Linux para Ingeniería:

Herramientas de Gestión de Paquetes de Software y Repositorios

Luis Garreta

luis.garreta@javerianacali.edu.co

Ingeniería de Sistemas y Computación
Pontificia Universidad Javeriana – Cali

13 de abril de 2018

Repositorios y Paquetes

En cualquier distribución, los paquetes son el elemento básico para tratar las tareas de instalación de nuevo software, actualización del existente o eliminación del no utilizado.

- **Un repositorio**
  Es un sitio centralizado donde se almacena y mantiene software en forma de paquetes que está listo para instalarse.

- **Un paquete de software**
  Es un conjunto de ficheros que forman una aplicación o una unión de varias aplicaciones relacionadas, formando un único fichero (denominado paquete), con un formato propio y comprimido, que es el que se distribuye, ya sea vía CD, DVD o mediante acceso a servicios de ftp o web a repositorios públicos.

Categorías de Agrupamiento de Paquetes

En el contenido de la distribución (sus CD/DVD o imágenes ISO) los paquetes suelen estar agrupados por categorías como:

1. Base: paquetes indispensables para el funcionamiento del sistema (útiles, programas de inicio, bibliotecas de sistema).
2. Sistema: útiles de administración, comandos de utilidad.
3. Desarrollo (developement): útiles de programación, como editores, compiladores, depuradores...
4. Gráficos: controladores e interfaces gráficas, escritorios, gestores de ventanas.
5. Otras categorías.

Pasos de Instalación de un Paquete

Para la instalación de un paquete, será necesario efectuar una serie de pasos:

- **1) Previo (preinstalación)**: comprobar que existe el software necesario (y con las versiones correctas) para su funcionamiento (dependencias), ya sean bi- bliotecas de sistema u otras aplicaciones que sean usadas por el software.
- **2) Descompresión del contenido del paquete**, copiando los ficheros a sus lo- calizaciones definitivas, ya sean absolutas (tendrán una posición fija) o, si se permite, reubicadas a otros directorios.
- **3) Postinstalación:** retocar los ficheros necesarios, configurar posibles paráme- tros del software, adecuarlo al sistema...

Tipos de Paquetes

- Veremos, a continuación, quizás los tres paquetes más clásicos de la mayoría de distribuciones:
  - ✓ TGZ
  - ✓ RPM
  - ✓ DEB
- Cada distribución suele tener uno por estándar y soporta alguno de los demás.
- También observaremos una serie de nuevas propuestas para sistemas de empaquetado "universales" que puedan ser usados de forma genérica por más de una distribución.

Paquetes TGZ

- Los paquetes TGZ son quizás los de utilización más antigua.
- Las primeras distribuciones de GNU/Linux los utilizaban para instalar el software, y todavía los usan varias distribuciones (por ejemplo, Slackware) y algunos UNIX co- merciales.
- Son una combinación de ficheros unidos por el comando tar en un único fichero .tar, que luego ha sido comprimido por la utilidad **gzip**, suele aparecer con la extensión .**tgz** o bien **.tar.gz**.
- Asimismo, hoy en día es común encontrar los **tar.bz2** que utilizan, en lugar de gzip, otra utilidad llamada **bzip2** que, en algunos casos, consigue mayor compresión del archivo.

Este tipo de paquete no contiene ningún tipo de información de dependencias, y puede presentar tanto contenido de aplicaciones en formato binario como en código fuente. Podemos considerarlo como una especie de colección de ficheros comprimida.

Proceso instalación paquetes TGZ: **Descomprimir**

El proceso básico con estos paquetes consiste en:

1. Descomprimir el paquete (no suelen utilizar path absoluto, con lo que se pueden descomprimir en cualquier directorio):

```
tar -xzvf fichero.tar.gz (o fichero.tgz)
```

ó

```
gunzip fichero.tar.gz
tar -xvf fichero.tar
```

2. Una vez tenemos descomprimido el tgz, tendremos los ficheros que con tenía; normalmente, el software debe incluir algún fichero de tipo **Readme** o **Install**, donde nos especificarán las opciones de instalación paso a paso, y tam- bién posibles dependencias del software.

Proceso instalación paquetes TGZ: **Instalar**

Una vez tenemos descomprimido el tgz, tendremos los ficheros que con tenía; normalmente, el software debe incluir algún fichero de tipo **Readme** o **Install**, donde nos especificarán las opciones de instalación paso a paso, y tam- bién posibles dependencias del software. :

1. **./configure**: se trata de un script que configura el código para poder ser com- pilado en nuestra máquina, y verifica que existan las herramientas adecuadas. La opción –prefix = directorio permite especificar dónde se instalará el softwa- re. Normalmente, se soportan muchas opciones adicionales de configuración según el software (con –help se mostraran las que acepta).
2. **make**: la compilación propiamente dicha.
3. **make install**: la instalación del software a un lugar adecuado, especificado previamente como opción al configure o asumida por defecto

Ejemplos de Paquetes TGZ

- **Si se trata de formato binario**, tendremos que tener en cuenta que sea adecua- do para nuestro sistema; por ejemplo, suele ser común alguna denominación como la que sigue (en este caso, la versión 1.4 de un paquete):

```
paquete-i686-pc-linux-gnu-1.4-installer.tar.gz
```

- **Si fuese en formato fuente**, suele aparecer como:

```
paquete-source-1.4.tar.gz
```

Paquetes RPM

Fedora/Red Hat: paquetes RPM

- El sistema de paquetes RPM creado por Red Hat supone un paso adelante, ya que incluye la **gestión de dependencias** y tareas de configuración del software.
- Además, el sistema guarda una pequeña **base de datos** con los paquetes ya instalados, que puede consultarse y se actualiza con las nuevas instalaciones.
- Los paquetes RPM, por convención, suelen usar un nombre como:

```
paquete-version-rev.arq.rpm
```

Las operaciones típicas con los paquetes RPM

- Gestión de dependencias: los paquetes RPM incorporan la idea de gestión de dependencias y de base de datos de los paquetes existentes.
- Información del paquete: se consulta sobre el paquete una información determinada
- Instalación:...
- Actualización: equivalente a la instalación, pero comprobando primero que el software ya existe rpm -U paquete.rpm. Se encargará de borrar la ins- talación previa.
- Verificación: durante el funcionamiento normal del sistema, muchos de los archivos instalados cambian
- Eliminación: borrar el paquete del sistema RPM. Si hay dependencias, puede ser necesario eliminar otros primero.

Tareas Proceso de Instalación Paquetes RPM

Tareas que se realizan antes/durante/después de la Instalación:

- Verificar las posibles dependencias.
- Examinar por conflictos con otros paquetes previamente instalados
- Realizar tareas previas a la instalación.
- Decidir qué hacer con los ficheros de configuración asociados al pa- quete si previamente existían.
- Desempaquetar los ficheros y colocarlos en el sitio correcto.
- Realizar tareas de postinstalación.
- Finalmente, almacenar registro de las tareas realizadas en la base de datos de RPM.

Paquetes DEB

Paquetes Debian/Ubuntu: DEB

Similar a la idea de REDHAT con los paquetes RPM pero ahora son paquetes DEB

Paquetes en Debian/Ubuntu

- Ubuntu divide todo el software en cinco secciones, llamadas componentes, para mostrar diferencias en licencias y la prioridad con la que se atienden los problemas que informen los usuario.
- Estos componentes son:
  - ✓ main ,
  - ✓ restricted ,
  - ✓ universe ,
  - ✓ commercial y
  - ✓ multiverse
- Por omisión, se instala una selección de paquetes que cubre las necesidades básicas de la mayoría de los usuarios de computadoras.

Características del repositorio Main

- Características:
  - ✓ Es el repositorio principal de Ubuntu.
  - ✓ En este repositorio hallamos los paquetes principales de la distribución.
  - ✓ Son libres u Open Source.
  - ✓ El mantenimiento del repositorio es realizado por Canonical[a]. .
  - ✓ Actualizaciones de seguridad y soporte técnico por parte del equipo de Canonical.
  - ✓ Incluyen los paquetes y las aplicaciones que la comunidad y los desarrolladores de Ubuntu consideran más importantes.
- Ejemplos de programas en este repositorio son:
  - ✓ Firefox, Evince, Nano, Etc.

[a] Recuerden que Canonical es la empresa encargada de desarrollar Ubuntu

Características del repositorio Universe

- Características:
  - ✓ Incluye la totalidad de paquetes libres y Open Source que no han sido incluidos en el repositorio Main.
  - ✓ Mantenidos por la comunidad y no por Canonical
  - ✓ Canonical no garantiza actualizaciones de seguridad ni corrección de bugs.
    - Esta dependerá de la comunidad.
  - ✓ Usar estos paquetes tienen un poquito de riesgo para el usuario.
  - ✓ En el caso que exista un software muy popular, o que esté bien soportado, se trasladará al repositorio main.
- Ejemplos de programas en este repositorio son:
  - ✓ Abiword, Vlc, Audacious, Pidgin, etc.

Características del repositorio multiverse

Características:

- Incluye software no libre y software open source con restricciones legales.
- Los paquetes de este repositorio no cumplen con las directivas del software libre.
- El soporte de este repositorio es realizado por la comunidad.
- Al tratarse de paquetes privativos o con restricciones legales, hace que las actualizaciones de seguridad y la resolución de problemas sea más lenta.

Ejemplos de software en este repo:

- Adobe Flash Plugin.
- Códecs para reproducir audio y vídeo.
- libdvdcss para reproducir DVD.
- Unrar.
- Ubuntu-restricted-extras.
- nautilus-dropbox.
- Etc.

Características del repositorio restricted

- Características:
  1. La totalidad de software es privativo.
  2. El mantenimiento de este repositorio es realizado por Canonical.
  3. El soporte que puede proporcionar Canonical es limitado.
  4. Al tratarse de software privativo, Canonical únicamente podrá proporcionar soluciones o actualizaciones cuando los creadores del software privativo reaccionen.
- Ejemplos de software de este repositorio:
  - ✓ Básicamente incluye drivers privativos.
  - ✓ Mediante el uso de estos drivers privativos, en algunos casos podremos hacer funcionar el siguiente hardware:
    1. La tarjeta de red wifi.
    2. La tarjeta gráfica del ordenador.
    3. El Bluetooth.
    4. etc.

Otros repositorios existentes en ubuntu: **PPA**

Solo en Ubuntu y en distribuciones derivadas de Ubuntu existen repositorios creados por terceros y que podemos añadir a nuestra distribución.

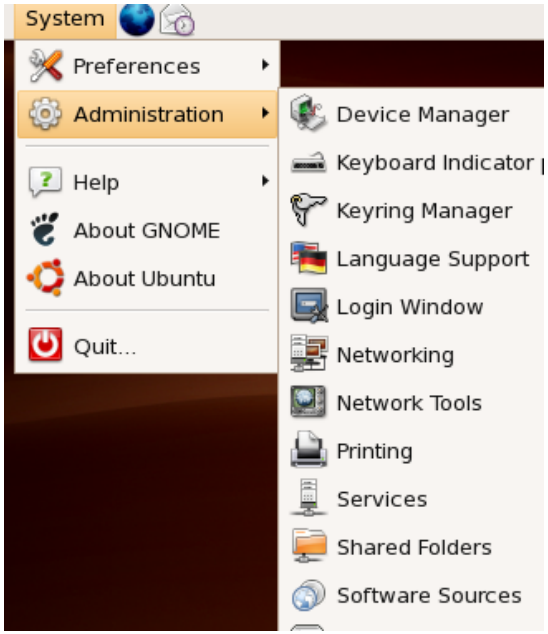
- Características:
  - ✓ Estos repositorios son los PPA (Personal Package Archive).
  - ✓ La utilidad principal de estos repositorios son:
    - Instalar un software que no se halla en los repositorios de nuestra distribución.
    - Actualizar un software que ya está presente en nuestro equipo a la última versión.
  - ✓ Estos repositorios son completamente ajenos a Ubuntu.
  - ✓ Mantenimiento y actualización dependen solo del dueño o gestor del repositorio ppa.
- Ejemplos:
  - ✓ algunos de los repositorios ppa existentes en Ubuntu (`https://www.ubuntuupdates.org/ppas`).

Herramientas

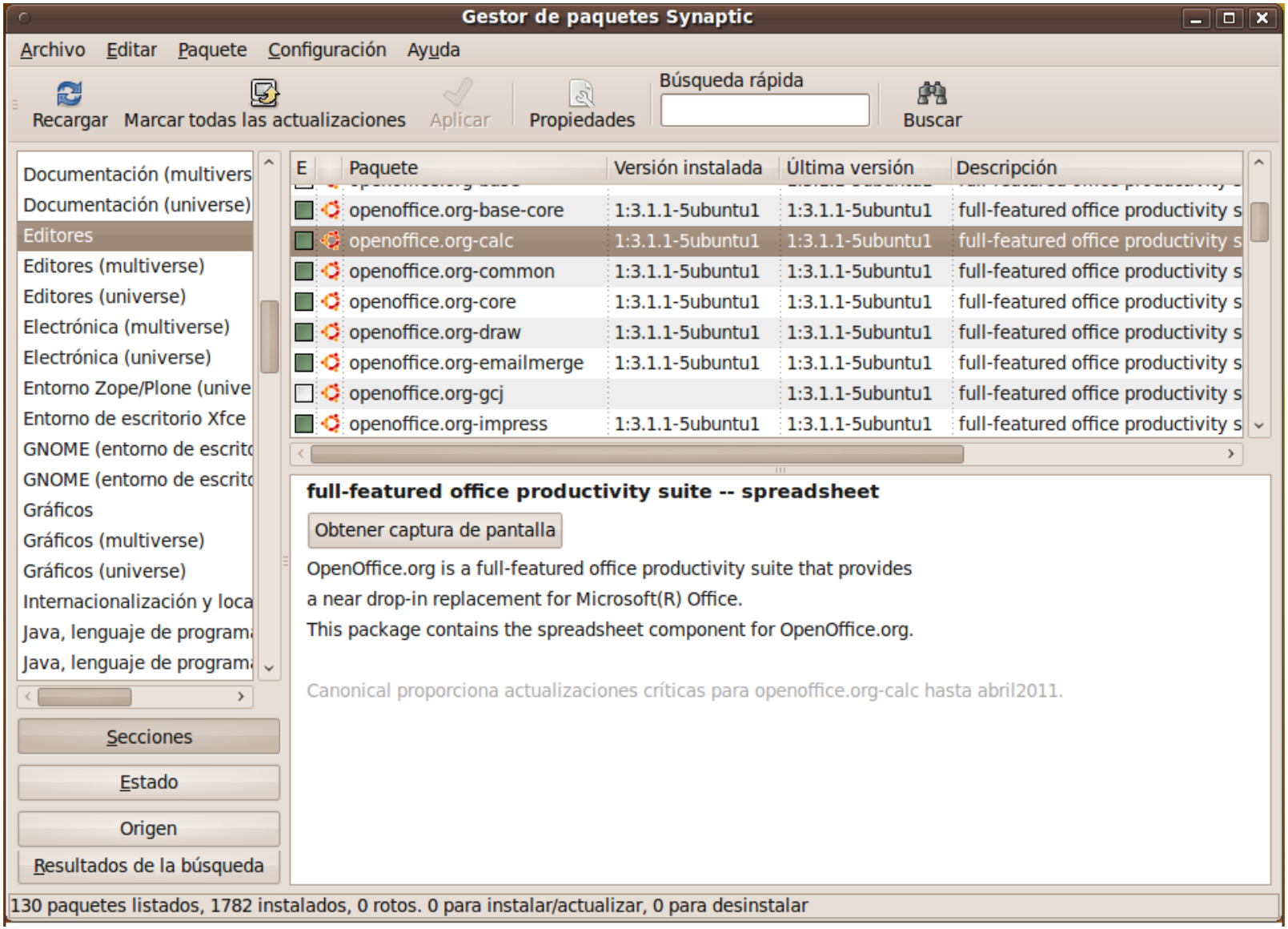
GUI de Manejador de Paquetes en Ubuntu

- Algunas GUIs para manejo de Software:
  - ✓ gnome-software, ubuntu-software
  - ✓ **synaptic:** la más éstandard (debian/ubuntu)
- Ahora las de línea de comandos incluyen:
  - ✓ apt, apt-get, aptitude, dpkg, apt-cache, otras

Manejo de Paquetes Mediante GUI: ubuntu-software



Manejo de Paquetes Mediante GUI: Synaptic



APT

Manejo de Paquetes Mediante Comandos: **Los programas aptitude, apt-get y apt**

APT es un proyecto gigante y su plan original incluía una interfaz gráfica:

- Está basado en una biblioteca que contiene la aplicación central y **apt-get** fue la primera interfaz — basada en la línea de órdenes — desarrollada dentro del proyecto.
- **apt** es un segundo frontend de linea de comandos proporcionado por APT el cual soluciona algunos errores de diseño de la orden apt-get.
- Varias otras interfaces gráficas aparecieron luego como proyectos externos: **synaptic**, **aptitude** (que incluye tanto una interfaz en modo texto como una gráfica — aún cuando no esté completa), wajig, etc.
- La interfaz más recomendada, **apt** es la que utilizaremos en los ejemplos de esta sección. Note, sin embargo, que la sintaxis de línea de órdenes de apt-get y de **aptitude** son muy similares.

Resumen comandos Manejo de Paquetes con APT

apt-get update	Actualizar lista de paquetes disponibles
apt-get upgrade	Actualizar todos los paquetes (hacer con cuidado en un servidor en producción)
apt-get dist-upgrade	Actualizar a la siguiente versión de la distribución (hacer con cuidado en un servidor en producción):
apt-cache search paquete	Buscar un paquete disponible:
apt-get install paquete	Instalar un paquete desde repositorio
apt-get remove paquete	Eliminar un paquete
apt purge paquete	Al momento de desinstalar un paquete, se borren, junto a él, todas las dependencias y archivos de configuración asociados.
dpkg -i paquete	Instalr paquete desde archivo
dpkg –configure -a	Intentar solucionar conflictos y dependencias rotas

Configuración de los Repositorios

- La configuración del sistema APT se efectúa desde los archivos disponibles en **/etc/apt**, donde **/etc/apt/sources.list** es la lista de fuentes disponibles.
- Podría ser, por ejemplo (archivo /et/apt/sources.list:

```
deb http://http.us.debian.org/debian stable main contrib non-free
debsrc http://http.us.debian.org/debian stable main contrib non-free
deb http://security.debian.org stable/updates main contrib non-free
```

- Aquí hay recopiladas varias de las fuentes "oficiales" para una Debian (stable en este caso), desde donde se pueden obtener los paquetes de software, así como las actualizaciones que estén disponibles.
- Básicamente, se especifica:
  - ✓ el tipo de fuente (web/ftp en este caso),
  - ✓ el sitio,
  - ✓ la versión de la distribución (stable),
  - ✓ y categorías del software que se buscará (libre, o contribuciones de terceros o de licencia no libre o comercial)

Información sobre un paquete

La herramienta apt-cache y aptitude dispone de opciones que nos permiten buscar información so- bre los paquetes, como por ejemplo:

1. Buscar paquetes sobre la base de un nombre incompleto:

```
apt-cache search nombre
```

2. Mostrar la descripción del paquete:

```
apt-cache show paquete
```

3. De qué paquetes depende:

```
apt-cache depends paquete
```