Tema 2: Mantenimiento y administración de paquetes

ADMINISTRACIÓN DE SISTEMAS OPERATIVOS

Curso 2020/2021

Cristian Martín Fernández

Contenido

- Administración de paquetes Ubuntu
- Actualizaciones del sistema
- Repositorios por defecto de Ubuntu
- Buscar y localizar paquetes
- Instalación y desinstalación de software
- Gestión de prioridades
- Creación de un nuevo paquete
- Hacer una réplica del sistema

¿Qué es un paquete?

- Conjunto de ficheros que **empaquetan** un determinado software (Upstream) para ser instalado
- Metadatos para indexar en los buscadores de paquetes
- Metadatos para uso del paquete y ayuda al usuario

Ventajas de los paquetes

- **Automatización en la instalación de software**
- **Gestión automática de dependencias**: los paquetes contienen metadatos que proporcionan información sobre qué otros archivos son necesarios para cada paquete.
- **Control de calidad**: proceso de empaquetado para probar y asegurarse de que el software sea estable y esté libre de errores que puedan afectar la calidad del producto y que el software no cause que el sistema se vuelva inestable.
- **Repositorios de paquetes**: permiten a los usuarios descargar sus paquetes de proveedores de confianza de forma cómoda.
- **Actualización del sistema operativo:** continuamente se están desarrollando parches y mejoras de los paquetes para solventar problemas de seguridad y mejorar la experiencia del usuario.

Administración de paquetes Ubuntu

- Pueden contener formato binario o código fuente
- Formato RPM (requieren de conversión en Ubuntu) o formato Debian
- Ubuntu usa los formatos Debian

Administración de paquetes Ubuntu

- En las versiones server se suele hacer con la línea de comandos en **modo consola** ya que estas versiones en Linux no suelen incorporar el entorno gráfico.
- La mayoría de los administradores de Ubuntu suelen utilizar las herramientas de la familia APT que manejan la administración de paquetes a bajo nivel.
- La herramienta original desarrollada para este propósito fue **apt-get**

Administración de paquetes Ubuntu

- En la versión de escritorio se suele utilizar un programa gráfico de administración de paquetes llamado **Synaptic**, que permite a los usuarios navegar a través de los archivos de paquetes.
- **Aptitude** es una alternativa utilizada con frecuencia a apt-get que proporciona tanto la interfaz interactiva como el uso de la mayoría de los comandos de apt-get en consola.
- Si se ejecuta Aptitude sin argumentos nos da una interfaz de texto a los usuarios de consola que nos podría recordar la interfaz gráfico Synaptic. Con este interfaz se puede recorrer la lista de resultados con el cursor y “marcar” los paquetes para su instalación

Metadatos para indexar en los buscadores de paquetes

- Nombre del software
- Nombre del autor principal y de la persona que crea el paquete
- La localización principal del software o al menos una que sea válida
- Arquitectura o requerimientos sobre los cuales se puede ejecutar el software con garantías

Metadatos para indexar en los buscadores de paquetes

- Información para clasificar sobre lo que hace el paquete y el uso que puede tener
- Descripción del software
- Información sobre la importancia o prioridad del paquete: esencial, opcional, esencial para usuarios de…, etc.

Metadatos para uso del paquete y ayuda al usuario

- Relación con otros paquete (**Depends**):
  - Software distinto al que se requiere para ser compilado.
  - Software distinto al que se requiere para instalarse.
  - Otros paquetes que se requieren para poder ejecutarse.

Metadatos para uso del paquete y ayuda al usuario

- Relación con otros paquetes:
  - Un software distinto que no puede ejecutarse o incluso instalarse de forma simultánea con el software de este paquete (**Conflicts**)
  - Puede romper otro paquete (**Breaks**)
  - Un software que puede usarse para reemplazar un tercero (**Provides**)
  - Un software distinto al que puede mejorar este software o paquete (**Recommends**) y otros opcionales (**Suggests**)

Mostrar información de un paquete

- Para mostrar información detallada de un paquete usamos la opción show de apt:
  - Ejemplo: `apt show aptitude`
- También podemos utilizar apt-cache:
  - Ejemplo: `apt-cache show aptitude`

Carácterísticas avanzadas:

- Los **archivos de configuración**, al contrario que los archivos normales o regulares, no pueden ser siempre reemplazados simplemente con una versión nueva cuando el paquete de software se actualiza.
- Otra importante característica incluida en los empaquetadores modernos es una estructura de los metadatos que permite fácilmente el **tratamiento de multi-idioma.**

Funciones básicas de la administración de paquetes

- La función más importante o más valorada del empaquetado es automatizar la **compilación del software**.
- Los paquetes con formato DEB proporcionan dos formatos:
  - Formato paquetes fuente (.dsc)
  - Formato paquetes binarios (.deb)
- Los paquetes están diseñados para ser compilados de forma no interactiva.

Funciones básicas de la administración de paquetes

- En el caso de paquetes oficiales de Ubuntu pueden ser compilados automáticamente en un rango de arquitecturas distintas por un software automático de compilación de paquetes llamado "**auto-compiladores**".
- Las dependencias necesarias en tiempo de compilación se declaran en los paquetes y pueden ser satisfechas de forma automática y con antelación.

Funciones básicas de la administración de paquetes

- Un ejemplo de estas dependencias es el caso de los paquetes fuentes, que deben ser compilados para cada arquitectura (ARM, AMD64, ix86) y son cargados en Ubuntu como fuente. Se compilan automáticamente en todas las arquitecturas soportadas por Ubuntu sin cambios en el paquete fuente.

Funciones básicas de la administración de paquetes

- Cualquier número de paquetes puede ser creado desde un solo paquete fuente.
- La creación de paquetes múltiples es muy útil para proyectos largos que publican mucho paquetes binarios, de documentación o de otra índole.
- La función principal del sistema de paquetes es ayudar a automatizar la instalación del software.

Funciones básicas de la administración de paquetes

- Cuando un paquete binario se instala:
  - Los “contenidos” del software pueden verificarse para asegurar la integridad del paquete. Además el origen del software puede comprobarse usando firma criptográfica mediante las claves que guarda nuestro equipo.
  - La dependencias del software pueden analizarse y el sistema es escaneado en la instalación para comprobarlo. Si no se satisface la instalación se detiene y se informa al usuario que podrá subsanarla.
- En los paquetes fuentes hay una fase de compilación previa para generar los archivos ejecutables.

Funciones básicas de la administración de paquetes

- En el caso de desinstalación de software, la lista de dependencias también contribuye a una eliminación limpia del software.
- **Aptitude** marca las dependencias y las elimina durante la desinstalación.
- Con APT hay que realizar un paso extra para eliminar estas dependencias en desuso (**autoremove**).

Funciones básicas de la administración de paquetes

- Al usuario que instala el paquete se le puede pedir **confirmación para las opciones de configuración** en algunos puntos del proceso de instalación.
- Las respuestas a estas preguntas pueden ser almacenarse en el sistema y usarse en la personalización de un archivo de configuración para el software que se está instalando.

Funciones básicas de la administración de paquetes

- Todos los contenidos del paquete pueden ser guardados en el sistema.
- **Los metadatos y la información de la cuenta que lo ha instalado se guardarán en una base de datos del sistema** con información actual de los paquetes instalados y sus estados actuales de instalación, configuración, etc.

Actualizaciones del sistema

- Otro campo donde es muy útil el empaquetado de paquetes es en **la actualización del software.**
- En este caso, la actualización se hace encima de la instalación existente y se borran los archivos que ya no sean necesarios.
- En el caso de actualización de software con bibliotecas compartidas también es muy útil para obligar a la actualización de las bibliotecas que sean necesarias

Actualizaciones del sistema

- Se recomienda realizar actualizaciones regularmente, ya que incluyen las últimas actualizaciones de seguridad.
- Cada sistema Ubuntu guarda una lista de repositorios de paquetes en `/etc/apt/sources.list`, que describe la lista de sitios donde los administradores de paquetes (APT y otros) buscarán actualizaciones del software instalado y del propio sistema. Estas ubicaciones pueden ser URL o directorios locales al sistema.

Actualizaciones del sistema

- Para actualizar la lista de repositorios y las nuevas versiones de paquetes usamos el comando:
  - `apt update`
- O bien el comando
  - `aptitude update`
- Estos comandos descargarán la última actualización de las listas de paquetes para todos los repositorios listados en los archivos /etc/apt/sources.list y comprobará cualquier firma criptográfica en esas actualizaciones con las claves guardadas en nuestro sistema

Actualizaciones del sistema

- Para instalar las nuevas versiones de paquetes:
  - `apt upgrade`
  - `aptitude safe-upgrade`
- La opción safe-upgrade puede instalar nuevos paquetes si son necesarios, upgrade no puede. Ninguno de los puede eliminar paquetes.
- También se puede actualizar un único paquete
  - `sudo apt install <package> --only-upgrade`

Actualizaciones del sistema

- Para indicarle a APT que utilice una distribución específica al buscar paquetes a actualizar se puede utilizar la opción -t, seguido del nombre de la distribución que desea (stable/testing/unstable). Por ejemplo:
  - `apt -t stable upgrade`
- Es conveniente siempre instalar la versión stable a no ser que sea un usuario experto o para pruebas

Actualizaciones del sistema

- APT también puede configurarse para descargar y actualizar automáticamente los paquetes con nuevas versiones.
- Pero no es una opción muy frecuente ni recomendada por la documentación oficial de Ubuntu, ya que podría producir una inestabilidad del sistema o en algunos de los servicios de éste.

Actualización completa del sistema

- Para actualizaciones más importantes, tales como el cambio de una versión mayor a la siguiente, se puede utilizar:
  - `apt full-upgrade`
- Con esta instrucción, APT completará la actualización aún si tiene que eliminar algunos paquetes obsoletos o instalar nuevas dependencias.
- Esta también es la orden utilizada por los usuarios que trabajan diariamente con la versión Unstable de Debian y siguen su evolución día a día.
- ”*Es tan simple que casi no necesita explicación: la reputación de APT está basada en esta excelente característica*”

Actualizaciones del sistema

- APT también se puede configurar, por ejemplo, para trabajar con un proxy o cómo descargar los paquetes en **/etc/apt/apt.conf.d/**
- Cada fichero representa un archivo de configuración repartido en múltiples archivos. En este sentido, todos los archivos en /etc/apt/apt.conf.d/ son instrucciones para la configuración de APT.

Repositorios por defecto de Ubuntu

- Divididos en diferentes secciones que podemos activar o desactivar en el fichero /etc/apt/sources.list. Dan acceso a más de 60.000 paquetes.
  - Main
  - Universe
  - Backports
  - Restricted
  - Multiverse
- Para añadir un nuevo repositorio
  - `sudo add-apt-repository <repository>`

Repositorios por defecto de Ubuntu

- **Main**
- Es el repositorio principal. Contiene aplicaciones constituidas por software libre que proviene de Ubuntu (Canonical).
- Contiene una lista de aplicaciones seleccionadas por los desarrolladores, la comunidad y los usuarios de Ubuntu consideradas más importantes, y que el equipo de distribución y seguridad de Ubuntu está dispuesto a ofrecer soporte.
- Cuando instalamos el software desde el repositorio main, podemos tener cierta confianza de que el software vendrá con actualizaciones de seguridad y soporte técnico. A menudo de 18 meses.

Repositorios por defecto de Ubuntu

- **Universe**
- Aquí encontraremos la mayoría del software libre mantenido por la comunidad.
- El software no está revisado ni mantenido por Ubuntu.
- Por tanto, puede no tener las garantías de seguridad ni de soporte técnico que ofrecen los paquetes del repositorio main. Aunque hay miles de paquetes que son continuamente mantenidos y actualizados.

Repositorios por defecto de Ubuntu

- **Backports**
- Nuevas versiones de paquetes mantenidos por la comunidad.
- Software normalmente no tan bien probado ni validado como el software de los repositorios anteriores.

Repositorios por defecto de Ubuntu

- **Restricted**
- Aquí se encuentra los paquetes con derechos de autor restringidos, a menudo controladores y drivers de hardware.
- No tienen una licencia totalmente libre.

Repositorios por defecto de Ubuntu

- **Multiverse**
- Contiene software que no tiene una licencia open source, y por tanto no es gratuito.
- La responsabilidad es del usuario de acceder a la licencia.
- El software no está revisado ni mantenido por Ubuntu.

Ver repositorios de un paquete antes de la instalación

- Podemos verlo con la opción de mostrar los metadatos del paquete:
  - `apt show <package>`
- También se pueden visualizar con la opción policy:
  - `apt policy <package>`

Añadir claves para repos de confianza

- **apt-key** sirve para gestionar la lista de claves que APT usa para autenticar paquetes.
- Los paquetes de repositorios autenticados mediante estas claves se consideran de confianza.
- Se pueden instalar paquetes sin confianza (aunque no es recomendable).
- Listar claves
  - `apt-key list`
- Instalar clave.
  - `apt-key add <key>`
- Se suele combinar con wget
  - `wget -qO - https://deb.opera.com/archive.key | sudo apt-key add -`

Buscar y localizar paquetes

- Usuarios de escritorio:
  - Añadir y quitar aplicaciones
  - Programa gráfico Synaptic
- Para los usuarios de consola: apt, apt-cache y aptitude
- Ejemplo: `$ apt search pager less`
- Ejemplo: `$ apt-cache search pager less`
  - Busca paquetes con las palabras claves “pager” y “less”

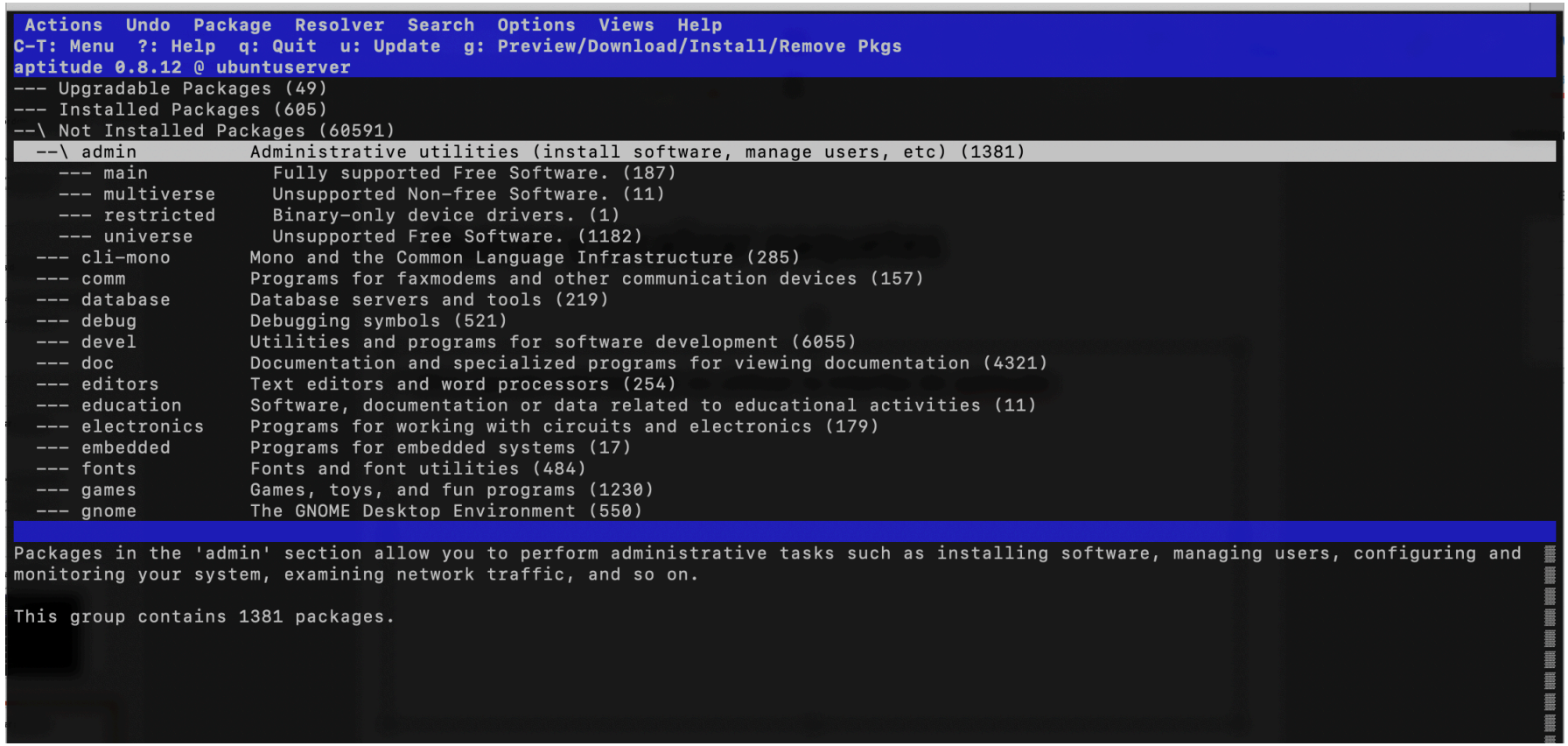
Buscar y localizar paquetes

- Apt y apt-cache. Mismo ouput, diferente estilo

```
cristian@ubuntuserver:~/paquete-1.0/debian$ apt-cache search pager less
less - pager program similar to more
wdiff - Compares two files word by word
console-log - Puts logfile pagers on virtual consoles
irssi-scripts - collection of scripts for irssi
libio-pager-perl - module to select a pager and pipe text to it
most - Pager program similar to more and less
mypager - pager for MySQL/PostgreSQL command line clients
texlive-latex-extra - TeX Live: LaTeX additional packages
cristian@ubuntuserver:~/paquete-1.0/debian$ apt search pager less
Sorting... Done
Full Text Search... Done
console-log/focal 1.2-2 all
  Puts logfile pagers on virtual consoles

irssi-scripts/focal 20200223 all
  collection of scripts for irssi

less/focal-updates,now 551-1ubuntu0.1 amd64 [installed,automatic]
  pager program similar to more

libio-pager-perl/focal 1.01-1 all
  module to select a pager and pipe text to it

most/focal 5.0.0a-4 amd64
  Pager program similar to more and less

mypager/focal 0.6.1-1 all
  pager for MySQL/PostgreSQL command line clients

texlive-latex-extra/focal 2019.202000218-1 all
  TeX Live: LaTeX additional packages

wdiff/focal 1.2.2-2build1 amd64
  Compares two files word by word
```

Buscar y localizar paquetes

- Otra opción interesante es utilizar la interfaz de aptitude:



Buscar y localizar paquetes

- **packages.ubuntu.com** ofrece una interfaz web que permite a los usuarios buscar determinados archivos en cualquier paquete de Ubuntu, sin necesidad de tener instalado el paquete que lo contiene.

Instalación y desinstalación de software

- Para instalar paquetes podemos usar la opción “install” de apt, apt-get o de aptitude, y es necesario tener privilegios de “root”.
- Para ello usaremos el comando “sudo” delante y así evitamos trabajar directamente como “root” que no está activado en Ubuntu por defecto.
- Se puede especificar la versión de un paquete:
  - `apt install <package> <package>=2.5`
- También es posible instalar y desinstalar programas en la misma línea:
  - `apt install +<package> -<package>`

Instalación y desinstalación de software

- A veces el sistema puede dañarse después de eliminar o modificar los archivos de un paquete.
- La forma más sencilla de recuperar estos archivos es reinstalar los paquetes afectados.
- Desafortunadamente, el sistema de empaquetado encuentra que éste ya está instalado y amablemente rechaza su reinstalación; para ello hay que usar la opción –reinstall
  - `apt --reinstall install <package>`

Instalación y desinstalación de software

- Para desinstalar podemos usar la opción “remove”:
  - `apt remove <package>`
- Elimina todo incluido los ficheros de configuración:
  - `apt purge <package>`
- Para limpiar paquetes que no se utilizan (ej.., dependencias ya no existentes):
  - `apt autoremove`
- Se puede ver por qué se utiliza un paquete con aptitude:
  - `aptitude why <package>`

Instalación y desinstalación de software

- Para ver TODOS los paquetes de Ubuntu disponibles:
  - `apt list`
  - `apt-cache pkgnames`
- Para ver los paquetes instalados:
  - `apt list --installed`

apt vs apt-get/apt-cache

apt	apt-get / apt-cache	Descripción
apt install	apt-get install	Instalar un paquete
apt update	apt-get update	Actualizar repositorios
apt upgrade	apt-get upgrade	Actualizar paquetes instalados
apt autoremove	apt-get autoremove	Eliminar paquetes no necesarios
apt remove	apt-get remove	Eliminar un paquete instalado
apt purge	apt-get purge	Eliminar paquete instalado y configuraciones
apt search	apt-cache search	Buscar en repositorios un paquete
apt show	apt-cache show	Ver detalles de un paquete
apt policy	apt-cache policy	Ver versión y repo de un paquete
apt list	apt-cache pkgnames	Mostrar el listado de paquetes disponibles

Gestión de prioridades de paquetes

- Es posible configurar APT para gestionar la actualización de paquetes a través de prioridades.
- Estas prioridades influenciarán el comportamiento de APT: para cada paquete, siempre seleccionará la versión con la prioridad más alta.
- Por ejemplo, podría desear extender una distribución con uno o dos paquetes más recientes de Testing, Unstable o Experimental, o bien, no actualizar un paquete en concreto porque es dependencia de otros.
- Para consultar la prioridad de un paquete se utiliza **apt policy <package>**

Gestión de prioridades de paquetes

- Prioridades de paquetes
  - **< 0 nunca** será instalado,
  - **1..99** solo será instalado si **ninguna otra versión** del paquete está ya instalada,
  - **100..499** solo será instalado si no hay **ninguna otra versión más nueva instalada o disponible en otra distribución** (100 por defecto para paquetes instalados)
  - **500....989** solo será instalado si no hay **ninguna versión más nueva instalada o disponible en la distribución de destino**, (500 por defecto para paquetes nuevos)
  - **990..1000** será **instalado a no ser que la versión instalada sea más nueva**,
  - **> 1000** se **instalará siempre**, incluso si fuerza a APT a degradarlo a una versión más vieja.

Gestión de prioridades de paquetes

- Permitir que el paquete Perl no se actualice y se mantenga en la versión 5.24

```
Package: perl

Pin: version 5.24*

Pin-Priority: 1001
```

Gestión de prioridades de paquetes

```
Package: *

Pin: origin my.custom.repo.url

Pin-Priority: 1


Package: my-specific-software

Pin: origin my.custom.repo.url

Pin-Priority: 500
```

Instalación y desinstalación manual de software

- Otra opción interesante es la herramienta de bajo nivel “dpkg” que permite trabajar con paquetes previamente descargados:
  - `dpkg -i <package>`, para instalar
  - `dpkg -r <package>`, para desinstalar
  - `dpkg -l <package>`, para información del paquete.
  - `dpkg --contents <package>`, lista archivos del archivo del paquete
  - `dpkg -L <package>`, Lista componentes del paquete
  - `dpkg -S` archivo, dado un archivo nos devuelve el paquete que lo contiene

Creando un nuevo paquete

- Antes de ver como se organiza un paquete vamos a descargarnos los fuentes de uno y hacer un unpack:
  - `sudo apt source hello`
- Es posible que nos de fallo ya que no están los sources activos, para ello debemos de activar los repositorios deb-src en /etc/apt/sources.list
- Hay que hacer **apt update** después de actualizar.

Creando un nuevo paquete

- Ahora vamos a ver como es el contenido de un paquete deb
  - `sudo apt download hello`
- Hacemos un unpack con dpkg-dev
  - `sudo dpkg-deb -R hello_2.10-2ubuntu2_amd64.deb ./hello-deb`

Creando un nuevo paquete

- Fichero descomprimido .deb

```
cristian@ubuntuserver:~/hello-deb$ ls -R .
.:
DEBIAN  usr

./DEBIAN:
control  md5sums

./usr:
bin  share

./usr/bin:
hello

./usr/share:
doc  info  man

./usr/share/doc:
hello

./usr/share/doc/hello:
changelog.Debian.gz  copyright  NEWS.gz

./usr/share/info:
hello.info.gz

./usr/share/man:
man1

./usr/share/man/man1:
hello.1.gz
cristian@ubuntuserver:~/hello-deb$
```

Creando un nuevo paquete

- Fichero de control del paquete hello

```
cristian@ubuntuserver:~/hello-deb$ cat DEBIAN/control
Package: hello
Version: 2.10-2ubuntu2
Architecture: amd64
Maintainer: Ubuntu Developers <ubuntu-devel-discuss@lists.ubuntu.com>
Installed-Size: 112
Depends: libc6 (>= 2.14)
Conflicts: hello-traditional
Breaks: hello-debhelper (<< 2.9)
Replaces: hello-debhelper (<< 2.9), hello-traditional
Section: devel
Priority: optional
Homepage: http://www.gnu.org/software/hello/
Description: example package based on GNU hello
 The GNU hello program produces a familiar, friendly greeting.  It
 allows non-programmers to use a classic computer science tool which
 would otherwise be unavailable to them.
 .
 Seriously, though: this is an example of how to do a Debian package.
 It is the Debian version of the GNU Project's `hello world' program
 (which is itself an example for the GNU Project).
Original-Maintainer: Santiago Vila <sanvila@debian.org>
cristian@ubuntuserver:~/hello-deb$
```

Creando un nuevo paquete binario

- El requisito mínimo para crear un paquete binario es el fichero de control y el fichero ejecutable.
- Para crear un nuevo paquete, primero debemos de crear una carpeta con el nombre de nuestro paquete y su versión, por ejemplo:
  - `mkdir hello-uma_1.0`
- Debemos de crear una carpeta DEBIAN donde crear nuestro fichero de control:
  - `mkdir hello-uma_1.0/DEBIAN`

Creando un nuevo paquete binario

- Hay que crear un nuevo fichero control y editar los metadatos del paquete
  - `nano DEBIAN/control`
- Y copiamos en el fichero de control unos metadatos de ejemplo:

```
Package: hello-uma

Version: 1.0

Architecture: all

Maintainer: Cristian Martin <cmf@lcc.uma.es>

Installed-Size: 112

Section: devel

Priority: optional

Description: Example hello package from UMA
```

Creando un nuevo paquete binario

- El fuente va a consistir en un fichero C++ simple (hello-uma.cpp)

```
#include <iostream>

int main(){

   std::cout << "Hello UMA!";

   return 0;

}
```

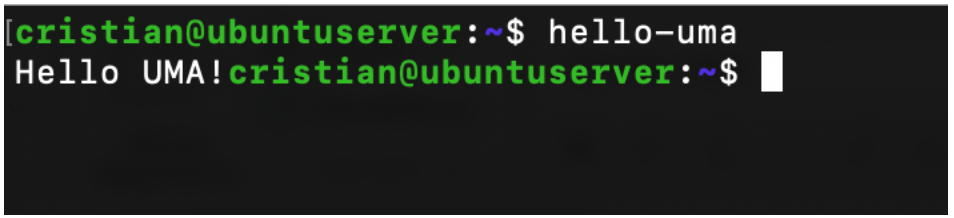
- Lo compilamos y generamos el ejecutable hello-uma:
  - `g++ hello-uma.cpp –o hello-uma`

Creando un nuevo paquete binario

- Una vez compilado el paquete lo movemos a la carpeta ejecutable (creándola previamente):
  - `mkdir –p usr/bin`
  - `mv hello-uma usr/bin`

Creando un nuevo paquete binario

- A partir de esto paso ya podemos empaquetar nuestro paquete con los requisitos mínimos con:
  - `dpkg-deb --build depend_1.0`
- Una vez generado podemos instalar nuestro paquete:
  - `sudo dpkg -i hello-uma_1.0.deb`
- Y ejecutarlo:



Creando un nuevo paquete

- Para la automatización en la creación de ficheros se puede utilizar el comando **dh_make** (es necesario una instalación previa):
  - $ dh_make -p package-name_1.1 --single --native --copyright mit --email email@uma.es
- Aquí es necesario la creación de un fichero de compilación Makefile
- Los archivos con extensión .ex son archivos de ejemplo que se pueden utilizar modificándolos (y eliminando la extensión)

Otros ficheros importantes

- Otros ficheros importantes aparte del fichero de control sobre todo para crear un nuevo paquete oficial:
  - debian/changelog → historial de cambios.
  - debian/copyright → copyright del paquete.
  - debian/rules -> reglas que se tomarán para crear el paquete.

Subir un paquete a un repositorio público de Ubuntu

- Primer hay que registrarse en Launchpap https://launchpad.net/ubuntu
- Hay que crear una clave PGP y subirla al servidor de ubuntu y luego registrar en vuestra cuenta dee Launchpad para firmar los paquetes:https://help.ubuntu.com/community/GnuPrivacyGuardHowto
- Y en vuestro perfil crear un nuevo Personal package archives (PPA).
- Publicar en un PPA requiere paquetes fuente: son más complejos de preparar y requieren de una fase de compilación.
- Es recomendable utilizar la opción **dh_make para crear la estructura de un proyecto y subirlo a un repositorio PPA de Ubuntu.**

Hacer una réplica del sistema

- En muchas ocasiones es necesario tener una instalación de un sistema con las mismas aplicaciones que otro sistema actual, o tener una copia de seguridad de todas las aplicaciones instaladas. En estos casos podemos hacer una réplica del sistema en tres pasos:

1. Obtenemos una lista de todos los paquetes instalados:
   - `dpkg --get-selections >package_list`

Hacer una réplica del sistema

2. Copiamos el fichero con el listado de dependencias en otra máquina y seleccionamos la lista de paquetes:
   - `sudo dpkg --set-selections <package_list`
3. Instalamos estas selecciones mediante el siguiente comando:
   - `apt-get dselect-upgrade`
   - Dselect–upgrade es una referencia a su predecesor de APT, dselect, pero que simplemente funciona para actualizar paquetes en el sistema e instalar cualquier paquete nuevo “marcado” para actualizarse en el proceso de –set-selections.

Bibliografía

- Debian Handbook https://debian-handbook.info/
- Documentación oficial de Ubuntu https://ubuntu.com/server/docs
- Packing Ubuntu: https://packaging.ubuntu.com/html/
- Building a Debian (`.deb`) source package, and publishing it on an Ubuntu PPA: https://saveriomiroddi.github.io/Building-a-debian-deb-source-package-and-publishing-it-on-an-ubuntu-ppa/