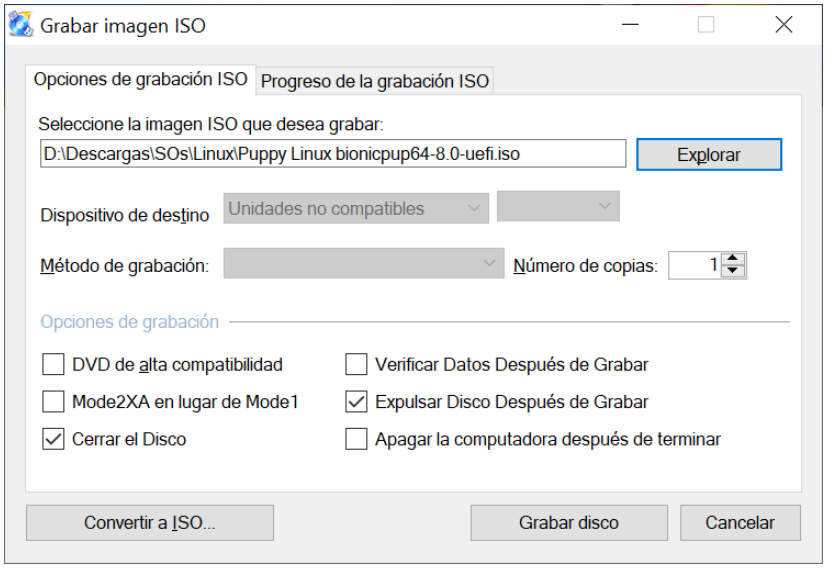
- **El programa de instalación** (Es el caso de las ISOs de los sistemas operativos Windows y algunos Linux). Algunas ISOs de Linux vienen en forma de Netinstall, que son ISOs de un tamaño menor con lo imprescindible para lanzar el programa de instalación y un paquete de software básico, necesitando una conexión a Internet para la descarga de software adicional.
- **Un sistema operativo Live CD**
  - Es un sistema operativo que se ejecuta directamente desde el disco, sin realizar una instalación en el disco duro (Es el caso de ISOs de algunos Linux pero también de sistemas operativos de rescate que hacen recopilación de utilidades de test de hardware y recuperación de datos)
  - Algunos Live CD dan la opción de lanzar un programa de la instalación en el disco duro

A esta alturas del curso ya habréis utilizado ficheros ISO en VirtualBox montándolos en unidades de DVD virtuales para instalar sistemas operativos en el módulo de “Implantación de Sistemas Operativos”, pero vamos a ver como sería el procedimiento para poder utilizar estas ISOs en un ordenador físico.

Grabación de ISO en soporte físico

En un ordenador físico tendremos que grabar o escribir el contenido de este fichero ISO en un dispositivo de almacenamiento físico, que puede ser:

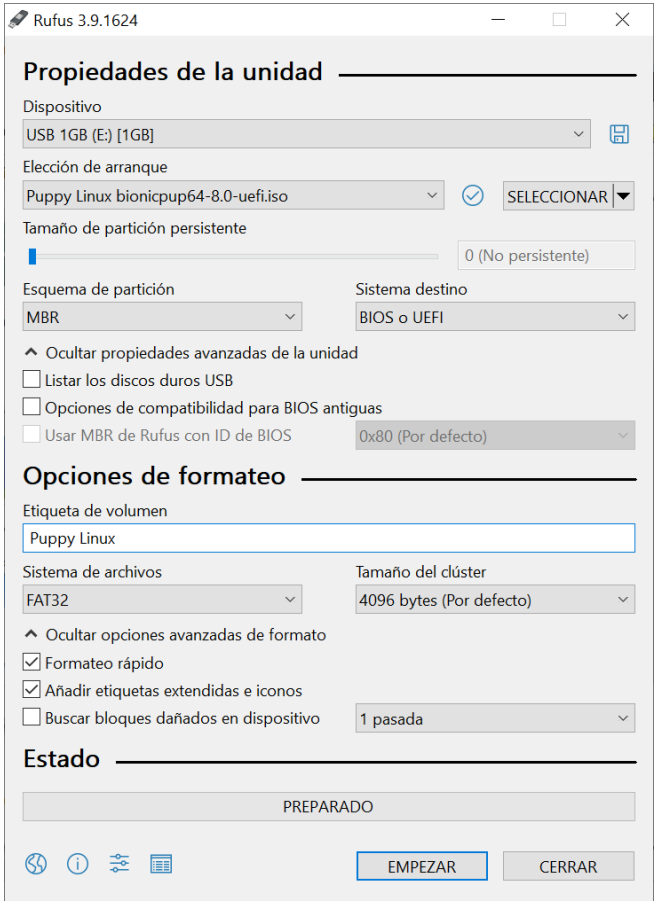
- **Un soporte óptico (CD o DVD)**, el que escribir el contenido del fichero ISO con algún software de grabación (Por ejemplo: Brasero en Linux o CDBurnerXP en Windows)
- **Un pendrive USB**, utilizando utilidades especializadas en crear "bootable USBs". Algunos ejemplos son Rufus (para sistemas Windows) o Unetbootin (Windows, Linux o macOS), pero otras muchas utilidades de este tipo.



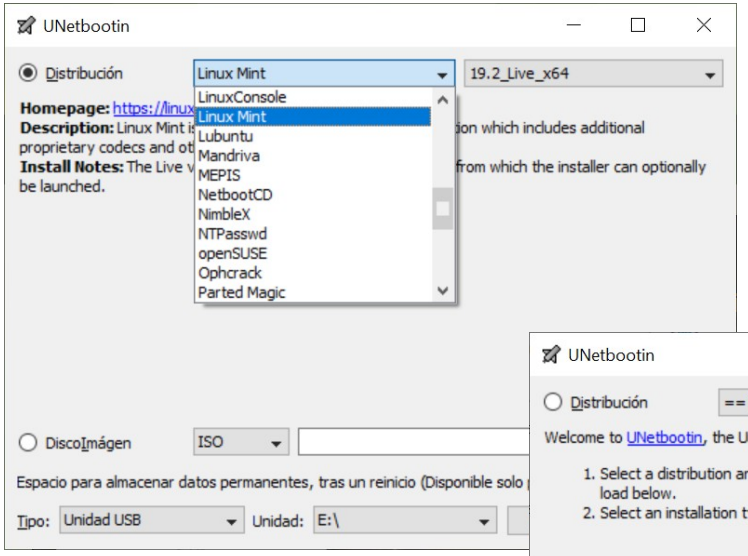
Grabación de un fichero ISO en un disco óptico con CDBurnerXP

Grabación de ISOs en pendrives USB

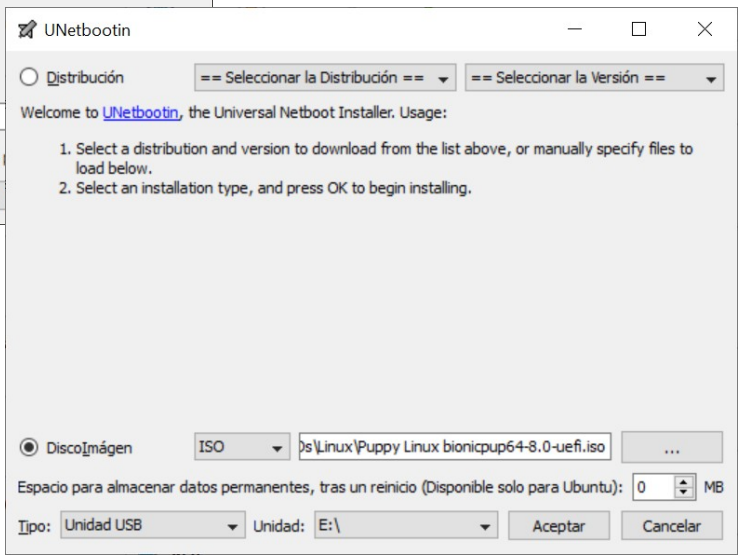
Tanto con Rufus como Unetbootin podremos grabar una ISO que tengamos en un pendrive, pero entre estas dos Unetbootin además nos permitirá automatizar la descarga de una selección de distribucciones Linux, sistemas operativos libres y alguna Live CD de recuperación:



Grabación de la ISO de Puppy Linux con Rufus



Grabación de la ISO de Linux Mint 19.2 con Unetbootin con la opción de descarga automática



Grabación de la ISO de Puppy Linux con UNetbootin

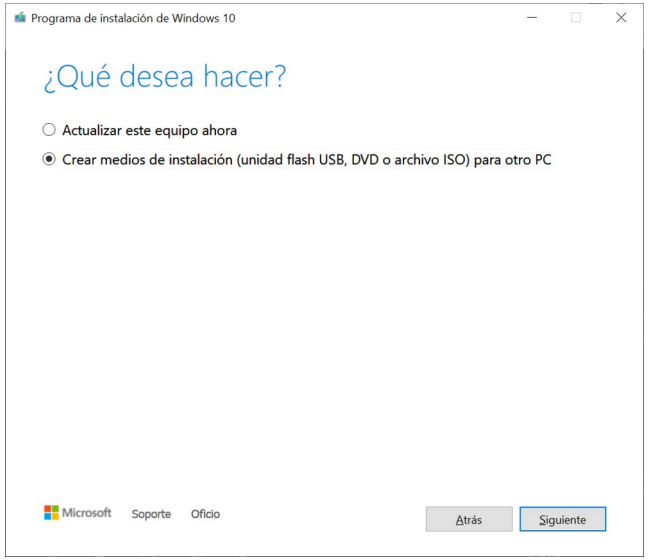
Obtener y grabar ISOs de Windows 10

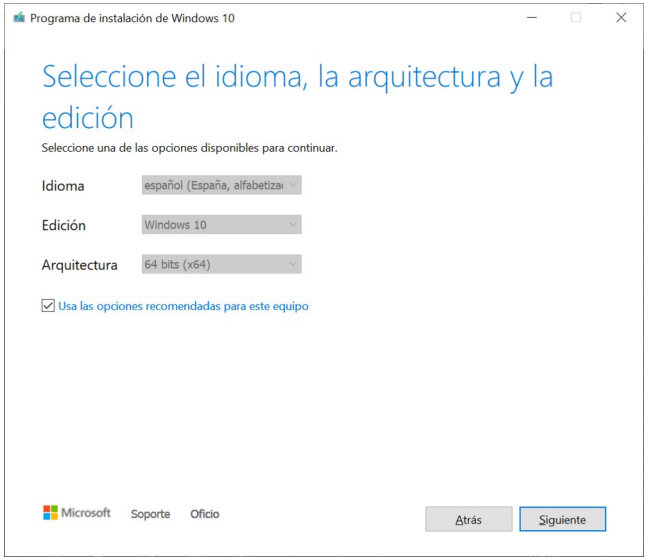
En el caso de Windows 10 Microsoft proporciona un software específico para obtener y grabar la ISO de instalación denominado Windows 10 Media Creation Tool. Este software permite:

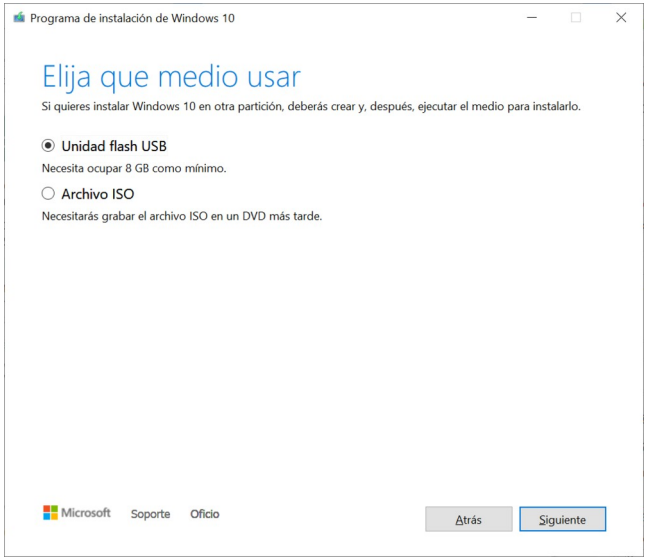
- Descargar la ISO y guardarla en disco (Después podremos escribirla en un USB utilizando Rufus o utilizarla para una máquina virtual)
- Opcionalmente grabarla directamente en un pendrive USB o un DVD

En caso de no tener un sistema operativo Windows (Linux o Mac OS) no podremos instalar el Windows 10 Media Creation Tool al ir a la página de descarga nos redirigirá a otra web donde descargar la ISO.

Si tenemos una licencia de Windows 10 para nuestro ordenador tras la instalación esta se activará automáticamente, sino podremos introducir la clave de licencia durante la instalación.

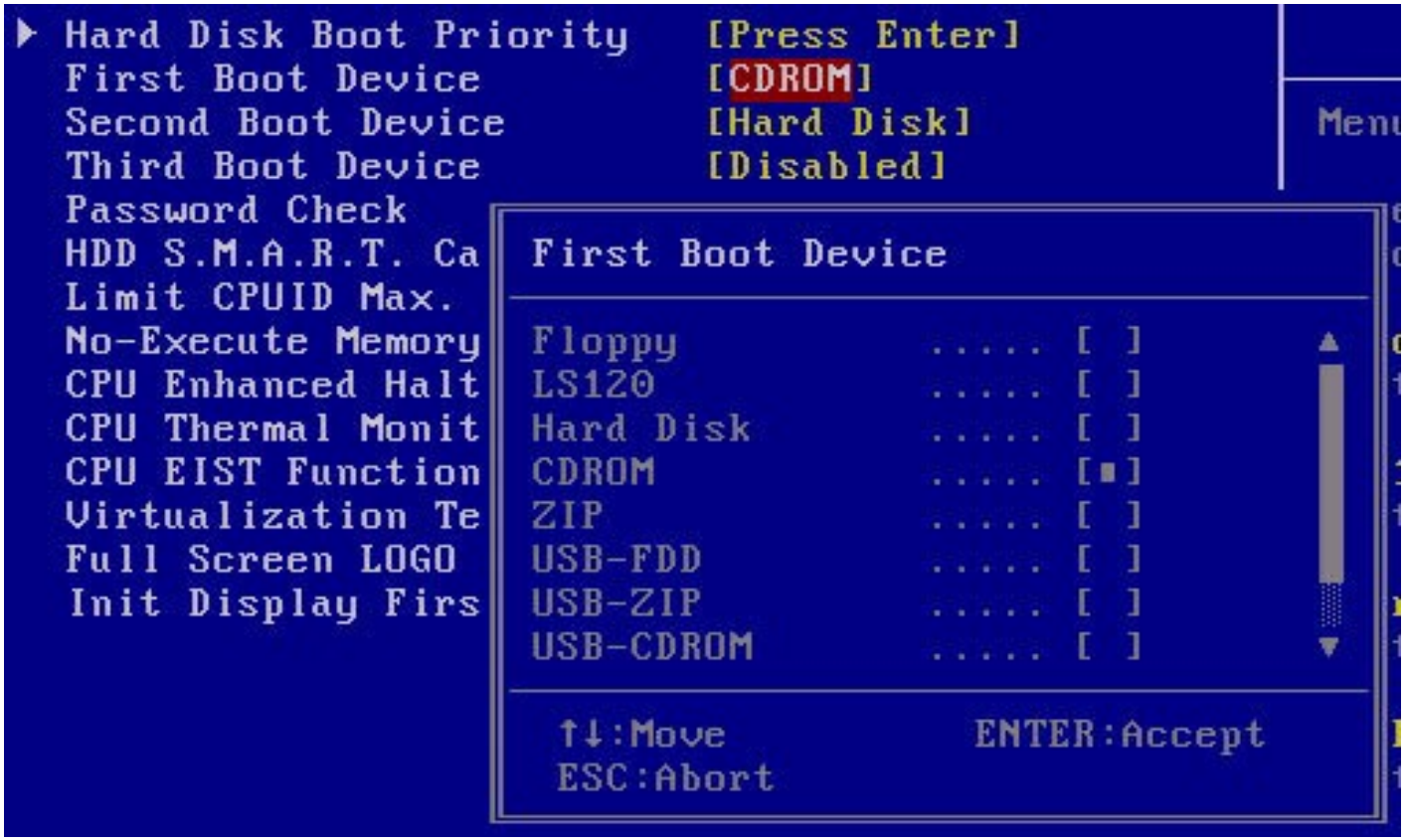






Configuración del orden de arranque en la BIOS

Una vez grabado tendremos que revisar el orden de arranque en la BIOS para asegurarnos que en el orden de arranque tenemos la unidad óptica o el almacenamiento USB antes que el disco duro. Esta configuración variará dependiendo de la BIOS de nuestro ordenador



Administración de software en sistemas Linux

Distribuciones Linux

- El kernel solo contiene la implementación de los servicios básicos del Sistema operativo. Por eso para poder distribuirlo e instalarlo es necesario acompañarlo de herramientas de instalación, configuración, utilidades, aplicaciones, … que es lo que se hizo al fusionar el kernel de Linux con GNU
- Este trabajo de combinar el kernel con software lo llevan a cabo las diferentes organizaciones que crean las **distribuciones Linux**
- Se llevan a cabo por personas, entidades o empresas que se dedican a hacer recopilaciones de software existente combinado opcionalmente con software propio de forma que facilitan la instalación y configuración de GNU/Linux
- Incluye mucho software GNU y la licencia GPL permite distribuirlas gratuitamente o a través de canales comerciales, pero lo que se paga es el trabajo de recolección, el software propio de la distribución que pueda incluir, una presentación más elaborada, costes de distribución y soporte técnico para el usuario, …
- De todas formas casi todas las distribuciones se pueden descargar gratuitamente desde Internet, aunque algunas cobran en concepto de herramientas adicionales propias y/o un servicio de soporte técnico

Distribuciones Linux según administración del software

- Puedes encontrar listas de distribuciones Linux existentes en la página web principal del sistema operativo Linux (www.linux.org) y rankings de popularidad en la web https://distrowatch.com
- La forma de administrar el software varía de una distribución Linux a otra, y encontraremos diferencias en:
  - El formato de ficheros en el que se empaqueta el software
  - Las herramientas utilizadas para administrar el software y sus dependencias
  - La capacidad de resolver dependencias entre paquetes durante la instalación
- Vamos a ver una clasificación de distribuciones basada en el formato de paquetes y software que emplean para administrar el software:
  - Basadas en RPM
  - Basadas en Debian
  - Basadas en Pacman
  - Basadas en Gentoo
  - Basadas en Slackware
  - Independientes

Distribuciones basadas en RPM (RPM-based)

- Se llaman así porque utilizan el sistema de gestión de paquetes **RPM Package Manager** basados en el **formato de fichero .rpm** como sistema de empaquetamiento de software
- Un fichero .rpm contiene todos los ficheros necesarios para instalar cualquier elemento de software
- Fue creado para ser utilizado en las distribuciones Red Hat pero fue adoptado por otras distribuciones
- Estas distribuciones utilizan los gestores de paquetes **dnf** y **yum** capaces de resolver dependencias y que opcionalmente pueden utilizar repositorios de paquetes en local, red o Internet. Entre estos dos dnf es más reciente y ofrece mejoras con respecto a yum , por lo que deberá ser nuestra primera opción de uso, pero de momento conviven ambos. Las distribuciones Suse tienen un gestor de paquetes propio, **zypper**.
- Las distribuciones más conocidas basadas en Red-Hat son:
  - Red Hat Linux
  - CentOS
  - Fedora
  - SUSE
  - Mandriva Linux

Distribuciones basadas en Debian (Debian-based)

- **Debian** es una distribución que surgió en el año 1993 fundada por Ian Murdock que enfatiza el uso del software libre
- Es desarrollada y mantenida por una comunidad de desarrolladores voluntarios
- Utiliza el **formato de fichero .deb** como sistema de empaquetamiento de software que también contiene todos los ficheros necesarios para instalar cualquier elemento de software
- La publicación de una nueva versión de Debian precisa de al menos 2 años, que supone que tenga un ciclo de desarrollo más alto que otras distribuciones
- Utiliza el gestor de paquetes **dkpg** pero es más habitual utilizar el **apt** u otros frontales gráficos basados en apt como **aptitude** y **synaptic**.
- Debian con su formato .deb **fue la primera distribución en automatizar la resolución de dependencias entre paquetes**

debian

Distribuciones basadas en Debian (Debian-based)

- Son muchas las distribuciones basadas en Debian entre las que podemos destacar por distintos motivos **Ubuntu** y **Kali Linux**
- **Kali Linux** es una distribución diseñada para seguridad informática, informática forense y tests de penetración siendo por esto la elegida por las comunidades de hackers
- En el caso de **Ubuntu** se publican revisiones publicadas con regularidad, tiene opción de soporte comercial y sirve a su vez de base para un elevado número de distribuciones:
  - Algunas con distribución oficial de Ubuntu como **Xubuntu**, **Edubuntu**, **Kubuntu**, **Lubuntu, Ubuntu Studio**, …
  - Otras que son desarrolladas y distribuidas por terceros como **Linux Mint** que ha llegado incluso a ganar en popularidad a Ubuntu por su facilidad de uso y soporte multimedia

Basadas en pacman y gentoo

- Distribuciones **basadas en pacman** (pacman-based)
  - **pacman es un gestor de paquetes que emplea el formato tar comprimidos en gzip o xz** para todos los paquetes. Además es capaz de resolver dependencias y automáticamente descargar e instalar todos los paquetes necesarios
  - Su principal distribución es **Arch Linux** en la que se basan la mayor parte de las otras distribuciones pacman como es el caso de **Manjaro Linux**, que le quita popularidad a Arch Linux por su mayor facilidad de uso
- Distribuciones **basadas en gentoo** (gentoo-based)
  - **Gentoo** es una distribución que permite un alto grado de configuración que permite conseguir configuraciones muy eficientes, pero esta configuración requiere conocimientos que hace que no esté al alcance de cualquier usuario. Utiliza el **gestor de paquetes Portage que emplea el formato tar comprimidos con bzip2 (.tbz2) para todos los paquetes**
  - **Chromium OS/Chrome OS** diseñados por Google y que utilizan el navegador Chrome como principal interfaz de usuario. Está basada en gentoo, pero **no utiliza Portage sino la Chrome Web Store**. Es el sistema operativo de los **chromebooks**

Basadas en slackware y distribuciones independientes

- Distribuciones **basadas en slackware** (slackware-based)
  - slackware es una distribución diseñada para ser muy estable y simple, siendo la que guarda más parecido con el Unix original. A diferencia de las otras no dispone de instalador gráfico ni de gestor de paquetes capaz de resolver dependencias. **Sus paquetes están en tar comprimidos en gzip o xz**
- Distribuciones **independientes**, que no encajan en ninguna de las anteriores y que tienen sus propios mecanismos de administración del software. Entre estas destacamos:
  - **Android**, diseñado para pantallas táctiles de smartphones y tablets
  - **Puppy Linux**, distribución que requiere muy pocos recursos hardware
  - **DD-WRT**, empleada como firmware de routers wireless y puntos de acceso

Administración de software en distribuciones Debian/Ubuntu

Repositorios de paquetes

- En los sistemas debian el software se distribuye en **ficheros con extensión .deb llamados paquetes**
- Los paquetes en ficheros .deb están alojados en **repositorios** que tienen una ubicación, típicamente en Internet pero que **también puede ser local o en red local**
- En el sistema podemos **configurar múltiples repositorios** indicando para cada uno de ellos su ubicación que podremos utilizar para consultar e instalar software
- La lista de paquetes disponibles en los repositorios configurados se almacena en una **caché local** que acelera la consulta de paquetes disponibles. La conexión a los repositorios se realiza en el momento de realizar la instalación de paquetes y cuando queramos actualizar esta caché
- Los **repositorios además de los paquetes almacenan las dependencias**, que se dan cuando cuando hay paquetes que requieren la instalación de otros adicionales para su correcto funcionamiento. Estas dependencias nos permiten automatizar la instalación de un paquete que necesitemos y todos los necesarios que no conozcamos
- Los repositorios actualizan los paquetes que contienen cada cierto tiempo, por lo que también **son la fuente para actualizar el software instalado** a versiones más recientes

Gestión de repositorios

Los repositorios que utiliza el sistema se configuran en ficheros de texto en la ruta **/etc/apt.** Podemos editar estos ficheros para añadir nuevos repositorios, aunque siempre es recomendable hacer una copia de seguridad previa o utilizar otras utilidades para añadirlos.

Añadir algunos repositorios puede requerir la instalación de claves con las que se indica que es un repositorio de confianza

Para cada entrada de un repositorio tenemos que ajustar:

- Si es de binarios (empiezan por deb) o de código fuente (deb-src)
- La URI (Unified Resource Identifier), que indica la ubicación de los ficheros
- El nombre de la versión, en el caso de la debian de la captura es la versión **stretch**
- El tipo de componentes del repositorio: **main** (Software libre), **contrib** (Software libre que depende de software no libre) y **non-free** (Todo tipo de software no libre)

Podemos ver un ejemplo en la siguiente captura del fichero sources.list de debian:

```
#

# deb cdrom:[Debian GNU/Linux 9.3.0 _Stretch_ - Official amd64 NETINST 20171209-12:10]/ stretch main

#deb cdrom:[Debian GNU/Linux 9.3.0 _Stretch_ - Official amd64 NETINST 20171209-12:10]/ stretch main

deb http://ftp.es.debian.org/debian/ stretch main
deb-src http://ftp.es.debian.org/debian/ stretch main

deb http://security.debian.org/debian-security stretch/updates main
deb-src http://security.debian.org/debian-security stretch/updates main

# stretch-updates, previously known as 'volatile'
deb http://ftp.es.debian.org/debian/ stretch-updates main
deb-src http://ftp.es.debian.org/debian/ stretch-updates main
```

Repositorios según distribución

Los repositorios disponibles tras una instalación pueden variar de una distribución a otra.

Debian y Ubuntu tienen sus propios repositorios, y otras distribuciones como Linux Mint, basada en Ubuntu, combina sus propios repositorios con los de Ubuntu.

En los repositorios de distribuciones basadas en Ubuntu la notación final que indica el tipo de software que aloja:

- **main** (sofware libre soportado oficialmente por canonical)
- **restricted** (software no libre soportado oficialmente)
- **universe** (software libre no soportado oficialmente)
- **multiverse** (software no libre y no soportado oficialmente).

Como vemos en el siguiente ejemplo de fuentes de repositorios de una Linux Mint un mismo repositorio puede alojar múltiples tipos de software:

```
  GNU nano 2.9.3                    official-package-repositories.list

# Do not edit this file manually, use Software Sources instead.

deb http://packages.linuxmint.com tara main upstream import backport #id:linuxmint_main

deb http://archive.ubuntu.com/ubuntu bionic main restricted universe multiverse
deb http://archive.ubuntu.com/ubuntu bionic-updates main restricted universe multiverse
deb http://archive.ubuntu.com/ubuntu bionic-backports main restricted universe multiverse

deb http://security.ubuntu.com/ubuntu/ bionic-security main restricted universe multiverse
deb http://archive.canonical.com/ubuntu/ bionic partner
```

Comando apt-get

Con el **comando apt-get o apt** gestionamos la mayor parte de todo lo que tenga que ver con los repositorios de paquetes. Vemos algunos ejemplos de uso:

- **apt-get update** → Actualizamos la caché de paquetes de los repositorios configurados en el sistema. Es necesario ejecutarlo siempre que modificamos el fichero sources.list
- **apt upgrade** → Actualiza todos los paquetes a la última versión disponible en el repositorio
- **apt-get install *NombrePaquete1 NombrePaquete2 ...*** → Instala los paquetes pasados como parámetros y sus dependencias asociadas solicitando previa confirmación
- **apt -y install *NombrePaquete* NombrePaquete2 …** → Instala los paquetes pasados como parámetros y sus dependencias sin pedir confirmación
- **apt-get remove *NombrePaquete1 NombrePaquete2 ...*** → Elimina los paquetes pasados como parámetros
- **apt purge *NombrePaquete*** → Elimina los paquetes pasados como parámetros y además borra sus ficheros de configuración. Útil cuando queremos reinstalar un software que no funciona porque su configuración es incorrecta o se corrompió
- **apt autoremove** → Elimina los paquetes que fueron instalados para satisfacer alguna dependencia que el sistema considera que ya no son necesarios

Comandos apt-cache y dkpg

El **comando apt-cache** nos permite hacer búsquedas sobre los paquetes en la caché local del sistema

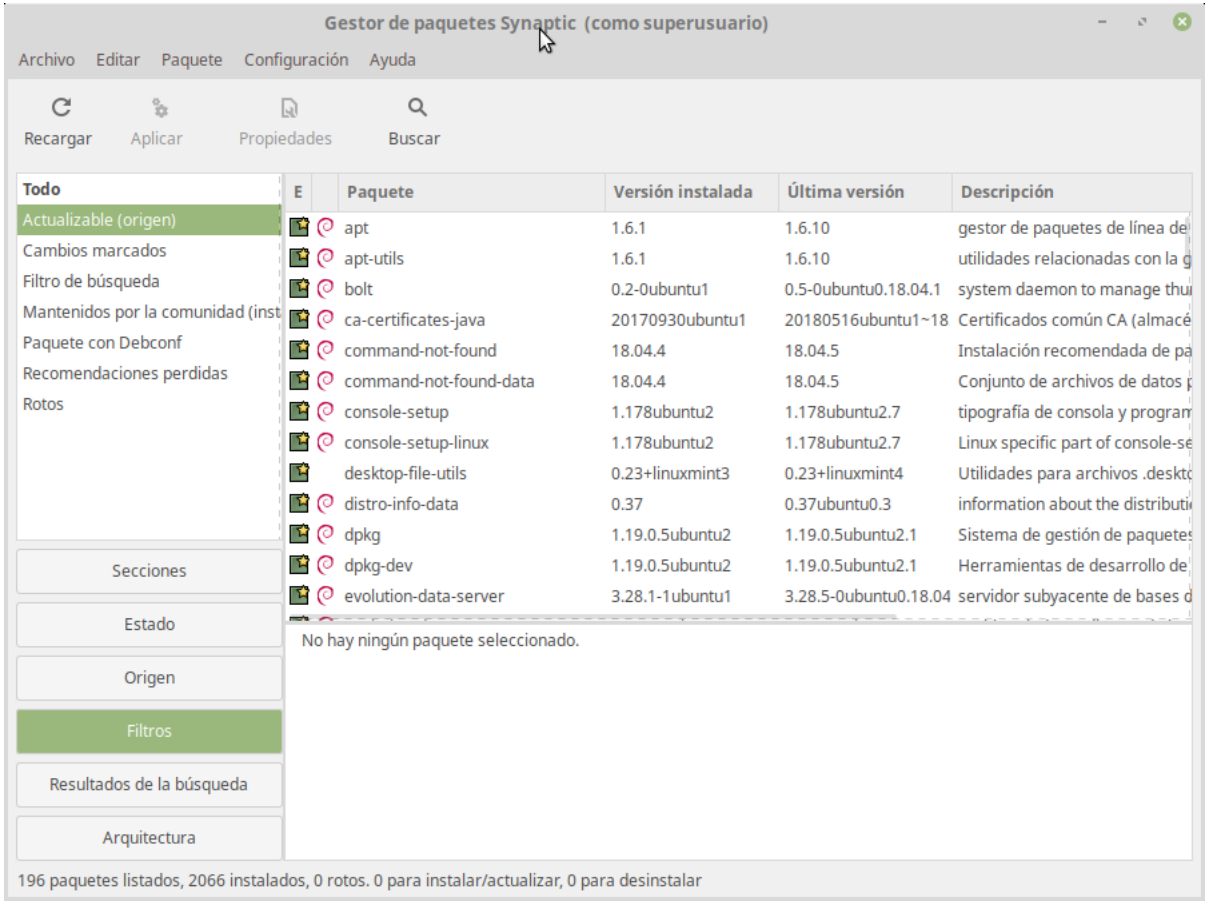
- **apt-cache search *PatronBusqueda*** → Busca entre los paquetes en la caché aquellos que coincidan con el patrón de búsqueda
- **apt-cache pkgnames** → Lista todos los paquetes disponibles
- **apt-cache show *NombrePaquete*** → Información extendidas del paquete pasado como parámetro
- **apt-cache stats** → Estadísticas generales de la cache

El **comando dpkg** (debian package) es otra opción con la que podemos gestionar los paquetes

- Es un comando a **más bajo nivel que apt-get**, ya que es el que utiliza apt-get a la hora de realizar la instalación final del .deb
- Utilizaremos dpkg cuando queramos instalar directamente ficheros .deb que tengamos, si echar mano de los repositorios configurados y tambien para obtener información de los paquete instalados:
  - **dpkg -i fichero.deb** → Instala el paquete pasado como parámetro
  - **dpkg -l** → Lista todos los paquetes instalados e información de cada uno
  - **dpkg -l NombrePaquete** → Lista el paquete indicado e información de este

Frontal gráfico de paquetes

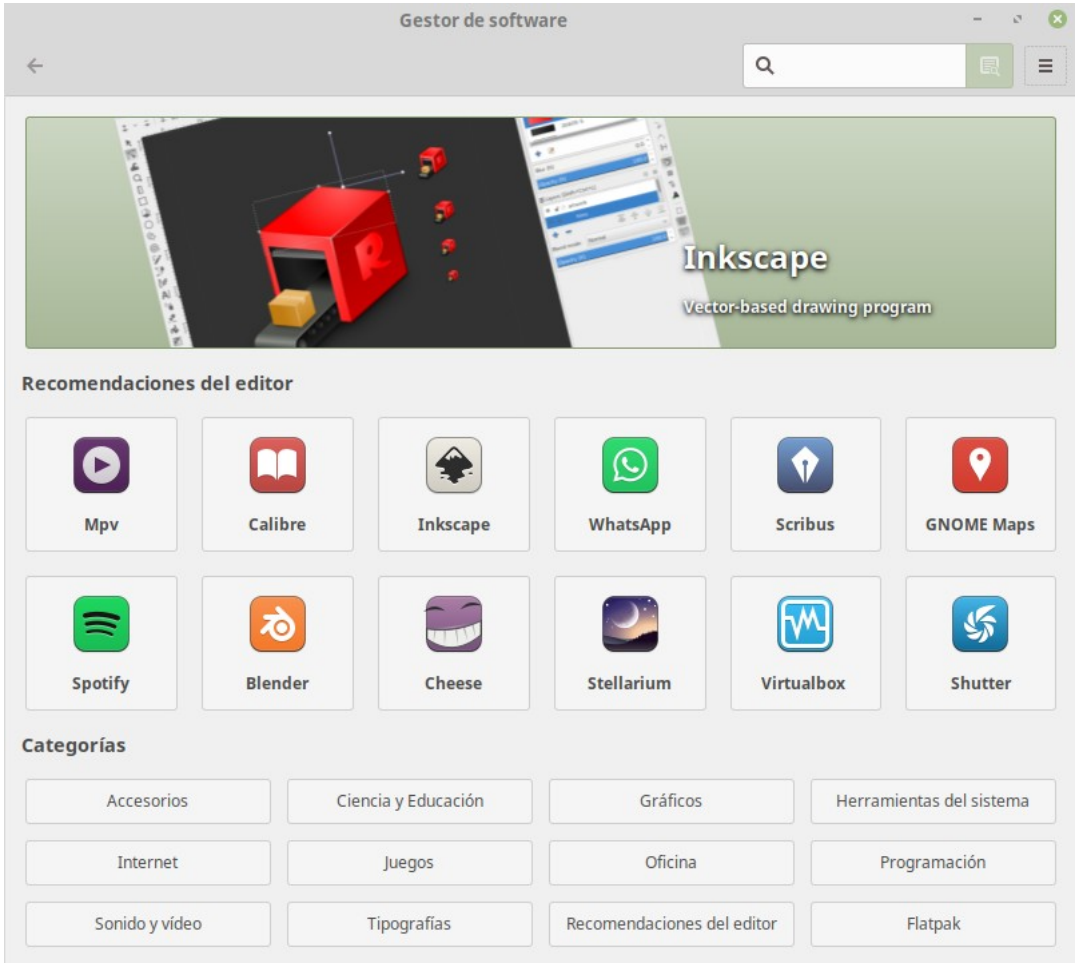
Si optamos por una instalación con interfaz gráfica también encontraremos algunos gestores de paquetes con interfaz gráfica en el que podremos buscar e instalar paquetes con la resolución automática de dependencias:



Gestor de paquete “Synaptic” en Linux Mint

Administrador gráfico de aplicaciones

Además también encontraremos gestores de software más orientados a la instalación de aplicaciones que de paquetes al estilo de una app store, aunque es un nivel superior que por detrás en realidad gestiona paquetes:



Aplicación “Gestor de Software” de la distribución Linux Mint

Administración de software en distribuciones RPM

Comandos de gestión de paquetes dnf y yum

Como mencionamos antes en las distribuciones RPM se utilizan los comandos yum y dnf, preferiblemente este último. dnf surgió como un fork de yum y utilizan los mismos parámetros, por lo que todos los ejemplos que pondremos aquí funcionan con cualquiera de los comandos:

- **dnf update** → Actualiza todos los paquetes a la última versión disponible en el repositorio
- **yum install *NombrePaquete1 NombrePaquete2 ...*** → Instala los paquetes pasados como parámetros y sus dependencias asociadas solicitando previa confirmación
- **dnf -y install *NombrePaquete* NombrePaquete2 …** → Instala los paquetes pasados como parámetros y sus dependencias sin pedir confirmación
- **yum remove *NombrePaquete1 NombrePaquete2 ...*** → Elimina los paquetes pasados como parámetros
- **dnf update *NombrePaquete1 NombrePaquete2*** ... → Actualiza los paquetes pasados como parámetros
- **yum clean all** → borra la caché de los repositorios almacenada en /var/cache/yum
- **dnf autoremove** → Elimina los paquetes instalados para satisfacer dependencias que ya no son necesarias

Configuración de repositorios

Los repositorios se configuran en ficheros con extensión .repo en la ruta **/etc/yum.repos.d**. Podemos tener varios ficheros cada uno con varios repositorios configurados.

Podemos gestionar estos repositorios con estos comandos:

- **dnf config-manager --set-enabled *NombreRepositorio*** → Activa el repositorio pasado como parámetro, que debe estar configurado en un fichero .repo
- **yum config-manager --set-disabled *NombreRepositorio*** → Desactiva el repositorio pasado como parámetro
- **dnf repolist** → Lista todos los repositorios de paquetes activados
- **dnf repolist all** → Lista todos los repositorios de paquetes (activados y desactivados)

Fichero de repositorios de distribución Fedora

Vemos el fichero fedora.repo para ver como es la notación de repositorios en una distribución fedora. El formato de estos no tiene nada que ver con los de distribuciones debian/ubuntu:

```
  GNU nano 4.3                                                fedora.repo
[fedora]
name=Fedora $releasever - $basearch
#baseurl=http://download.fedoraproject.org/pub/fedora/linux/releases/$releasever/Everything/$basearch/os/
metalink=https://mirrors.fedoraproject.org/metalink?repo=fedora-$releasever&arch=$basearch
enabled=1
metadata_expire=7d
repo_gpgcheck=0
type=rpm
gpgcheck=1
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-fedora-$releasever-$basearch
skip_if_unavailable=False

[fedora-debuginfo]
name=Fedora $releasever - $basearch - Debug
#baseurl=http://download.fedoraproject.org/pub/fedora/linux/releases/$releasever/Everything/$basearch/debug/tree/
metalink=https://mirrors.fedoraproject.org/metalink?repo=fedora-debug-$releasever&arch=$basearch
enabled=0
metadata_expire=7d
repo_gpgcheck=0
type=rpm
gpgcheck=1
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-fedora-$releasever-$basearch
skip_if_unavailable=False

[fedora-source]
name=Fedora $releasever - Source
#baseurl=http://download.fedoraproject.org/pub/fedora/linux/releases/$releasever/Everything/source/tree/
metalink=https://mirrors.fedoraproject.org/metalink?repo=fedora-source-$releasever&arch=$basearch
enabled=0
metadata_expire=7d
repo_gpgcheck=0
type=rpm
gpgcheck=1
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-fedora-$releasever-$basearch
skip_if_unavailable=False
```

Agrupaciones de paquetes

En los sistemas con paquetes RPM además se hacen agrupaciones de paquetes relacionados entre sí con una misma temática. Algunos ejemplos son:

- Herramientas de desarrollo
- Software educacional
- Producción de audio
- Herramientas de administración
- ...

Podemos gestionar estos grupos con los siguientes comandos:

- **yum grouplist** → Lista todos los grupos de paquetes disponibles, los instalados
- **dnf groupinstall 'MySQL Database'** → Instala el grupo MySQL Database
- **dnf groupupdate 'MySQL Database'** → Actualiza el grupo MySQL Database
- **dnf groupremove 'MySQL Database'** → Se elimina el grupo MySQL Database

El comando rpm

El comando rpm nos permite la administración de paquetes a más bajo nivel procesando ficheros rpm de forma autónoma sin la gestión automática de las dependencias.

Será el que tendremos que utilizar para instalar software que no tengamos en los repositorios configuraciones y que nos proporcionen directamente en la forma de fichero rpm pero ofrece opciones adicionales para eliminar, actualizar, verificar y consultar el paquete.

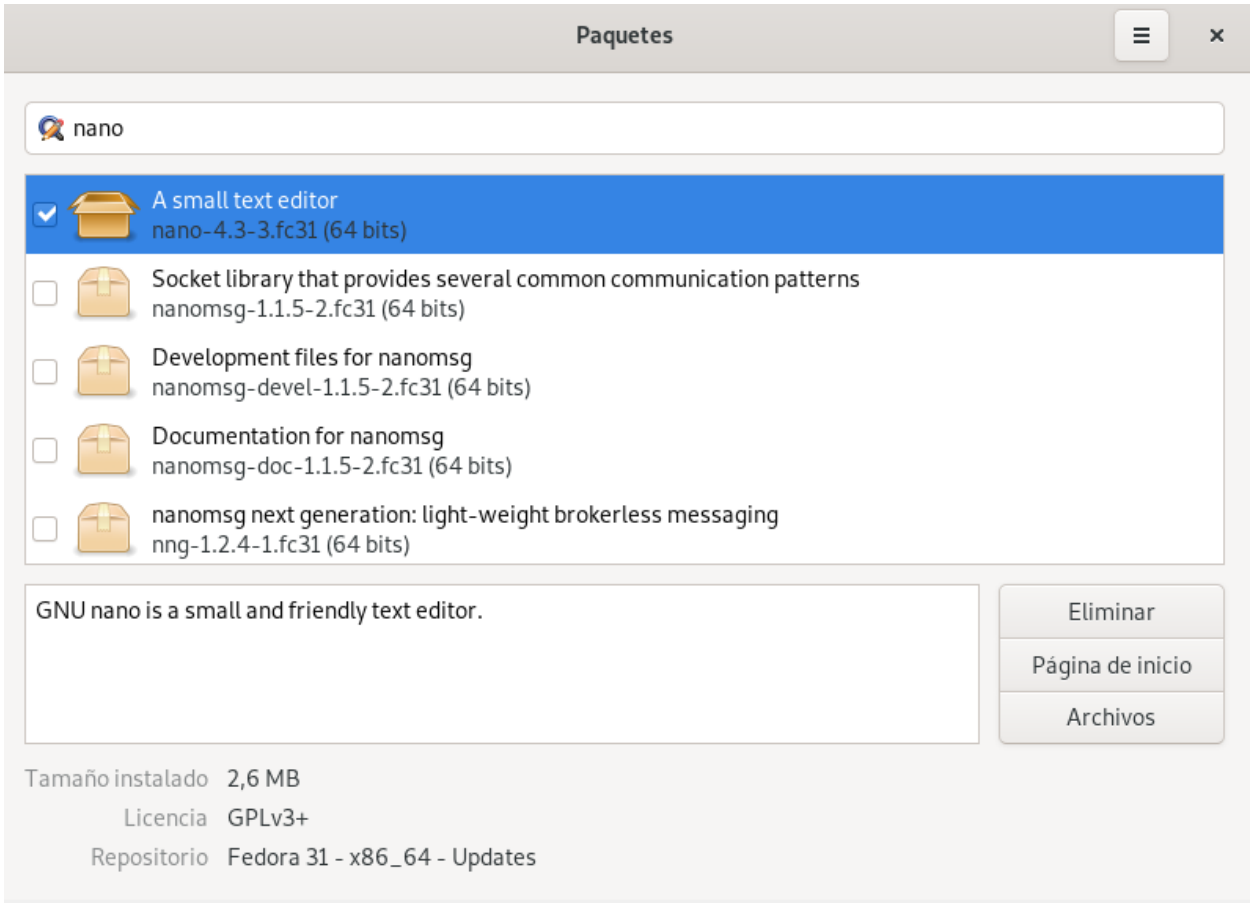
Es equivalente al dpkg de las distribuciones debian/ubuntu.

Vemos algunos ejemplos de uso del comando:

- **rpm -ivh pidgin-2.7.9-5.el6.2.i686.rpm** → Instala el paquete pasado como parámetro (opción i), mostrando información en detalle (opción v) con barras de progreso (opcion h)
- **rpm -qpR BitTorrent-5.2.2-1-Python2.4.noarch.rpm** → Comprobar dependencias del paquete pasado como parámetro
- **rpm -ivh --nodeps BitTorrent-5.2.2-1-Python2.4.noarch.rpm** → Instalar un paquete sin verificar dependencias
- **rpm -q BitTorrent** → Comprueba si se instaló algún paquete con el nombre pasado como parámetro
- **rpm -ql BitTorrent** → Listar los ficheros instalados por un paquete
- **rpm -evv BitTorrent** → Eliminar un paquete ya instalado

Frontal gráfico de paquetes

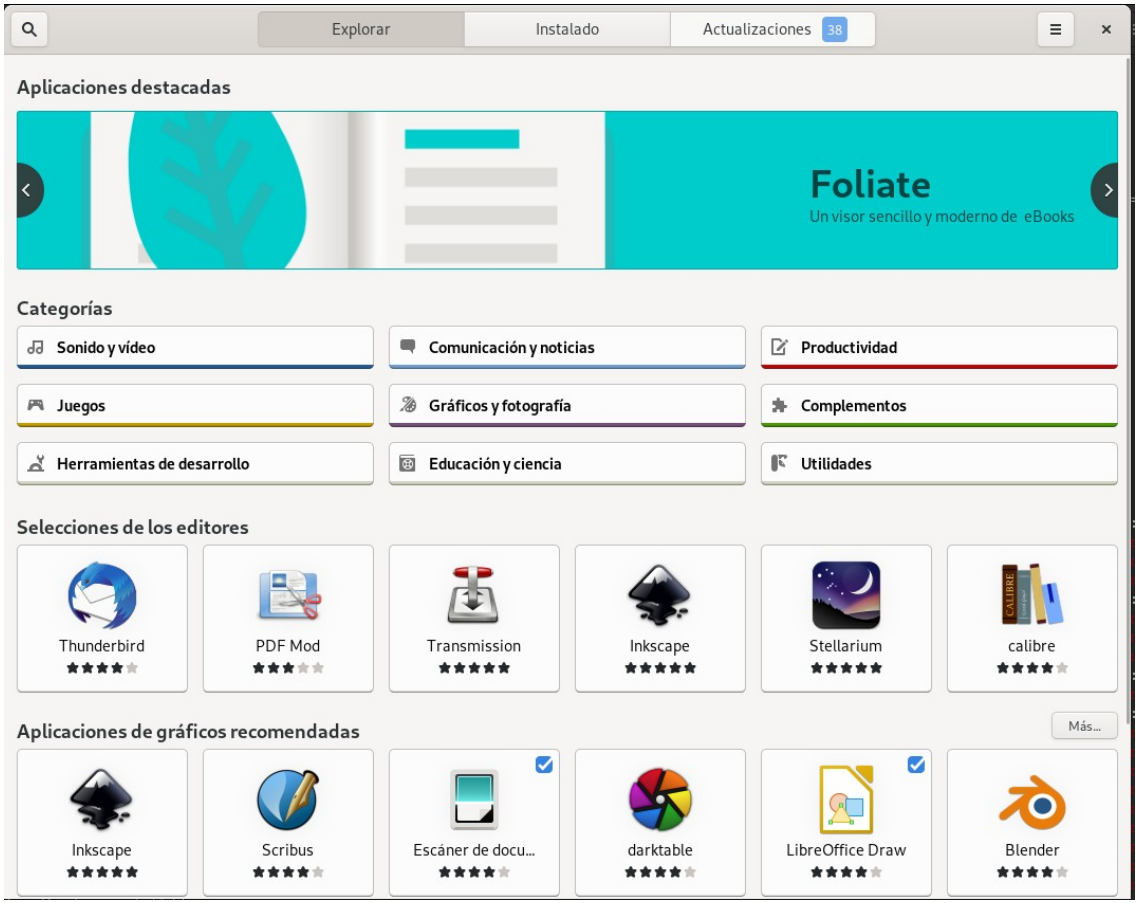
Fedora también dispone de un frontal gráfico para la gestión de paquetes si optamos por una instalación con interfaz gráfica:



Aplicación “paquetes” de Gnome

Administrador gráfico de aplicaciones

También encontraremos gestores de software más orientados a la instalación de aplicaciones que de paquetes:



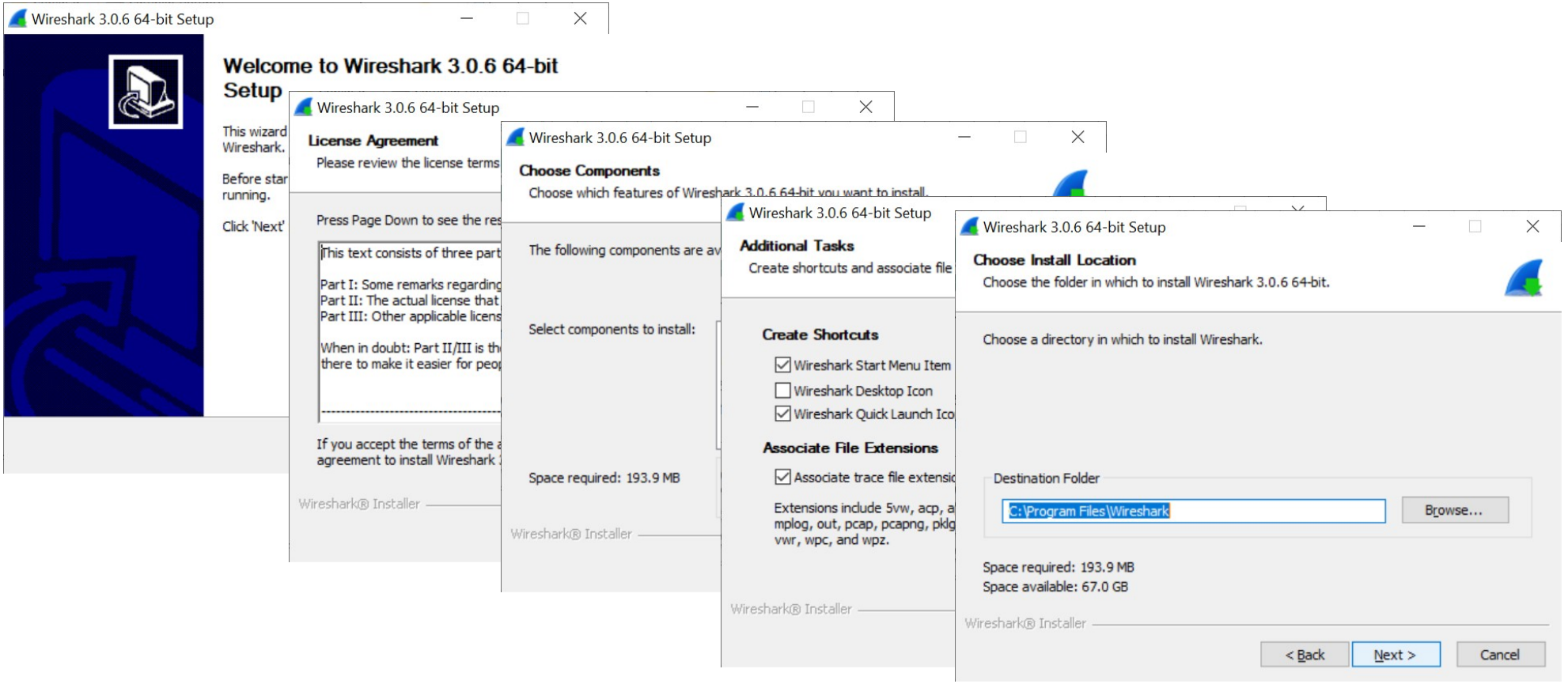
Aplicación “Software” de la distribución Fedora

Instalación y administración de software en Windows

Paquetes de instalación de aplicaciones “tradicionales”

Tal como comentamos en los sistemas Windows el software “tradicional” viene empaquetado en las siguientes formas:

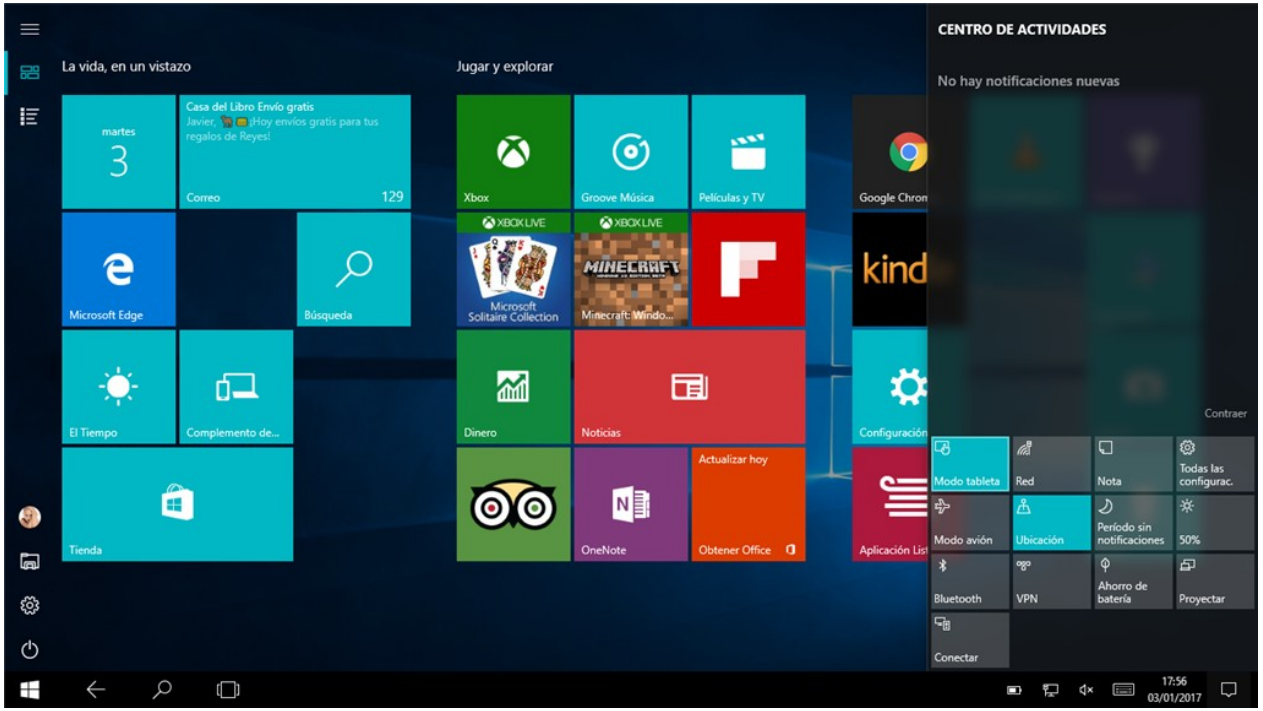
- **Paquetes ejecutables con extensiones .exe o .msi**, los cuales ejecutaremos y por lo general nos guiarán a través de un asistente en el que podremos ajustar opciones de instalación. Este automatizará la creación de accesos directos y registrará el programa en el sistema operativo
- **Aplicaciones portables** que podremos poner en cualquier ubicación del disco. En este caso si queremos accesos directos tendremos que crearlos manualmente



Aplicaciones “Modern” y la interfaz “Metro”

Windows además tiene otro tipología de aplicaciones, las “Modern”, que son aplicaciones diseñadas para la interfaz “Metro” introducida en Windows 8, la cual se conserva en el modo tableta de Windows 10 y parcialmente en el rediseño del menú inicio.

Cualquier versión de Windows anterior a la 8 no soporta este tipo de aplicaciones, y estas no se distribuyen en forma de paquetes con programa de instalación, sino a través de la Microsoft Store.



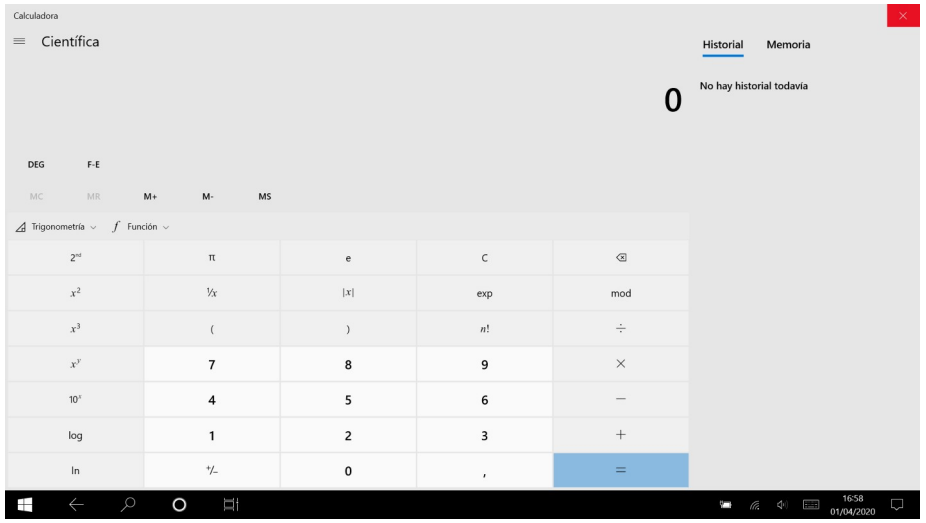
Windows 10 en modo tableta, con la interfaz Metro

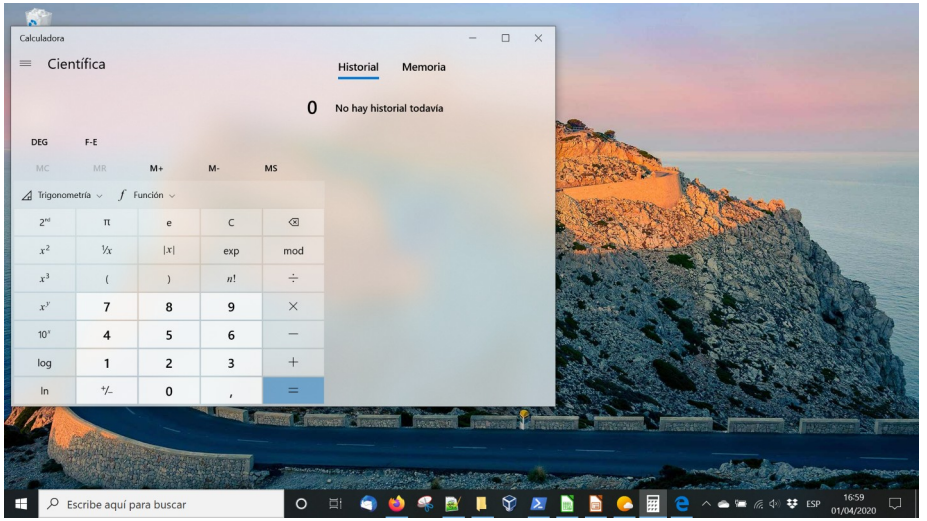
Aplicaciones “Modern”

Estas aplicaciones Modern suelen estar **diseñadas para ocupar toda la pantalla en modo tableta.** En este modo **las aplicaciones Modern no se pueden minimizar**.

Además con estas podemos interactuar con cualquier dispositivo de entrada, ratón/teclado o pantalla táctil, por lo que son operativas independiente de si trabajamos con un PC o una tablet, lo que encaja con el doble modo de trabajo de Windosw 10: Modo tableta ON y Modo tableta OFF.

Un ejemplo de aplicación Modern es la calculadora que viene con Windows 10. En modo tableta ocupa toda la pantalla y no se puede minimizar. Cuando no estamos en modo tableta se comporta como cualquier aplicación normal:

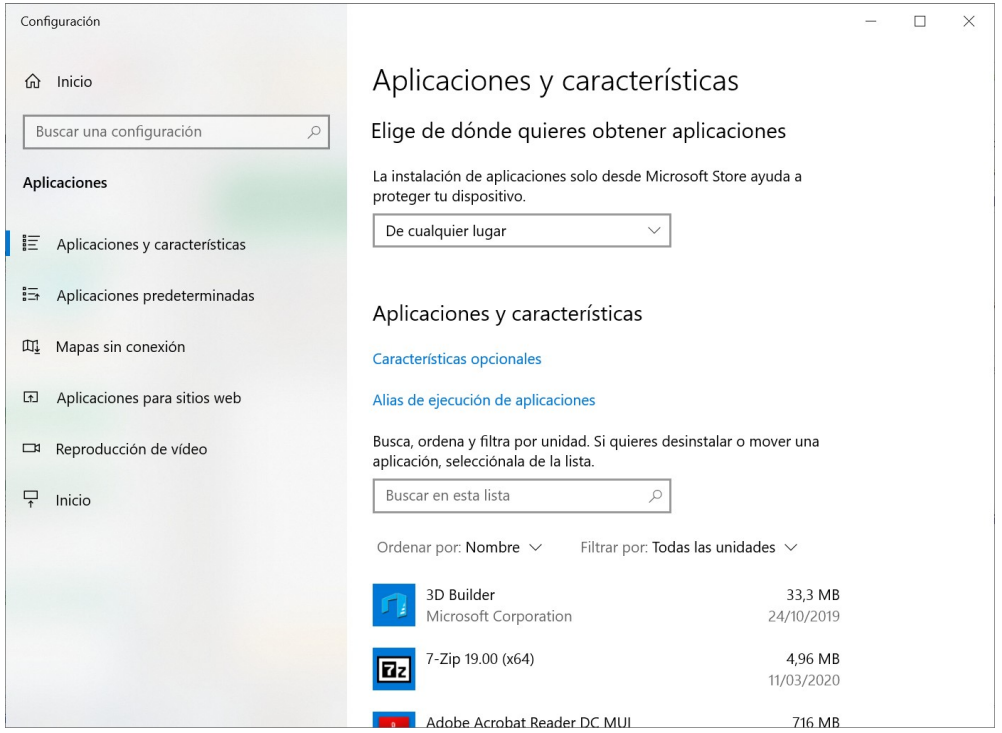


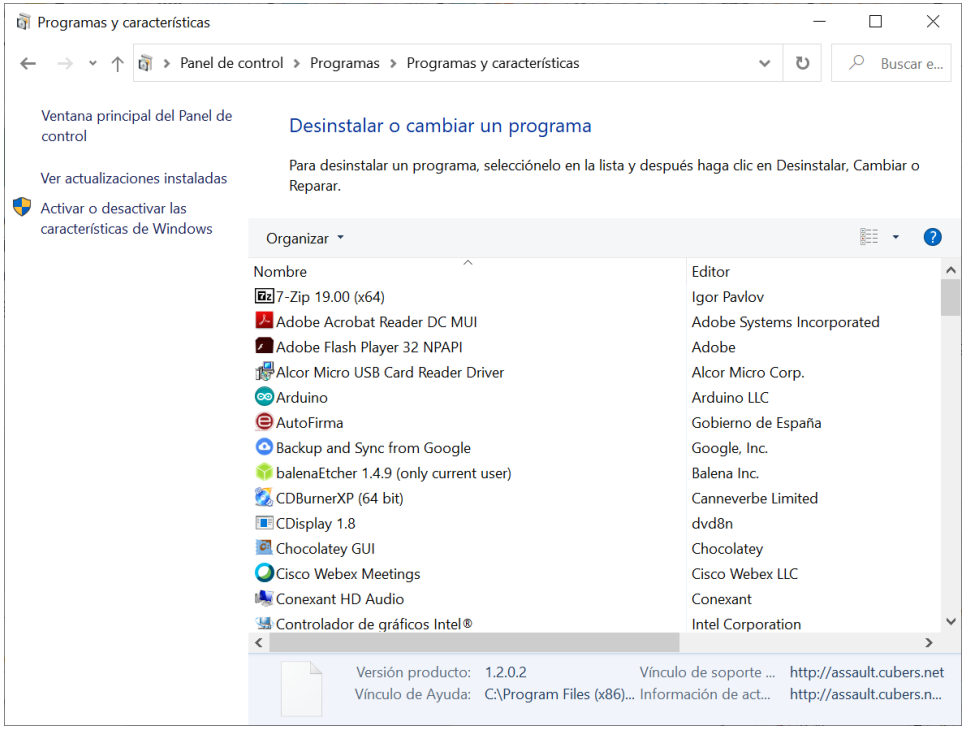


Utilidades de administración de software en Windows

Windows integra herramientas desde la cual podemos administrar el software instalado y al que podemos acceder vía:

- **Configuración → Aplicaciones y características**: Con esta podemos administrar aplicaciones clásicas y Modern
- **Panel de control → Programas → Programas y características**: Con esta podemos administrar únicamente aplicaciones clásicas



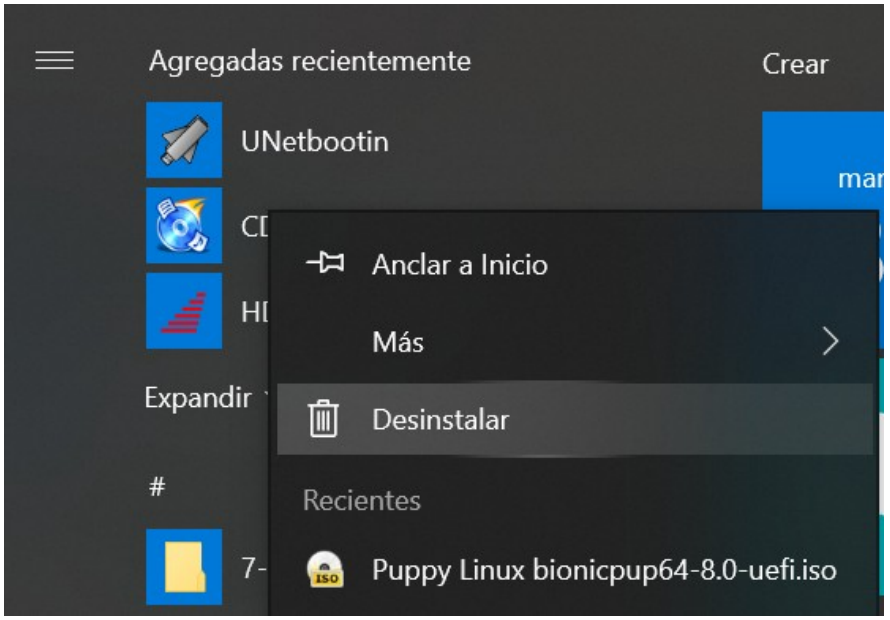


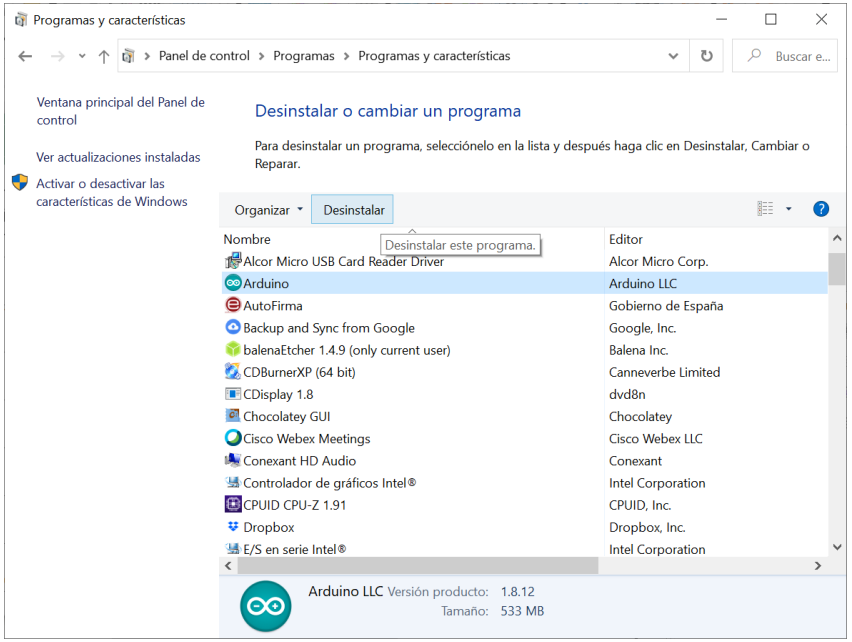
Borrado de aplicaciones

A la hora de desinstalar programas lo mejor es hacerlo desde estas herramientas de administración de software ya que nos da más garantías de un borrado completo.

También podremos lanzar la desinstalación directamente desde el menú inicio en el menú contextual sobre el acceso directo a a la aplicación.

Deberemos evitar en todo caso los borrados manuales de aplicaciones directamente del disco, salvo aplicaciones portables para las que sí basta con borrar la carpeta que contiene todos los ficheros de la aplicación portable y aquellos accesos directos que hayamos creado nosotros manualmente

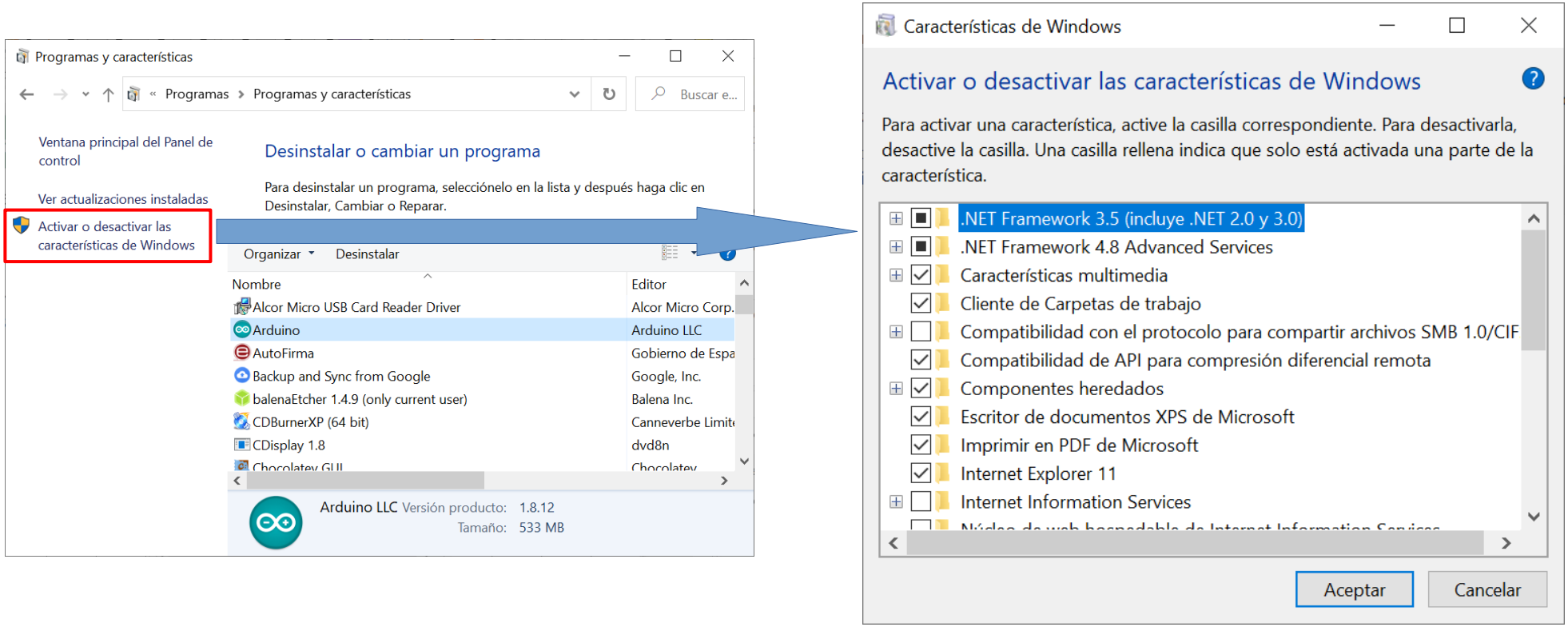




Características

Además de las aplicaciones instaladas y administradas desde estas herramientas integradas en Windows tenemos la opción de “Activar o desactivar características de Windows”.

Las características son componentes software preinstaladas con Windows que debemos activar para poder utilizarlas y que podremos desactivar si no las necesitamos, con el fin de ahorrar recursos:



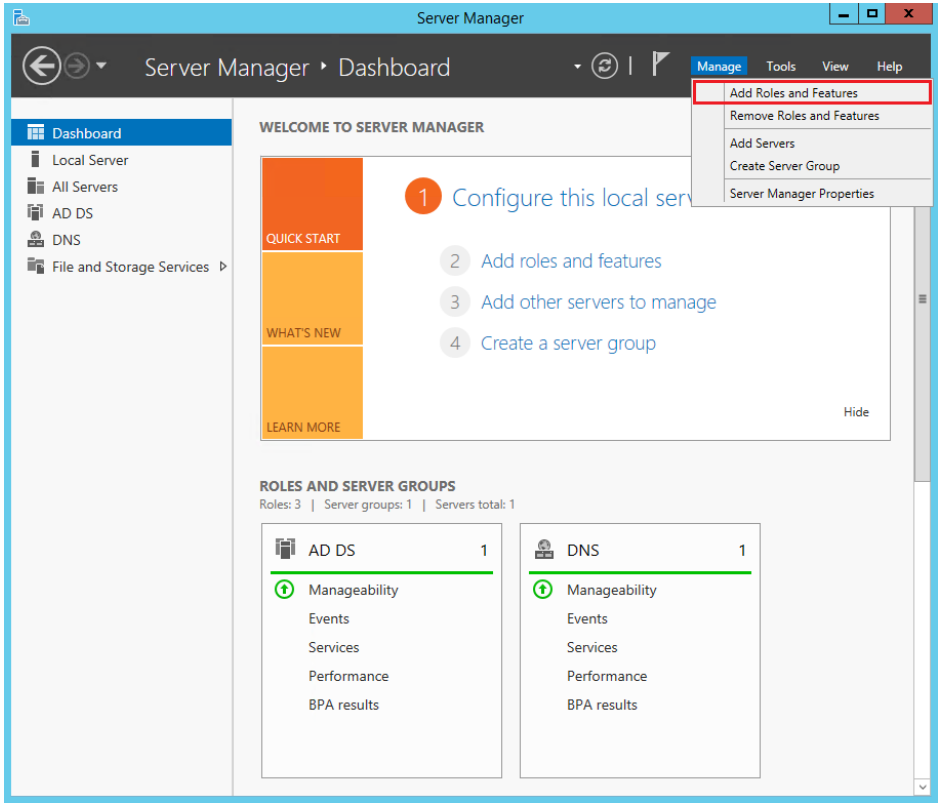
Roles y características en sistemas Windows Server

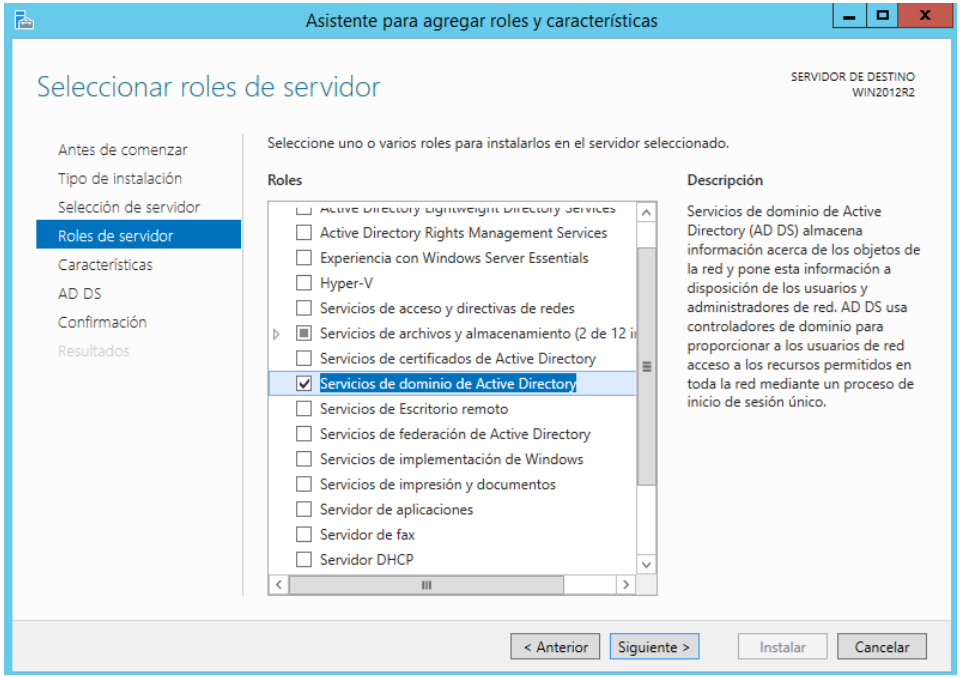
En los windows Server además de los administradores de de programas y aplicaciones, disponen de roles y características, componentes software preinstalados también con Windows:

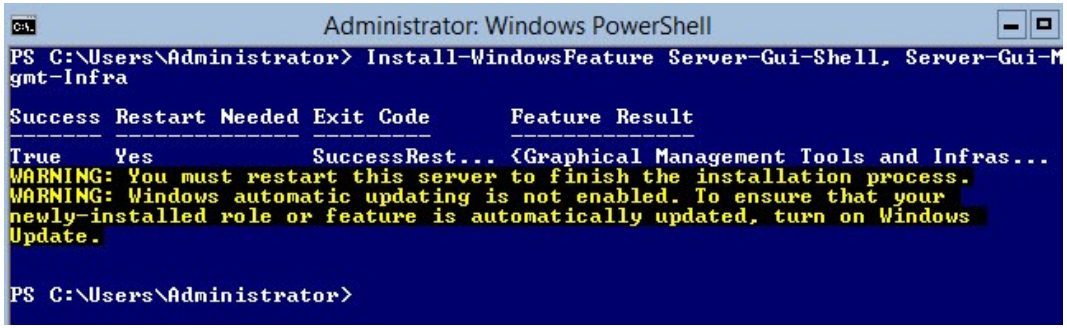
- Los **Roles** agrupan un conjunto de software que permiten habilitar el funcionamiento de servicios como:
  - Servicios de Directorio activo
  - Servicios de Escritorio remoto
  - Servidor DNS
  - Servidor DHCP
  - Hyper-V (para virtualización bare-metal hypervisor)
  - Servidor Web
  - ...
- Las **Características** son programas software que complementan o aumentan la funcionalidad de los Roles. Algunos ejemplos son:
  - .NET Framework 3.5/4.5
  - Windows Powershell
  - Servidor y cliente telnet
  - Servidor SMTP

Instalación de roles y características en Windows Server

Los roles y características específicos de los Windows Server por lo general se instalan desde el Server manager, pero también podemos instalarlos por la línea de comandos Powershell







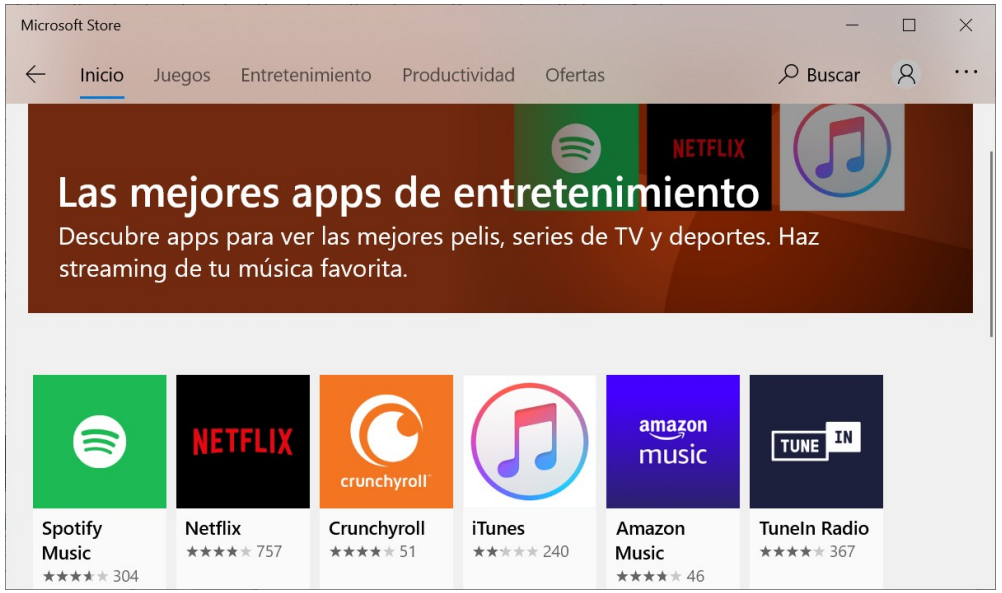
Microsoft Store

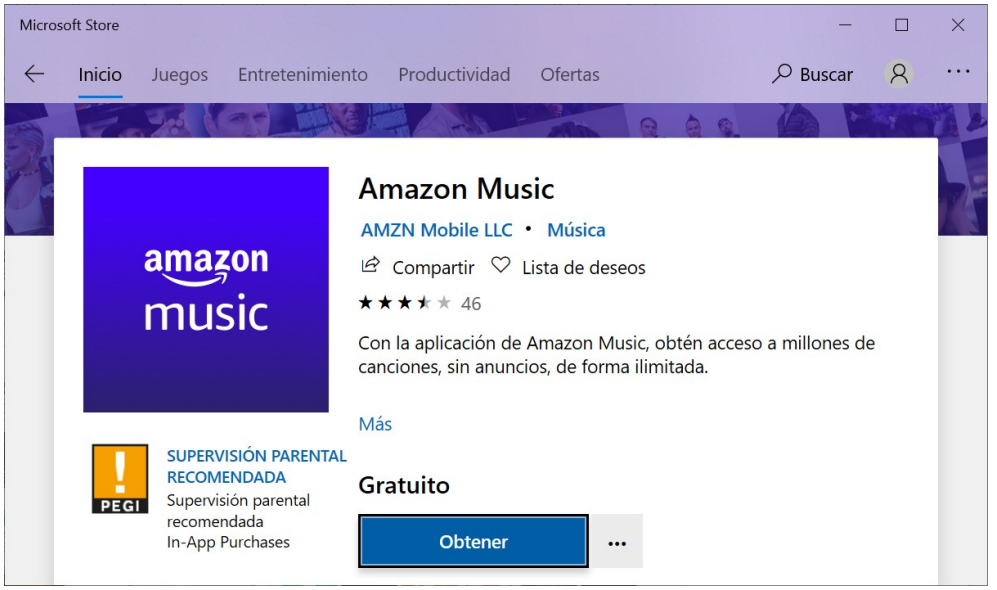
Microsoft Store es la una plataforma de apps para los sistemas Windows similar a la App Store de IOS o Google Play que tenemos en los smartphones.

Tiene un catálogo de aplicaciones gratuitas y de pago y es el **único repositorio oficial para aplicaciones "Modern"**, aunque desde la Microsoft Store también podemos instalar aplicaciones tradicionales.

La instalación desde la Microsoft Store es desatendida, sólo tenemos que seleccionar la aplicación y se instalará automáticamente. En caso de necesitar alguna configuración adicional esta se hará en el primer inicio de la aplicación.

Microsoft Store además es capaz de **detectar las nuevas versiones del software instalador y actualizarlo automáticamente**.





Comparativa de opciones de instalación de software

Instalación tradicional tradicional en paquetes .exe o .msi

- Desventajas:
  - **Instalación no desatendida**, tenemos un programa de instalación que nos guía por varios pasos que requieren nuestra interacción
  - Las actualizaciones no se gestionan de forma automática, salvo que el propio desarrollador del software lo haya integrado en su aplicación
- Ventajas
  - Una **gran variedad de software** se distribuye en este formato

Instalación desde la Microsoft Store

- Desventajas:
  - **Poca variedad de software**. Ejemplo: Software libre tan popular como LibreOffice o Gimp no lo encontraremos en la Microsoft Store
- Ventajas
  - **Actualizaciones automáticas de software** integrada dentro de la propia Store cuando los desarrolladores suben una nueva versión

Instalaciones desatendidas en Windows

Instalación de paquetes .msi con el comando msiexec

Comentamos que uno de los inconvenientes de los paquetes de software .msi y .exe es que **las instalaciones no son desatendidas**, esto nos obliga a ir siguiendo paso a paso las pantallas del proceso de instalación haciendo algún tipo de intervención: Pulsar algún botón, seleccionar alguna opción, introducir una ruta alternativa de instalación, …

Para esto tenemos una solución que es el **comando msiexec**, al que le podemos pasar el nombre de un fichero .msi y tiene múltiples parámetros.

La notación estándar del comando es la siguiente:

```
msiexec.exe <opciones de instalación> <ruta al paquete msi> [parámetros]
```

La **opciones de instalación** más típicas son:

- **/i** → Instalación normal
- **/a** → Instalación con privilegios de administración
- **/x** → Desinstalación

Con **los parámetros** entre otras cosas podemos indicar el nivel de interacción del usuario durante la instalación y ajustar el reinicio automático tras la instalación:

```
msiexec.exe /i <ruta al paquete msi> [/quiet][/passive][/q{n|b|r|f}]
[/norestart][/promptrestart][/forcerestart]
```

Instalación desatendida de paquetes .msi con msiexec

Vemos el significado de los parámetros según su tipo:

Ajustes de interfaz de usuario:

- **/quiet** → Instalación sin interacción del usuario
- **/passive** → Instalación desatendida con barra de progreso
- **/q** → Ajuste de interfaz de usuario
  - **n** → Sin interfaz de usuario
  - **n+** → Sin interfaz de usuario salvo un mensaje al final
  - **b** → Interfaz de usuario básica
  - **r** → Interfaz de usuario reducida
  - **f** → Interfaz de usuario completa

Ajustes de reinicio:

- **/norestart** → Sin reinicio
- **/promptrestart** → Consultar si se quiere reiniciar
- **/forcerestart** → Reinicio automático

Combinando adecuadamente estos parámetros podemos realizar una instalación desatendida a partir de un fichero msi como en el siguiente ejemplo:

msiexec /i "libreoffice_6.1.3_win_x64.msi" /qn /norestart

Gestor de software Chocolatey

Chocolatey es un complemento para powershell que nos permite instalar software por línea de comandos funcionando de una forma similar a los gestores de paquetes de Linux

Esto nos permite realizar la instalación de software tradicional en Windows **de forma desatendida**.

En entornos Windows proporciona una gran ventaja que ya teníamos en entornos Linux y soluciona algunas limitaciones de la instalación del software:

- Nos ahorramos el paso de localizar, descargar e instalar cada una de las herramientas software que queramos instalar y que estén en el repositorio de chocolatey
- Podemos actualizar el software instalado automáticamente evitando tener que realizar una descarga
- Esto supone un ahorro de tiempo para cualquier usuario, pero desde el punto de vista de un administrador este ahorro es mucho mayor, sobretodo en casos en los que tenga que administrar varias máquinas

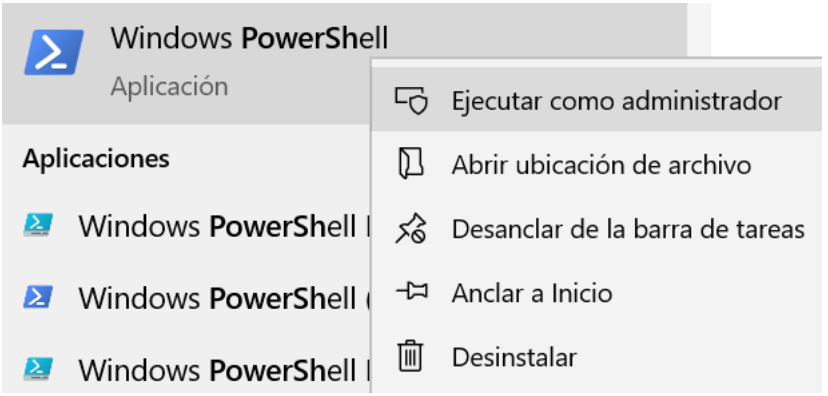
Comando choco

El funcionamiento de chocolatey es muy similar al apt-get de las distribuciones linux tipo debian/ubuntu y del dnf/yum de las RPM pero en este caso se utiliza el **comando choco desde una consola powershell**

Vemos algunos ejemplos de uso del comando:

- **choco install jre8** → Instala el Java Runtime Environment
- **choco install libreoffice -y** → Instala libreoffice sin pedir confirmación
- **choco upgrade firefox** → Actualiza Firefox
- **choco uninstall notepadplusplus.install** → Desinsatala el notepad++
- **choco search xnview** → Busca software a instalar con el texto xnview
- **choco upgrade all -y** → Actualiza todo el sotware administrado con chocolatey

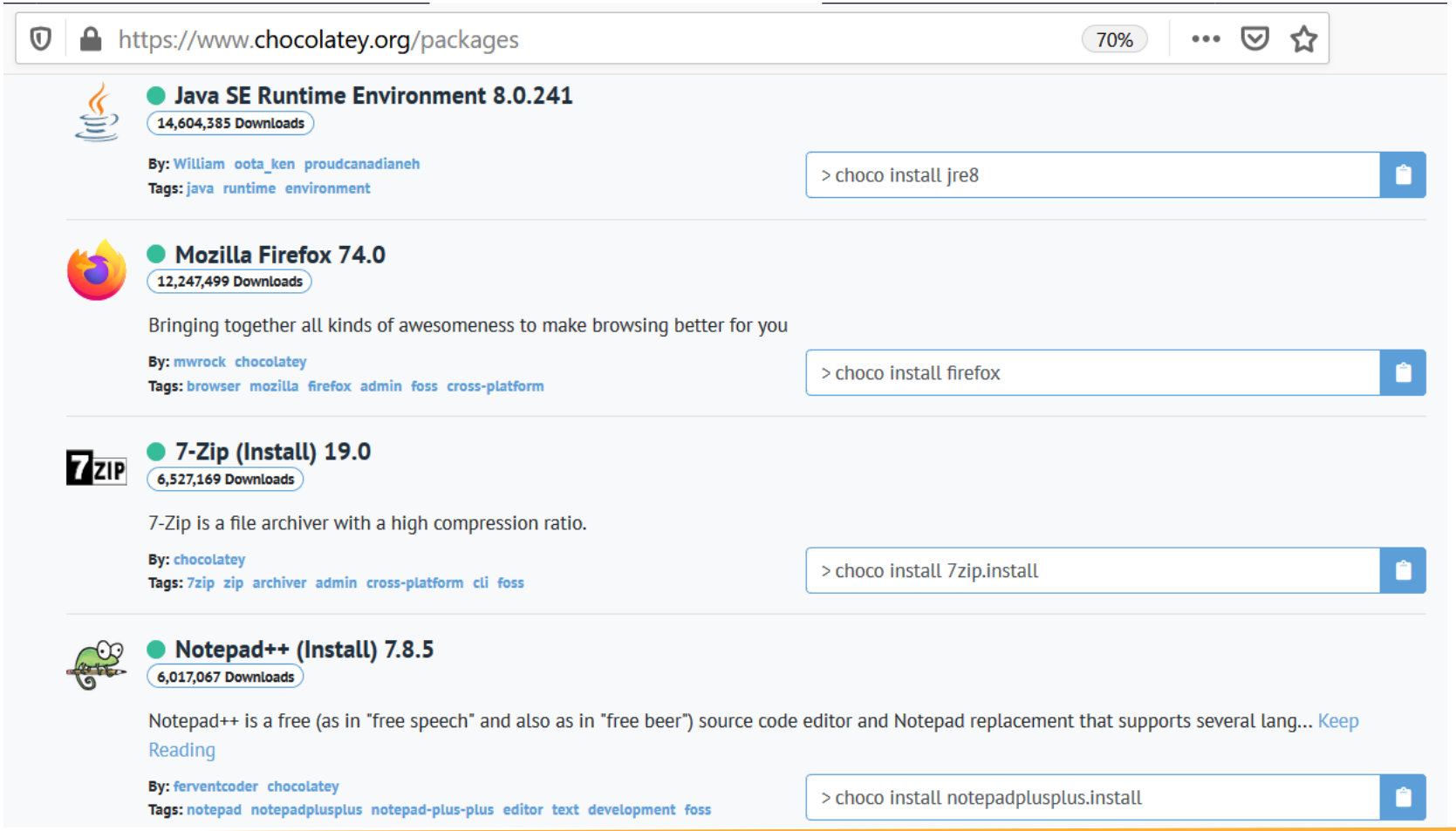
Para ejecutar el comando choco deberemos **abrir una consulta de Powershell como Administrador**, al requerirse privilegios para la instalación de software:



Repositorio de chocolatey

El repositorio de chocolatey dispone sólo de software gratuito entre el que encontramos muchas herramientas de uso habitual. Podemos consultar el disponible en https://chocolatey.org/packages .

Aquí podremos ver el nombre del paquete que tendremos que pasar al comando choco para realizar operaciones de instalación, actualización o desinstalación:



ChocolateyGUI

Chocolatey dispone de un gestor de paquetes gráfico denominado **chocolateyGUI** en el que podremos consultar el catálogo, instalar y desinstalar aplicaciones gestionadas con chocolatey e incluso lanzar una actualización de todo el software.

Lo podremos instalar con el comando **choco install chocolateygui**