# Descubriendo Linux:
## ¿Qué debes saber si no sabes nada?

Mª Dolores Nogueras y Antonio Gómez

**Es curioso cómo damos por sentado, cuando hablamos de Linux, que todos los que nos oyen o leen tienen la formación mínima para entender todo aquello que estamos diciendo. Uno de los principales escollos, a nuestro entender, para realizar el salto mortal sin red que el usuario cree que supone descubrir Linux es la falta de información general sobre sus características y posibilidades. Este artículo pretende ofrecer una introducción a tan maravilloso mundo, desde la perspectiva de la práctica educativa. ¿Qué debemos saber de Linux para presentárselo a nuestros alumnos?**

es@lpmagazine.org

Cuando empecé a interesarme por Linux como alternativa de trabajo desde mi PC, era (eso creía) un usuario experimentado en informática. Sin tener una formación específica ortodoxa, una mezcla de intuición e inconsciencia me guiaban a través de los problemas que me iba encontrando con la generación y administración de documentos, realización de prácticas de todo tipo con el alumnado (desde la consabida realización del primer trabajo en procesador de textos, a la programación de tarjetas controladoras desde los primeros Windows 3.0), webquests, etc... Intuición porque la llegada de los entornos gráficos de escritorio a principios de los 90 daban más facilidad para orientar al usuario sobre qué estaba pasando en cada momento. Inconsciencia porque, en el fondo (era joven y aventurero), me importaba un pito qué pasaría a continuación. Si sucedía lo que yo esperaba, genial. Si había algún error, con suerte, habría guardado el archivo correspondiente. Si no había suerte, todo se había perdido, y tocaba empezar de cero. Y normalmente no me importaba. Era genial. Pensar que una máquina como mi ordenador (un Pentium a ¡100! MHz), normalmente, exigiría como medida extrema un simple formateo y reinstalación del software, y sólo como último recurso, era hipnóticamente atrayente para un legendario "manazas" como el que suscribe, que sólo pulsando un interruptor podía averiar el sistema eléctrico y dejar sin suministro a la casa de sus padres (y les puedo asegurar que no miento).

Pero ya estoy divagando, y sólo estamos en la introducción. El caso es que cuando me empecé a interesar por Linux, hará ya unos cinco o seis años, me encontré con miríadas de información. Ése es uno de sus principales atractivos, al tiempo que, en ocasiones, un obstáculo: la libertad para difundir, compartir y modificar la información. Sumémosle el hecho de que la mayoría de

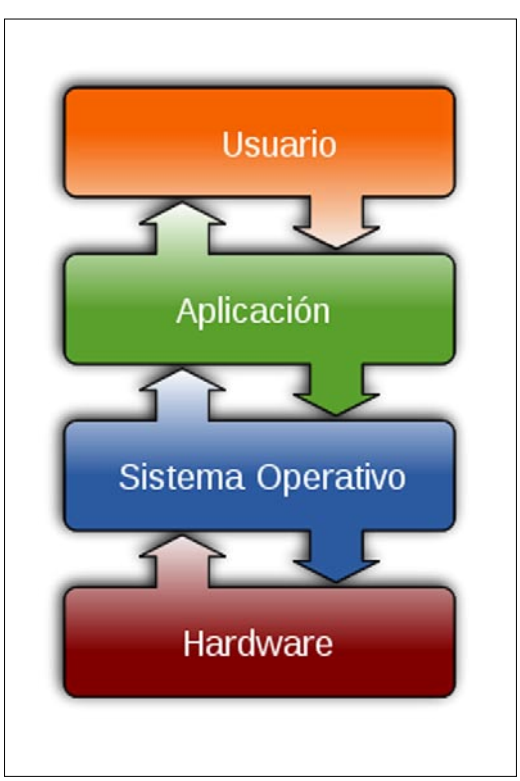


**Figura 1.** Un sistema operativo opera en una capa intermedia entre las aplicaciones controladas por el usuario (software) y el equipo (hardware). Fuente: Wikipedia

la información está generada por expertos, y orientada al consumo por parte de expertos. En mi caso particular, superé ese primer problema mezclando a partes iguales dos de los más enternecedores aspectos de mi personalidad: cabezonería e ilusión.

Hoy en día, sin atreverme a considerarme un "gurú", ni mucho menos (ya puedo oír las carcajadas de fondo de algunos lectores que ya han visto nuestro trabajo en anteriores números), sí podríamos decir que sabemos, al menos, por donde empezar en este mundillo, así como ayudar a introducirse en él a compañeros profesores y maestros, así como a alumnos, que nunca hayan tomado contacto con el software libre. A lo largo de este artículo, que sólo pretende ser un primer paso en un viaje tan adictivo como largo, trataremos de orientar al profesor medio sobre el uso de Linux, así como el modo en que creemos que debería presentarse a un alumnado estándar entre los doce y los dieciocho años, que normalmente tiene ya nociones básicas e incluso medias sobre el uso de un ordenador. Hablaremos de lo que es un sistema operativo, la organización de archivos en Linux, el concepto de multiusuario, los escritorios Gnome y Kde, el sistema de instalación de nuevo software y repositorios, e incluso haremos una breve mención al sistema de consola BASH.

## Clarificando conceptos: El sistema operativo

Empecemos por el principio. ¿Qué es un sistema operativo? ¿Lo tenemos claro? ¿Estamos transmitiendo a los alumnos este concepto de forma correcta?

En un principio, un sistema operativo es software. Es el primer programa que se carga en el ordenador, y su misión es actuar como intermediario (interfaz) entre el software que utiliza el usuario y el hardware que compone el equipo. Explicado así, el sistema operativo no abarca todas las herramientas que esperamos encontrar cuando lo instalamos en nuestro PC. Estamos acostumbrados a esperar que un sistema operativo (como los denostados *WINDOWS* de Microsoft) incluya otras herramientas: un procesador de textos sencillo, un visualizador de imágenes, un explorador de archivos, etc., integrados, además, en una aplicación gráfica (lo que denominamos comúnmente un *escritorio*) que permite la interacción del usuario con el ordenador de manera sencilla, ratón mediante. Sin embargo, dichas herramientas son complementarias. El sistema operativo como tal se denominaría *NÚCLEO*.

Una buena (reconozcámoslo, muy buena, genial) estrategia de marketing y publicidad por parte de Microsoft, hizo que en los años 90, a ojos del usuario inexperto, Windows fuera, no un sistema operativo, sino *el sistema operativo*. La ecuación era sencilla: me he comprado un ordenador, tengo que instalar unos programas para trabajar o jugar, y para que esos programas funcionen, el ordenador debe tener instalado el sistema operativo *Windows* (el 3.1 al principio, el 95 de infausto recuerdo después, algo mejorado por el 98, hasta llegar a XP; a partir de ahí, los redactores de este artículo hemos perdido la pista a las hazañas de Microsoft, porque descubrimos el mundo del software libre; eso sí, aún nos estamos riendo del fiasco *Windows Vista*). Hoy en día, aunque de un modo desesperantemente lento, el usuario medio va asimilando que en realidad, no estamos hablando de una elección obligada, y que existen múltiples opciones alternativas.

Volviendo un poco a la perspectiva del usuario inexperto, de todos modos, es normal esperar que aparte de dicho núcleo, se instalen de modo complementario otras herramientas, juzgadas imprescindibles por el aficionado novato. Por ello, en el mundo de GNU/Linux, se habla de *DISTRIBUCIONES* o *SABORES*: conjunto de programas complementarios al núcleo de Linux, que suelen incluir, como mínimo, un sistema de escritorio que suponga una interfaz gráfica; un procesador de textos, un explorador de archivos, y un programa navegador de Internet. También suele ser común incluir una consola o terminal, que permita al usuario más experto un trabajo a más bajo nivel con el ordenador, así como algún sistema de ayuda y/o documentación con toda la información relativa al software incluido.

## Algo de historia. ¿De dónde proviene GNU/Linux?

Efectivamente. Una duda común a todos los aficionados que se inician en este mundo da vueltas al origen de esta forma de software. ¿Software libre siempre quiere decir gratis? ¿Puede haber profesionales que se dediquen a esto sin retribución alguna? ¿De qué viven entonces? ¿O quizás estamos hablando del nada confiable trabajo de simples aficionados que juegan a emular las hazañas de

**Figura 2.** Richard Stallman, padre del proyecto GNU, tiene fama de ser notablemente excéntrico. No utiliza normalmente un entorno gráfico en su ordenador, no utiliza Internet salvo para enviar correos...

**Figura 3.** Linus Torvalds completa el tándem formado por todas las herramientas GNU al concebir un núcleo compatible con ellas y licenciarlo como software libre

sus idolatrados programadores, hordas de *freaks* con acné, gafas de pasta arregladas con esparadrapo, y poseedores de títulos como *mago elfo de nivel 15*?

Vale. Vale. Respiremos. Empecemos por el principio. Y eso implica hablar forzosamente de Unix, por un lado, y de *Richard Stallman*, por el otro.

En los años sesenta, setenta y ochenta, los primeros ordenadores no estaban, desde luego, diseñados para su uso por parte de gente que no fuera un programador experto. Eran prototipos carísimos, enmarcados en experimentos de Universidades y proyectos de I+D de grandes multinacionales, que si bien representan un porcentaje infinitesimal de la potencia de computación de nuestros equipos actuales, suponían un paso de gigante en el progreso de las Tecnologías de la Información (concepto que ni siquiera existía como tal). Había un mismo equipo para decenas de programadores y trabajadores, cada uno de los cuales trabajaba con su propia cuenta de usuario, introducía su contraseña, trabajaba con sus datos, y cerraba su sesión, cediéndole su turno al siguiente investigador. Faltaban aún muchos años para concebir siquiera que pudiera haber un ordenador doméstico a disposición de un usuario con pocos conocimientos informáticos en su propio hogar.

En este marco brilló con luz propia el sistema operativo Unix, que cobró rápidamente prestigio de ser un sistema estable. Nacido originariamente en el seno de los Laboratorios Bell de AT&T y General Electric, se define como un sistema operativo portable (se puede ejecutar en múltiples plataformas), multiusuario y multitarea. Es de base comercial, es decir, se considera como software de pago. Sin embargo, los sistemas operativos con base GNU/Linux, base de este artículo, imitan (clonan) el comportamiento de este sistema, y en un principio, no se exige remuneración económica. ¿Cómo hemos llegado a esta situación?. Sigamos nuestro repaso histórico...

Stallman trabajaba en el *MIT (Masachussets Institute Technology)* como programador en el Laboratorio de Inteligencia Artificial, ya en el primer año de sus estudios de Físicas. En aquellos tiempos, el software era libre de origen, es decir, se compartía por sistema toda la información. La constante presión de la cultura comercial, sin embargo, erosionó esta concepción hasta el punto de que algunos compañeros de Stallman fundaron su propia compañía, *Symbolics*, al objeto de ir sustituyendo el software libre utilizado por el Laboratorio por sus propios productos comerciales (esto es, de pago). Stallman se opuso todo lo que pudo, pero acabó quedándose solo. Como reacción, en 1983, inició el movimiento GNU.

GNU es el acróstico de *GNU is Not Unix*, en un juego de palabras, todavía no sabríamos decir si intraducible, o simplemente un sinsentido que trata de ser humorístico. El objetivo básico de dicho proyecto consiste en conseguir un sistema operativo totalmente libre. Para ello, en un alarde de energía y mostrando una voluntad de trabajo titánica, se dedicó, con ayuda de quien quiso colaborar, a reescribir en lenguaje C varias herramientas que repiten las funciones del sistema operativo UNIX. Actualmente destacan, entre dichas herramientas, el compilador para lenguaje C *gcc*, el terminal de consola *BASH*, el procesador de textos *Emacs* o el famoso *GIMP*.

Pero con todo ello, entre todas estas herramientas que puede incluir el sistema operativo libre con el que soñaba Stallman, seguía faltando un *núcleo*, un *kernel* que se encargara de la comunicación entre software y hardware del ordenador. Ahí es donde entra Linus Torvalds.

Estudiante finlandés de 22 años, Torvalds estaba muy interesado en el por entonces reciente sistema operativo *Minix*, de Andrew Tannenbaum, de tipo comercial y con propósitos didácticos. Como parte de sus estudios, se inspiró en dichas ideas para desarrollar un núcleo desarrollado en lenguaje C, que en su primera versión, en 1991, podía ejecutar el terminal de consola *BASH* y el compilador *GCC*.

Sólo faltaba el paso definitivo, dado en 1992. Torvalds adoptaba para su trabajo una licencia *GPL (General Public License)*, encuadrada en el movimiento GNU, por la que se otorgan derechos de libre modificación, redistribución, copia y uso ilimitados para todos los usuarios.

No hace falta extenderse mucho más para que el lector deduzca por su cuenta que ambos proyectos se complementaban en un sistema operativo acorde con la filosofía del software libre. Había nacido *GNU/Linux*.

## ¿Todo es gratis?

Bueno, no. A veces lo parece, máxime cuando dentro del concepto *Free Software* hacemos la traducción del inglés, *free*, que es un vocablo polisémico entre los anglosajones. Puede significar *gratis*, pero también *libre*, este último significado es el que realmente deberíamos tomar.

Concretamente, el objetivo último del software libre es el de poner a disposición de cualquier interesado toda la información, el

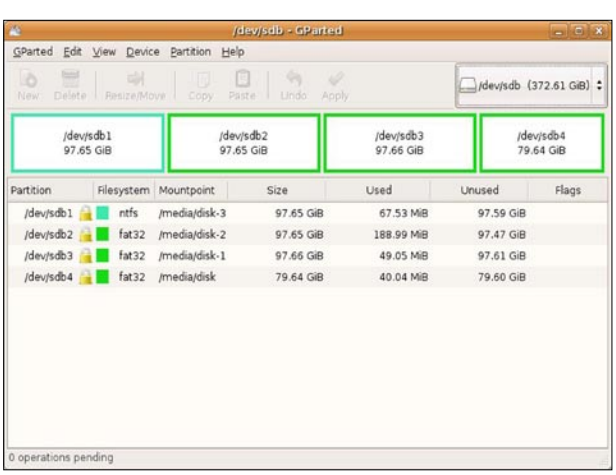


**Figura 4.** Gparted nos ayuda a organizar nuestros discos duros en particiones con distintos formatos. En la imagen, en el disco duro calificado como sdb, hay cuatro particiones, en este caso, todas para Windows

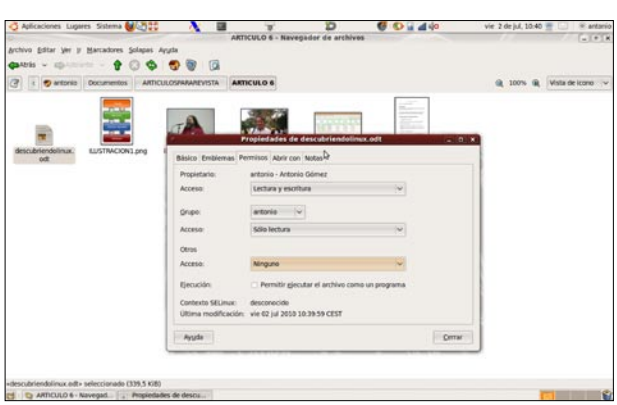

**Figura 5.** Podemos modificar de modo sencillo los permisos de un archivo desde la pestaña Propiedades del escritorio

código fuente, si se prefiere, más la documentación complementaria relacionada, de la utilidad informática de la que estemos hablando en ese momento. De ahí el término *Open Source*. Pero eso no conlleva una obligada relación *software libre-gratuidad de la aplicación*, si bien suele ser lo más normal.

Cuando surge esta duda al introducir en el mundo GNU/Linux a nuestros alumnos, muchas veces presentamos esta metáfora: *el funcionamiento de una lavadora o de un frigorífico puede o no interesarnos, pero si queremos saber cómo funciona, nadie nos lo oculta. Si una persona con suficiente interés quiere construir su propio electrodoméstico, sólo tendría que comprar los materiales y la herramienta necesaria para ello. No es un secreto de estado el conjunto de principios físicos, eléctricos y mecánicos en los que se basa el funcionamiento de estos aparatos, aunque otra cosa será el que nos compense más pagar a una empresa especializada por adquirir un producto fabricado por profesionales, con todas las garantías.*

Precisamente, una concepción equivocada de este mundo suele llevar al profano a una conclusión errónea: *de acuerdo, la colaboración desinteresada de la comunidad hacker implica el desarrollo de múltiples herramientas muy interesantes, pero en el mundo real, no creo que puedan compararse con aquellas desarrolladas por una empresa que trabaja por un beneficio económico, y que por lo tanto invertirá dinero y recursos en el desarrollo de dichas aplicaciones.*

¿Por qué decimos que es una concepción equivocada? Porque es falso que no pueda haber un beneficio económico detrás del software libre. Hay varias compañías multinacionales que impulsan y desarrollan proyectos *Open Source* de distintos calados y características. Sin ir más lejos, tenemos a Mark Shuttleworth, propietario de *Canonical Ltd*, empresa madre de Ubuntu, o el caso de la empresa *Sun*, desarrolladora de la suite ofimática *OpenOffice.org*.

En un artículo calificado por varios profesionales como imprescindible, *Setting Up Shop: The Business of Open-Source Software*, Frank Hecker distingue varios modelos de negocio alrededor del software libre. No es objeto de nuestro texto extendernos sobre este tema, pero sí que podemos mencionar un par de ejemplos bastante ilustrativos al respecto:

- **Support Seller**: en este caso, el beneficio económico se obtiene, no de la venta de la aplicación informática, sino de servicios de soporte y mantenimiento relacionados con el producto.
- **Loss Leader**: el producto de software libre es una versión operativa que permite al público conocer las posibilidades de la aplicación. Sin embargo, se ofrecen como privativas otras versiones mejoradas que añaden múltiples mejoras atractivas para nuestros potenciales clientes.
- **Sell it, free it**: la empresa vende un producto como privativo, a cambio de un beneficio económico. Cuando amortiza el programa, desarrolla una versión mejorada de dicho producto privativo a partir de los beneficios obtenidos. A continuación, libera (convierte en *Open Source*) la versión amortizada de la aplicación.

**Figura 6.** Compiz en Gnome proporciona efectos de escritorio espectaculares que nada tienen que envidiar a otras aplicaciones similares

## Funcionamiento y características de un sistema operativo GNU/Linux

Hemos presentado, hasta aquí, la filosofía de base del movimiento en torno al software libre. Pero lo que realmente interesa al aficionado que no ha trabajado nunca con GNU/Linux es saber cómo funciona un sistema operativo de este tipo y cómo funcionan los elementos que constituyen su base, sea la distribución que sea: Debian, Ubuntu, Mandriva, Fedora, Slackware, y decenas (cientos) más.

Como ya se ha mencionado antes, una distribución o sabor es el conjunto del núcleo o *kernel* Linux con varias herramientas complementarias GNU. Por lo tanto, y aunque muchos lectores expertos pueden no estar de acuerdo con esta idea, nos atrevemos a afirmar que para el usuario medio o incluso experto, acabaría siendo indiferente la distribución con la que se empezara a trabajar: al final, seguirá utilizando como base el mismo kernel, y reunirá, entre las herramientas con las que viene la propia distribución, y las que consiga a través de los repositorios o de la red, el conjunto de aplicaciones informáticas con las que realmente necesite trabajar. Así, por ejemplo, podemos empezar con una distro (distribución) Ubuntu, que popularmente se concibe como una de las más amigables para el no iniciado, y que incorpora por defecto un escritorio *Gnome*, y cambiarlo por el más espectacular y agradable a la vista *KDE*. Si no nos gusta el navegador *Firefox* que incorpore, podemos cambiarlo por *Opera, Chrome,...* las posibilidades son infinitas.

En los próximos apartados, trataremos de hacer un repaso de las características comunes a todas las distribuciones con base GNU/Linux, a efectos de proporcionar una base teórica con la que el nuevo aficionado pueda sentirse más seguro en su viaje por nuestro


**Figura 7.** KDE, con el software explorador tanto de archivos como para Internet, Konqueror, es considerado tradicionalmente como más atractivo gráficamente y con más opciones de configuración

mundo. Empezaremos con la preparación del disco duro, su forma de organizar los archivos, los tipos de usuario y los correspondientes permisos de lectura/escritura, y a continuación pasaremos a analizar los dos escritorios más populares (al menos para nosotros): KDE y GNOME. Analizaremos el sistema de instalación de aplicaciones en el S.O. en base a los repositorios, para terminar echando una ojeada sobre el terminal BASH (uno de los más populares), que suele ser la bestia negra que distinguirá al usuario *noob* (novato) del que realmente empieza a avanzar como *linuxero*. Pasen y vean.

## Particiones y formatos lógicos en GNU/Linux. Instalación del sistema operativo

Hablaremos en estas líneas del modo que tiene GNU/Linux de organizar y localizar la información en los distintos archivos en sus sistemas de almacenamiento, sean discos duros, lápices USB, disquetes (¿alguien los utiliza todavía?), etc.

En primer lugar, hay que aclarar que el no iniciado debe olvidarse por completo de lo que estaba acostumbrado a ver en Windows. El mítico *Mi PC* nos ofrecía en una ventana una imagen rápida de los distintos sistemas de almacenamiento antes mencionados, nombrados alfabéticamente: *C:/*, *D:/*, etc., para discos duros, *A:/* para disquetes, y así sucesivamente.

En Linux, por sistema, se ignoran los discos que no se utilicen. Los que se utilizan, se dice que *están montados* (es decir, se destina parte de los recursos del ordenador a acceder a su contenido). El resto, simplemente se ignoran, logrando una mayor eficiencia en la gestión de la memoria. Pero volveremos más tarde sobre ello.

El caso es que los discos duros, sobre los que normalmente se asentarán los sistemas operativos, se organizan según un *formato lógico* que asignará unos sectores del disco a unos tipos de archivo en particular. Cuando queremos utilizar más de un sistema de archivos en un mismo disco duro (separar software de datos, o instalar más de un sistema operativo), debemos dividir previamente el disco duro en más de una *partición* (división dentro del disco duro).

Windows utiliza los formatos lógicos FAT, FAT16, FAT32 y NTFS. Linux, en cambio, utiliza los formatos EXT2, EXT3 y EXT4, entre otros.

La instalación de cualquier sistema operativo GNU/Linux pasará por la división del disco que la va a alojar, al menos en dos particiones distintas: la partición raíz (*/*), y la partición *swap* o memoria de intercambio. En el siguiente apartado nos ocuparemos de la primera. En cuanto a la memoria de intercambio, baste decir que es un apartado dentro del disco duro que se reserva para procesos poco activos, pero presentes, dentro de la memoria, de modo que estén disponibles para cuando se necesiten, pero al mismo tiempo no ocupen espacio dentro de la memoria RAM con la que trabaja directamente el microprocesador.

En GNU/Linux, herramientas como Gparted nos ayudan a organizar y administrar particiones y formatos lógicos dentro del disco.

## Sistema de archivos en GNU/Linux, Superusuario y usuarios, Permisos

Como se decía en el anterior apartado, asignábamos a los *formatos lógicos* la responsabilidad de organizar la información dentro de la partición en un determinado sistema de archivos y carpetas. En concreto, los formatos EXT* disponen de un árbol de directorios estandarizado e inamovible en la partición raíz (/):

- /bin: para guardar comandos de usuario,
- /boot: información sobre el arranque del sistema,
- /dev: dispositivos presentes en la máquina (montados y no montados),
- /etc: archivos de configuración de las aplicaciones,
- /home: carpeta reservada para alojar los distintos documentos de cada usuario. Hay una carpeta por usuario, que es el único con permisos de escritura en dicha carpeta,
- /lib: para alojar librerías necesarias en distintas aplicaciones,
- /media: a través de esta carpeta, se puede acceder a la información de los dispositivos presentes en /dev que hayan sido previamente montados,
- /mnt: actualmente no tiene mucha utilidad. Es el predecesor de /media,
- /proc: aloja información sobre el estado de los procesos y el kernel,
- /root: reservado para el administrador del sistema.
- /sbin: aloja comandos disponibles sólo para el administrador (root),
- /tmp: carpeta disponible para archivos temporales,
- /usr:en esta carpeta se alojan todos los archivos que se juzgan

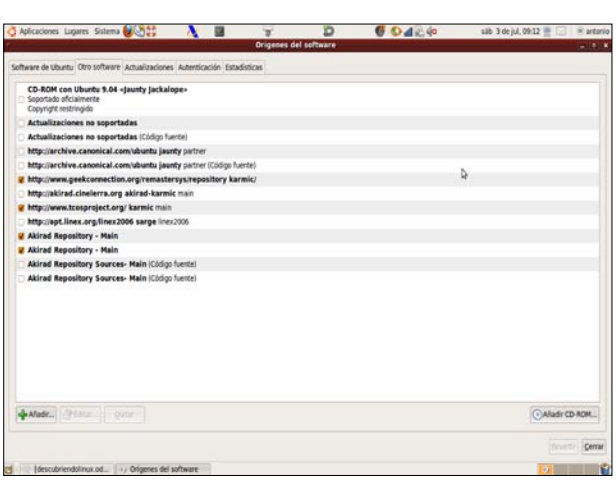


**Figura 8.** En todos los sistemas operativos GNU/Linux podemos añadir nuevas direcciones de repositorios

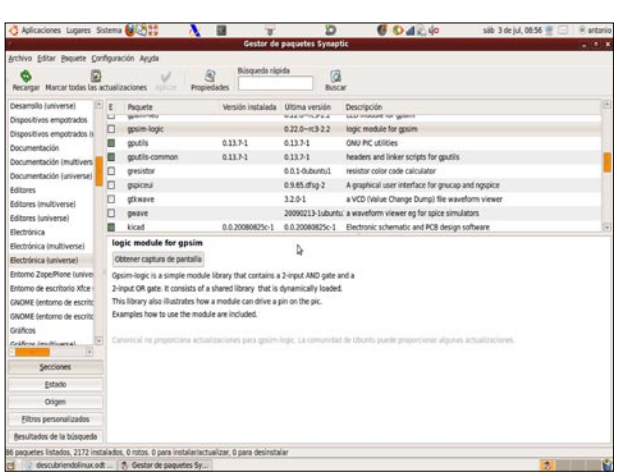


**Figura 9.** Los gestores gráficos de paquetes como Synaptic dan cumplida información de los paquetes instalables, instalados, rotos,... así como de las dependencias cumplidas y por cumplir

relativos a aplicaciones del usuario, documentación, etc. En fin, todo lo que no es encuadrable en el resto de carpetas,
- /var: en esta carpeta se aloja información de tipo variable: diarios o logs de sistema, cuentas de correo, sitios web alojados en un servidor dentro del ordenador, etc.

El sentido de esta organización es el de tener asignado un lugar específico para cada tipo de archivo. En consecuencia, aunque un usuario, sea gráficamente o a través de consola, pueda acceder a otras carpetas aparte de la que corresponde realmente a su cuenta (*/home/nombredeusuario/*), no podrá estropear nada accidentalmente o a propósito, puesto que sólo el usuario con permisos de superusuario o administrador (el consabido *root*) tendrá atribuciones para modificar archivos en las carpetas. Además, `siempre` se le solicitará la introducción de su contraseña de superusuario, lo que añade un grado más de seguridad a dicho sistema de archivos. En otro orden de cosas, el avispado lector puede empezar a imaginar ya la dificultad de intentar introducir *malware* o software malicioso en un ordenador organizado de este modo. Aquí, un virus tiene mucho más difícil el instalarse por su cuenta oculto en el seno de otro programa, entre otras razones, que luego se verán, porque se vería obligado a denunciarse a sí mismo al pedir al usuario la contraseña de administrador para obtener permiso de instalación. De todos modos, y como se verá después, la filosofía de trabajo a la hora de instalar nuevo software también varía bastante. Eso sí, ¡cuidado!. No pretendemos decir con esto que GNU/Linux sea invulnerable a ataques malintencionados, porque la vulnerabilidad siempre está sujeta a las reacciones del propio usuario. Sí podemos afirmar que será un sistema estable, que bien utilizado, dificultará muchísimo el acceso externo por parte de personas no autorizadas a nuestro equipo.

Hablemos ahora de los permisos de usuario. En GNU/Linux, sólo un usuario tiene atribuciones de lectura y escritura sobre las carpetas y archivos fuera de la carpeta de su propia cuenta en */home*: el administrador o superusuario, denominado *root*. Cualquier intento de creación, modificación o borrado de un archivo, así como de instalación de nuevo software, exigirá la introducción previa de la contraseña de superusuario. A nivel de consola, cualquier comando que suponga trabajar a este nivel, deberá ser precedido del término *sudo*, que indica que la orden tecleada corresponde a un superusuario que espera se le pida el password para identificarse.

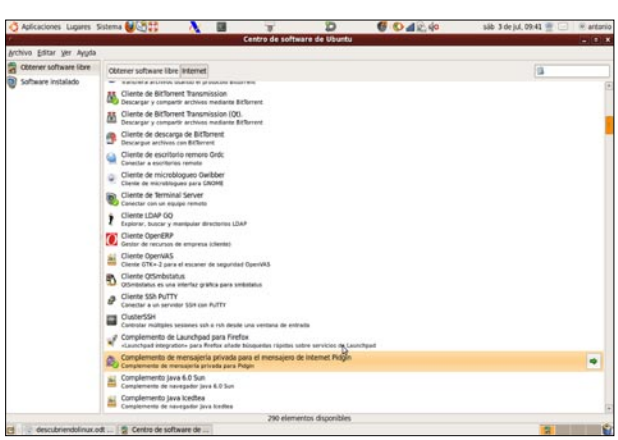

**Figura 10.** La mayoría de las distribuciones suelen incorporar alguna utilidad sencilla de instalación de nuevo software mediante paquetes

Fuera del *root*, en GNU/Linux se consideran tres tipos de identificación de usuario:

- Propietario del archivo o carpeta,
- Grupo de usuarios al que pertenece el propietario,
- Otros usuarios no enmarcados en los grupos anteriores.

Cada uno de los cuales puede tener, o no, permiso para:

- Leer el archivo,
- Escribir o borrar el archivo,
- Ejecutar el archivo.

Cada archivo tiene asignado un permiso de lectura/escritura/ejecución para cada uno de los tipos de usuario antes mencionado, según esta terminología numérica: 0 para ningún tipo de permiso, 1 para lectura, 2 para escritura, y 4 para ejecución. Según los permisos que se asignen a cada tipo de usuario, que se suman entre sí, tendremos una tríada que oscilará entre 000 y 777.

Comprendemos que es un concepto algo difícil al principio; por eso queremos ilustrarlo con un ejemplo. Supongamos que tenemos dos grupos de usuarios asignados en un PC con kernel Linux: *profesores* y *alumnos*. En el primer grupo, tenemos los usuarios *profesor1* y *profesor2*. En el segundo grupo, tenemos los usuarios *alumno1* y *alumno2*. Imaginemos que el usuario *profesor1* es poseedor del archivo *examenhistoria.txt*, en su carpeta de cuenta */home/profesor1*. Por otro lado, el usuario *alumno1* es el propietario del archivo ejecutable *chuletasparaexamen.sh*, en su carpeta */home/alumno1*.

En un principio, el archivo *examenhistoria.txt* tiene los permisos 710. ¿Qué permisos tendrán los otros usuarios sobre dicho archivo? Analicémoslo. Según lo expuesto, el sistema numérico indica los permisos, por orden, para el propietario, para el grupo, y para otros usuarios. Por lo tanto:

- El propietario, *profesor1*, tiene asignado el número 7, que corresponde a los permisos sumados 1+2+4. Esto es, *profesor1* tiene permisos de lectura, escritura y ejecución (aunque el archivo es de texto, no es un ejecutable en este ejemplo) sobre *examenhistoria.txt*

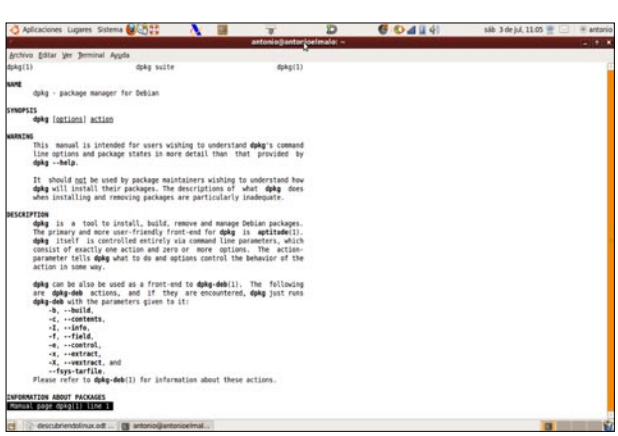

**Figura 11.** Aunque algo incómodo de utilizar al principio, el terminal de consola demuestra ser mucho más potente que su homólogo gráfico. En la ilustración, el comando man proporciona toda la información sobre el gestor de paquetes dpkg

- Los otros usuarios del grupo de *profesor1*, *profesores*, tienen el permiso 1 sobre ese mismo archivo. Es decir, como no se puede dividir este permiso como suma de otros números, el otro usuario del mismo grupo (*profesor2*) tiene permisos de lectura sobre *examenhistoria.txt*. El usuario *profesor2* podrá leer, aunque no modificar, dicho archivo.
- Los usuarios *alumno1* y *alumno2*, que no pertenecen al grupo antes mencionado, tienen permiso 0 (ningún permiso) sobre el archivo. No podrán leer, ni mucho menos modificar o ejecutar, un archivo con ese número de permiso.

Siguiendo con el ejemplo, el archivo *chuletasparaexamen.sh*, un ejecutable que (por ejemplo) generara "apuntes de apoyo" para el próximo examen del alumno, tiene asignado el número de permiso 541. En base a eso:

- El propietario, *alumno1*, tiene el permiso 5, como resultado de sumar 4+1. Esto es, *alumno1* puede ejecutar este archivo, así como leerlo para poder analizar su estructura de programación.
- Los otros usuarios del mismo grupo *alumnos*, *alumno2* en este caso, tendrán el permiso 4. Esto es, *alumno2* también podrá generar "apuntes de apoyo", si bien no podrá leer ni mucho menos modificar dicho archivo.
- Curiosamente (y a efectos ilustrativos de nuestro ejemplo), el permiso asignado a otros usuarios es 1, que correspondería a la lectura del archivo para un posible análisis, aunque no parezca lógico darle un permiso que no tienen los usuarios del mismo grupo que el propietario.

Gráficamente, (usando escritorios como Gnome o Kde), podemos ver y, si somos propietarios, modificar estos permisos mediante el ya familiar proceso de hacer clic derecho–>Propiedades–> Hacer clic en la pestaña *Permisos*. Si trabajamos a través de terminal, el comando utilizado es *chmod* (ejemplo: chmod 777 chuletasparaexamen.sh dará todos los permisos a todos los usuarios, si nos hemos identificado como *alumno1*. Si no somos tal usuario, se nos dará un mensaje de error).

## Entornos de escritorio

Las distribuciones GNU/Linux actuales cuentan con un sistema gráfico que permite al usuario un manejo amigable, fácil e intuitivo del ordenador. Con base en el software gráfico X Window System, orientado a Unix, y que va en la actualidad por la versión X11, los entornos de escritorio cuentan con un sistema de ventanas de idéntico (prácticamente) funcionamiento a los S.O. Windows y MAC OS (minimizado, maximizado, cierre de ventanas, menús desplegables, teclas de atajo de teclado, fondos de escritorio, colores, etc.).

Muy interesante resulta recalcar que, a diferencia de los otros sistemas operativos no libres, queda en manos del usuario elegir qué entorno de escritorio desea utilizar. Si no queda satisfecho con el que viene por defecto con la distribución elegida (por ejemplo, Gnome para Ubuntu, o KDE para Mandriva), puede utilizar el sistema de repositorios (en próximos apartados se explica) para instalar otro u otros de su agrado. Puede incluso mantener en convivencia más de un entorno, eligiendo cuál le apetece usar en cada inicio de sesión de usuario.

El sistema de trabajo por menús de todos los escritorios, salvando las lógicas distancias, es altamente intuitivo, incluso para el usuario no iniciado. Normalmente, se hará una división de menús por tipo de posibilidad (separando aplicaciones, explorador de archivos y utilidades de configuración del sistema). Algunos ejemplos de entornos de escritorio son:

- KDE y GNOME (ampliamente mencionados desde el principio del artículo),
- Window Maker,
- Ice Wm,
- Blackbox,
- Flushbox.

Tradicionalmente, se ha sostenido que KDE es gráficamente más rico y vistoso que Gnome, además de otorgar muchas más posibilidades de configuración al usuario, si bien exige más recursos al sistema, y Gnome es mucho más estable. En la actualidad, aplicaciones como *Compiz* añadidas al software original, mejora la apariencia gráfica y acerca visualmente ambos escritorios.

## Instalación de nuevo software, Filosofía de los repositorios, Gestión de paquetes, Synaptic y Rpmdrake

La instalación de nuevo software en un S.O. GNU/Linux centra su base en la utilización de los *repositorios*. Un repositorio viene a ser un sitio con una dirección web determinada donde se almacena y mantiene información digital, normalmente en forma de archivos informáticos.

Cuando nuestro ordenador está equipado con un sistema operativo libre, dispone de una lista de direcciones de repositorios, normalmente en el archivo /etc/apt/sources.list (el lector puede consultarlo, si bien no modificarlo directamente, a causa del establecimiento de permisos de archivo mencionado en anteriores apartados).

El sistema de repositorios utilizado en GNU/Linux permite la instalación de nuevo software mediante la gestión de paquetes. Según esta definición, un *paquete* consistiría en un conjunto de software, normalmente encapsulado en un único fichero, que incluiría información adicional como el nombre completo, la identificación del distribuidor, una descripción de sus funciones, y sobre todo, la relación de otros paquetes que se requieren para el correcto funcionamiento del software en cuestión. Lo que nos lleva al concepto de *dependencia*. En GNU/Linux, como en cualquier otro sistema operativo, si nos hace falta un software determinado para conseguir que otro funcione, y no lo tenemos instalado, lógicamente no funcionará. Pero si en el paquete que estamos instalando se hace una referencia a dicho software del que *dependemos*, es posible que el propio proceso de instalación localice, descargue (cuando hablamos de Internet, un 90% de las veces) e instale esos paquetes complementarios.

De este modo, el usuario no necesita saber nada sobre métodos de instalación, localizar el archivo dentro del conjunto que inicia el proceso, decidir la carpeta donde se guardará el programa... El propio sistema operativo, una vez se le ha indicado el software que se desea, buscará en su base de datos el nombre del paquete, localizará el repositorio donde se localiza dicho paquete, lo descar-

gará a memoria, leerá la información contenida en él y realizará la instalación, resolviendo las dependencias que dicho paquete señale (esto es, descargando e instalando los programas complementarios que dicho paquete declare necesitar).

Aunque uno de los repositorios normalmente suele ser el propio CD de instalación que hemos utilizado para instalar el sistema operativo que estemos utilizando, lo más normal es que la instalación de nuevos programas mediante paquetes requiera la utilización de Internet.

Los paquetes en Linux suelen tener la extensión *.deb, para sistemas operativos provenientes del sistema Debian (Debian, Ubuntu, Molinux, y varios más) y *.rpm, para los derivados del originario Red Hat (Fedora, Mandriva, Suse,...).

En ocasiones, el usuario que empieza a adquirir algo de experiencia requiere la instalación de un software que no está incluido en los paquetes universales que por defecto incorpora su distribución. En esos casos, una breve investigación en la red le permitirá conocer la dirección del repositorio que necesita utilizar, sea mediante consola, identificándose como *root* y modificando el archivo */etc/apt/sources.list*, sea utilizando el menú de configuración de opciones del sistema donde se haga referencia a tal posibilidad. Por ejemplo, en los sistemas operativos derivados de la distribución Debian, como Ubuntu o Molinux, dicha posibilidad se encuentra en *Sistema->Administración->Orígenes del software.*

Todos los entornos de escritorio incorporan una utilidad directa de instalación de nuevo software de manera gráfica, utilizando los repositorios. Sin embargo, el software, diríamos, básico en que se apoyan todas estas utilidades sigue siendo el elemento gestor de paquetes; por ejemplo, *Synaptic* en Gnome, *Rpmdrake* para Mandriva o *Yast* en OpenSuse.

Naturalmente, estas aplicaciones gráficas son implementaciones desde el escritorio de aplicaciones a más bajo nivel, escritas normalmente en C, que pueden correrse directamente bajo consola. El ejemplo con el que los redactores de este artículo nos sentimos más cómodos es con *dpkg*, el gestor de paquetes *deb*, que pueden utilizar las aplicaciones *apt-get* o *aptitude*, con la opción *install,* (otros son *yum* o *urpmi*). Veamos un ejemplo de consola (de la que hablaremos a continuación); supongamos que deseamos instalar un videojuego llamado *crafty*:

```
antonio@antonioelmalo:
   ~$ sudo aptitude install crafty
```

El comando sudo indica que nos deseamos identificar como usuario *root*, con permisos de administración:

```
[sudo]password for antonio:******
```

Introducimos nuestra contraseña; a continuación, se nos va informando del progreso de la instalación:

```
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
Leyendo la información de estado extendido
Inicializando el estado de los paquetes... Hecho
```

Se nos comunica la relación de paquetes que se procederán a instalar, para resolver todas las dependencias:

```
Se instalarán los siguiente paquetes NUEVOS:
  crafty crafty-books-medtosmall{a} gnuchess{a}
gnuchess-book{a} xaw3dg{a}
  xboard{a}
0 paquetes actualizados, 6 nuevos instalados, 0 para
eliminar y 0 sin actualizar.
Necesito descargar 5829kB de ficheros. Después de
desempaquetar se usarán 13,5MB.
¿Quiere continuar? [Y/n/?] y
```

A partir de aquí, la consola devuelve la relación de direcciones de repositorios desde donde se va descargando cada paquete, así como el progreso de la instalación. Si al finalizar el proceso, hubiera cualquier problema, el terminal nos lo comunicaría.

El usuario no iniciado, de todos modos, siempre contará con alguna herramienta de corte gráfico más simple que no proporciona tanta información, posiblemente densa y excesiva para el principiante, y que cumplirá igualmente con su objetivo. Una de nuestras preferidas, en las últimas versiones de Ubuntu, se encuentra en el menú de Gnome *Aplicaciones->Centro de software de Ubuntu*.

## El terminal de consola BASH

Como decíamos anteriormente, la consola suele ser la "bestia negra" que evita a toda costa el principiante. No es que no se sienta cómodo con ella, es que simplemente no sabe utilizarla. Es lógico que sea así. El terminal de consola es un intérprete de comandos que realiza, a bajo nivel, todo lo que el usuario puede conseguir mediante el interfaz gráfico a base de ventanas. Eso sí, puede realizar aún más funciones, con decenas, cientos de posibilidades de configuración adicionales.

BASH es el acrónimo de *Bourne Again SHell*, un juego de palabras que combina el apellido Bourne (el escritor original del terminal de consola para Unix que BASH emula) y el término "renacido". Este programa fue escrito en el seno del proyecto GNU, y suele ser el terminal de consola que viene por defecto con la mayoría de las distribuciones. (Recordamos, no obstante, al lector, que siempre puede prescindir de este programa, e instalar otro de su preferencia con iguales o parecidas atribuciones; en todo este texto estamos remarcando las casi infinitas posibilidades de configuración de un sistema operativo con base en el software libre).

Cuando se pone en marcha el terminal de consola, aparece una pantalla de texto, con un cursor parpadeante, precedido de dos términos separados entre sí por el símbolo @:

```
antonio@antonioelmalo:~$
```

El primer término hace referencia al usuario que ha puesto en marcha la terminal (*antonio*), y el segundo hace referencia al nombre del equipo (*antonioelmalo*). Donde está situado el cursor, el usuario irá tecleando diversos comandos con las opciones que correspondan. El terminal irá obedeciendo cada orden, ofreciendo la información correspondiente de vuelta. Algunos comandos básicos en consola son:

- `ls`: devuelve la relación de archivos en la carpeta en la que estamos.
- `cd ruta`: cambia a la carpeta especificada en ruta.
- `mkdir nombrecarpeta`: crea una carpeta en la ruta en la que nos encontramos con el nombre especificado. Si el nombre especificado es precedido de una ruta, se creará en la ruta señalada.
- `rmdir nombrecarpeta`: borra la carpeta especificada.
- rm nombrearchivo: borra el archivo especificado.
- `./nombrearchivo.sh`: ejecuta el archivo ejecutable especificado con extensión .sh.
- `su nombreusuario`: indica al terminal que vamos a conmutar a otro usuario con el nombre indicado. Se nos pedirá, en consecuencia, que introduzcamos la contraseña correspondiente. Veamos un ejemplo:

```
antonio@antonioelmalo:~$su alumno01
Contraseña: ******
alumno01@antonioelmalo:/home/antonio
```

Como puede verse, el terminal indica ahora que estamos en el equipo *antonioelmalo* como el usuario *alumno01*. Además, se nos dice que continuamos en la ruta */home/antonio*, donde no tenemos atribuciones de escritura en un principio. Para ir a nuestra carpeta de cuenta, deberíamos teclear *cd /home/alumno01*.

No es objeto de este texto abundar mucho más en las posibilidades de la consola, aunque podemos asegurar que la sensación de libertad y de total dominio de lo que está pasando en el equipo irá animando al núbil no iniciado a explorar esta herramienta, que poco a poco le irá demostrando su potencia.

Para terminar, mencionar otras posibilidades que dan un mayor dominio de la consola al usuario: la tecla *Tabulador* suele completar el texto a medias que el usuario está tecleando, cuando se refiere a una ruta o comando contenidos en el ordenador. Por ejemplo, si tecleamos *cd /home/antonio/Doc*, y a continuación la tecla *Tabulador*, BASH completará el texto de modo que obtendremos *cd /home/antonio/Documentos*. El comando *man*, de manual, proporciona al interesado toda la información al respecto de un comando sobre el que quiera investigar. De hecho, se suele abrir una mini enciclopedia explorando todas las opciones y posibilidades que dicho comando ofrezca. Una vez abierta esta posibilidad (por ejemplo, en *man ls*), el espaciador va cambiando de página. También podemos movernos por el manual correspondiente con las teclas de cursor, *Avanzar página* y *Retroceder página*. La tecla *q* finalizará el proceso.

## En la red

- Movimiento por el software libre GNU
  *http://www.gnu.org/home.es.html*
- Centro de Excelencia por el Software Libre de Castilla la Mancha
  *http://www.ceslcam.com/*
- Modelos de negocio basados en el software libre, según Franck Hecker *http://hecker.org/writings/setting-up-shop*
- Sitio web de Debian
  *http://www.debian.org/index.es.html*
- Sitio web de Ubuntu
  *http://www.ubuntu.com*
- Sitio web de Molinux
  *http://www.molinux.info*
- Sitio web de Mandriva
  *http://www2.mandriva.com/es/*
- Sitio web de OpenSUSE
  *http://es.opensuse.org*

## Conclusiones

Está claro que es imposible condensar en un solo artículo de revista todo lo que el usuario interesado debería saber sobre lo que comúnmente se llama el *Mundo Linux*. Pero está claro que por algún sitio hay que empezar, y nosotros hemos hecho un repaso de aquellos conceptos que consideramos constituyen la base de este mundo, y que en su día echamos de menos que se nos explicara. Desde la perspectiva de la práctica educativa, es particularmente delicado iniciar en un universo tan versátil, y por lo mismo, tan ambiguo, a nuestros jóvenes. No hay que olvidar que, si bien su contacto con la Informática es mucho más intenso y con pulso más firme que en el caso de sus padres, también se suele iniciar, tanto en el entorno escolar como en el doméstico, en el seno del software privativo, que hace de una marca específica y de sus derivados su bandera. Si, con estas humildes líneas, logramos ampliar los horizontes del lector aficionado, y le hacemos comprender que no tiene que aceptar el uso obligado y necesario de un producto determinado de una marca concreta, sino que siempre puede buscar software alternativo; si logramos que entienda que aprender no quiere decir conocer los entresijos de un producto determinado, sino saber qué se desea conseguir, y tener autonomía para investigar y obtener de manera independiente el software necesario para ello; si conseguimos, simplemente, darle una base que le permita progresar en su conocimiento del *Free Software*, entonces... entonces habremos logrado nuestro objetivo. Muchas gracias por leernos hasta el final. ¡Hasta otra!.

## Sobre los autores

María Dolores Nogueras Atance, licenciada en Ciencias Químicas, es profesora de Tecnologías en la actualidad, pero también ha pasado algunos años como profesora de Formación Profesional en Laboratorio. Su irrupción en el mundo informático ha sido algo tardío, y debido sobre todo a la estrecha relación de dicho mundo con la materia que actualmente imparte. Sin embargo, ha sabido retomar el ritmo y pone a prueba y se esfuerza por aprender toda nueva herramienta informática que caiga en sus manos y que pueda tener algo que ver con la educación.

Antonio Gómez García es Ingeniero Técnico Industrial de Formación, y lleva más de diez años dedicando su actividad profesional a la Educación Secundaria y Bachillerato en institutos. Profesor de Tecnologías y de Tecnologías de la Información, ha trabajado como asesor TIC en el Centro de Profesores de Puertollano, y dedica gran parte de su tiempo al software libre y su introducción en el sistema educativo. Desde esa filosofía, ha colaborado ya en varias actividades de formación de padres, profesores y alumnos sobre seguridad en Internet. En la actualidad, es Responsable de Medios Informáticos en el IES Eduardo Valencia, de Calzada de Calatrava (Ciudad Real). Agradecerá cualquier aporte que queráis realizar en administra *dor@eduardovalencia.no-ip.org*