

# Manifiesto anti distros

**Diego Saravia**<sup>1</sup>

Primera versión: 10 de marzo del 2005<sup>2</sup>

## 1. Introducción

La mayoría de las distros -tal como las conocemos- son al software libre como los cuadros a la pintura y los temas de 7 minutos a la música.

Una forma o formato de distribución diseñado para convertir al Software Libre en producto y capitalizarlo.

Lo que ocasiona la aparición de burocracias, especialmente las comerciales, con metas innecesarias salvo para ellas mismas, y que como toda burocracia busca formas de hacerse imprescindible e impone normas que retrasan o ponen la innovación real en los márgenes del sistema.

El problema de que distro usar, es lo primero que se plantea una persona u organización que se acerca al movimiento. Un problema que suele ocasionar luchas comerciales y discusiones religiosas. Un problema que crea empresas de servicios que solo son aptas para trabajar con una distro.

Y es un problema artificial que solo aparece en aquellas personas que se acercan al movimiento, pero no que debiera ser un problema para el técnico competente.

Los problemas de elección reales e importantes son las otras cientos de opciones a tomar, como que escritorio usar, que sistema de archivos en red, que paquete de oficina, etc, etc..

Una vez decididas, usar una distro u otra no tiene la más mínima relevancia, tanta como elegir de donde instalar los paquetes.

Discutir que distro usar es entrar en el juego perverso de las empresas comerciales que asocian su distro con dinero y servicios.

Debemos insistir en que la distro es lo de menos, lo que importa es que la gente conozca los conceptos y que se pueda interoperar de la mejor forma.

¿Es el trabajo de hacer distros o subdistros malo? No, por supuesto que no. Es muy útil tener repositorios (distros) particulares, específicas para cada empresa o adaptadas a los médicos o a las escuelas. Debemos facilitar este trabajo y que cualquiera puede hacerse su distro. Pero esto no tiene porque generar ramas incompatibles de software.

Me parece genial que cada empresa usuaria, cada organización, cada “Linux User Group” (LUG), cada persona, tenga su distro, con su menú, su selección de paquetes y sus fondos de pantalla. Ese no es el problema.

Por lo tanto debemos rediseñar el concepto de distro.

---

<sup>1</sup>URLs: <http://www.ututo.org.ar/dsa>; <mailto:dsa@unsa.edu.ar>. Inenco; Dpto. Física; Fac. Cs. Exactas; Universidad Nacional de Salta. Hipatia y Solar,

Con aportes de las comunidades de Conexion Social: <http://www.migrandovenezuela.org>, y Solar: <http://www.solar.org.ar>

<sup>2</sup>24 de agosto de 2005

Versión 0.60. Se puede encontrar la última versión de éste trabajo en sus diferentes formatos y sus fuentes en: <http://www.hipatia.info/docs/distros/> o en <https://softwarelibre.unsa.edu.ar/dsatex/distros.dir/>. Primera versión: 10 de marzo del 2005

Si Ud. quiere contribuir con modificaciones o agregados puede enviar un correo a Diego Saravia con los mismos: <mailto:dsa@unsa.edu.ar>. Por razones legales y prácticas al enviar sugerencias ud. transfiere el copyright de sus sugerencias a Diego Saravia. No siempre será posible registrar el origen de cada aporte. Diego Saravia se compromete a permitir que la última versión de este material este universalmente disponible bajo licencias libres. Verifique que ud. tiene derecho a enviar ese material en estas condiciones. Ante cualquier duda consulte y envíe la información relevante por Correo.

## 2. Definiciones y elementos

**REPOSITORIOS:** consistentes en uno o mas medios conteniendo paquetes. Deben contener el paquete necesario para ejecutar un sistema autónomo y todos los necesarios para ejecutar lo que se desee para el objeto del repositorio.

**DISTROS:** mecanismos, formas, organizaciones creadas para distribuir los paquetes de los repositorios. Materializadas al menos en un medio (en CDROM, diskette o Internet) con un conjunto de paquetes conteniendo ejecutables y archivos de configuración. Probablemente este preparado para ejecutarse e instalarse.

Así pues el trabajo que caracteriza a un distro es la compilación de paquetes propios, la forma de empaquetarlos y como subdivide el software en paquetes.

**SUBDISTROS:** distros derivadas de otras, que toman los paquetes del repositorio de la distro madre, constituyendo otros repositorios y alterando algunos paquetes, a veces solamente el fondo de pantalla, otras veces scripts, menús, etc..

**GENERADORES DE DISTROS:** mecanismos, procedimientos, protocolos, normas (como las “Guidelines” de “Debian”[Debian:SW, Nguyen:CCD-04]) o software para generar distros, como “Linux from Scratch”[LFS:SW] o “Gentoo”[Gentoo:SW] y hasta cierto punto “Metadistros”, o el paquete “apt-get”[Welton:C-00] source de “Debian”.

**PAQUETE:** conjunto de software que no se puede o conviene subdividir, cada paquete tiene ejecutables, archivos de configuración, librerías provistas y requeridas, además de otras cosas, pero estos estarán identificados a los efectos de su incorporación en los menús y en los instaladores, actualizadores y retardadores

**DEPENDENCIAS:** archivos que necesita tener un paquete pero que están en otros, para funcionar (ver dependencias de compilación).

Lo interesante es que la división del software necesario para una computadora en paquetes trae el fenómeno de la dependencia. Entonces no es importante hablar solo del paquete sino de la forma en que se subdivide el software y se arma la red de dependencias.

La forma de dividir los paquetes es también característica de las distros.

**MANEJO DE PAQUETES:** sistema que permite manejar los paquetes en un menú, ejecutar y configurar lo que deba hacerse además de instalar, actualizar y retardar un paquete

**SISTEMA AUTÓNOMO:** es un sistema distribuido en un paquete que contiene: núcleo, imagen y software adicional que se instala y puede auto-manejarse y repararse.

**SISTEMA QUE COMPILA:** sistema que tiene los paquetes ejecutables que permiten crear paquetes ejecutables o simplemente paquetes. Un directorio donde se agrupan las fuentes y un superrepositorio donde se colocan los paquetes producidos.

**SUPER-REPOSITORIO:** de un sistema que compila (o de varios sincronizados) conteniendo todos los paquetes en muchas versiones.

**FUENTES:** Url del proyecto original de cada soft, puede adicionarse parches con lo necesario para que cada paquete funcione como aquí se indica. Siempre se harán esfuerzos para que estos parches sean incorporados al proyecto original. No hay unicidad entre paquetes fuentes y paquetes ejecutables. Es decir 3 fuentes pueden producir 8 ejecutables.

**SOFTWARE LIMITADO A UNA DISTRO:** como parte del mecanismo perverso mencionado algunos han confeccionado software con la idea de que solo funcione en una distro. Por ejemplo Yast. O el conjunto de scripts que inicia una computadora en muchas distros. O la forma de hacer un discos compactos “vivos”, o el sistema “apt-get” en Debian. Por suerte ya hay gente intentando que Yast funcione en Debian o Ututo[Ututo:SW] por ejemplo. Pero hay que entender que este software no es mas que paquetes. Artificialmente limitado, pero un paquete mas al fin.

Se suele asociar una distro con estos paquetes, pero esto es un error. Un error mas inducido por este modelo.

### 3. Discusión y propuesta

¿Cual es el problema con las distros tal como las conocemos?

1. Que favorecen la aparición de un trabajo adicional e independiente centrado en la distro y no en el software original: “el empaquetamiento”. Así, se alienta a un conjunto de desarrolladores a centrar su esfuerzo en el concepto de distro en vez de concentrarse en mejorar cada paquete. Con lo cual se desperdicia y se multiplica el trabajo sin mayor diversidad real.

¿Esta mal que empaquetar sea importante? No, es un trabajo muy importante. El problema es que se use y se enfoque en separar a la gente por distros. Es importante que se enfoque en cada paquete. Me gustaría hablar de los empaquetadores del Open Office, y no de los empaquetadores Debian o SUSE[Suse:SW].

2. Que tras el concepto de distro se construye software que solo opera en el marco de esa distro.
3. Algunas distros comerciales, en su afan de vender CDROMs, hacen mucho mas complejo la actualización entre versiones, lo que muchas veces resulta en la necesidad de construir rpm específicos por cada versión de la distribución.
4. Que imponen al usuario novato un problema innecesario como esencial: ¿que distro usar?.

#### La solución

¿Es la solución elegir una distro y abandonar las otras?

No, la solución es cambiar el modelo actual de distros. Sacar el foco de allí y concentrarse en los paquetes. Pensar la forma de disolver el concepto actual de distros y hacer mas fácil la vida a la gente.

Hacer que el empaquetamiento sea un trabajo vinculado estrechamente con el desarrollo del software que empaqueta.

Concentrarse en mejorar los miles de paquetes originales para que interoperen mejor entre si y se instalen directamente desde cada uno de sus sitios. Para que tener 1000 personas trabajando en paquetes Debian, que directamente mejoren los “Makefile” de los originales, Así devuelven al proyecto lo que usan, me parece mas racional y funcional al modelo de software libre. Lo mismo para las otras distros. Los “Makefile” de cada proyecto puede producir los paquetes necesarios para que se usen, tales como deb, rpm, tgz, sh o incluso paquetes completos o que separen documentación, ejecutables y desarrollo.

Armar repositorios que los agrupen y discos compactos que distribuyan esos repositorios, originados en el sitio base de cada proyecto.

Hacer que la diferencia entre un paquete deb y un rpm sea solo de formato, como puede ser la diferencia entre un gz y un bz2. Ambos se abren con un tar -algo<sup>3</sup>.

Hacer un “particionador” general, un instalador general y un vivificador (para CDROMs vivos) general.

Lo ideal es que el paquete pueda generarse e instalarse los “Makefile” de cada software original. Así cada desarrollador y los empaquetadores vinculados producirán los “Makefile” apropiados para que se generen los paquetes que se instalen apropiadamente en una computadora.

Diseñar instaladores, sistemas de menús y demás que puedan interoperar con los paquetes tal como estén en sus repositorios.

### 4. Respuestas a las preguntas frecuentes. FAQ

**¿Entonces cobrar es malo? ¿Qué hace la gente con las facturas a final de mes? La misma competencia se puede dar entre cooperativas. ¿O es que el Software Libre sólo puede serlo gracias al subsidio estatal?**

Para nada es malo. La cuestión es que las formas que se usen para cobrar no aten a la gente.

---

<sup>3</sup>seria bueno que haya un flag que autodetecte que es

El concepto de distro actual esta comercialmente pensado para atar a la gente con el proveedor de esa distro. En cierta forma el concepto de producto armado de esa forma no es demasiado compatible con la filosofía del software libre. Pues te ata a determinada empresa o grupo.

La filosofía de prestación de servicios en un entorno libre implica que puedas cobrar (mucho o poco) por tu servicio, pero que ese servicio no obligue al servido a seguir atado a ti.

La división de comunidad–empresas en grupos excluyentes, cada una con una distro es absolutamente artificial y la podemos superar. Es tan difícil aprender Debian, si sabes bien SUSE, como aprender “postfix” si sabes “sendmail”. O sea el contenido de conocimiento adicional que representa una distro es poca, comparado con el conocimiento de base que uno puede tener con relación al sistema GNU/Linux.

**Coincido contigo, ahora, lo cierto es que elegir una distribución u otra tendrá fuertes implicaciones. No en balde los creadores de las distribuciones vieron cada uno un beneficio en “forkear” y crear una propia.**

No es “la elección de distro” la única opción a tomar, diría que es la menos importante.

Es falso el problema de elegir “la distro”. Es originado por aquellos que hacen de “la distro” su producto.

Para instalar un sistema con GNU/Linux, se deben tomar cientos de decisiones. Elegir “la distro” no te hace avanzar mucho en ello.

Que usar: “KDE”, “gnome”, “blackbox”?, “sendmail” o “postfix”?, que particiones usar?, que fondo de pantalla?, que tiene el menú?, que paquetes selecciono para cada estación y servidor?, “nfs”, “samba”, “openafs” o “gfs” ?, “nis”, “ldap”, “kerberos” o una combinación?, “openwebmail” o “squirrel” u otros?, “firefox” o “konqueror” o ...?, “evolution”, “kmail”, o sólo usar “webmail”?, instalo las máquinas desde un repositorio o preparo un CDROM?, y cientos de posibilidades más.

**No estoy de acuerdo, las variedad distribuciones son justamente un punto fuerte en GNU/Linux. Me parece bárbaro crear estándares, consorcios y ese tipo de cosas. Pero matar la diversidad, sería un error estratégico muy grande. Si creo que se deberían crear estándares tan buenos que permitan convertir un RPM a un DEB de manera que se pueda usar para fines productivos, por poner un ejemplo.**

Entonces estamos de acuerdo.

Si eso pasara y tuviésemos conversión transparente de deb a rpm y de rpm entre si y eventualmente deb entre si (Ubuntu[Ubuntu:SW] - Debian por ejemplo), las distros no serían relevantes, y ese es el punto, justamente.

Y no planteo crear un consorcio, solamente meter la idea en las cabezas de los desarrolladores y empaquetadores para que poco a poco la cosa vaya por ese camino, bueno creo que ya esta pasando, a los sumo estoy dándome cuenta de una tendencia, no se si otro lo habrá pensado en estos términos o descripto. Nada nuevo bajo el sol, solo que puesto en palabras fuertes para hacer un poco de olas.

La aparición del concepto de distro fue un aporte muy grande al movimiento, un paso necesario, pero creo que ya podemos abandonarlo.

Como en la selección natural (explosión cámbrica), se produjo una explosión de diversidad, apenas se vio la importancia del empaquetamiento y se uso ese fenómeno para crear marcas comerciales: RedHat[RedHat:SW]. La respuesta de la comunidad no se hizo esperar y fue excelente: Debian.

Pero llega un momento cuando ya se conoce mucho sobre un tema que la innovación se mueve a otro lugar y es razonable que la diversidad ceda y queden las buenas prácticas y los estándares sobrevivientes. Esto empieza a pasar en el área del empaquetamiento. A nadie se le ocurriría plantear un sistema de paquetes que no sea tan útil como el “apt-get”, simplemente es hora de que esos sistemas, los varios que puedan quedar, manejen todos los formatos de paquetes e interoperen con cualquier repositorio.

**frente al muchas veces trillado cuestionamiento de ¿para qué otra? La respuesta es por qué no? Si la distribución es mala o poco popular, desaparecerá o la usará poca gente. Por otro lado nadie puede asegurar que un fulanita que hace una distro XYZ quiera sumarse a una única distribución, así que no se pierde nada al fin y al cabo. La libertad esta en poder tener la fuente, la diversidad hace a la acción de libertad.**

De acuerdo, pero si la base de las distros es interoperable todos ganaremos mucho. O sea, lo que combato es el concepto de paquetes no interoperables, mas que lo que tu llamas distro y yo subdistro, y que en el futuro pueda quedar como distro.

Los humanos poco a poco sacamos conclusiones sobre cuales son las “buenas prácticas” y tendemos a ellas y creo que una buena práctica para la comunidad sería trabajar en un esquema de paquetes compatibles

entre los diferentes sistemas. ¿Es esto muy difícil de lograr? ¿requiere consensos, discusiones, consorcios? NO, ESO ES LO BUENO, solo requiere centrar el empaquetamiento en el desarrollo de cada paquete original. Esta sucediendo naturalmente.

**No entendí muy bien el tema de convertir RPM a DEB y supongo que viceversa.**

Los dos tipos de paquetes básicamente guardan la misma información.

El problema es como dividir el universo de software en paquetes y como organizar las dependencias.

Eso hace que sea difícil convertir un paquete en otro y que un rpm no sea compatible incluso con otro. Los de RedHat con los de SUSE por ejemplo.

Otro factor que entorpece el trabajo es la necesidad de los distros comerciales de vender CDROMs, que hace que no provean formas fáciles de actualizar todo a las nuevas versiones, sino que proveen parches dentro de cada versión. Esto crea una nueva barrera artificial para los paquetes para la SUSE 9.0 y otros para la 9.3, incompatibles entre sí.

Debian lo pudo resolver sin problemas con sus “upgrades”, pues no tiene interés en vender CDROMs.

**¿y perder lo mejor de Debian? ... El problema de “Makefile” es que no verifica y baja dependencias, es una herramienta “estática” (si las cosas no están donde vos pedís, las cosas no andan ... si, si, usa “autotools” de última**

**Usar debs de otros distros?, como cuales?, si todos usan los de Debian ? ... Podrías decir Ubuntu, pero no aplica. El “main” de Ubuntu es propio de Ubuntu, si bien está basado originalmente en paquetes de Debian, siguen su propio camino. En el caso de Universe se toma de Debian y se compila. Si se agregan cosas nuevas, ahora está Debian Ubuntu que pretende hacer lo contrario, tomar lo que tiene Ubuntu y aplicarlo en Debian.**

No, no. Me refiero a poner en los “Makefile” originales la posibilidad de hacer los paquetes deb o rpm o tgz o todos ellos.

**ahh, creo que ahora voy entendiendo a donde a puntas. Sería algo como hacer : “make deb”, “make rpm”. Eso suena bien y razonable. Un poco complejo tal vez para el programador, debería haber herramientas (macros “m4”, “alien” que funciones más “inteligentemente”, entre otras cosas).**

Exacto. Lo pueden hacer los empaquetadores también. Lo ideal sería “make dist-rpm” o make “dist-deb”

Incluso una opción intermedia sería “dist-rpm-RedHat”, “dist-rpm-SUSE” o “dist-deb-Ubuntu” llegado el caso, aunque sin duda lo mejor sería un rpm y un deb que hagan paquetes “iguales” en el sentido de las dependencias y totalmente transformables uno en otro.

**Tu idea se llama Gentoo**

Quizás se llame “autotools” y “make dist”.

Estoy de acuerdo en que Gentoo dio un paso en esa dirección, un paso más que “Linux from scratch”.

Y Ubuntu complementa la idea pues provee lo que Gentoo no quiso, ejecutables. Así Ubuntu es una distribución y Gentoo un mecanismo para hacer distribuciones.

Pero creo que hay margen para avanzar más y si muchos nos metemos el tema en la cabeza podemos llegar en algún tiempo a anular la diversidad que es como azúcar sintáctica, es decir diversidad inconducente.

Toda esta historia empezó con los rpm, gran invento, pero que permitió separar a RedHat y construirla como empresa, lo que derivó en la respuesta .deb y el “apt”, base de identidad de Debian y su gran invención.

Ahora, sería bueno apuntar a consolidar esas ideas y absorberlas en el movimiento global, de tal forma que en el futuro sea indiferente que instales un paquete desde deb o rpm, así como es indiferente que instales desde un tar con bz2 o gzip, o que puedes usar en una consola gráfica gnome y en otra kde en la misma computadora.

También es interesante apuntar a consolidar los instaladores, que puedan interoperar Yast con Webmin y otros en cualquier distro.

Es decir ortogonalizar el espacio decisorio.

En cierta forma sería justo decir que mi idea se llama slackware[[Slackware:SW](#), ?] o incluso SLS[[Wikipedia:SLSD](#)]. Tomar los ejecutables tal como los generan los desarrolladores, empaquetarlos y meterlos a un repositorio.

El concepto de paquete con dependencias es una sofisticación importante pero debió haber seguido en los paquetes originales y no como capa que justifique el concepto de distribución.

Si cada paquete expone los archivos de librerías y de configuración que provee y los que requiere ... Cada repositorio podría tener su base de datos con esa información, que podría ser consultada por los manejadores

de paquetes. Nada nuevo bajo el sol. Podría también combinarse con los `./configure ...`. Es decir de la misma manera que el programador del paquete programa las “autotools” para saber que “include” y librerías pedir, puede esa misma información usarse a la hora de instalar el paquete.

Así no es necesario indicar el paquete X requiere el paquete Y. Sino X requiere `yyy.1.25.so` por un lado y por otro Y provee `yyy.1.25.so`.

Entonces los repositorios serian mucho mas versátiles.

**Buscar una solución intermedia a corto plano no me parece mal, como es el caso de tener mantenedores de paquetes.**

Desde mi punto de vista lo ideal es no concentrar en distros la capa de mantenedores de paquetes, sean `deb` o `rpm`, sino incorporar ese trabajo en los paquetes originales.

Y que Debian y cualquier distro se arme en base a los “Makefile” de los sitios originales o incluso los `deb` o los `rpm` de otras distros.

**¿Y que “ese” GNU/Linux tenga repositorios con todos los paquetes y fuentes GNU que andan por ahí? si mi interpretación es correcta ... entonces**

**Según lo que entendí en otro correo, no es del todo correcta. El plantea que el paquete de “upstream” (el que hace el programa) ya esté listo para poder meterse en cualquier distro.**

Concentrar el esfuerzo en las fuentes originales.

Lo que traerá como consecuencia que las distros dejen de ser relevantes y sean solo repositorios distintos con cambios menores y selecciones de diferentes paquetes. Eventualmente alguna especializada en un procesador o arquitectura.

Siempre sera bueno tener un CDROM listo para instalar una distro musical o una interesante para los médicos, o una compilada para 486, que se yo.

Pero todo tendra a estandarizarse más, al menos en la base.

**El problema es que “upstream” no suele empaquetar, por costumbre o desinterés. Es por eso que los mantenedores aparecen. No se si está bien o mal, pero a mi personalmente me agrada andar haciendo paquetes, hay desafíos muy interesantes.**

Sin duda, entonces devolvamos esas contribuciones al paquete original.

La voluntad dentro de Debian no falta, para hacer lo que pides, los parches o “bugs” son reportados a “upstream”, el problema suele ser que hay “upstreams” y “upstreams”. Algunos son menos “buena onda” y otros son copados y ayudan aplicando estos parches, otros se resisten. Como ejemplos se me ocurre ahora `Gtk/Gnome` (también vale aclarar que varios “Debian developers” (DD) son “Gnome developers” y como mal “upstream” esta el de “centericq” por ejemplo (tipo totalmente ególatra que no acepta un solo parche).

Hay muchas veces que los “Makefile” rompen los “policy” (acá entraríamos en otro tema, “un-generic policy” por ejemplo :D). Un caso que estoy tratando justo ahora es un paquete para Ubuntu. El autor tiene en el “Makefile” que instale los documentos “INSTALL”, “ChangeLog”, etc en `/usr/share/nombre`. El “policy” de Debian dice que “INSTALL” no debe copiarse, pues no tiene sentido. “ChangeLog” debe ir comprimido con `gzip` o `bzip2`, por lo que tuve que parchar el “Makefile” para que el paquete cumpla dicho “policy”. Este es solo un pequeño ejemplo y fácil de resolver porque usa “autotools”, pero hay casos donde el autor provee directamente el “Makefile” hecho a mano y es mucho mas difícil sacar un paquete que cumpla todo al pie de la letra.

¿Porque no mejorar los “Makefile” de cada proyecto?

Seria bueno generar un parche con todos los cambios propuestos, incluso un “Makefile” decente, cuando no exista. Entonces no existiría un paquete Debian fuente, sino un parche Debian fuente. Si el autor original no lo acepta, seguirán estando esos dos un buen tiempo.

**Ojo, los DD no solo hacen paquetes, también hacen la distro completa (son los únicos autorizados a hacer “commits” directos de códigos del proyecto). Lo que si hacen los DD es aprobar los “uploads” de aquellos que no son DDs, para garantizar su calidad.**

Claro, pero que es una distro si no la suma de paquetes. Para lo que tu dices ¿porque no hacer paquetes generales?

Estoy de acuerdo en la idea, sería ideal (y hasta utópico) poder plantearlo a todas las distros y que sea aceptado (hasta diría que sería una mini-guerra civil :D). Desgraciadamente no lo tenemos y personalmente creo que nunca lo tendremos (o por lo menos no creo vivir para verlo :).

Hay un sitio que busca hacer rpm genéricos, sería interesante plantear una cooperación Debian y ese sitio para sacar rpm y deb con las mismas dependencias y forma de subdividir un paquete.

[OpenPKG:SW]

**Opinión personal: las cosas están como están porque a cada uno le gusta hacerlo a su manera.**

Y si.

Ese es el punto, que la gente ha focalizado la “marca” en la cuestión distro, las comerciales por cuestiones de dinero, pues instalaron la idea de distro-producto.

Y Debian como respuesta a ese fenómeno concentro su esfuerzo en la propia marca comunitaria. Si 1000 personas se agruparon en Debian con su idea, también pueden ponerse de acuerdo en otras.

Creo que hoy en día se podría concentrar el esfuerzo en disolver ese foco y plantearlo en otro lado, un buen instalador general -algo como 123L que propuse- algún manejador de paquetes genérico, metadistros, etc

Es cuestión de ir proponiendo la idea y poco a poco irá sucediendo -si la idea sirve, claro- a medida que mas gente “compre” esa idea.

**pero coincido que en general todas las subDebian no son mas que eso sub distros, quizás Ubuntu pueda llegar a ser otra distro.**

**Knoppix pondría como otra distro bastante alejada de Debian. Si bien usa muchísimo, el autor ha hecho grandes cambios en la parte de “booteo” (mucho relacionado con la detección de hardware principalmente).**

Claro y sería bueno que eso sea -y probablemente lo sea- un paquete mas en algún momento pueda adoptar Debian u otra distro.

Así como alguien quiere portar Yast para Debian y Arturo lo está haciendo para Ututo.

Una idea interesante es “sourceinstall”[Fontana:GSI-05], aunque si se leen sus contras se nota que sería bueno que interopere con un repositorio y que sea automatizable

tal como hace “apt get source”

**Aún si hubiese un solo particionador y un solo instalador y un solo todo, habría distros.**

Depende de que llames como distro.

Yo en general llamo distro a un conjunto de paquetes ejecutables Así por ejemplo Debian es una distro, con cientos de variantes que se diferencian en al menos un paquete: el fondo de pantalla, el menú y algunos agregados.

Es decir hay tres o cuatro distros importantes que concentran trabajo real al menos de compilación: SUSE, RedHat, Debian, Ututo, Ubuntu, Mandriva[Mandriva:SW] quizás alguna más.

Algunas de ellas además se toman el trabajo de hacer y mejorar muchos paquetes, pero eso lo tomo como trabajo de hacer un paquete. SUSE con Yast, o los scripts de arranque, etc. O Debian con “apt”. Sería bueno que los desarrolladores de esos paquetes se saquen su “distro” de la cabeza y los piensen para cualquier pc con GNU/Linux. Su aparición fue necesaria, no había otra forma de hacerlo, ahora hemos progresado mucho.

Debian, SUSE, RedHat, Gentoo, Ututo y seguramente otras, tienen además el trabajo de empaquetado, que mantienen y construyen, lo que es un trabajo enorme, lo que propongo es que ese trabajo se vuelque en los paquetes originales y sea compartido por todos.

¿Que quedaría para las distros? El trabajo de compilar, y decidir que paquete meten en el CDROM y hacer los menús.

O sea un trabajo duro: “compilar”, pero relativamente simple y un trabajo estético y de toma de decisiones.

Entonces las distros tendrían mucha menos relevancia.

Por un lado aparecerían miles de distros, una por cada empresa por ejemplo.

Pero por otro lado todas las distros serían mucho mas compatibles. Hasta es probable que solo quede un conjunto de rpms y otro de debs y que finalmente sean equivalentes incluso a nivel de dependencias.

**Si yo quiero un SO “listo para instalar” que tenga básicamente software para oficina, y vos quieres hacer uno centrado en software para programadores, ya tenemos dos distros...**

Claro, lo armas y listo, o te lo bajas armado,

O repositorios con diferentes paquetes en un CDROM. Hoy ya con cualquier distro puedes hacer eso.

**Repositorio podría haber uno solo (difícilmente, pero buéh imaginemos), pero la idea de distro esta bastante asociada a un medio de distribución físico: no puedo meter en un solo CDROM (o DVD)**

**todo el repositorio de todo el software libre que existe, y aún si distribuyera no se cuantos DVDs con todo, no sería práctico.**

Claro la cuestión es no centrar el asunto en la “distro”, que deje de ser relevante.

**Obviamente que todo sería mucho más cómodo si esa fuese la ÚNICA diferencia entre distros (y no existiesen las otras: escritorio gnome/kde/etc, paquetes dpkg/rpm/tar, etc)**

En varias distros actuales tienes eso como opciones, son parte de los cientos de opciones que tienes al instalar un equipo.

**Sobre la inutilidad de la guerra de las distros**

Con respecto al tema guerra de distros, creo que estas discusiones son muy pertinentes.

Ayudan a todos nosotros a conocer las impresiones que podemos tener sobre cada distro y sus fortalezas y debilidades. Así poder lograr sacar las buenas ideas y utilizarlas en todas.

## 5. Instalador independiente de la distro. Sistema 123L

**Especificaciones estándar para distribuir sistemas GNU/Linux**

El proceso de instalación de GNU/Linux ha sido siempre uno de los factores más complejos a la hora de su difusión. Se propone un esquema que solucione muchos de esos problemas.

Características

- Modular, con pasos o módulos cortos, ortogonales y fácilmente repetibles e investigables si surgen problemas, con test que permitan determinar si el paso se cumplió y porque no si hubo fallos.
- 3 pasos fundamentales, por lo que se puede denominar 1 2 3 y listo, o PIA: “particionar”, instalar, arrancar.
  - **PENSAR.**
  - 1: “**Particionar**”.
  - 2: **Instalar** GNU/Linux autónomo.
  - 3: **Arrancar** instalar el arranque (“boot”) e iniciar el sistema.
  - 4. **LISTO**, usar y/o hacer crecer GNU/Linux autónomo hasta la configuración deseada.

Los 3 pasos, controlados por una “Makefile”.

- Uso mínimo de interpretes y librerías, para que pueda funcionar en sistemas GNU/Linux/ chicos.

Debe poder ser ejecutado desde cualquier tipo de sistema y circunstancia. Desde distros vivas completas, o una distro viva instalador específico, o un GNU/Linux ya instalado en la máquina para instalar el GNU/Linux objetivo o para reparar o modificar algo.

El sistema debe poder instalar cualquier GNU/Linux autónomo.

¿Que es un GNU/Linux autónomo?

Un sistema GNU/Linux que tenga lo mínimo imprescindible para “bootear” y administrarse solo, en particular este “Makefile”.

Consistirá en un paquete: núcleo + imagen inicial+ programas básicos.

El núcleo, la imagen, y los programas básicos deberán caber en un diskette cada uno, y deberán poder ser copiados de allí. O sea el paquete núcleo+imagen+pb debe poder estar en tres diskettes o menos.

Todo estará contenido en algún paquete, la distro viva instaladora, sus árboles, etc.. Todo se debe poder constituir agrupando paquetes y procesándolos con scripts auto-contenidos.

La instalación del GNU/Linux autónomo consiste en copiar el núcleo, la imagen e instalar en las particiones designadas los programas mínimos para su funcionamiento y autonomía.

Este conjunto es un paquete y se actualiza junto. Cada vez que se instala, actualiza o retrasa se hace con el “Makefile” del sistema 123L.

El núcleo+imagen inicial+pb: sistema GNU/Linux autónomo es un solo paquete.

Los paquetes son las unidades mínimas que deben instalarse sin poder separarse, si se instala, actualiza o retrasa se hace en bloques.

El sistema “vivo” instalador también. Puede instalar su propio núcleo+imagen+pb, o instalar otras de otros diskette. Esto significa que el núcleo del diskette y su imagen del sistema vivo instalador deben ser en sí un sistema GNU/Linux autónomo.

El sistema podrá montar un conjunto de arboles opcionales (directorios y puntos de montaje) de existir.

Este sistema autónomo ofrece siempre múltiples consolas, una de ellas ingresa automáticamente siempre un usuario (se puede además hacer manual) que despliega un programa básico de instalación, actualización y retraso de paquete. Este programa básico usa como dato el nombre del paquete (el mismo que se utiliza en el menú, para agrupar a todos los ejecutables y archivos de configuración del mismo) y sin información adicional debe poder ejecutar el programa pre-designado del mismo, o instalar el paquete si no lo está. Si hay en repositorios disponibles una actualización ofrecerla. Con comandos adicionales se podrá cambiar la versión para atrás, o eliminarlo. Cualquiera de estas opciones debe llevar a una actualización o retraso de sus dependencias.

Pueden existir varios paquetes con el mismo nombre, además de las versiones puede tener características diferenciales, como procesador para el que están optimizado, o diferentes opciones (“ldap” con “kerberos” por ejemplo)

- Una vez instalado (en el paso listo) el sistema ya funciona y tendrá un sistema de incorporación de paquetes y actualización (o retraso). Este sistema funcionara contra cualquier repositorio, sea por red o por CDROM.
- El sistema 123L se podrá instalar en cualquier GNU/Linux.
- En todo momento el sistema debe saber que repositorios tiene disponibles. Si coloco un CDROM con un repositorio el sistema debe notarlo.

Los repositorios serán coherentes, es decir tendrán paquetes que interoperarán entre si y todo paquete tendrá en el repositorio todo lo que necesita para funcionar. Los repositorios podrán ser múltiples, es decir basarse en varios medios. Un CDROM puede basarse en un repositorio en la red. Si pongo un CDROM de este tipo y no hay red, ese repositorio no se activa.

- El sistema de actualización debe copiar todos los paquetes necesarios para dejar operable al sistema en un cache local seguro antes de proceder. (no red, no CDROM) Si debe instalar mas que lo que puede almacenar el cache deberá dividir la tarea adecuándose al tamaño del cache.

Así:

- Eliminar los paquetes que no se vayan a usar.
- Copiar los primeros paquetes coherentes instalarlos.
- Copiar otra tanda y volver a repetir.

Nunca el sistema debe quedar con su sistema autónomo incompleto o con paquetes no ejecutables.

Las tareas pautadas de este tipo deben quedar en un “Makefile”, cosa de que cualquier problema se pueda retomar desde el paso faltante.

Así, si se determinan 10 operaciones de cache e instalación, cualquiera de las 10 deben poder retomarse después de una colgada.

Todos los archivos de configuración (los indicados o los modificados a mano) de un paquete desinstalado deben preservarse, eventualmente el paquete podrá tener medios de decidir como auto-configurarse con esto.

Existirá un directorio con estos archivos viejos, clasificados por nombre del super-repositorio origen, nombre del paquete, versión del paquete y versión del cambio de la configuración usando un sistema simplificado a la “cvs”.

Así estas distros tendrán un sistema elemental de control de cambios.

- los datos de usuario: “home”, “var”, “root”, “srv”, “usr/local” y archivos de configuración alterados, se preservan en todo cambio.

Para sistemas vivos se pueden preparar esquemas especiales para almacenar estos directorios.

- Los paquetes deben tener deltas. No hay versiones de los sistemas. El super-repositorio contiene todos los paquetes en todas las versiones y los provee junto con sus deltas. Es posible construir muchísimos sistemas diferentes.

## 6. Sistema compilador

Cada sistema compilador debe imponer un nombre a sus paquetes, de haber varios coordinados pueden usar el mismo nombre. Por defecto un sistema genera paquetes con su propio nombre.

Cada repositorio usa paquetes producidos en sistemas con mismo nombre. Cada GNU/Linux instalado usa paquetes con el mismo nombre.

Cambiar el nombre implica eliminar todos los paquetes incluido el base e instalar otro.

Los archivos de configuración son preservados en el directorio especial de las particiones.

Cuando se producen paquetes para auto-consumo el super-repositorio local es un medio mas del repositorio en uso y se producen paquetes del mismo nombre

## 7. Sistema de Menú

Capa paquete contendrá la información sobre en que rama del menú colocar sus entradas

Son entradas, una por defecto con el programa principal.

Otras con otros ejecutables del paquete

Otras con los archivos de configuración editables, enlazadas con el editor de referencia. Lo ideal seria que todos los archivos de configuración de todos los paquetes tengan un formato único al estilo de variables “bash”<sup>4</sup>. Mientras tanto un sistema de administración puede tener un conjunto de estos archivos en paralelo con medios de edición de los mismos y de generación–lectura de los archivos de configuración reales.

Otra con la posibilidad de remover el paquete.

Otra con la posibilidad de actualizarlo cuando exista.

Cada paquete además tendrá que librerías provee.

Así listara: librerías provistas, ejecutables provistos, archivos de configuración provistos.

Un sistema para manejar menús podrá acomodar todo el software de los repositorios disponibles. Si se pide ejecutar algo no instalado se instala.

El menú podrá ser ejecutado en un navegador “web”, adaptado para ejecutar, y en terminales de texto o gráficas.

download, patch and untar comconfigure (leer script a partir variable) ejecuta configure make make pkg make install (aunque no use paquete toca base de datos) make execonfig (a partir variables bash)

urpmi mandrake

dpkg-deb dpkg dselect apt

apt4rpm apt-rpm

ebuild un paquete emerge dependencias portage sofisti

Referencias varias: [Wikipedia:LD] [LinuxISO:SW] [Wikipedia:Ptg] [FHS:SW] [Wikipedia:CLD] [Distrowatch:SW] [LWN:LDL] [Distromania:SW] [LinuxQuestions:DR] [FrozenTech:LL] [LMP:SW] [LDC:SW] [FreeStandars:SW] [LinuxBase:SW] [Gentoo:WWP] [Gentoo:EH] [Gentoo:API] [Hamilton:SUT] [Bailey:MR-97] [Vaughan:GAA-00] [Kitenet:ACD] [Cortes:ACP] [apt4rpm:FAQ] [Kojima:RPA] [Bodnar:WWR] [Raymond:SRP] [FreeBsd:FPH] [Miller:RMC] [Crasta:EPW] [GNU:GL] [Uekawa:DLP:02] [LinuxQuestions:LRC] [Vairagade:MPS-03] [Poirier:PSR-01] [Autoconf:AMA] [Yamato:AP] [Sweet:SDU-99] [rpmorg:SW] [Banks:M] [Worth:SAT] [Knight:SUG-04] [Debian:HMP] [Jackson:DPM-96] [Benson:P]

<sup>4</sup>Tarea enorme

## 8. Derechos y estándares

Este documento:



- puede ser utilizado por cualquiera bajo los términos de la GFDL. No contiene secciones invariantes.

<http://www.gnu.org/copyleft/fdl.html>



- cumple los estándares de la w3c en su versión html.

<http://www.w3c.org>

## Referencias

[apt4rpm:FAQ] apt4rpm. Apt-rpm questions and answers.

<http://apt4rpm.sourceforge.net/faq.html>.

[Autoconf:AMA] Autoconf. The autoconf macro archive.

<http://ac-archive.sourceforge.net/>.

[Bailey:MR-97] Edward C. Bailey. Maximum rpm, taking the red hat package manager to the limit, 1997.

<http://www.openpkg.org/doc/book/maximum-rpm.html/>.

[Banks:M] Greg Banks. Maketool.

<http://www.alphalink.com.au/~gnb/maketool/index.html>.

[LinuxBase:SW] Linux Base. Sitio web.

<http://www.linuxbase.org/>

[http://en.wikipedia.org/wiki/Linux\\_Standard\\_Base](http://en.wikipedia.org/wiki/Linux_Standard_Base).

[Benson:P] Dave Benson. Pkgwrite.

<http://ffem.org/daveb/pkgwrite/>.

[Bodnar:WWR] What's wrong with rpm?, 2002.

<http://distrowatch.com/dwres.php?resource=article-rpm>.

[LDC:SW] Linux Distribution Chooser. Sitio web.

<http://www.zegeniestudios.net/ldc/index.php>.

[Cortes:ACP] Carlos Cortes. Alien: Conversor de paquetes deb, rpm, tgz y slp en linux.

<http://bulma.net/body.phtml?nIdNoticia=1186>.

[Crastra:EPW] James Crastra. Elf prelinking and what it can do for you.

<http://www.crastr.us/james/articles/prelink.php>

[http://linuxcommand.org/man\\_pages/prelink8.html](http://linuxcommand.org/man_pages/prelink8.html).

[Debian:HMP] How to make deb packages.

<http://linuxdevices.com/articles/AT8047723203.html>.

[Debian:SW] Debian. Sitio web.

<http://www.debian.org/>.

[Distrowatch:SW] Distrowatch. Sitio web.

<http://distrowatch.com/>.

- [Distromania:SW] Ditromania. Sitio web.  
<http://www.distromania.com/>.
- [Fontana:GSI-05] Claudio Fontana. Gnu source installer, 2005.  
<http://www.gnu.org/software/sourceinstall>.
- [FrozenTech:LL] FrozenTech. Livecd list.  
<http://www.frozentech.com/content/livecd.php>.
- [LFS:SW] Linux from scratch. Sitio web.  
<http://www.linuxfromscratch.org/>.
- [Gentoo:EH] Gentoo. Ebuild howto.  
<http://www.gentoo.org/proj/en/devrel/handbook/handbook.xml?part=2&chap=1>.
- [Gentoo:API] Gentoo. A portage introduction.  
<http://www.gentoo.org/doc/en/handbook/handbook-x86.xml?part=2&chap=1>.
- [Gentoo:SW] Gentoo. Sitio web.  
<http://www.gentoo.org/>.
- [Gentoo:WWP] Gentoo. Working with portage.  
<http://www.gentoo.org/doc/en/handbook/handbook-x86.xml?part=3>.
- [GNU:GL] GNU. Gnu libtool.  
<http://www.gnu.org/software/libtool/manual.html>.
- [Hamilton:SUT] Bruce Hamilton. A sysadmin's universal translator (rosetta stone).  
<http://bhami.com/rosetta.html>.
- [LinuxISO:SW] Linux ISO. Sitio web.  
<http://www.linuxiso.org/>.
- [Jackson:DPM-96] Ian Jackson y Christian Schwarz. Debian policy manual, 1996.  
<http://www.debian.org/doc/debian-policy/>.
- [Kitenet:ACD] Joey Kitenet. A comparison of the deb, rpm, tgz, and slp package formats/comparing linux/unix binary package formats.  
<http://debian-br.sourceforge.net/txt/alien.html>  
<http://www.kitenet.net/~joey/pkg-comp/>.
- [Knight:SUG-04] Steven Knight. Scons user guide 0.96, 2004.  
<http://www.scons.org/doc/HTML/scons-user/book1.html>.
- [Kojima:RPA] Alfredo K. Kojima. An rpm port of apt, 2000.  
<http://freshmeat.net/articles/view/192/>.
- [LinuxQuestions:DR] LinuxQuestions. Distro reviews.  
<http://www.linuxquestions.org/reviews/index.php>.
- [LWN:LDL] LWN. The lwn.net linux distribution list.  
<http://old.lwn.net/Distributions/index.php3>.

- [Mandriva:SW] Mandriva. Sitio web.  
<http://www.mandriva.com/>.
- [Miller:RMC] Peter Miller. Recursive make considered harmful.  
<http://www.pcug.org.au/~millerp/rmch/recu-make-cons-harm.html>.
- [Nguyen:CCD-04] Bihn Nguyen. A constructive critique of debian linux, 2004.  
<http://desktoplinux.com/articles/AT7588639943.html>.
- [OpenPKG:SW] OpenPKG. Sitio web.  
<http://www.openpkg.org/>  
<http://www.openpkg.org/doc/articles/sysadmin/article.html>.
- [Poirier:PSR-01] Dan Poirier. Packaging software with rpm, 2001.  
<http://www-128.ibm.com/developerworks/library/l-rpm1/index.html>  
<http://www-128.ibm.com/developerworks/library/l-rpm2/index.html>  
<http://www-128.ibm.com/developerworks/library/l-rpm2/index.html>.
- [LMP:SW] The Linux Mirror Project. Sitio web.  
<http://www.tlm-project.org/>.
- [FreeBsd:FPH] The FreeBSD Documentation Project. Freebsd porter's handbook, 2000.  
[http://www.freebsd.org/doc/en\\_US.ISO8859-1/books/porters-handbook/index.html](http://www.freebsd.org/doc/en_US.ISO8859-1/books/porters-handbook/index.html).
- [LinuxQuestions:LRC] Linux Questions. Library related commands and files.  
[http://wiki.linuxquestions.org/wiki/Library-related\\_Commands\\_and\\_Files](http://wiki.linuxquestions.org/wiki/Library-related_Commands_and_Files).
- [Raymond:SRP] Eric Steven Raymond. Software release practice howto, 2000.  
<http://en.tldp.org/HOWTO/Software-Release-Practice-HOWTO/>.
- [RedHat:SW] RedHat. Sitio web.  
<http://www.redhat.com>.
- [rpmorg:SW] rpmorg. Sitio web.  
<http://www.rpm.org/>.
- [FreeStandars:SW] Free standars group y Wikipedia. Sitio web.  
<http://www.freestandards.info/>  
[http://en.wikipedia.org/wiki/Free\\_Standards\\_Group](http://en.wikipedia.org/wiki/Free_Standards_Group).
- [Slackware:SW] Slackware. Sitio web.  
<http://www.slackware.com/>.
- [Suse:SW] Suse. Sitio web.  
<http://www.suse.com>.
- [Sweet:SDU-99] Michael Sweet. Software distribution using the esp package manager, 1999.  
<http://www.easysw.com/epm/documentation.php>.
- [Ubuntu:SW] Ubuntu. Sitio web.  
<http://www.ubuntulinux.org/>.

- [Uekawa:DLP:02] Junichi Uekawa. Debian library packaging guide, 2002.  
<http://www.netfort.gr.jp/~dancer/column/libpkg-guide/libpkg-guide.html>.
- [Ututo:SW] Ututo. Sitio web.  
<http://www.ututo.org/>.
- [Vairagade:MPS-03] Mugdha Vairagade. Manage packages using stow, stow offers a flexible, capable alternative to rpm, 2003.  
<http://www-128.ibm.com/developerworks/linux/library/l-stow/>.
- [Vaughan:GAA-00] Gary Vaughan, Ben Elliston, Tom Tromey, y Ian Taylor. Gnu autoconf, automake and libtool, 2000. ISBN 1-57870-190-2  
<http://linux.duke.edu/~mstenner/free-docs/autobook-1.3/autobook.html>  
<http://sources.redhat.com/autobook/>.
- [Welton:C-00] N. Welton, David. Comentario sobre apt-get, 2000.  
<http://lists.debian.org/debian-devel/2000/debian-devel-200012/msg00969.html>.
- [Worth:SAT] D.J. Worth y C. Greenough. A survey of available tools for developing quality software using fortran 95, 2005.  
[http://www.sesp.cse.clrc.ac.uk/Publications/tools\\_report/](http://www.sesp.cse.clrc.ac.uk/Publications/tools_report/).
- [Wikipedia:CLD] Wikipedia. Comparison of linux distributions.  
[http://en.wikipedia.org/wiki/Comparison\\_of\\_Linux\\_distributions](http://en.wikipedia.org/wiki/Comparison_of_Linux_distributions).
- [Wikipedia:LD] Wikipedia. Linux distribution.  
[http://en.wikipedia.org/wiki/Linux\\_distribution](http://en.wikipedia.org/wiki/Linux_distribution).
- [Wikipedia:Ptg] Wikipedia. Portage.  
[http://en.wikipedia.org/wiki/Portage\\_\(software\)](http://en.wikipedia.org/wiki/Portage_(software)).
- [Wikipedia:SLSD] Wikipedia. Sls distribution.  
[http://en.wikipedia.org/wiki/Softlanding\\_Linux\\_System](http://en.wikipedia.org/wiki/Softlanding_Linux_System).
- [Yamato:AP] Masatake Yamato. Autopack.  
<http://www.gyve.org/~jet/autopack/>  
<http://lists.gnu.org/archive/html/automake/2001-11/msg00004.html>  
<http://lists.debian.org/debian-devel/2005/03/msg00433.html>  
[http://autoconf-archive.cryp.to/ax\\_dist\\_rpm.html](http://autoconf-archive.cryp.to/ax_dist_rpm.html).
- [FHS:SW] Filesystem Hierarchy Standard y Wikipedia. Sitio web.  
[http://en.wikipedia.org/wiki/Filesystem\\_Hierarchy\\_Standard](http://en.wikipedia.org/wiki/Filesystem_Hierarchy_Standard)  
<http://www.pathname.com/fhs/>.