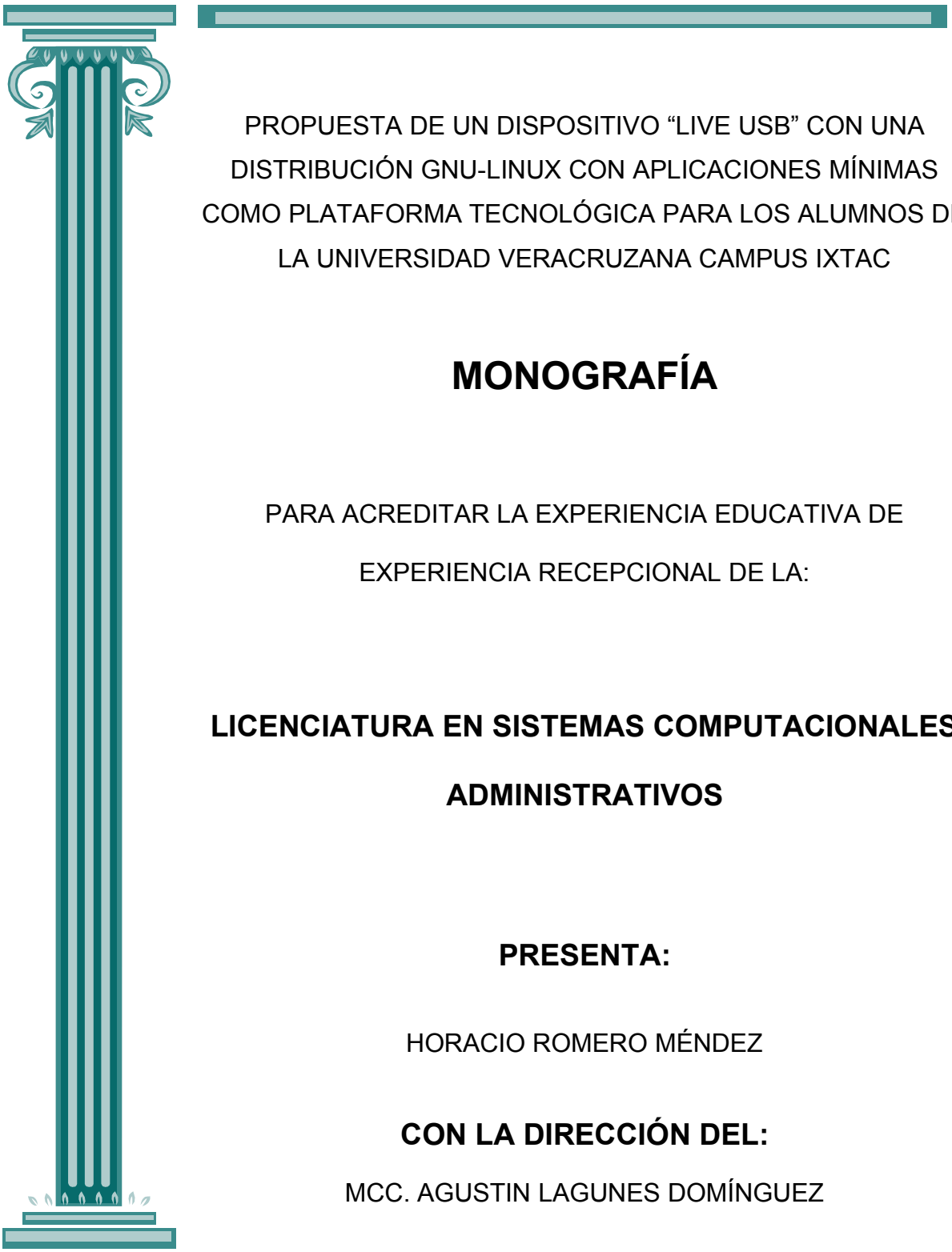
UNIVERSIDAD VERACRUZANA

Facultad de Contaduría y Administración. Campus Ixtac



PROPUESTA DE UN DISPOSITIVO “LIVE USB” CON UNA DISTRIBUCIÓN GNU-LINUX CON APLICACIONES MÍNIMAS COMO PLATAFORMA TECNOLÓGICA PARA LOS ALUMNOS DE LA UNIVERSIDAD VERACRUZANA CAMPUS IXTAC

MONOGRAFÍA

PARA ACREDITAR LA EXPERIENCIA EDUCATIVA DE

EXPERIENCIA RECEPCIONAL DE LA:

LICENCIATURA EN SISTEMAS COMPUTACIONALES ADMINISTRATIVOS

PRESENTA:

HORACIO ROMERO MÉNDEZ

CON LA DIRECCIÓN DEL:

MCC. AGUSTIN LAGUNES DOMÍNGUEZ

IXTACZOQUITLAN, VER. JUNIO DE 2010

Índice

Introducción....3
Capítulo I. Software libre....6
1.1 La definición de software libre....6
1.1.1 Categorías del software libre....8
1.1.2 Categorías del software no libre....10
1.2 Las licencias y patentes....12
1.2.1 Licencias para software....13
1.2.2 Licencias para documentación....16
1.2.3 Licencias para trabajos....16
1.2.4 Patentes....17
1.3 El proyecto GNU y la FSF....17
1.3.1 GNU....18
1.3.2 FSF....19
1.4 Mach y Hurd....19
1.4.1 Mach....20
1.4.2 Hurd....21
1.5 De Unix a Linux....23
1.5.1 Unix....24
1.5.2 Linux....25
1.6 GNU/Linux....30
1.7 Distribuciones....31
Capítulo 2. Software libre en la educación....36
2.1 Software libre en las escuelas....36
2.2 Proyecto LULA....40
2.3 La Universidad Veracruzana....44
Capítulo 3. Live USB....49
3.1 Lives....49
3.2 La distribución....50
3.3 Linux Mint....51
3.3.1 instalación....51
3.3.2 Uso y configuración....67
3.3.3 Creación del liveUSB....75
Conclusiones....91
Bibliografía....93

Introducción

Es indubitable la influencia que ha ejercido la tecnología a nuestro modo de vida, desde avances científicos cada vez más importantes, hasta hacer a nuestra vida diaria más cómoda.

Es irónico entonces que mientras vivimos en una sociedad de información, seamos una sociedad no informada y siendo que conocimiento no siempre equivale a comprensión, entonces tendemos a quedarnos con la tecnología que todos los demás usan y nos enseñan, sin buscar alternativas por cuenta propia, muchas de las cuales podrían ser mejores y beneficiarnos.

Aunque es lógico que no en todos los campos puedan implementarse cambios pues puede afectar significativamente su producción y ritmo de trabajo. Como por ejemplo el campo laboral, donde un cambio de tecnología (física o lógica) puede traer mayores pérdidas que ganancias si se llegase a implementar, al menos esto a corto plazo.

El campo educacional, también se vería afectado, tanto por el lado docente, administrativo, así como por el lado estudiantil, aunque el impacto sería mayor si el cambio sería a nivel software que de hardware.

Es por ello que se plantea una transición parcial progresiva, en la cual se busca no perder la compatibilidad, y para aquellos que no lleven a cabo no se vean perjudicados utilizando una memoria USB o Pendrive, y un sistema operativo libre, que permita su modificación sin violar derechos de autor, y además se ve constantemente enriquecido por su propia comunidad.

Todo esto será reforzado por manuales, tutoriales y video-tutoriales, que buscan una amigable transición, y que a su vez, será un apoyo para la comunidad de cada uno de los elementos que serán tomados en dicha implementación.

Aunque las necesidades de cada carrera son diferentes entre si, lo es también las necesidades de cada grado, por lo que las personas que estén a cargo del proyecto y de darle continuidad, no solo tendrán nuevos conocimientos sobre su área, sino que adquirirán conocimiento sobre todas las áreas, así como una experiencia enriquecedora sobre su misma área.

Esto puede marcar un precedente, puesto que la implementación del software libre en las universidades ha sido la propuesta de diversos estudiantes que como principal obstáculo ha sido el impacto que puede tener dicho cambio ante estudiantes de áreas ajenas al mismo. Por lo mismo, no se ha podido implementar tal propuesta tanto en ámbito laboral, como gubernamental (con algunas excepciones).

Otro efecto colateral es la oportunidad que tiene el estudiante de conocer alternativas a software dominante en el mercado, que le dará una visión más amplia sobre el área, el cual se ve seriamente afectado por un único sistema operativo, así como una única suite ofimática. De él será la decisión de qué software seguir utilizando, pero esa decisión será tomada ya con conocimiento de causa, podrá proponer nuevas alternativas durante su estancia laboral que beneficien tanto a la empresa, como a su experiencia laboral.

Vemos que tiene diversos beneficios el uso de software libre tanto para los usuarios (alternativas gratuitas, sin violar patentes, visión más amplia) como para los desarrolladores del mismo software (uso y difusión de su trabajo), dando la oportunidad que la comunidad estudiante a su vez aporte con tutoriales personalizados, así como con ideas y sugerencias.

Por todo lo anterior, es necesario entender cada una de las partes que conforman el proyecto.

Capítulo I

El software libre

Capítulo I. Software libre

Para entender las bondades que ofrece el software libre, en necesario comprender lo que es y lo que no es. En éste capítulo se abordará ese tema, además de como ha evolucionado a lo largo de los años para convertirse en un producto profesional.

1.1 La definición de software libre

Para nuestro idioma, no existe una ambigüedad en torno a la palabra “libre”, pero para el idioma inglés “free” tiene varias acepciones, entre las cuales son “Libre” y “Gratis”.

El software libre no implica que sea gratis, mas bien implica una libertad, la de ejecutar, copiar, distribuir, estudiar, cambiar y mejorar el software. En síntesis significa que los usuarios tienen las cuatro libertades esenciales.

1. La libertad de ejecutar el programa sea cual sea el propósito.
2. La libertad para modificar el programa para ajustarlo a tus necesidades. (Esto implica que exista un libre acceso al código fuente).
3. La libertad de redistribuir copias, ya sea de forma gratuita, ya sea a cambio del pago de un precio.
4. La libertad de distribuir versiones modificadas del programa, de tal forma que la comunidad pueda aprovechar las mejoras introducidas. [1]

[1] STALLMAN, Richard. “Software libre para una sociedad libre” Traficantes de sueños Madrid (España), 2004 P. 19

Por lo anterior, entendemos entonces que el software libre es aquel cuyos usuarios gocen de las 4 libertades, son libres de redistribuir copias con o sin modificaciones de forma gratuita, o pidiendo un pago por ello. Asimismo se permite utilizar un programa por cualquier individuo u organización en cualquier sistema informático, con cualquier fin y sin verse obligado de notificar al desarrollador o a alguna entidad en concreto.

Así se le confiere la libertad de ejecutar el programa, lo cual significa ”la libertad para cualquier tipo de persona u organización de usarlo en cualquier tipo de sistema de computación, para cualquier tipo de trabajo y propósito, sin estar obligado a comunicarlo a su programador, o alguna otra entidad específica. Como usuario es libre de ejecutar un programa para sus propósitos; y si lo distribuye a otra persona, también es libre para ejecutarlo para sus propósitos, pero [...] no [se] tiene derecho a [imponerse] [...] propios propósitos”[2].

Richard Stallman menciona además “La libertad de redistribuir copias debe incluir las formas binarias o ejecutables del programa, así como el código fuente; tanto para las versiones modificadas como para las no lo están. (Distribuir programas en forma de ejecutables es necesario para que los sistemas operativos libres se puedan instalar fácilmente). Resulta aceptable si no existe un modo de producir una formato binario o ejecutable para un programa específico, dado que algunos lenguajes no incorporan esa característica, pero debe tener la libertad de redistribuir dichos formatos si encontrara o programara una forma de hacerlo.”[3]

Lo anterior es la base de todo software que se considere libre, el acceso al código fuente, lo que permite mejorarlo, además de encontrar posibles fallas en el mismo.

Mientras que en el software proactivo, su principal característica es el no poder acceder al código fuente, más que sólo las personas que tienen derecho a ello,

[2] http://www.gnu.org/philosophy/free-sw.es.html, 28-04-2010, 14:23
[3] http://www.gnu.org/philosophy/free-sw.es.html, 14-03-2010, 02:08

por lo tanto si se desea hacer una mejora, hay que pedirla al programador, y esperar que él la realice y distribuya su actualización.

1.1.1 Categorías del software libre

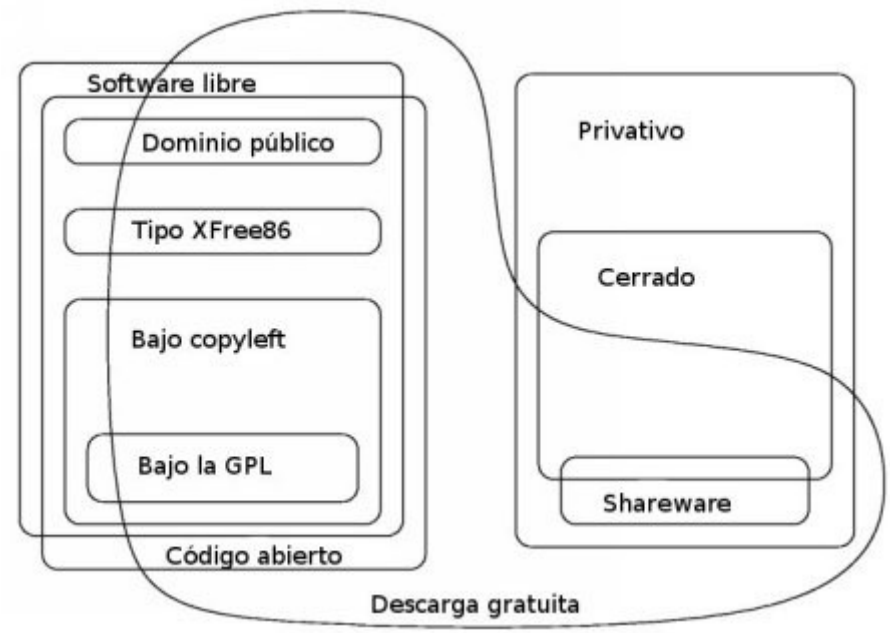


Ilustración 1: Categoría del software libre

- Software libre

  Ésta categoría es para el software que puede ser usado por cualquier persona, además puede ser distribuido y copiado, sin importar que sea el original o una modificación, ya sea pagando por él o no. Normalmente para que esto se cumpla se debe tener acceso al código fuente.

  Precisamente ésta es la base para la creación de proyecto GNU [4]

[4] GNU es un conjunto de programas libres, es explicado a detalle en la sección 1.6

"Si un programa es libre, puede ser potencialmente incluido en un sistema operativo libre tal como GNU o los sistemas GNU/Linux libres." [5]

- Software de código abierto (Open Source)

  La frase "Código abierto" es usada por muchas para indicar que es software libre, aunque no es precisamente lo mismo.

  El sitio oficial de la GNU indica lo siguiente:

  "Ellos [(El software libre de código abierto)] aceptan algunas licencias que nosotros consideramos demasiado restrictivas, y hay licencias de software libre que ellos no han aceptado. Sin embargo, las diferencias entre lo que abarcan ambas categorías son pocas: casi todo el software libre es de código abierto, y casi todo el software de código abierto es libre." [6]

- Software de dominio público

  Oficialmente el software que no se encuentra protegido por derechos de autor, se le considera Software de dominio público.

  Dado que éste tipo de software no está protegido por el copyleft, puede ocurrir que existan copias o versiones que no sean completamente libres.

  Esto conlleva a que en algunos casos a que un programa sea de dominio público sin que se pueda tener acceso al código fuente del mismo. Y ya que el acceso al código fuente es un requisito para considerarse a un software como libre, la Free Software Foundation[7] no lo considera como software libre propiamente dicho.

  Su postura oficial es: "En algunos casos, un programa ejecutable puede ser de dominio público sin que su código fuente esté disponible. Este software no es software libre, porque para que lo sea es preciso que el código fuente sea accesible. Por su parte, la mayoría del software libre no es software de

[5] http://www.gnu.org/philosophy/categories.es.html, 13-06-2010, 08:10
[6] http://www.gnu.org/philosophy/categories.es.html, 13-06-2010, 08:10
[7] Abordado en el punto 1.3

dominio público; está protegido por derechos de autor, y los poseedores de estos han dado permiso legal para que cualquiera lo empleé libremente usando una licencia de software libre." [8]

Debido a la confusión que genera la frase "Dominio público" con el término "Libre" o "Gratis" no debería usarse como sinónimo entre ambas, ya que "Dominio público" refiere más bien a "Sin derechos de autor".

- Software libre protegido con copyleft

  El copyleft protege al software impidiendo que todas las copias o modificaciones del mismo se les agregue alguna restricción o requisitos adicionales, además obliga a que el código fuente de las modificaciones sea público.

- Software libre no protegido con copyleft

  Al contrario de la anterior categoría, en ésta si se permite que se distribuya y / o modifique un software agregándole restricciones o requisitos adicionales.

  Por ello puede suceder que un programa libre al no tener copyleft, tenga versiones que no sean del todo libres.

- Software de GNU

  Todo software liberado bajo es auspicio del Proyecto GNU[9] se le denomina "Software de GNU", así mismo a los programas y paquetes que también sean cobijados por el Proyecto GNU tienen un nombre similar, y aunque no el 100% de ese software esta bajo la categoría de copyleft, si es software libre.

1.1.2 Categorías del software no libre

- Software semilibre

[8] http://www.gnu.org/philosophy/categories.es.html, 10-04-2010, 03:10
[9] http://www.gnu.org/philosophy/categories.es.html#TOCGNUsoftware 10-04-2010, 03:10

Ésta categoría se encuentra a la mitad de la definición del software libre y software no libre. Aunque no se tiene acceso al código fuente (característica del software no libre) si permite que se use, copie, distribuya y / o se modifique, siempre y cuando sea sin propósitos lucrativos.

- Software privativo

  Éste tipo de software tiene restringido tanto su uso, redistribución y su modificación, para ello se debe estar autorizado. Cuando se infringen tales términos se incurre en la llamada "piratería".

- Freeware

  Éste término es muy utilizado en sitios de descarga en Internet, aunque no tiene una definición muy clara[10]. Dicho termino se usa comúnmente para paquetes que son distribuibles pero no modificables, por ello su código fuente no está disponible lo que lo convierte en software no libre, lo cual hace un poco confuso su nombre.

- Shareware

  Ésta categoría es también muy común en sitios de descargas, y se le acuña al software que se le tiene permitido redistribuir copias del mismo pero que para usarlo debe pagarse una licencia.

  El sitio oficial de GNU da dos razones por las cuales shareware no se le puede considerar como software libre o semilibre:

  1. Para la mayoría del shareware, el código fuente no está disponible; por lo tanto, no se puede modificar el programa de alguna manera.

  2. No se puede hacer una copia de shareware e instalarla sin pagar un cargo por licencia, incluso en el caso de individuos que lo utilicen para actividades sin ánimo de lucro[11]

[10] http://www.gnu.org/philosophy/categories.es.html#TOCfreeware, 07-05-2010, 1:09
[11] http://www.gnu.org/philosophy/categories.es.html#TOCshareware, 09-05-2010, 1:52

- Software privado

  En opinión de Richard Stallman “El software privado, o a medida, es software desarrollado para un usuario (generalmente una organización o una compañía). Este usuario lo tiene en su poder y lo utiliza, y no lo libera al público ni como código fuente ni como binario”. [12]

  Ejemplos de ésta categoría de software los podemos ver en empresas que desarrollan software para su uso exclusivamente interno

- Software comercial

  En contraste con la anterior definición, el software comercial se crea con la finalidad de obtener dinero de su utilización, y aunque la mayoría de éste tipo de software es privativo, existe también software libre comercial como software libre no comercial.

1.2 Las licencias y patentes

Como una forma de evitar que el software libre se modificara y se convirtiese en privativo, surge el *copyleft*, el cual utiliza la ley del *copyright* pero en un sentido contrario.

Con ello se autoriza tanto la ejecución del mismo, así como su copiado, su modificación y con esto, su distribución, dando así todas las libertades a un programa y evitar que una persona o empresa impida la distribución de la misma, o que al distribuirse se le sean añadidos restricciones que entren en conflicto o nulifiquen las libertades originales del mismo programa o software.

Para esto último existe un tipo de *copyleft* denominado GPL (General Public License) el cual es utilizado para proteger todo el software del proyecto GNU. Existe otro llamado LGPL (Library General Public License), es cual fue utilizado

[12] http://www.gnu.org/philosophy/categories.html#TOCPrivateSoftware, 09-05-2010, 2:56

para la librería C de GNU y permite que ésta sea utilizada con software propietario.

1.2.1 Licencias para software

En algunas ocasiones, algún software tiene algunos requerimientos y/ o necesidades que otros, la Free Software Foundation ha trabajado en crear licencias para las necesidades de software, por lo regular, el nuevo software puede tener alguna de las licencias ya creadas, aunque en algunos casos puede crearse licencias especiales.

Las licencias ya creadas son expuestas a continuación

- **Licencias de software libre compatibles con la GPL**
    - La Licencia Pública General de GNU (GNU GPL)
    - La Licencia Pública General Reducida de GNU (GNU LGPL)
    - La Licencia de Guile
    - La licencia de las unidades de ejecución del compilador de ADA de GNU
    - La Licencia X11
    - La Licencia Expat
    - La Licencia de Copyright ML Estándar de New Jersey
    - El dominio público
    - La Licencia General e Cryptix
    - La licencia SBD modificada
    - La licencia de Zlib
    - La licencia de la “Biblioteca de Funciones Estándard” de iMatix
    - El aviso y la licencia de software del W3C
    - La licencia de la base ed datos de Berkeley
    - La licencia de OpenLDAP
    - La licencia de Python

- La licencia de Perl
- La licencia Artística con aclaraciones
- La licencia artística 2.0
- La licencia pública de Zope v2.0
- La licencia de código abierto de Intel
- La licencia de javascript de Netscape
- La licencia de eCos v2.0
- La licencia del Forum Eiffel v2
- La licencia de vim v6.1 o posterior

- **Licencias de software libre incompatibles con la PGL**
  - La Licencia Pública General de Affero
  - La licencia Pública Arphic
  - La licencia BSD original
  - Commom Development and Distribution License (CDDL)
  - La licencia de OpenSSL
  - La licencia libre académica (AFL)
  - La licencia de software abierto v1.0
  - La licencia de Apache v1.0 y v1.1
  - La licencia de Apache v2.0
  - La Licencia Pública de Zope versión 1
  - La licencia de xinetd
  - La Licencia de Python 1.6b1 y versiones posteriores hasta la 2.0, más la 2.1.
  - La antigua licencia de OpenLDAP, versión 2.3
  - La Licencia Pública de IBM, versión 1.0
  - Licencia Pública Común versión 1.0
  - La licencia de Phorum, versión 2.0
  - La Licencia Pública del Proyecto LaTeX (LPPL)
  - La Licencia Pública de Mozilla (MPL).

- La Licencia de Código Fuente Abierto de Netizen (NOSL), versión 1.0.
- La Licencia Pública de Interbase (IPL), versión 1.0.
- La Licencia Pública de Sun.
- La Licencia de Código Fuente Abierto de Nokia
- La Licencia Pública de Netscape (NPL)
- La Licencia de Código Fuente Abierto de Jabber, versión 1.0
- La Licencia para el Código Fuente según los Estándares Industriales de Sun, versión 1.0
- La Licencia Pública Q (QPL), versión 1.0
- La licencia FreeType
- La Licencia de PHP, versión 3.0.
- La Licencia Zend, versión 2.0.
- La Licencia de Plan 9 (junio de 2003)
- La Licencia de Fuente Pública de Apple (APSL), versión 2

- **Licencias de software no libre**

Todas las licencias que son incompatibles con la GNU GPL son automáticamente no libres. Y ya que cada compañía tiene su propio tipo de licencia se mostrarán las licencias más comunes.

- La Licencia Artística (Original).
- La Licencia de Fuente Pública de Apple (APSL), versión 1.x
- La Licencia B de Software Libre de SGI, versión 1.1
- La Licencia "Sun Community Source".
- La antigua licencia de Plan 9
- La Licencia Pública Abierta
- La Licencia Pública de la Universidad de Utah
- La Licencia Pública eCos, versión 1.1
- La Licencia de Código Fuente de Sun Solaris (Foundation Release), versión 1.1

- La Licencia de YaST
- Las licencias de Daniel Bernstein's
- La "Licencia Pública Libre de Aladdin"
- La licencia de Scilab
- La Licencia Pública de AT&T
- La Licencia "Jahia Community Source"
- La licencia de ksh93
- La Licencia de Código Fuente Compartido de Microsoft
- La Licencia "Hacktivismo Enhanced-Source Software License Agreement" (HESSLA).
- La licencia Squeak.

1.2.2 Licencias para documentación

- **Licencias de documentación libre**
  - La Licencia de Documentación Libre de GNU.
  - La Licencia de Documentación de FreeBSD
  - La Licencia de Documentación Común de Apple, versión 1.0
  - La Licencia de Publicación Abierta, versión 1.0.

- **Licencias de documentación no libre**
  - La Licencia de Contenido Abierto, versión 1.0.
  - La Licencia de Directorio Abierto (también conocida como Licencia dmoz.org).

1.2.3 Licencias para trabajos

- La Licencia Pública General de GNU.
- La Licencia de Documentación Libre de GNU.
- La Licencia de la Ciencia del Diseño (DSL)

- La Licencia Libre del Arte

1.2.4 Patentes

Las patentes en ocasiones puede confundirse con el copyright. Para tener una idea más clara se muestran diferencias entre ambas.

El copyright no protege sino más bien regula. Las patentes protegen las ideas.

El copyright se aplica automáticamente mientras que las patentes son publicadas por una oficina de patentes por solicitud.

Las patentes cuestan mucho y la solicitud tarda años en ser estudiada**.**

El copyright dura durante mucho tiempo, hasta 150 años en algunos casos. Las patentes duran 20 años. copyright protege la copia, la patente es un monopolio sobre el uso de alguna idea.

1.3 El proyecto GNU y la FSF

Mientras Richard Stallman trabajaba en el Laboratorio de Inteligencia Artificial del MIT (Instituto de Tecnología de Massachusetts) en 1971, pertenecía a una comunidad que compartía software que tenía mucho tiempo de existir.

Era ahí donde se usaba un sistema operativo de tiempo compartido llamado ITS (sistema de tiempo compartido incompatible), diseñado y escrito en ensamblador para el PDP (Procesador de Datos Programados).

En ese entonces el software compartido no se le llamaba software libre, pues ese término no existía. En síntesis se permitía que personas de otras universidades o empresas portaran y usaran un programa.

Todo esto fue permitido durante años hasta que a comienzos de los 80 fue descontinuado la PDP lo que conllevó a que casi todos los programas se volvieran obsoletos. Y fue así que el laboratorio adquirió un nuevo PDP-10 en 1982 con la desventaja de que era un sistema de uso compartido no libre.

Esto contravenía los principios éticos de la comunidad Hacker (Compartir el conocimiento) de la cual Stallman formaba parte, lo que hizo que buscara una forma de crear software libre sin restricciones.

1.3.1 GNU

Con todos los inconvenientes de la adquisición del PDP-10, Stallman decidió crear un sistema compatible con Unix para que fuera portable y sus usuarios pudieran migrar, llamándolo GNU, el cual es un acrónimo recursivo de “Gnu’s Not Unix” (GNU No es Unix).
Y ya que un sistema operativo no sólo es el núcleo, se previó agregar aplicaciones diversas como procesadores de comandos, compiladores, depuradores, editores de texto, ensambladores, intérpretes y programas de correo entre otros.

Siendo que el desarrollo de un sistema complejo es un proyecto de gran envergadura. Stallman decidió adaptar y usar las fragmentos existentes de software libre. Así, TeX sería el principal compaginador de texto, y se utilizaría el sistema X Window como sistema de ventanas para evitar escribir otro.

Esto da como resultado que el proyecto GNU comience en 1984 y que no todo el software sea GNU. Se incluyen programas hechos por terceros pero éstos son también, libres.

1.3.2 FSF

El proyecto GNU fue creciendo a tal grado que, en octubre de 1985 se creó la FSF (Free Software Foundation). Una organización sin ánimos de lucro enfocada al desarrollo del software libre.

Y contrario a lo que se piensa, se puede obtener dinero con el software libre; la propia FSF subsiste gracias a los ingresos obtenidos de la venta de copias de programas, código fuente, manuales, todo esto respetando la licencia y permitiendo su libre uso y modificación.

Una de las metas de la organización era crear un Sistema Operativo, también libre, por lo cual se diseñaron 2 paquetes muy importantes: la librería C y la Shell.

La librería C, es la que permite que todos los programas se comuniquen con el sistema operativo, fue desarrollado por Roland McGrath, mientras que el Shell llamado BASH fue creado por Brian Fox.

1.4 Mach y Hurd

Mach y Hurd son los sistemas operativos del proyecto GNU. En términos generales, un sistema operativo es el componente de software encargado de la interrelación de los programas con los componentes del equipo y del equipo con el usuario.

Una de las principales funciones del sistema operativo es la de servir de máquina virtual, ya que éste interactúa con el hardware, permitiendo al programador utilizar una serie de llamadas al sistema.

La otra gran función es la de controlar el acceso y la utilización de los recursos del sistema.

Además, también controla quien utiliza esos recursos y en qué momento, con esto se garantiza que todas las peticiones a los dispositivos se completen con éxito.

Un tipo de sistemas operativos reciente es el denominado Cliente/Servidor, que se conoce también como Microkernel, y puede ejecutarse en la mayoría de las computadoras. Éste sistema sirve para una gran variedad de aplicaciones, por ello se llama de propósito general.

El núcleo del sistema establece la comunicación entre los clientes y los servidores. Éstos deben tener mecanismos de protección y seguridad que serán filtrados por el núcleo del sistema que controla a su vez el hardware.

Uno de los precursores de éste tipo de sistemas operativos fue Mach y actualmente el proyecto GNU desarrolla el kernel Hurd, que ya se encuentra disponible.

1.4.1 Mach

GNU Mach fue desarrollado en 1990 por Carnegie Mellon University, en la Universidad de Utah, y es el micronúcleo del sistema GNU. Un micronúcleo proporciona solamente una funcionalidad limitada, como anteriormente se indicó; el mínimo nivel de abstracción que se precisa por encima del hardware para ejecutar el resto del sistema operativo en el espacio de usuario.

Ventajas de Mach

El sitio oficial[13] de Hurd enlista las diferentes ventajas que tiene Mach, mismas que se muestran a continuación:

- Es software libre. Cualquiera puede utilizarlo, modificarlo, y redistribuirlo bajo los términos recogidos en la Licencia Pública General de GNU (GPL).

[13] http://mirror.gnu.cype.es/software/hurd/gnumach.es.html 26-05-2010, 20:27

- Construido para perdurar. Como micronúcleo que es, GNU Mach no implementa muchas de las características que se pueden encontrar en un sistema operativo, sino que solamente dispone del mínimo indispensable que se precisa para implementar encima de él un sistema operativo completo. Esto significa que el mantenimiento de la mayoría del código del sistema operativo es ajeno a GNU Mach, y mientras que dicho código puede rediseñarse por completo, el código del micronúcleo, en comparación, permanecerá estable.
- Es modular. Mach es muy adecuado para SMP y técnicas de clústers de red. El soporte multihilo se implementa a nivel de núcleo, y el propio núcleo saca partido de esa característica. La transparencia de red a nivel de IPC hace que los recursos del sistema estén disponibles en todas y cada una de las máquinas.
- Existe. El micronúcleo Mach es software real y que funciona actualmente. No se trata de un proyecto de investigación o de una propuesta. No se tiene que esperar en absoluto para comenzar a utilizarlo o a desarrollarlo. Mach se ha utilizado en numerosos sistemas operativos en el pasado, normalmente como base para un único servidor UNIX. En el sistema GNU, Mach es la base de un sistema operativo multiservidor funcional, el Hurd.

GNU Mach es usado como micronúcleo en el sistema GNU/Hurd. Además es compatible con otras distribuciones de Mach.

1.4.2 Hurd

El Hurd de GNU es el proyecto de reemplazo de GNU para el núcleo (`kernel') de Unix. En un principio se iba a denominar Alix.

El proyecto GNU define a Hurd como “una colección de servidores que se ejecutan en el micronúcleo Mach para implementar archivos de sistema,

protocolos de red, control de acceso a archivos y otras características implementadas en el núcleo de Unix o núcleos similares (como Linux)"[14].

Hurd significa Hird of Unix-Replacing Daemon, y Hird a su vez significa Hurd of Interfaces Representings Depth, es decir, sus nombres se forman con un doble acrónimo recursivo.

Ventajas del Hurd

El sitio oficial [15] de Hurd enlista las diferentes ventajas que tiene Hurd, mismas que se muestran a continuación:

- Es software libre. Cualquiera puede usarlo, modificarlo, y redistribuirlo bajo los términos de la GNU General Public License (GPL).
- Es compatible. EL Hurd provee un entorno de usuario y una programación amigable. Para todas las intenciones y los propósitos, es un núcleo moderno del tipo Unix. Usa la GNU C Library, cuyo desarrollo está cercano a estándares como ANSI/ISO, BSD, POSIX, Single Unix, SVID, y X/Open.
- Está creado para sobrevivir. Tiene una estructura orientada a objetos que le permite evolucionar sin comprometer su diseño. Esta estructura ayuda al Hurd para que sobrepase un rediseño total y modificaciones sin tener que ser completamente reescrito.

- Es escalable. La implementación es multitarea para que así se ejecute eficientemente en procesadores simples y multiprocesadores simétricos. Las interfaces del Hurd están diseñadas para permitir clústeres de red transparentes (colectivos), aunque esta característica no ha sido implementada todavía.

[14] http://gnu.gds.tuwien.ac.at/software/hurd/hurd.es.html 21-05-2010, 21:28
[15] http://gnu.gds.tuwien.ac.at/software/hurd/hurd.es.html 21-05-2010. 21:28

- Es extensible. El Hurd es una plataforma atractiva para aprender sobre el núcleo o para implementar nuevas ideas en la tecnología del mismo. Cada parte del sistema está diseñada para ser modificada y extendida.
- Es estable. Es posible desarrollar y probar nuevos componentes del núcleo Hurd sin reiniciar la máquina (ni siquiera accidentalmente). La ejecución de los componentes del núcleo no interfiere con otros usuarios, y no se requiere de algún privilegio especial del sistema. El mecanismo para las extensiones del núcleo es seguro por el diseño: es imposible imponer cambios para los otros usuarios a menos que se tenga su autorización o que sea el administrador del sistema.
- Existe. El Hurd es software real que funciona. No es un proyecto de investigación o un propósito. No se tiene que esperar para empezar a usarlo y a desarrollarlo.

El Hurd, al igual que GNU Mach, la GNU C Library y otros programas GNU y non-GNU en el sistema GNU, proveen un sistema operativo completo. Pero aún no se encuentra listo para su uso en producción. Es por ello que se ha usado en su lugar el kernel Linux.

1.5 De Unix a Linux

En los años 40 y 50 las computadoras personales, ya que al usarla se apoderaba de ella durante todo ese tiempo. Las maquinas eran enormes, pero solo una persona podía usarlas en un momento dado.

Años después (En la década de los sesentas) se utilizaban mucho los sistemas por lotes, y trabajaba con tarjetas perforadas. Y Una vez que se reunía los trabajos, el operador los introducía en la máquina como un solo lote y normalmente tardaba una hora o más para devolver las salidas.

En tales circunstancias, la depuración era muy tardada, pues incluso por una sola coma fuera de lugar hacía que varias horas del programador fuesen desperdiciadas. Para eliminar tal problema, se inventó el tiempo compartido en Dartmouth College y en MIT.

Éste sistema sólo ejecutaba BASIC y tuvo éxito antes de desaparecer. El sistema del MIT, CTSS, era de uso general y tuvo un éxito enorme en la comunidad científica. En poco tiempo, los investigadores del MIT se unieron a Bell Labs y General Electric para diseñar un sistema de segunda generación, el cual fue llamado MULTICS.

A pesar que Bell Labs fue socio fundador del proyecto, se apartó poco después de él. A lo cual, uno de los investigadores, llamado Ken Thompson, decidió escribir un MULTICS austero en código ensamblador en una minicomputadora PDP-7 que ya nadie usaba.

1.5.1 Unix

El sistema MULTICS funcionó bien por lo cual Brian Kemighan, lo llamó UNICS. Aunque después sufrió un cambio de ortografía a UNIX. Poco tiempo después se unió Dennis Ritchie, y junto con él, todo su departamento.

Thompson decidió rescribir UNIX, y utilizó un lenguaje de alto nivel que fue creado por él, al que llamó B, y BCPL simplificado, y su vez éste era una simplificación de CPL). Pero debido a carencias de B, el intento fue un fracaso. Por lo que Ritchie diseñó el lenguaje C, y juntos, Thompson y Ritchie rescribieron UNIX en C.

Todo esto conllevó a que años más tarde, en la década de los 80, se usaran dos versiones diferentes del sistema, y por un lado, ambas incompatibles con UNIX, que serían tanto la 4.3BSD y System V Release 3. Aunque se hicieron varios intentos por estandarizarlo que no tuvieron éxito, la IEEE standars Borad lo logró, el resultado fue el sistema llamado POSIX, que significa “Sistema Operativo

Portátil" (Portable Operating System); y el sufijo IX es para que tuviera un parecido a la palabra UNIX.

Poco después entra MINIX, que sería una labor colectiva, a lo que muchas personas solicitaban funcionalidades, pero el autor por lo regular se negaba a realizar cambios, pues su objetivo era que el sistema fuera muy pequeño para que los estudiantes pudieran verlo en un semestre en la Universidad.

1.5.2 Linux

La negativa a añadir funcionalidades pedidas por los usuarios hizo que un joven estudiante finlandés, llamado Linux Torvalds, decidiera crear otro clon de UNIX, al que llamó Linux, el cual tendría funciones de las que MINIX no tenía.

El 3 de julio de 1991 acontecieron las primeras discusiones sobre Linux en un grupo de noticias comp.os.minix. La primera versión (0.01), se liberó el 25 de agosto, ni siquiera podía ejecutarse, solo incluía los principios del núcleo, y estaba escrita en ensamblador pero asumía que se tenía acceso a un sistema Minix para poder compilarlo. Se desarrolló en forma cruzada en una máquina MINIX y tomó algunas ideas de ella.

Ya por el 5 de octubre de ese año se anunció lo que fue la primera versión del código fuente (0.02). Con esta versión Linus pudo ejecutar Bash y gcc pero nada mas eso funcionaba.

En marzo de 1992 se alcanzó la versión 0.95 y por vez primera se podía ejecutar el sistema X-windows.

Para diciembre de 1993 el núcleo ya alcanzaba la versión 0.99 y la 1.0.0 llegó hasta el 14 de marzo de 1994, y fue hasta el 9 de Mayo 1996 que Tux fue propuesto como la mascota oficial del sistema Linux.

La serie 2x que es la actual ha recorrido un largo camino, el 9 de junio de 1996 fue lanzada, y la serie 2.2.x lo fue hasta 3 años después, el 25 de enero de 1999 mientras que la serie 2.4.x no fue, sino hasta el 4 de enero del 2001. Por último la actual serie (2.6x) se lanzó el 17 de diciembre del 2003, 7 años después que la primera serie de 2x.

En la ilustración 2 pueden verse las diferentes variantes de Unix desde el lanzamiento de la primera en la década de los 70 y la posición de Linux en esta historia.

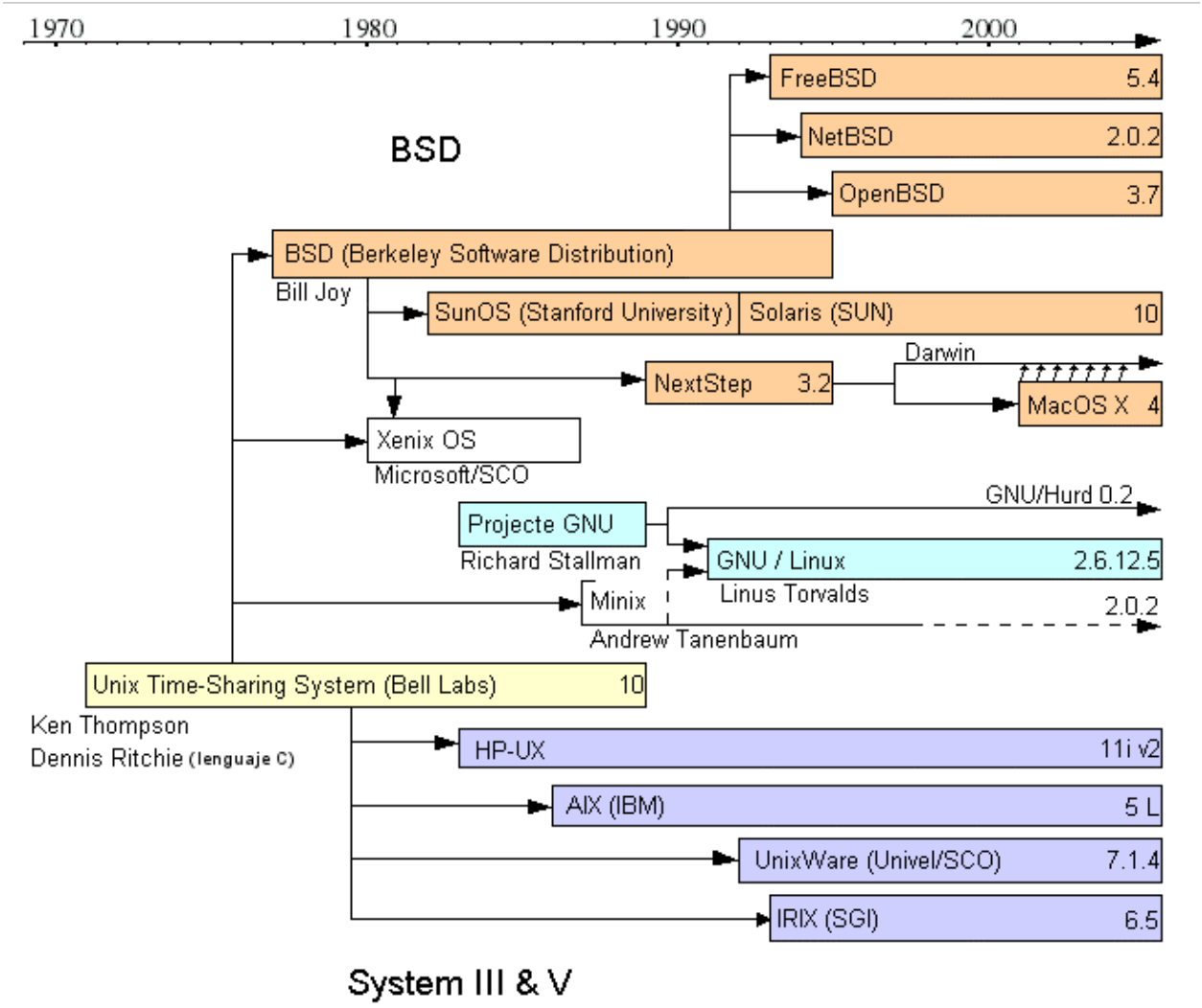


Ilustración 2: Unix y sus variantes

Características [16]

Multitarea: La palabra multitarea describe la habilidad de ejecutar varios programas al mismo tiempo. LINUX utiliza la llamada multitarea preventiva, la cual asegura que todos los programas que se están utilizando en un momento dado serán ejecutados, siendo el sistema operativo el encargado de ceder tiempo de microprocesador a cada programa.
Multiusuario: Muchos usuarios usando la misma máquina al mismo tiempo.
Multiplataforma: Las plataformas en las que en un principio se puede utilizar Linux son 386-, 486-. Pentium, Pentium Pro, Pentium II, Amiga y Atari, también existen versiones para su utilizacion en otras plataformas, como amd64, Alpha, ARM, MIPS, PowerPC y SPARC.

Multiprocesador: Soporte para sistemas con más de un procesador está disponible para Intel, AMD y SPARC.

Funciona en modo protegido 386.

Protección de la memoria entre procesos, de manera que uno de ellos no pueda colgar el sistema.

Carga de ejecutables por demanda: Linux sólo lee del disco aquellas partes de un programa que están siendo usadas actualmente.

Política de copia en escritura para la compartición de páginas entre ejecutables: esto significa que varios procesos pueden usar la misma zona de memoria para ejecutarse. Cuando alguno intenta escribir en esa memoria, la página (4Kb de

[16] http://www.linux-es.org/sobre_linux 21-05-2010, 16:48

memoria) se copia a otro lugar. Esta política de copia en escritura tiene dos beneficios: aumenta la velocidad y reduce el uso de memoria.

Memoria virtual usando paginación (sin intercambio de procesos completos) a disco: A una partición en el sistema de archivos, con la posibilidad de añadir más áreas de intercambio sobre la marcha.

La memoria se gestiona como un recurso unificado para los programas de usuario y para el cache de disco, de tal forma que toda la memoria libre puede ser usada para cache y ésta puede a su vez ser reducida cuando se ejecuten grandes programas.
Librerías compartidas de carga dinámica (DLL's) y librerías estáticas.

Se realizan volcados de estado (core dumps) para posibilitar los análisis post-mortem, permitiendo el uso de depuradores sobre los programas no sólo en ejecución sino también tras abortar éstos por cualquier motivo.

Compatible con POSIX, System V y BSD a nivel fuente.

Emulación de iBCS2, casi completamente compatible con SCO, SVR3 y SVR4 a nivel binario.

Todo el código fuente está disponible, incluyendo el núcleo completo y todos los drivers, las herramientas de desarrollo y todos los programas de usuario; además todo ello se puede distribuir libremente. Hay algunos programas comerciales que están siendo ofrecidos para Linux actualmente sin código fuente, pero todo lo que ha sido gratuito sigue siendo gratuito.

Control de tareas POSIX.

Pseudo-terminales (pty's).

Emulación de 387 en el núcleo, de tal forma que los programas no tengan que hacer su propia emulación matemática. Cualquier máquina que ejecute Linux parecerá dotada de coprocesador matemático. Por supuesto, si el ordenador ya tiene una FPU (unidad de coma flotante), esta será usada en lugar de la emulación, pudiendo incluso compilar kernel propio sin la emulación matemática y conseguir un pequeño ahorro de memoria.

Soporte para muchos teclados nacionales o adaptados y es bastante fácil añadir nuevos dinámicamente.
Consolas virtuales múltiples: varias sesiones de login a través de la consola entre las que se puede cambiar con las combinaciones adecuadas de teclas (totalmente independiente del hardware de video). Se crean dinámicamente y se puede tener hasta 64.

Soporte para varios sistemas de archivo comunes, incluyendo minix-1, Xenix y todos los sistemas de archivo típicos de System V, y tiene un avanzado sistema de archivos propio con una capacidad de hasta 4 Tb y nombres de archivos de hasta 255 caracteres de longitud.

Acceso transparente a particiones MS-DOS (o a particiones OS/2 FAT) mediante un sistema de archivos especial: no es necesario algún comando especial para usar la partición MS-DOS, ésta parece un sistema de archivos normal de Unix (excepto por algunas restricciones en los nombres de archivo, permisos, y esas cosas). Las particiones comprimidas de MS-DOS 6 no son accesibles por el momento, y no se espera que lo sean en el futuro. El soporte para VFAT, FAT32 (WNT, Windows 95/98) se encuentra soportado desde la versión 2.0 del núcleo y el NTFS de WNT desde la versión 2.2 (Este último solo en modo lectura).

Soporte en sólo lectura de HPFS-2 del OS/2 2.1

Sistema de archivos de CD-ROM que lee todos los formatos estándar de CD-ROM.

TCP/IP, incluyendo ssh, ftp, telnet, NFS, etc.

Appletalk.

Software cliente y servidor Netware.

Lan Manager / Windows Native (SMB), software cliente y servidor.
Diversos protocolos de red incluidos en el kernel: TCP, IPv4, IPv6, AX.25, X.25, IPX, DDP, Netrom, etc.

1.6 GNU/Linux

Como se menciona anteriormente, desde 1984 se venía trabajando en un Proyecto con la meta de desarrollar un sistema libre de tipo Unix, llamado GNU, y cuando se escribió el sistema Linux, el GNU estaba ya casi terminado.

GNU no era, y no es ahora, un proyecto para desarrollar paquetes específicos, más bien su propósito es desarrollar un sistema libre de tipo Unix.

A principios de los años 90 se había preparado todo el sistema salvo el núcleo. El desarrollo de este núcleo se retrasó (el GNU Hurd empezó a funcionar de manera fiable en 2001), pero no hubo que esperarlo, porque Linux ya estaba disponible.

Así cuando Linus Torvalds escribió Linux, se pudo colocarlo junto al sistema GNU para obtener un sistema libre completo: esto es entonces una versión basada en Linux del sistema GNU; es decir, el sistema GNU/Linux.

De esta unión, los principales obstáculos fueron el Hardware y las bibliotecas, ya que los fabricantes de hardware en pleno uso de su derecho de proteger las

especificaciones de sus productos, dificultan el soporte del mismo, y los programadores se ven en la necesidad de buscar soluciones para ello, como la ingeniería inversa.

Por otro lado las bibliotecas no libres afecta a todos los programas en los cuales es utilizada, limitándolos a las restricciones de sus licencias, un claro ejemplo de esto fue el GUI (Graphical User Interface) denominado Qt que fue incorporado a un escritorio que ganó gran aceptación, llamado KDE.

Y si bien los sistemas libres GNU/Linux no pueden emplear dicha librería, algunas distribuciones comerciales decidieron si hacerlo dando como resultado un sistema con más posibilidades, pero menos libertad.

A lo cual la comunidad del Software Libre reaccionó con GNOME (GNU Network Object Model Envireonment) liderado por Miguel de Icaza en 1997, apoyado por la Red Hat Software; proporcionando entre sus ventajas de ser completamente libre, la de ser compatible con varios lenguajes.

1.7 Distribuciones

Una distribución Linux o también como suele llamársele, distribución GNU/Linux, o simplemente distro, es un sistema operativo tipo UNIX (UNIX-like) que contiene software de aplicaciones.

Un sistema operativo es en términos sencillos, una serie de programas que permiten al usuario darle un uso a la computadora; además que da la posibilidad de cargar o instalar otros programas adicionales, o, si no, limitarse a usar lo que venga incluido en el sistema operativo mismo.

Pero para ello, lo primero que hace falta es un kernel o núcleo, o sea un programa de arranque, que contengan controladores pues con ellos el núcleo podrá manejar y controlar los elementos de una computadora, así como los periféricos.

El kernel de Linux es una colección de software libre (y en ocasiones software no libre) creado por personas o grupos alrededor del mundo.

Si alguna persona, grupo, o empresa, con o sin ánimo de lucro toma programas ya realizados y los distribuye en conjunción con el sistema operativo, se crea una distribución. Existen más de 300 distribuciones Linux en desarrollo activo.

En una distribución se puede hacer lo siguiente[17]:

1. Retomar el kernel del Linux más todo lo que deseé añadir que se pueda compilar bajo ese kernel (que es normalmente todo código compilable bajo un entorno UNIX);
2. Compilarlo de manera que sea instalable y ejecutable en una computadora personal;
3. Añadir utilidades adicionales, ya estén publicadas con el mismo tipo de licencia, ya lo estén con licencias diferentes (en este último caso de acuerdo con los propietarios de tales utilidades);
4. Distribuir los soportes donde estén estampadas copias de las 3 operaciones previas; la distribución se hará por cualquier medio: papel, CD-ROM, espacio web, etc;
5. Si se desea, exigir un precio por esos soportes más eventualmente otro por los servicios adicionales de consultoría y apoyo que ofrezca al comprador (o un precio único inclusivo de ambos conceptos).

El resultado es una distribución del sistema operativo GNU-Linux.

Al comienzo los usuarios de Linux debían ser expertos en UNIX, conocer las librerías y ejecutables, para poder arrancar y ejecutar Linux, además de conocer muy bien la configuración y la colocación de los archivos en el sistema.

[17] http://lp.jurid.net/linux/distribu.htm, 22-04-2010, 17:50

Así, las distribuciones de Linux parecieron poco después de que el kernel de Linux fue usado por vez primera, y una de las primeras distribuciones de Linux fue MCC Interim Linux en 1992, por Owen Le Blanc del Manchester Computing Centre (de ahí su nombre MCC), la cual no tenía un ambiente gráfico; ese mérito lo obtuvo la Universidad Texas A&M con Tamu Linux.

Al año siguiente aparece la primera distribución comercial llamada LGX, por Yggdrasil Computing, Incorporated además que fue la primera distribución basada en CD-ROM.

Softlanding Linux System (SLS), fundada por Peter MacDonald en 1992 se consideró como la primera distribución que fue usada de una manera más amplia, pero ocurrió algo parecido con MINIX, el propietario de SLS rechazó hacer reparaciones a algunas de sus fallas, por lo que Patrick Volkerding creó Slackware, la distribución más antigua y en desarrollo activo en estos días la cual fue liberada el 16 de julio de 1993, y éste último se convirtió en la distribución dominante, usada por casi todo el mundo y para el 16 de agosto de 1993, ve la luz el proyecto Debian y que a la fecha, es de las de mayor importancia

A partir de 1994 nacen 2 de las distribuciones importantes, SuSe Linux (basado en Slackware) y RedHat, las cuales han servido como base para otras distribuciones.

Dos años después nace Conectiva (de origen brasileño) y Mandrake (de origen francés) las cuales se fusionaron en el 2005 dando lugar a Mandriva que fue muy popular por su innovación mientras que Conectiva fue importante por ser la primera distribución de Linux desarrollada en Latinoamérica.

Para 1999 Corel Linux aparece, pero por problemas financieros desapareció y fue adquirida por Xandros en el 2001.

En 2002 sale a la luz Gentoo, por Daniel Robbins el cual está basado en la distribución Enoch, ese mismo año aparece también Puppy, una mini distribución, la cual sirvió como base para DamnSmall, pero basada en Debian.

Ya para 2003 se crean distribuciones muy conocidas como Knoppix, Morphix, Mepis y Kanotix, basadas en Debian, mientras que Red Hat RedHat cambia a un esquema comercial, y por ello se inicia el proyecto Fedora, la rama libre de su Red Hat.

Tan solo un año más tarde aparece lo que es una de las distribuciones con más éxito, Ubuntu, por Canonical Ltd., es cual está basada en Debian y nació por iniciativa de algunos programadores de este proyecto y del proyecto Gnome. Aunque han salido ramificaciones por el tipo de escritorio que utiliza, Kubuntu que usa KDE, Xubuntu que usa XFCE, Lubuntu con escritorio LXDE, además de Edubuntu enfocada a ambientes educativos, Ubuntu Estudio centrada en aplicaciones Multimedia, Gobuntu que usa solo Software Libre.

Capítulo II

Software libre en la educación

Capítulo 2. Software libre en la educación

Las ventajas del software libre le permiten ser utilizado en diversos ámbitos, inclusive en el educativo, el cual está siendo cubierto casi en su totalidad por software privativo.

Ya en varios países de América Latina, hay proyectos tanto de llevar linux a las entidades de gobierno, como escolares; para éste trabajo, nos enfocaremos en éste último.

2.1 Software libre en las escuelas

Alrededor del año 2000, los usuarios de Microsoft Office, y Microsoft Windows, ya no sentían la necesidad de actualizar sus productos a las nuevas versiones, pues sentían que las que ya tenían cumplían sus objetivos, por lo que Dan Kegel menciona en “The case for Linux in Universities”[18], que desde el punto de vista de Microsoft, esto era un problema porque disminuía las ganancias y respondió de las siguientes formas:

- Microsoft impulsó a sus clientes corporativos a alquilar el software en vez de comprarlo, estableciendo una fecha límite del 31 de julio de 2002 para cambiar a lo que llamaba "Licenciamiento 6". Esto aumentó los costos

[18] http://www.kegel.com/linux/edu/ 26-03-2010, 22:26
http://www.rau.edu.uy/universidad/csdi/docs/kegel.htm 26-03-2010, 22:40

anuales para la empresa promedio entre 30 y 100% y puesto que el cliente ya no es propietario del software, no puede usarlo una vez vencido el contrato de alquiler.

- Microsoft se volcó a la protección contra copias para hacer cumplir sus acuerdos de licencia. Esto significa que cosas que parecían legítimas a los usuarios, por ejemplo usar la misma copia de Windows en casa y en la oficina, ya no son posibles.

Ante esto, Jordi Adell y Lolanda Bernabé, en su escrito "Software libre en educación"[19] indica que Richard Stallman enumera[20] una lista de razones para utilizar software libre en las escuelas.

1. El software libre se puede copiar y redistribuir a precio de coste. La Administración educativa puede dotar de software a todos sus centros docentes a muy bajo precio y dedicar los recursos ahorrados a otros temas necesarios para la educación: más ordenadores, formación del profesorado, desarrollo de software libre educativo, etc. En los países menos desarrollados, el software libre puede ayudar a dotar de infraestructura tecnológica a sus escuelas y a paliar la "brecha digital" con el mundo desarrollado. Los vendedores de software privativo, que saben de la importancia de la educación para sus futuras ventas, pueden ofrecer software a muy bajo coste o gratuito a las escuelas. Pero se trata en realidad de una estrategia comercial para captar futuros clientes y para formarlo en sus productos a costa del erario público.

2. La escuela ha de enseñar a los estudiantes valores y estilos de vida que beneficien a toda la sociedad. La escuela ha de promover el uso de software libre por la misma razón que promueve el reciclaje: porque nos

[19] http://elbonia.cent.uji.es/jordi/wp-content/uploads/docs/Software_libre_en_educacion_v2.pdf, 13-05-2010, 16:10

[20] ADELL, Jordi & BERNABÉ, Y. Software libre en educación, Madrid, McGraw-Hill P. 17

beneficia a todos. Si los estudiantes usan el software libre y aprenden que es mejor que el privativo, cuando sean adultos seguirán usando el software libre. Eso permitirá a la sociedad liberarse de los abusos y del control de las multinacionales que controlan el software privativo.

3. El software libre favorece que los estudiantes aprendan cómo funcionan los ordenadores y el propio software. Los futuros programadores se inician en la programación durante la adolescencia. Es una etapa clave en la que necesitan buenos modelos y ejemplos para modificar, copiar y "jugar" con ellos. Necesitan desafíos. El software libre, al permitir el acceso al código fuente del programa, les facilita enormemente el aprendizaje. El software privativo es una "caja negra" que no aporta nada para satisfacer su curiosidad y sus ansias de saber. El mensaje que les envía el software privativo es "el conocimiento es una mercancía, lo que quieres saber es un secreto comercial, aprender está prohibido por la ley". El software privativo mantiene a la gente alejada del conocimiento, sacraliza la tecnología y contribuye interesadamente a la ignorancia tecnológica que tan buenos resultados económicos les proporciona a las empresas que lo comercializan.
4. Pero, aunque muchos adolescentes no sientan curiosidad por cómo están hechos los programas de ordenador, hay valores generales que persigue la educación que están en claro conflicto con el mensaje que transmite el software privativo. Las escuelas deben enseñar hechos, conceptos, principios y procedimientos, pero también valores. La misión de la escuela es enseñar a las personas a ser buenos ciudadanos, a cooperar con los demás, a ser solidarios. Esta es la base de la sociedad. En informática, cooperar significa, entre otras cosas, compartir software, poder hacer copias a todos los compañeros de clase, llevarse a casa el software que se usa en la escuela. Y todo eso, con el software privativo es un delito.

5. Enseñar a los estudiantes a usar software libre y a participar en la comunidad de usuarios / desarrolladores de software libre es una lección

cívica llevada a la práctica. También enseña a los estudiantes que el ideal es el modelo de servicio público y la solidaridad, no el modelo del beneficio a cualquier precio de los magnates.

Además también se obtienen las siguientes ventajas:

1. Coste 0 de las aplicaciones y el Sistema operativo (S.O.): El establecimiento / estado no tendrá que pagar licencias y por lo tanto se liberaran gastos extras que les permitirán invertirlos en otra área o en la compra de mejor equipamiento. El coste 0 también se refiere a todas las aplicaciones incluidas en todas las distribuciones GNU/Linux.

2. Este ahorro también se aplica a las licencias. El software libre no requiere la compra de las mismas como en otro tipo de S.O., y en efecto permite la instalación totalmente gratuita de miles de programas (y cuantas veces se quiera) que muchas veces son mejores a sus pares en otros sistemas operativos tales como Windows o Mac OS X.

3. Recuperar Hardware obsoleto: Como en los colegios no se tienen computadoras muy actualizadas, al querer utilizar una nueva versión de un S.O. privativo (Windows, Mac OS X), este le exige más. Esto obliga a gastar dinero extra o directamente a tirar el “viejo” equipo. Para recuperar este tipo de computadoras existen distribuciones Linux especializadas en ello.

4. Los Alumnos podrían disponer de CD’s o DVD’s cargados con el software que quieran o necesiten, sin ninguna restricción. En algunas universidades se necesita utilizar ciertos programas, y el compartir es algo que no existe en el mundo del software privado.

5. Al usar software libre, los educadores también deberán aprender, y esto implica que, tanto el docente como los alumnos, aprendan distintos sistemas operativos a la vez.

6. Se enseña que en la vida no solo hay un camino, siempre existen alternativas.

7. Aprender que no siempre lo más caro es lo mejor, en contraste al consumismo, que se impone en la sociedad.

8. Comprender, de verdad, el poder de la colaboración a la hora de crear cosas grandes y de interés general.

2.2 Proyecto LULA

LULA significa 'Linux de Universidades Latinoamericanas' el cual es una iniciativa no lucrativa coordinada por la Cátedra Telefónica de la Universidad de Extremadura. Tiene como misión favorecer la integración del Software Libre en la docencia universitaria y así facilitar el intercambio de material didáctico entre universidades de Latinoamérica.

Historia

Todo comenzó con la participación de las universidades que integran el Campus Virtual Latinoamericano (CAVILA). En la actualidad cuenta con la colaboración de personal docente perteneciente a otras universidades latinoamericanas.
El siguiente listado fue tomado del sitio oficial de LULA[21]

Universidades de CAVILA que participan

- Universidad de Extremadura (España)
- Universidad Federal de Santa María (Brasil)
- Universidad de Guadalajara (México)

[21] http://lula.unex.es/index.php?seccion=participantes, 19-05-2010, 17:59

- Universidad Nacional de Córdoba (Argentina)
- Universidad Nacional de Entre Ríos (Argentina)
- Universidad Nacional de La Plata (Argentina)
- Universidad de Porto (Portugal)
- Universidad de Santiago de Chile (Chile)

Personal docente perteneciente a otras universidades y centros educativos

- Universidad Piloto de Colombia (Colombia)
- Universidad Católica Boliviana San Pablo (Bolivia)
- Universidad Nacional Autónoma de México (México)
- Universidad Austral de Chile (Chile)
- Instituto Politécnico Nacional (México)
- Instituto de Capacitación para el Trabajo del Estado de Hidalgo (México)
- Universidad Panamericana (México)
- Universidad Santo Tomás (Colombia)
- Universidad Santo Tomás (Chile)
- Universidad de La Frontera (Chile)
- Universidad de El Salvador (Argentina)
- Universidad Rafael Landívar (Guatemala)
- Universidad de Murcia (España)
- Universidad Politécnica de Valencia (España)
- Universidad Nacional de Costa Rica (Costa Rica)
- Universidad Pontificia Bolivariana (Colombia)
- Universidad de Girona (España)
- Universidad Tecnológica de Honduras (Honduras)
- Universidad Nacional Micaela Bastidas de Apurímac (Perú)
- Universidad Nacional de la Patagonia San Juan Bosco (Argentina)
- Universidad Dr. José Matías Delgado (El Salvador)
- Universidad Autónoma Juan Misael Saracho (Bolivia)
- Universidad del Cauca (Colombia)
- Universidad Autónoma de Asunción (Paraguay)

- Universidad Nacional Mayor de San Marcos (Perú)
- Universidad Centroccidental Lisandro Alvarado (Venezuela)
- Universidad de Buenos Aires (Argentina)
- Universidad Nacional de Tucumán (Argentina)
- Universidad Autónoma Metropolitana Unidad Iztapalapa (México)
- Universidad de San Carlos de Guatemala (Guatemala)
- Instituto Tecnológico Superior Zacatecas Occidente (México)
- Universidad Autónoma de Santo Domingo (República Dominicana)
- Universidad Nacional de Misiones (Argentina)

Propósito

El propósito de éste proyecto es crear y mantener una distribución de Linux el cual contenga las aplicaciones educativas de software libre que sean muy usados en Latinoamérica.

Siendo los profesores los encargados de nominar programas que puedan ser parte de la distribución, mientras que la Cátedra Telefónica, es la encargada de decidir cuáles de ellas sí formarán parte.

Principales aportaciones y beneficios[22]:

- El profesorado puede indicar el software específico que requiere para la docencia diaria.
- Todo el software docente en un único soporte: Los alumnos tienen a su alcance las aplicaciones desde el primer momento.
- Intercambio de material educativo entre universidades: El profesorado puede conocer las aplicaciones que usan sus colegas de docencia.
- Integración de Software Libre en entornos educativos: Libertad de uso, independencia de distribuidor, ahorro en costes de licencia, etc.

[22] http://lula.unex.es, 09-03-2010, 14:22

- Marco de colaboración entre universidades mediante la compartición de experiencias y la posibilidad de coordinar proyectos en común.

Otros beneficios

Ahorro de Costos.

El Ahorro de dinero para las universidades públicas en cuanto a la informática es un aspecto muy positivo ya que el estado no subvenciona esta área (cuando debería hacerlo) y entonces se debe usar el dinero de la cooperadora, en caso de que exista.

Esta es una de las principales razones por las que es conveniente el uso del software libre ya que supone un gran ahorro de dinero, incluso en los países más ricos, y las escuelas andan escasas de ello. La institución tiene los mismos derechos que un usuario normal a ver, modificar, copiar y distribuir el software.

Reducción de la brecha Tecnológica

En los países subdesarrollados, las bajas condiciones económicas muchas veces restringen aspectos educativos.

Con el software libre no solo le permite a este tipo de países desarrollarse, sino que, a través de esto, permite la reducción de la brecha tecnológica existente con el resto del mundo y además consiente en avanzar en su conocimiento.

Las ventajas de poder leer el código abierto, trae consigo el interés de muchas personas en aprenderlo y esto genera personas con alto nivel de calificación para contribuir al desarrollo de un país.

Postura de la UNESCO

La UNESCO apoya el software libre en toda América Latina. Apoya la extensión y la diseminación del conocimiento humano, a través de su portal de software libre, brinda acceso a documentos y sitios Web que hacen referencia a este movimiento.

La utilización de aplicaciones de código abierto permite a los docentes incluir formas de aprendizaje interactivo en los temas de la currícula. Modelos, simulaciones, gráficos, multimedia, comunicación, etc. requieren de herramientas de software libre.

2.3 La Universidad Veracruzana

El propósito de éste trabajo no es precisamente el de estar en el proyecto L.U.L.A, pues aunque tiene diversas ventajas, ya antes mencionadas, la distribución resultante sería demasiado genérica.

Por lo que la idea es crear una distribución modificada para la propia Universidad Veracruzana.

Dicha Universidad se remonta a los años cuarenta. Específicamente en 1944, que fue cuando inició sus actividades con el propósito de reunir y coordinar las actividades de un grupo de escuelas que se encontraban dispersas en educación media superior, tales como escuelas oficiales artísticas, profesionales, especiales y de estudios superiores de la entidad.

Además Retoma actividades de escuelas secundarias de bachilleres tales como las escuelas de enfermeras y parteras de Orizaba, Xalapa y Veracruz, a la vez que crea facultades Jurídicas y de Bellas Artes, el Departamento de Arqueología, la Escuela Superior de Música y la radiodifusora de la Universidad XEXB.

En la siguiente década la UV inicia una etapa de conformación institucional, al fundar facultades e impartir nuevas carreras en Xalapa, y en otras ciudades, dicha etapa se extiende hasta 1968, cuando se decreta separar las enseñanzas media y media superior de la Universidad Veracruzana.

Para la década de los setenta la UV se expande y consolida con la regionalización universitaria, además se crean facultades y los primeros programas de posgrado, pero para la década de los ochenta disminuye dicha expansión mientras que se

aprueban nuevos planes y programas de estudio y desaparecen formalmente el ciclo de iniciación universitaria.

Como resultado la Universidad tiene presencia en cinco de las regiones económicas más importantes de la entidad, así como además cuenta con planteles en Xalapa, Veracruz, Boca del Río, Orizaba, Córdoba, Río Blanco, Amatlán, Nogales, Camerino Z. Mendoza, Poza Rica, Tuxpan, Minatitlán, Coatzacoalcos, y Acayucan, entre otras.

Su organización académica es integrada por una estructura de áreas académicas, facultades, programas educativos e institutos de investigación, además las direcciones generales de las áreas académicas: Artes, Ciencias Biológico-Agropecuarias, Ciencias de la Salud, Económico-Administrativa, Humanidades y Técnica, coordinan las actividades realizadas por las facultades y programas educativos.

Mientras que la Dirección General de Investigaciones coordina los planes y las actividades de los institutos y centros de investigación, y la Dirección General de Difusión Cultural opera las labores de los grupos artísticos y los programas de actividades culturales.

Según datos obtenidos del sitio oficial de la Universidad Veracruzana[23]:

En el campus Xalapa funcionan 32 facultades, 20 institutos, 6 centros de investigación, un Centro de Iniciación Musical Infantil, un Centro de Idiomas, un Departamento de Lenguas Extranjeras, tres Centros de Autoacceso, tres talleres Libres de Arte, una Escuela para Estudiantes Extranjeros, un Laboratorio de Alta Tecnología, una Unidad de Servicios de Apoyo a la Resolución Analítica, un Hospital Escuela y una Unidad de Servicios Bibliotecarios y de Información (USBI).

[23] http://www.uv.mx, 04-04-2010, 13:04

En Veracruz, 13 facultades, 4 institutos y dos centros de investigación, un Centro de Iniciación Musical Infantil, un Centro de Idiomas, dos Centros de Autoacceso, un Taller Libre de Arte y una USBI, en Orizaba-Córdoba, 8 facultades, dos centros de Idiomas, dos Centros de Autoacceso.
En Poza Rica-Tuxpan, 13 facultades, un Centro de Idiomas, dos Centro de Autoacceso, dos Talleres Libres de Arte y una USBI. Y en Coatzacoalcos–Minatitlán, 8 facultades, una Escuela de Enfermería, un Centro de Idiomas, dos Centros de Autoacceso y dos USBI.

Campus

La universidad cuenta con los siguientes campus

- Campus Xalapa
- Campus Poza Rica - Tuxpan
- Campus Coatzacoalcos-Minatitlán-Acayucan
- Campus Orizaba-Córdoba
- Campus Veracruz

DES Económico Administrativa

Como podemos darnos cuenta, la Universidad Veracruzana está en constante expansión, así que se ha enfocado el proyecto a un sólo lugar, y es precisamente a la DES Económico Administrativa, que cuenta con una también larga historia, pues desde 2 de Enero de 1976 entró en vigor su ley orgánica, mientras que para 1977 se crea la primera carrera de Contaduría y Administración, además se comienzan labores de la carrera de Administración de Empresas Turísticas en la Facultad de Veracruz.

Gracias al apoyo de los obreros del Sindicato de Obreros y Artesanos Progresistas “Rafael Moreno” que donaron el terreno y edificio, se iniciaron labores el día 7 de marzo de 1977

Desde entonces la Universidad no ha dejado de avanzar, en 1995 se implementa la carrera de Administración de Empresas, para septiembre de 1997, se comienza a impartir la carrera de Sistemas Computacionales Administrativos, mientras que a partir de julio del 2009 se le nombró DES Económico Administrativo

Capítulo III

Live USB

Capítulo 3. Live USB

Sería no muy práctico el pedirle a los alumnos de la Universidad, que en sus computadoras, instalasen un sistema operativo basado en Linux, o que éste se instalase en el centro de cómputo de la misma Universidad, pues sólo una carrera es afín a la computación.

Peo por otro lado, la creación de un Live USB, permite que los estudiantes, lo utilicen contactándolo a la computadora, sin alterar los archivos que tienen en la computadora. Además pueden optar por no utilizarlo, y seguir ocupando el sistema operativo favorito.

3.1 Lives

Un dispositivo live, puede ser arracanble sin necesidad de instalarse en una computadora, los más extendidos fueron los livesCD, o livesDVD, donde solo bastaba encender una PC, con el disco insertado para poder utilizar el sistema operativo que éste contenía.

Ventajas principales

- El usuario no puede alterar la configuración ni los programas que contenga éste disco.
- El booteo desde el disco compacto está ampliamente soportado por las computadoras.
- Son baratos.

Desventajas principales

- El usuario no puede guardar su información en el disco.
- Las netbooks no disponen de un lector de discos.
- Fácilmente rayables si no se manejan con cuidado.

Con el tiempo aparecieron las LiveUSB, que utilizaban como base, una memoria USB, o también llamada pendrive, con lo que facilitó y popularizó aún más, la utilización de sistemas operativos en estas modalidades.

Las principales ventajas que se tienen al utilizar un LiveUSB son

- El usuario puede almacenar su información, con lo cual no necesita de elementos externos.
- Son pequeñas y pueden llevarse en un bolsillo o un llavero, lo que las hace muy portables.
- Son más rápida que un CD o DVD.

Por otro lado sus principales desventajas son:

- El usuario puede alterar la configuración del sistema, lo que en algunos casos llevaría inclusive a que no pudiera arrancar el sistema operativo.
- La mayoría de las computadoras portátiles no lo arrancan automáticamente.
- Son más caros que los CDs o DVDs.

3.2 La distribución

La distribución que se utilice debe ser muy amigable con el usuario ya que la mayoría de los alumnos de la DES Económico Administrativa, no cursan carreras relacionadas directamente con la computación, por lo que en el capítulo I se puede leer, podríamos elegir entre 3 distribuciones principales.

- Slackware
- Debian
- Red hat

Todas ellas tienen sus pros y contras, pero principalmente, Debian es reconocida precisamente por ser amigable con el usuario final.

De ella se desprende la distribución Ubuntu, que goza mucha popularidad en todo el mundo, y de ésta última se desprende a su vez una distribución llamada Mint, la cual es básicamente un Ubuntu modificado, pero dirigida principalmente a los usuarios en general, además que no descuida su interfaz, por lo que es una combinación de una facilidad de uso con una vista elegante.

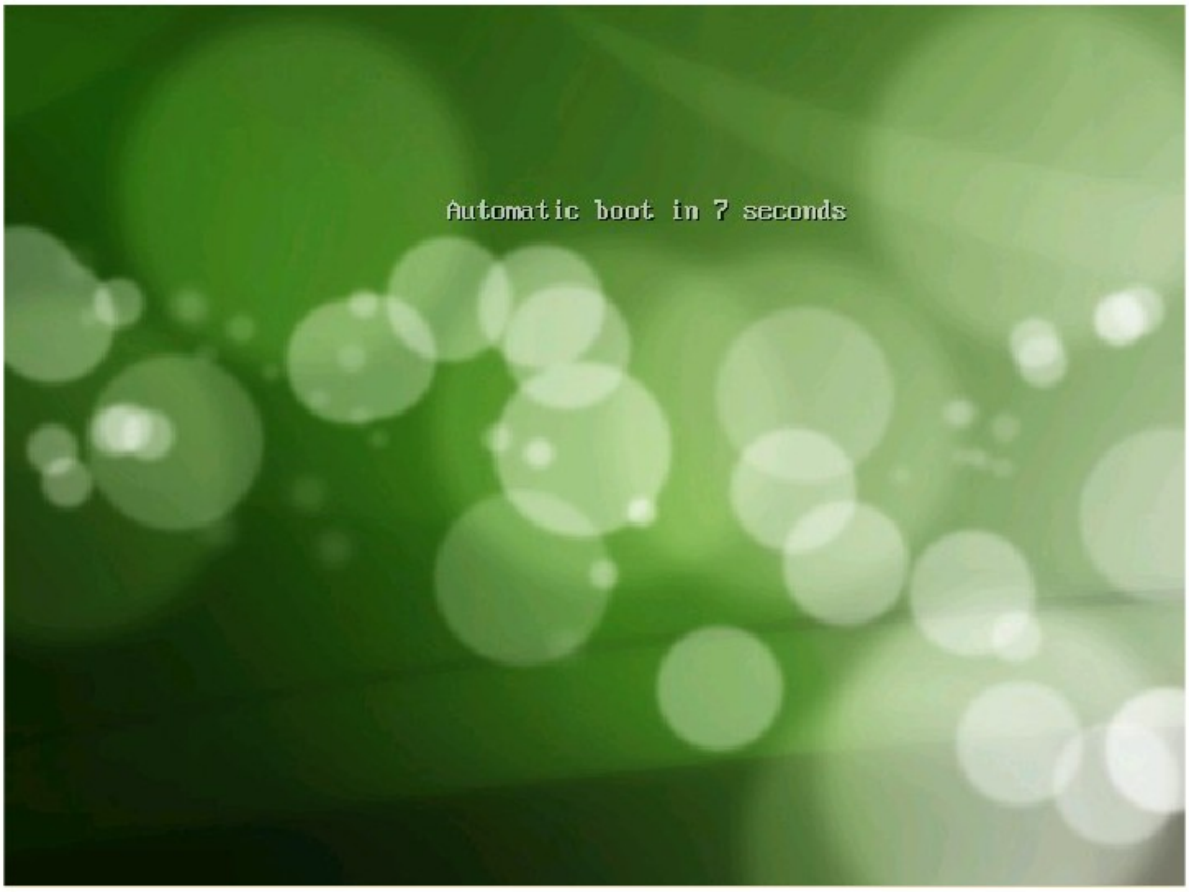
Por otro lado, también es posible crear una distribución desde las mismas fuentes de Linux, aunque sería un trabajo en el que que se necesite invertir mucho tiempo y esfuerzo, además de experiencia, estos elementos pueden usarse para configurar una distribución ya hecha, para adecuarlo a los estudiantes de la Universidad Veracruzana.

3.3 Linux Mint

Como primer paso, se debe conseguir el sistema operativo, el cual es completamente gratuito, su página principal es http://www.linuxmint.com. La versión más reciente es la 9, pero aún no está completamente traducida, por lo que es mejor utilizar la edición 8.

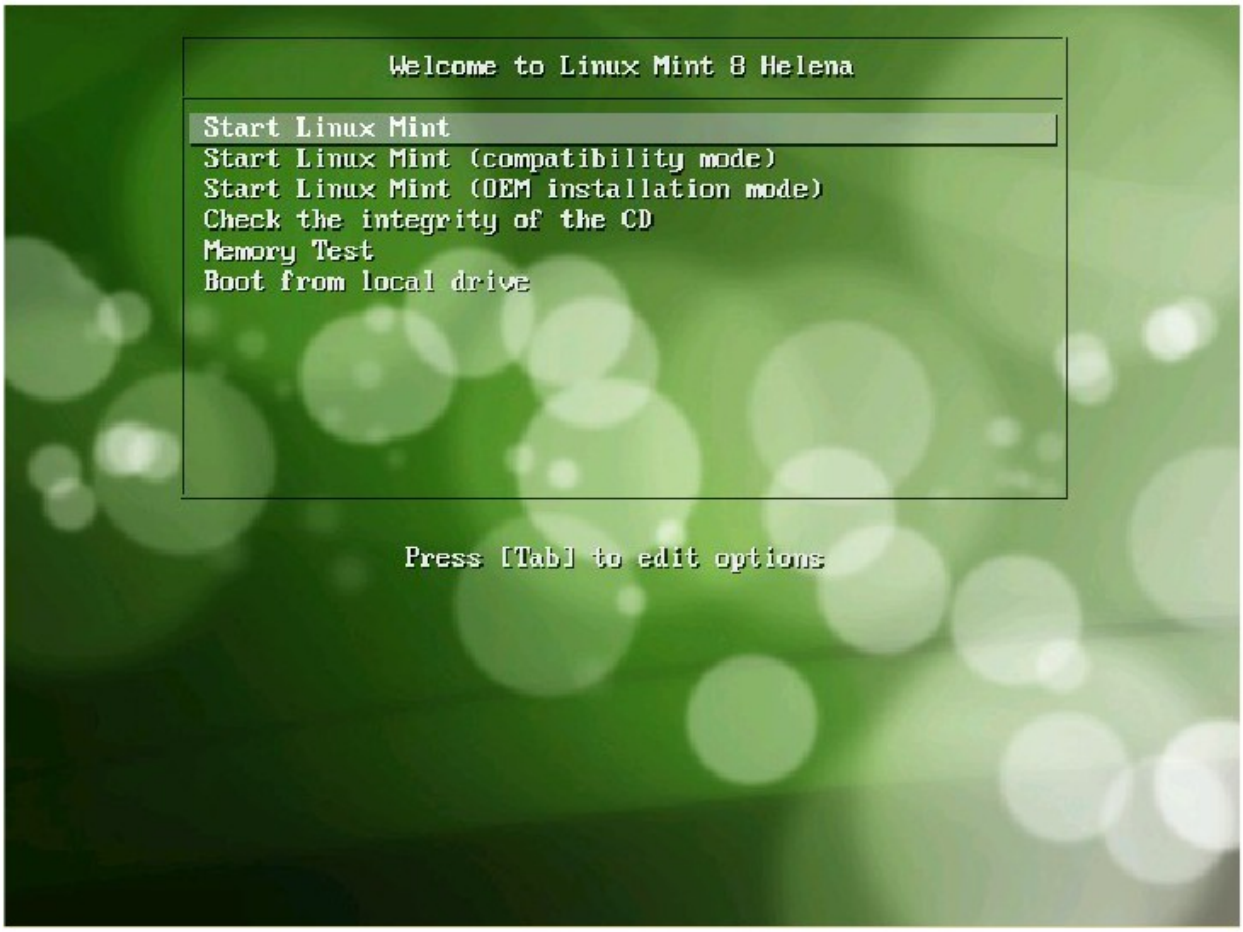
3.3.1 instalación

Una vez que se descarga el instalador, y se graba a un dvd, procedemos a arrancarlo, tal como se muestra en la pantalla 1.



Pantalla 1. Inicio

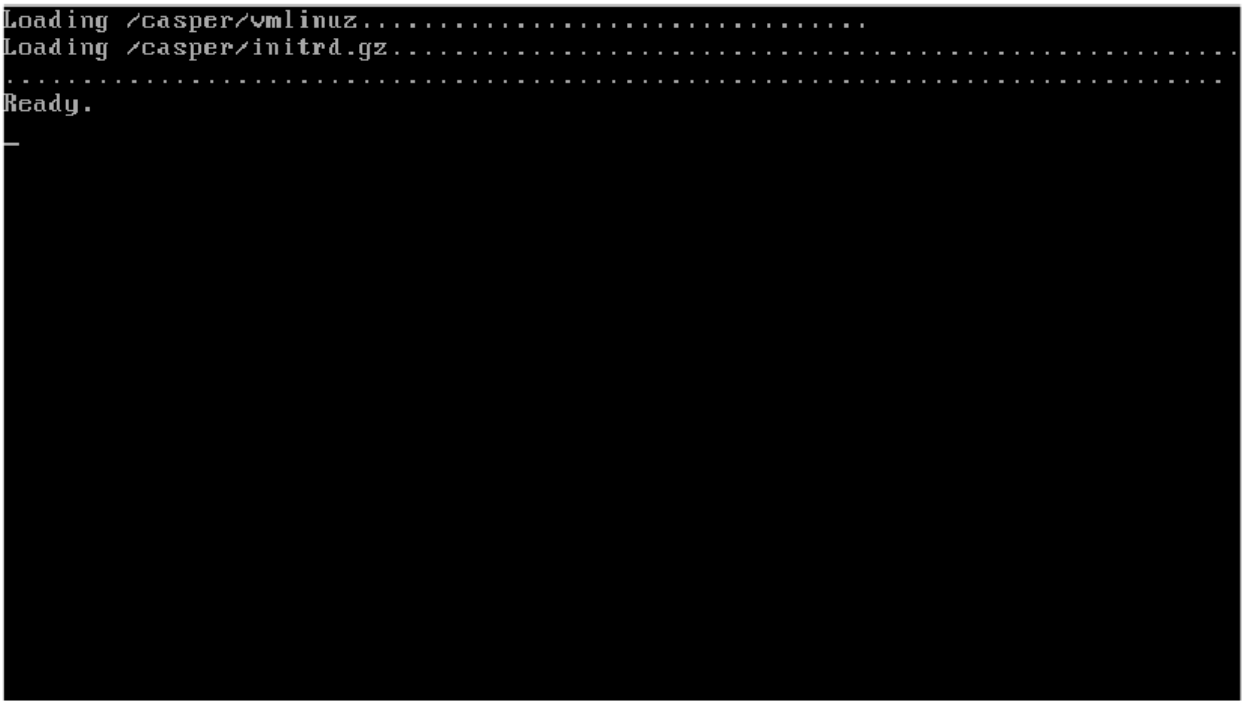
Ya sea que esperemos el tiempo que nos indica, o presionemos la tecla “enter”, nos aparecerá la siguiente pantalla.



Pantalla 2. GRUB

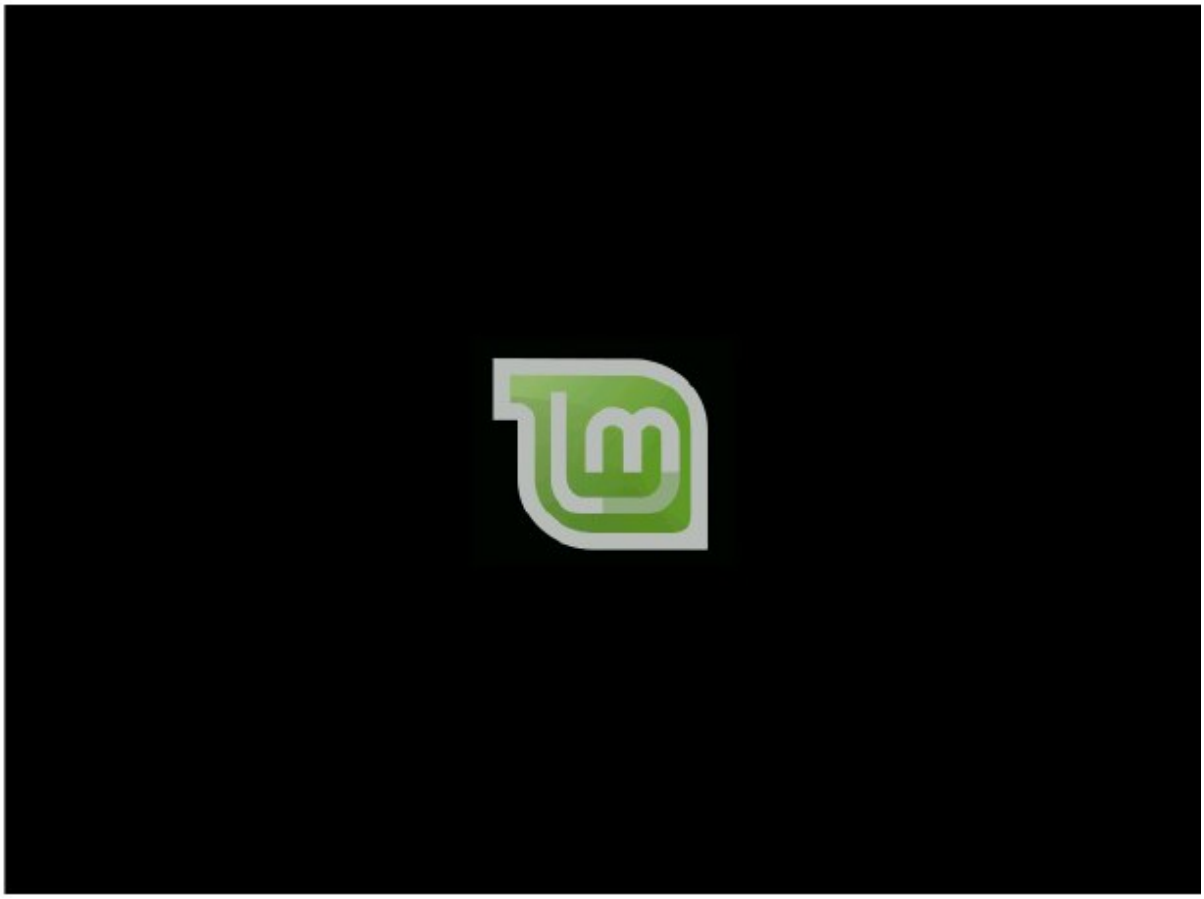
Al seleccionar la primera, Linux Mint procederá a arrancar, precisamente en modo Live. Ésta particularidad es beneficiosa, pues nos permite probarlo un poco, antes de decidir si deseamos instalarlo o no.

Mientras arranca, podemos ver la siguiente imagen.



Pantalla 3. Cargador

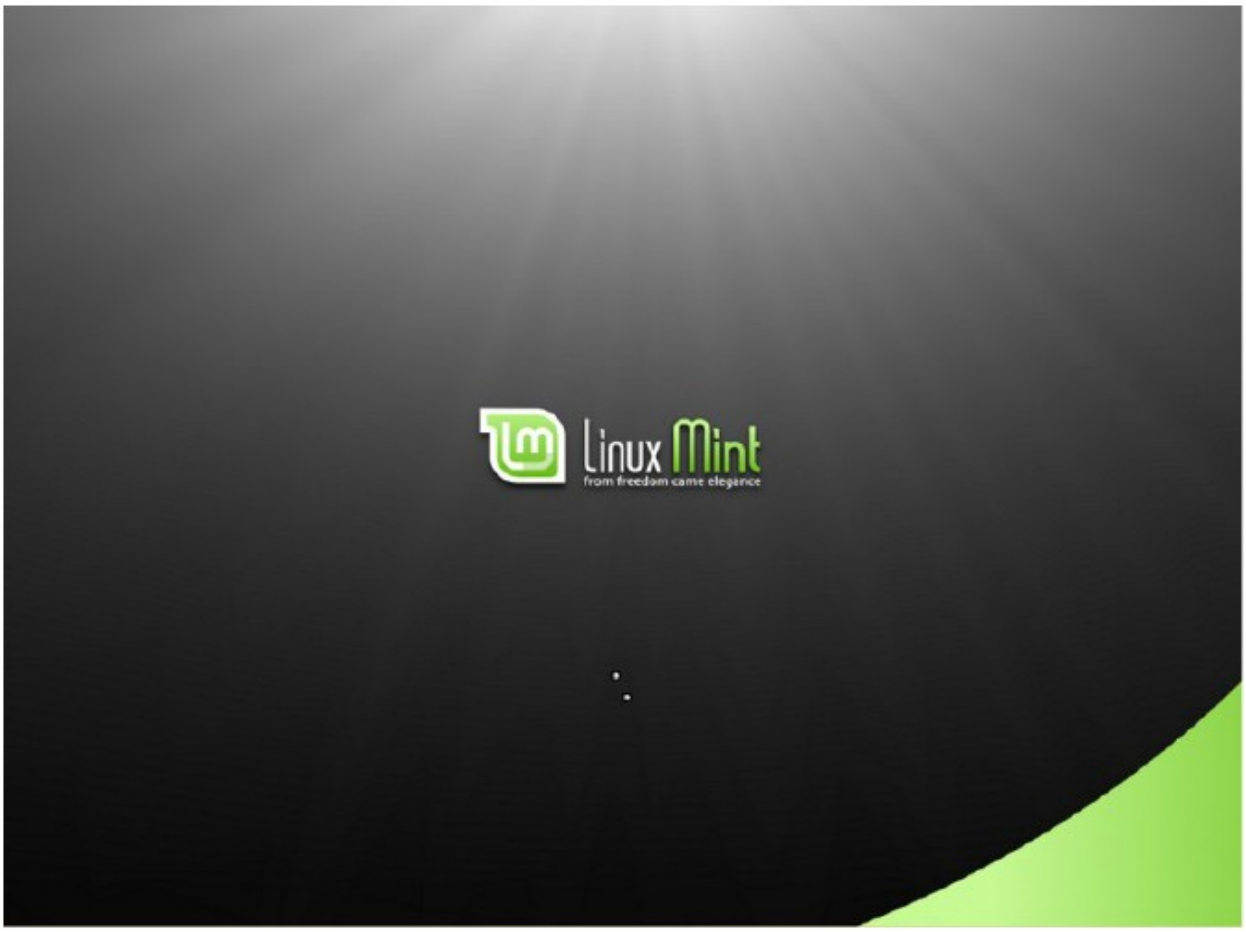
Después de unos segundos, nos muestra la pantalla de arranque, es también conocida como Splash screen, la cual muestra el logotipo de Linux Mint.



Pantalla 4. Splash

En ese momento se está iniciando el sistema operativo, en términos técnicos, ya cargó el kernel, así como los programas básicos del propio sistema. Por lo que falta arrancar el servidor X, así como el escritorio.

Todo esto se hace mientras nos muestra la siguiente pantalla.



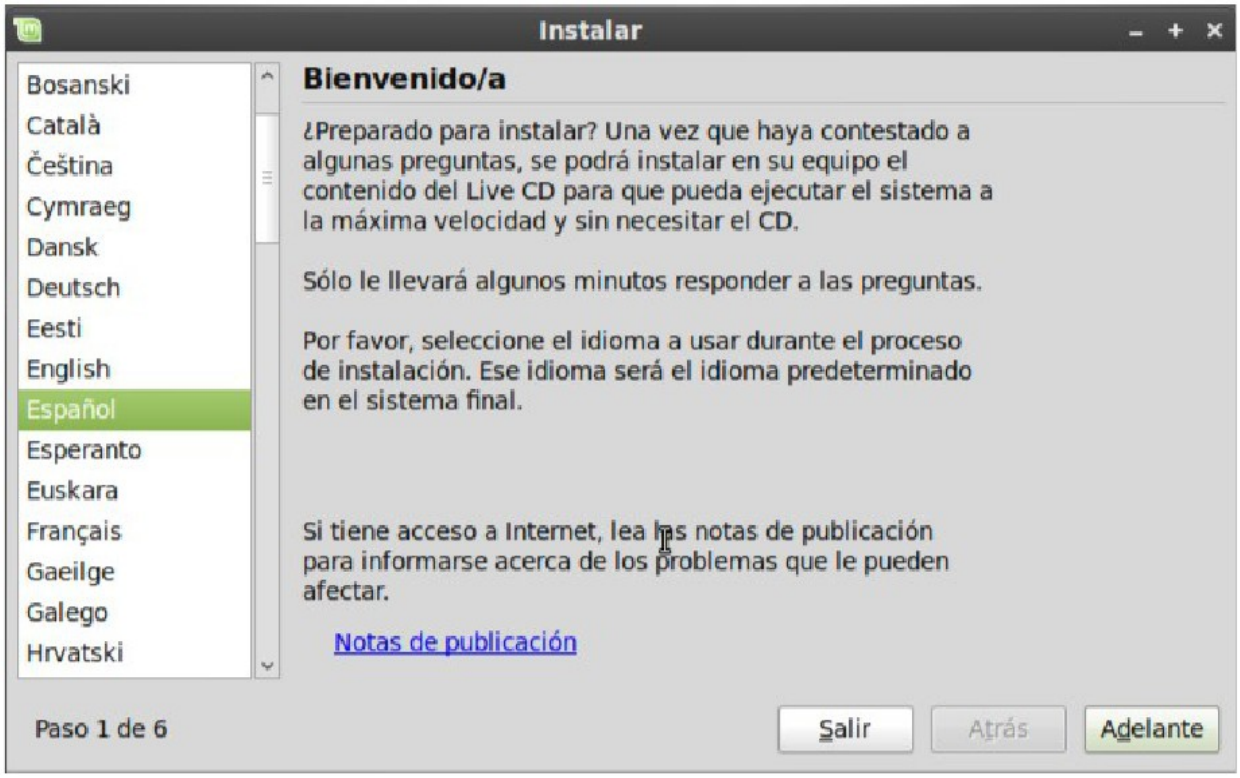
Pantalla 5. Login

Una vez que es cargado el escritorio, el cual en el caso de Linux Mint es GNOME, nos muestra la siguiente pantalla, permitiéndonos usar la distribución, o iniciar la instalación del mismo.

Pantalla 6. Escritorio

Al dar doble clic en “Install Linux Mint 8”, se inicia el proceso de instalación en la computadora, para la cual es necesario seguir una secuencia, que se explica a continuación.

En primer lugar, se elige el idioma.



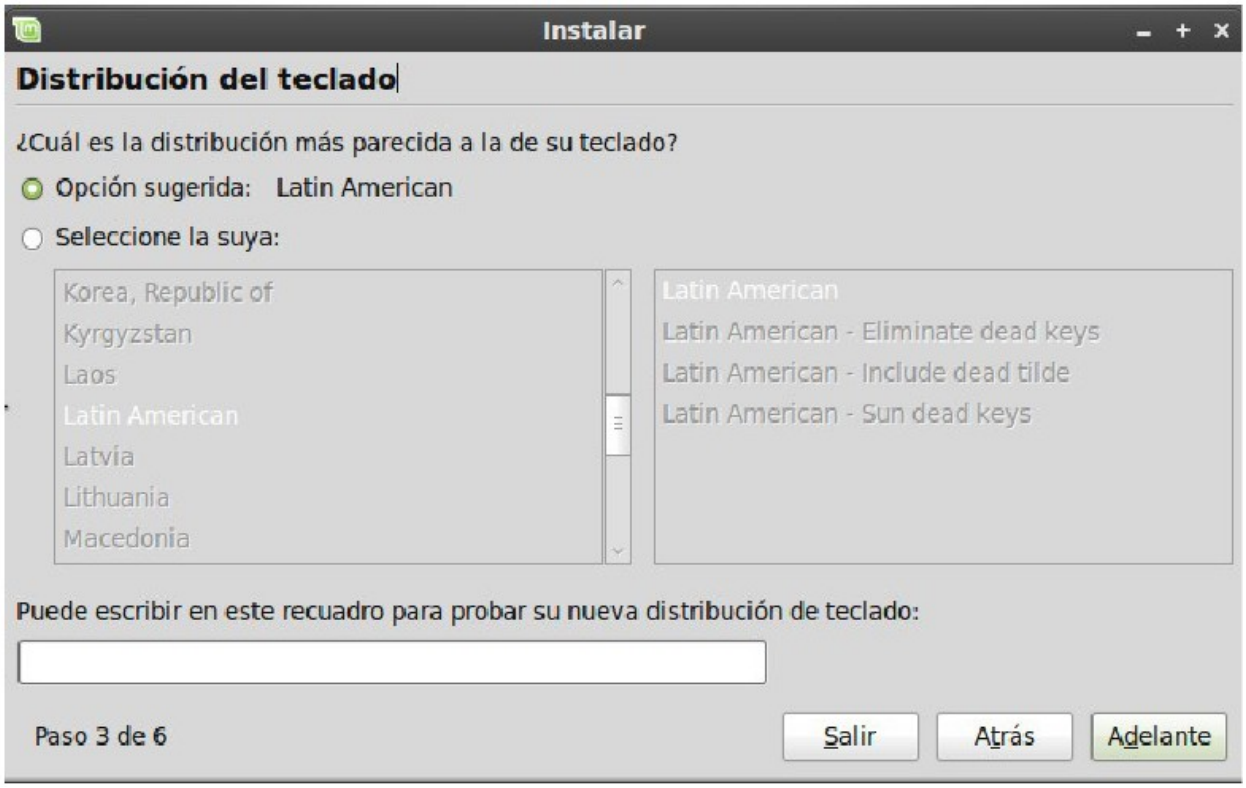
Pantalla 7. Idioma

Después se selecciona la ubicación, esto ayuda a configurar automáticamente elementos como el uso horario, así como la moneda, entre otras cosas.

En nuestro caso elegimos México, por lo que automáticamente se configura el uso horario a GMT -6.

Pantalla 8. Zona

Con esta elección, automáticamente se configura la distribución del teclado a Latino América, esto es importante pues al no seleccionarse correctamente, tendríamos problemas con los caracteres latinos como la ñ, o los acentos.

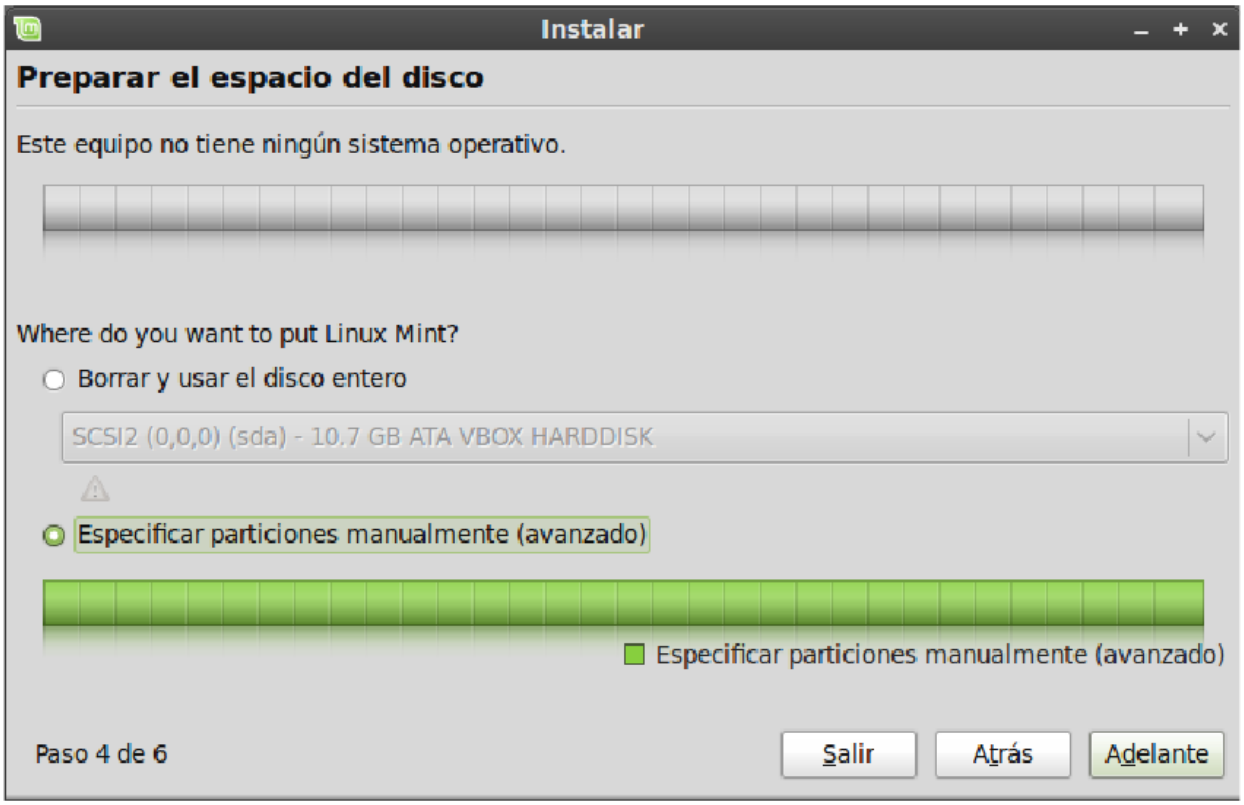


Pantalla 9. Teclado

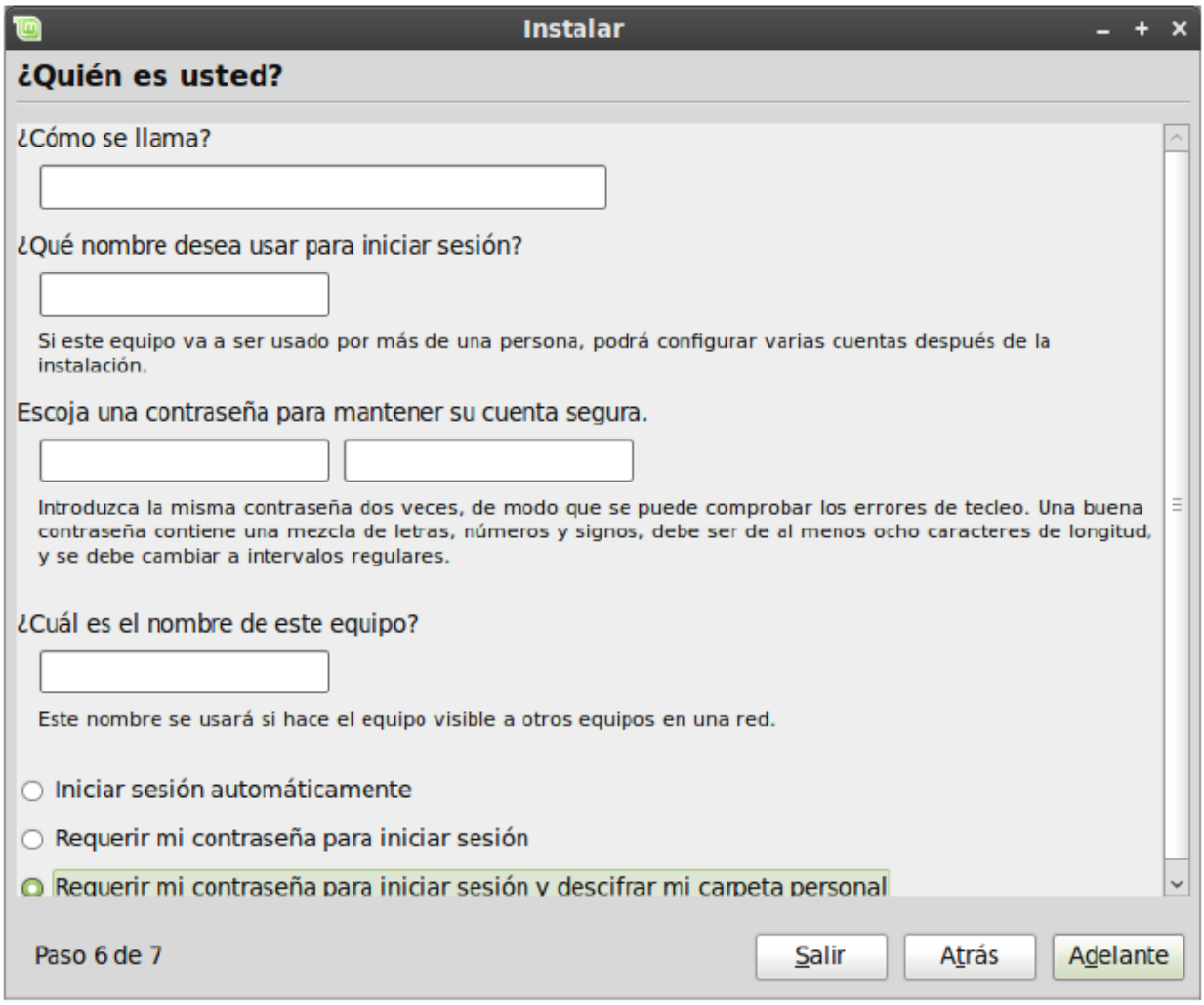
Una vez hecha la selección, comienza el proceso de particionar el disco.

Por consiguiente, nos muestra las opciones para elegir en donde instalar Linux Mint, dependiendo de si tenemos otros sistemas operativos ya instalados o no, será la opción que debamos elegir.

Pantalla 10. Particionador

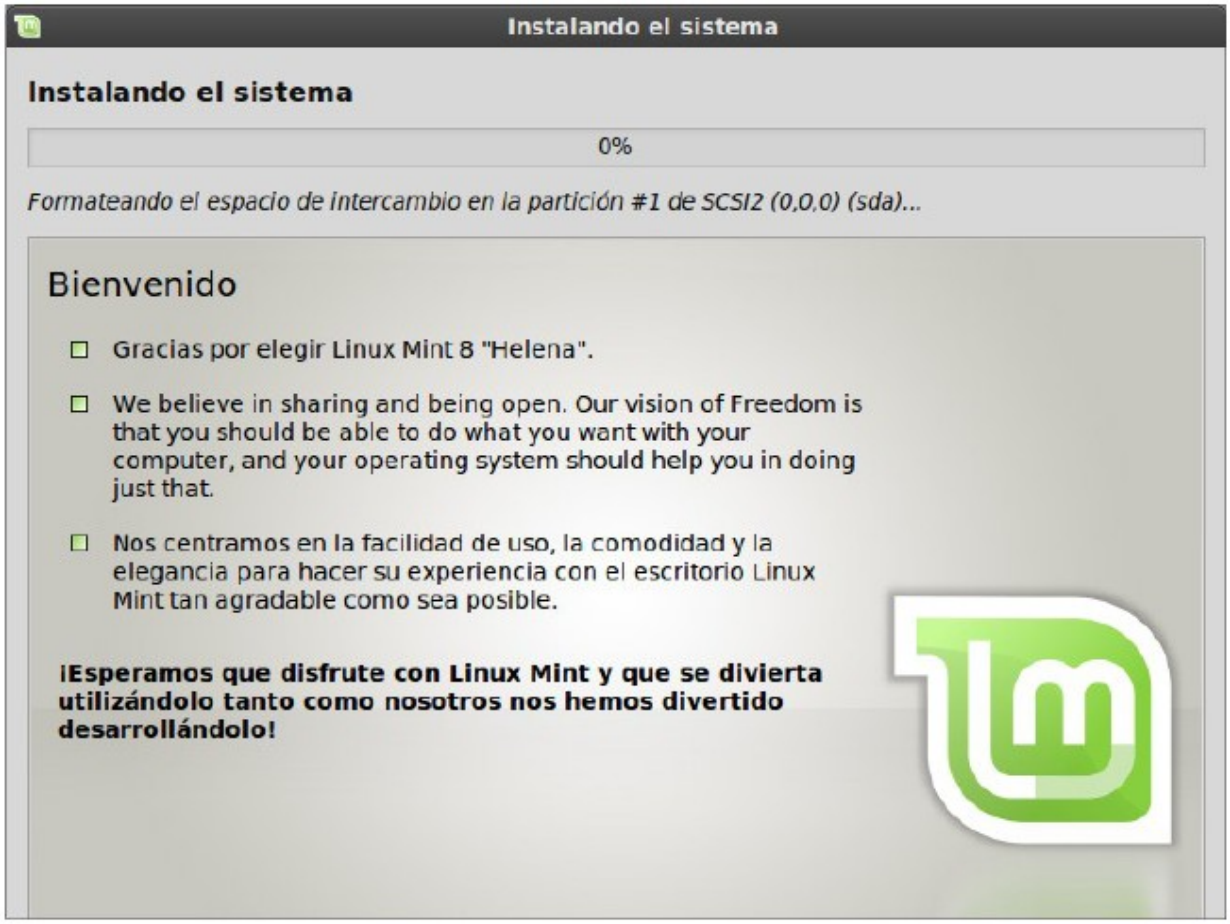


En la siguiente pantalla se toman los datos generales del usuario.



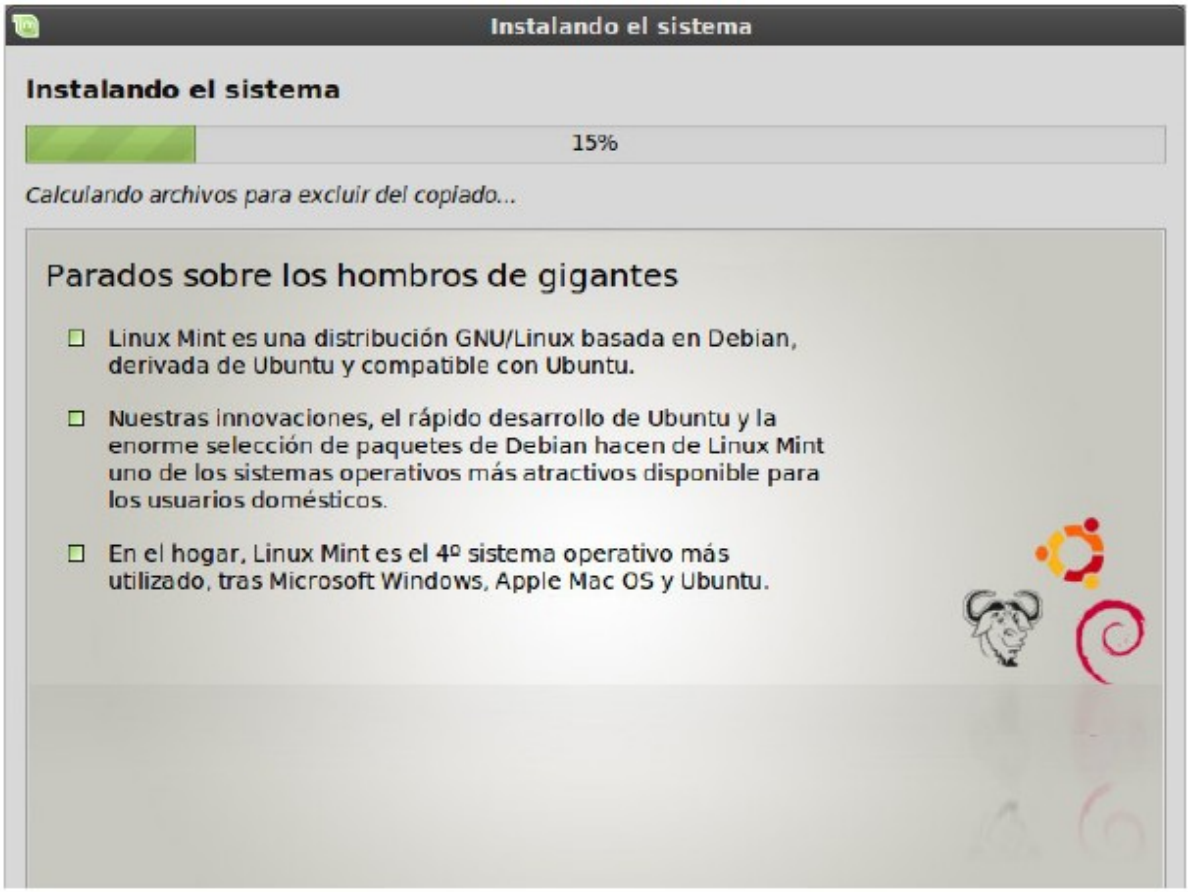
Pantalla 11. Datos

Por lo que una vez concluidos esos pasos, se procede a iniciar el proceso de instalar el sistema operativo en la computadora



Pantalla 12. Instalación

Se instala el sistema.

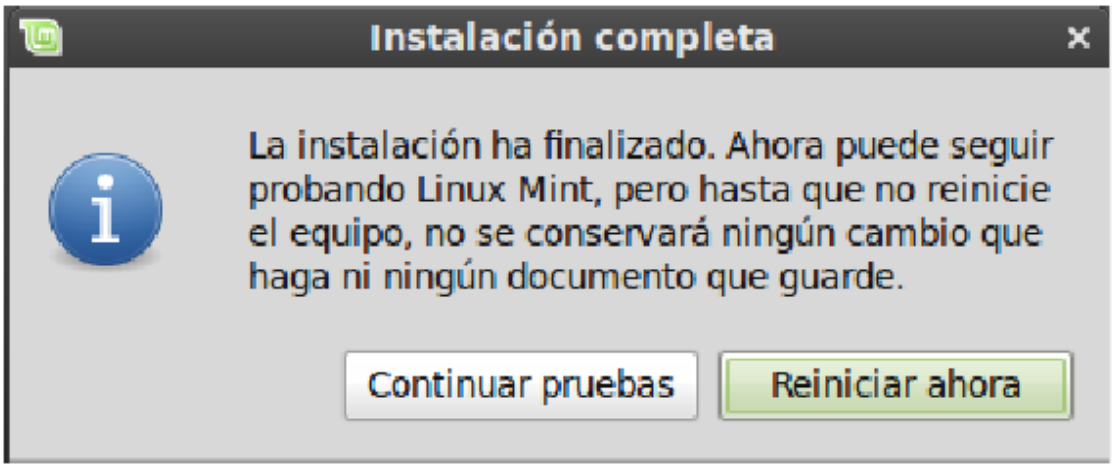


Pantalla 13. Instalación del sistema

Así como los paquetes de idiomas.

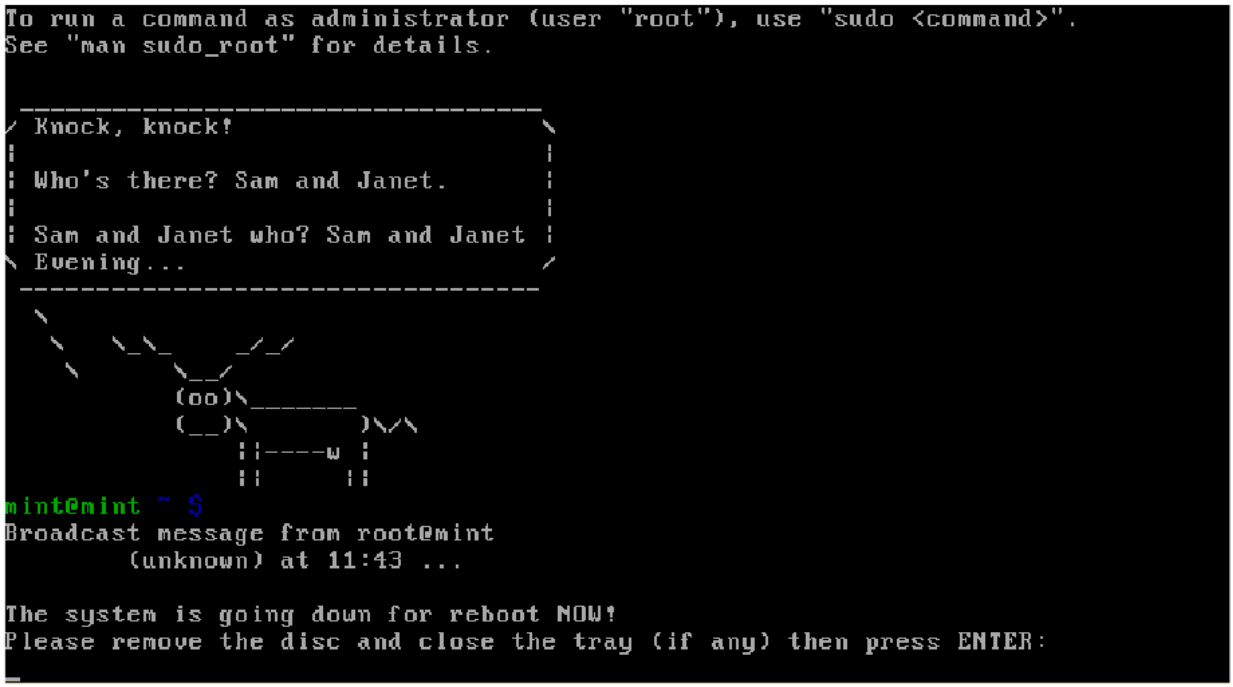
Pantalla 14. Instalación de idiomas

Una vez que concluye la instalación, nos pregunta si deseamos seguir utilizando el liveDVD, o si queremos reiniciar.



Pantalla 15. Instalación completa

Por lo que seleccionamos “Reiniciar ahora” y nos muestra la siguiente pantalla, en ese momento se expulsa el dvd de la unidad lectora y debemos presionar la tecla “Enter”



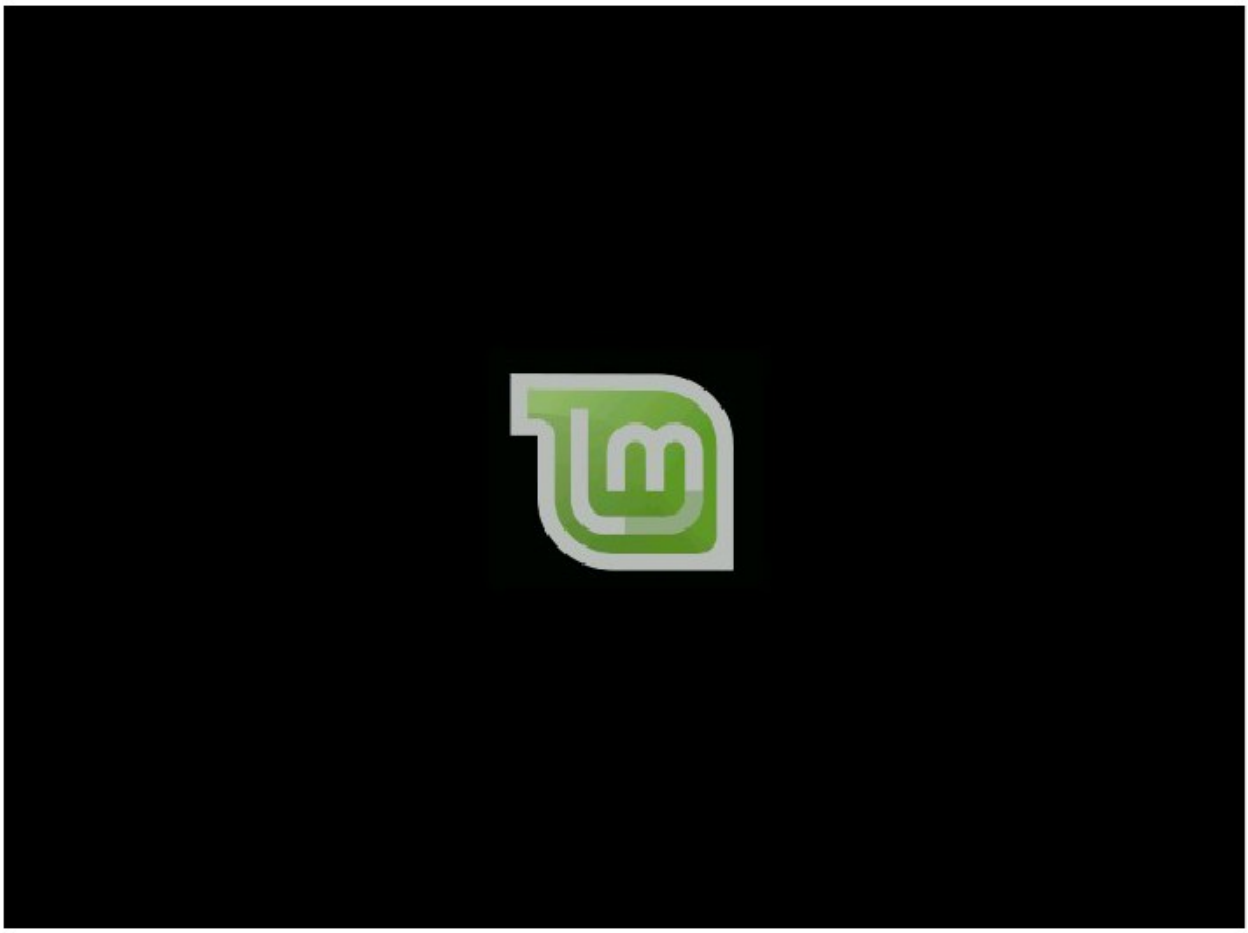
Pantalla 16. Reinicio

3.3.2 Uso y configuración

Una vez que reiniciamos el sistema, nos aparece una pantalla en la cual podeos elegir cuál de los sistemas operativos queremos utilizar. Dicha pantalla es la GNU GRUB.

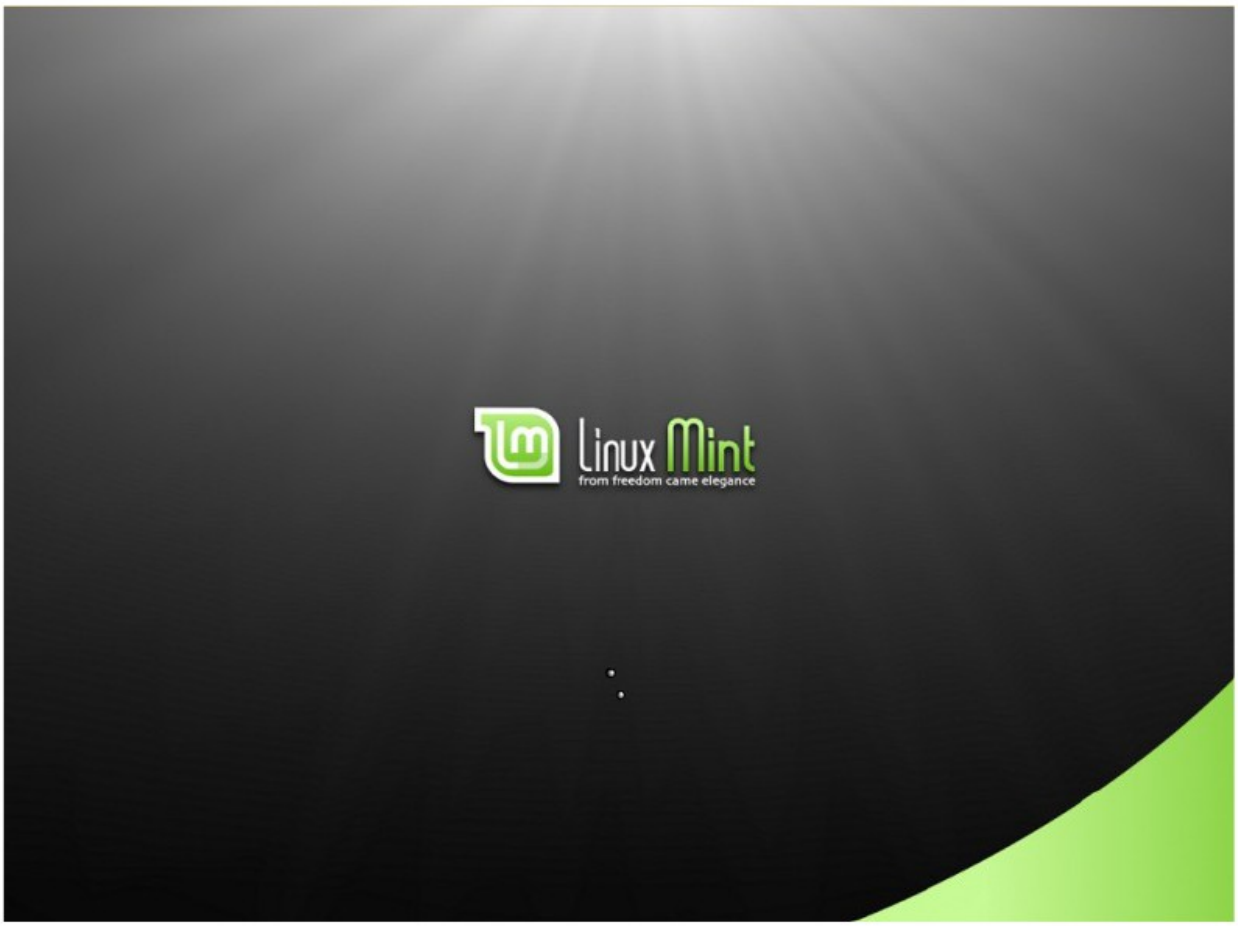
Pantalla 17. GRUB instalado

Al seleccionar la primera opción, nos muestra de nuevo la siguiente pantalla de arranque.



Pantalla 18. Logotipo

De igual forma nos muestra de nuevo la pantalla donde prepara el escritorio.



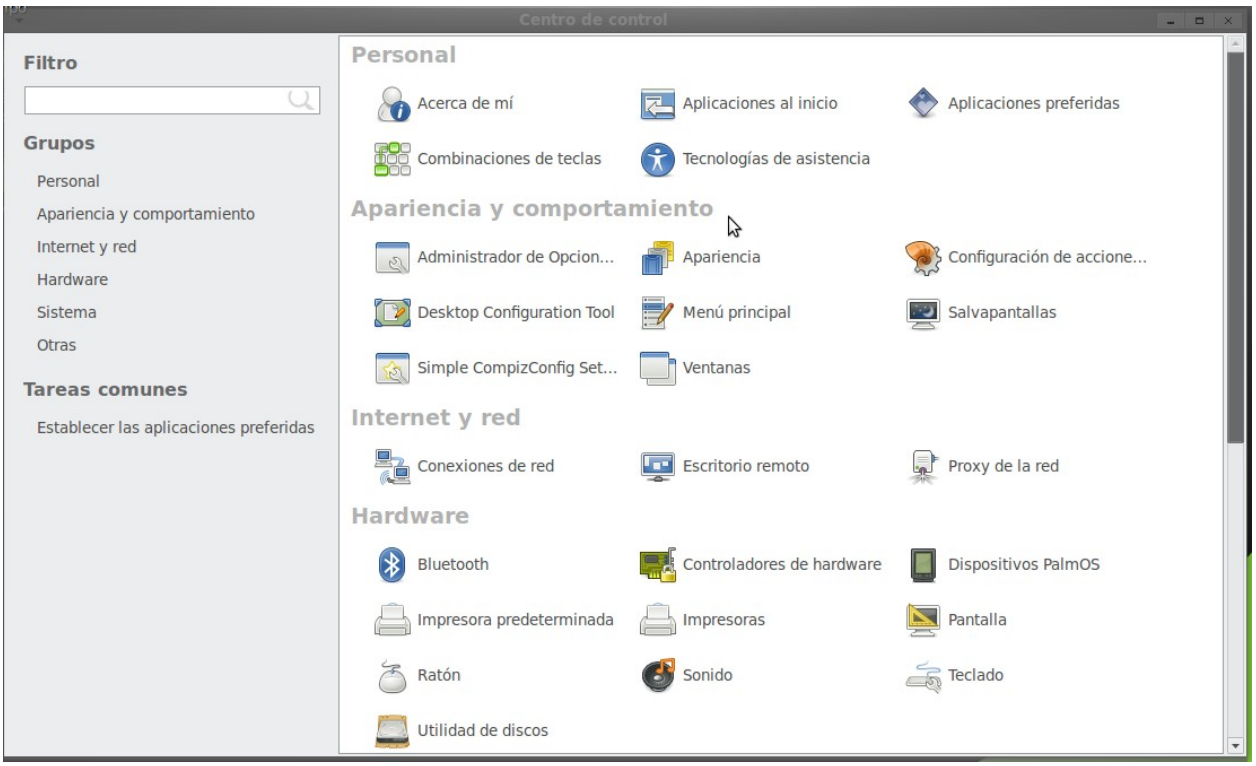
Pantalla 19. Mint instalado

Después nos muestra de nuevo el escritorio, pero esta vez ya está instalado en la computadora.

Pantalla 20. Escritorio Mint

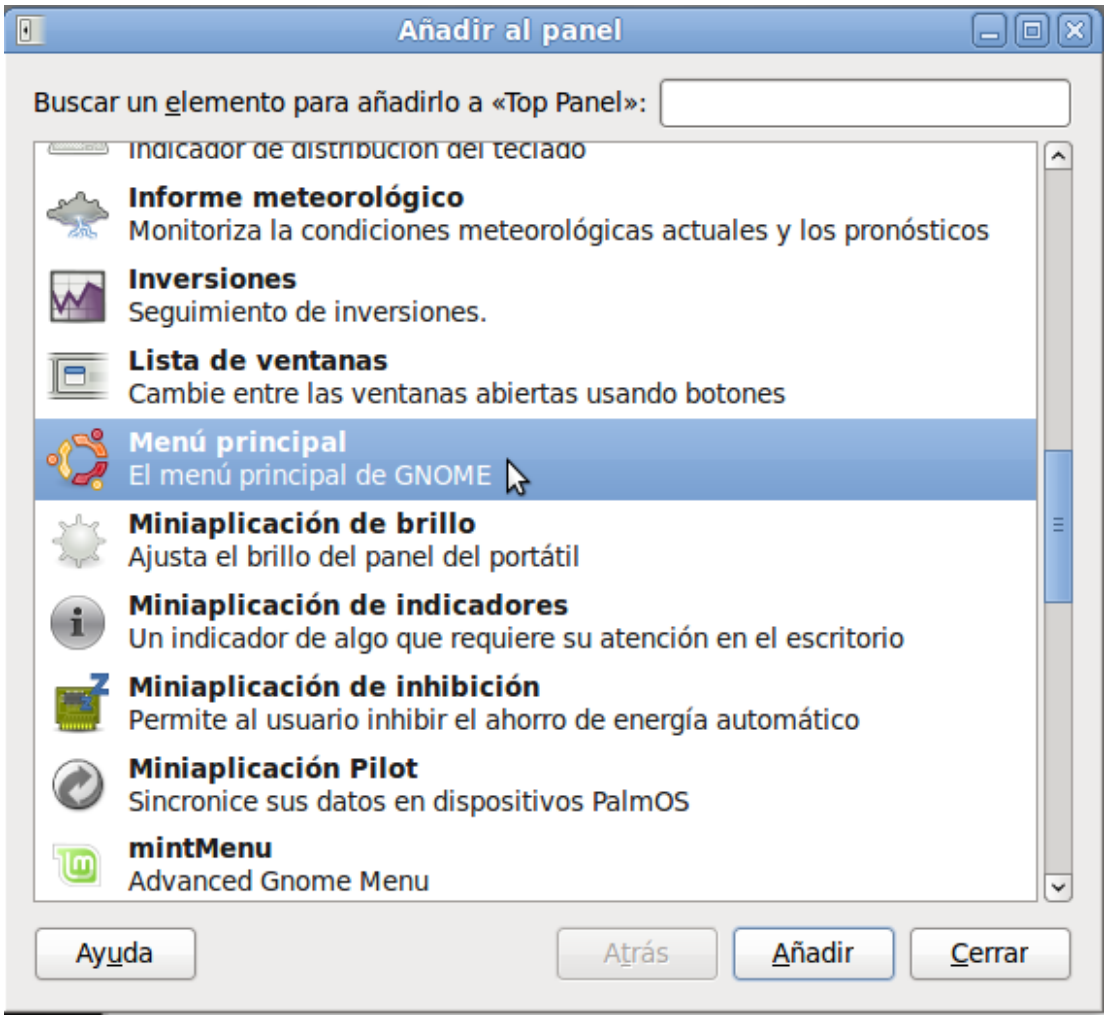
Una vez que tenemos el sistema operativo funcionando, se puede iniciar el proceso de personalización, que puede ser desde cambiar colores, pantalla de acceso, fondo de escritorio, así como instalación y configuración de programas que se desea que lleve el liveUSB.

La configuración puede modificarse desde el centro de control



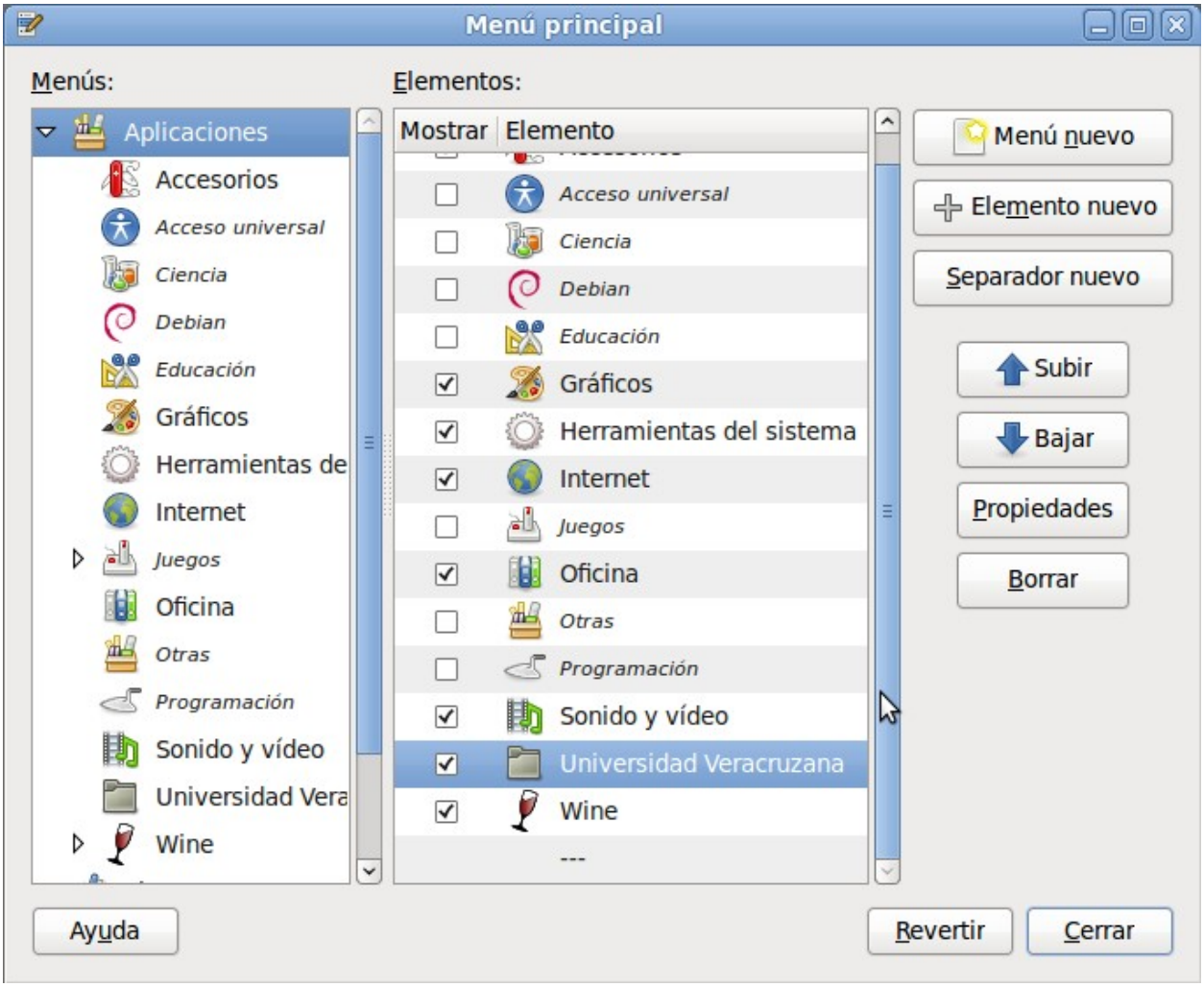
Pantalla 21. Centro de control

También se pueden agregar o quitar módulos de la barra de tareas, esto puede servir por si se desea utilizar el menú de inicio clásico de GNOME, en lugar del menú personalizado de Mint.



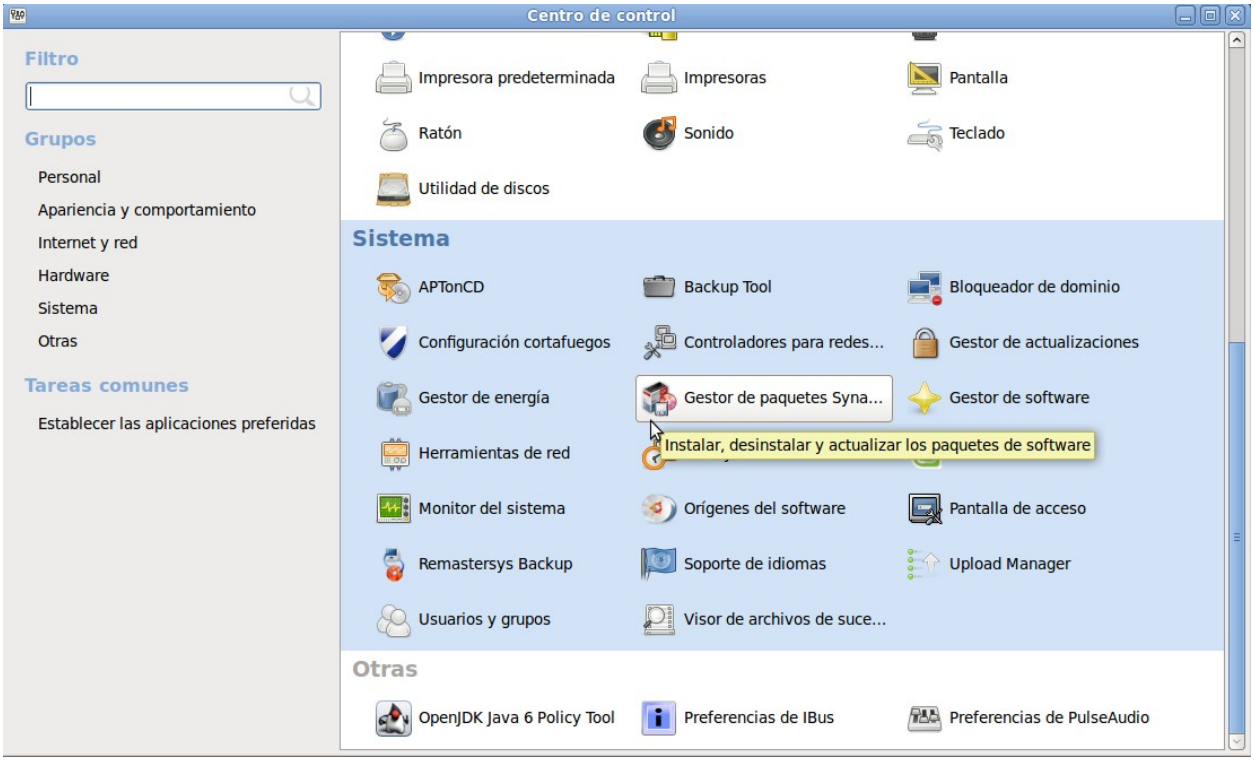
Pantalla 22. Gadgets

Otro beneficio es que Linux, nos permite la modificación de dicho menú, así, se pueden agregar elementos personalizados, y dentro de cada ellos accesos a programas o a documentos, todo esto muy útil para poder separar los semestres y las materias.



Pantalla 23. Configuración del menú principal

Desde el mismo centro de control podemos agregar los programas que necesitemos, para que una vez que esté hecho el LiveUSB, ya estén instalados y configurados.



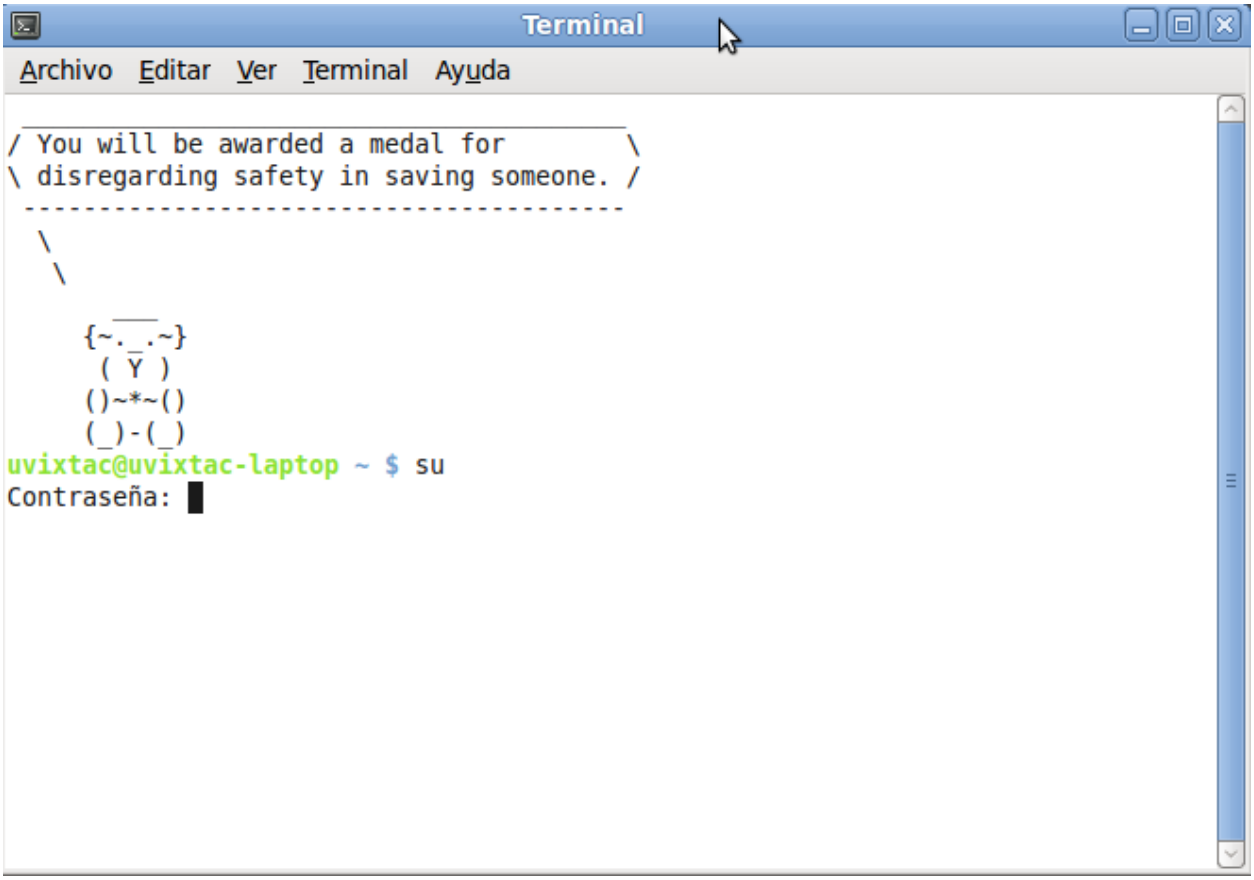
Pantalla 24. Instalar programas

3.3.3 Creación del liveUSB

Una vez que se tiene configurado el equipo tal y como se desea, puede procederse a la creación de un liveUSB, el cual será una copia del sistema operativo, así como sus configuraciones y programas.

Para ello necesitamos la aplicación llamada remastersys, la cual, se debe agregar a los repositorios de Linux Mint, siguiendo los siguientes pasos.

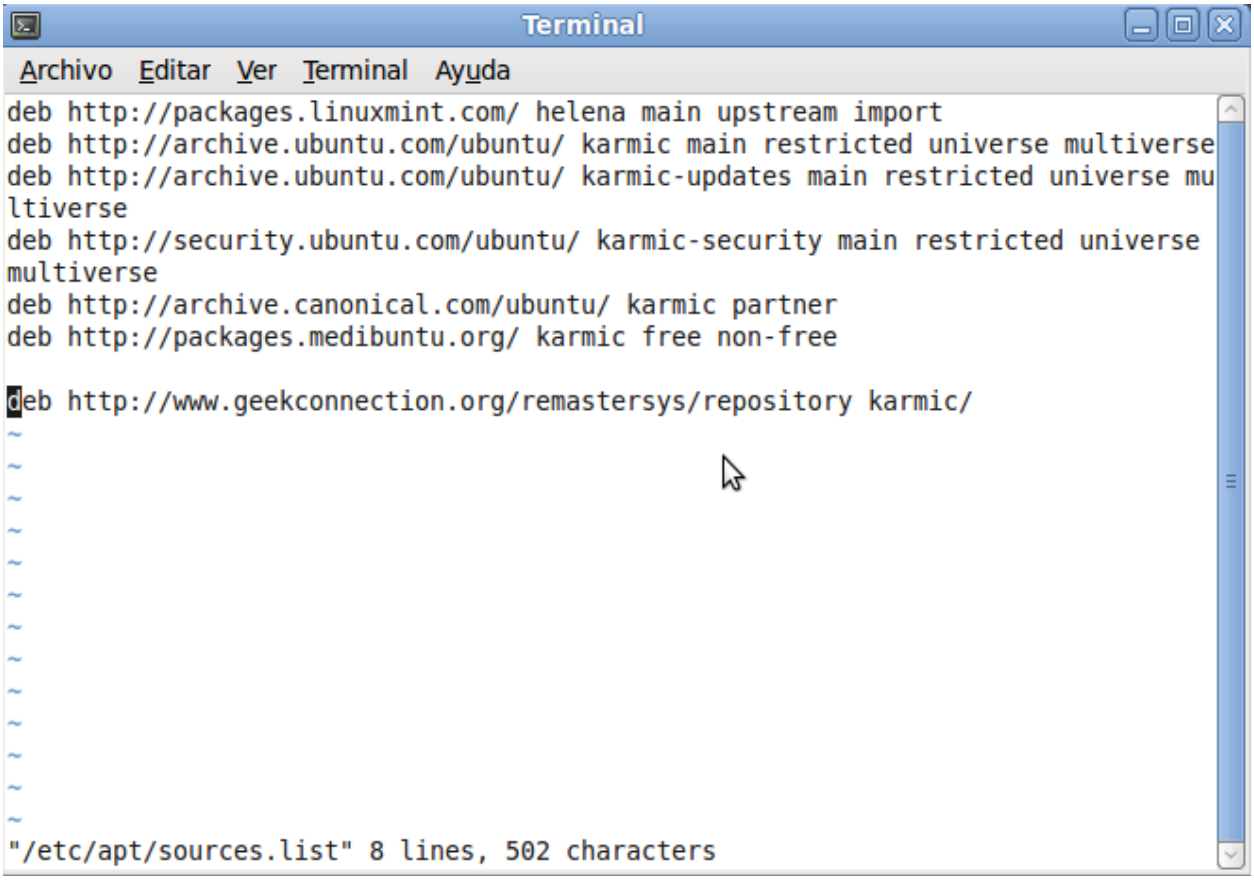
Iniciar la terminal, y acceder como súper usuario.



Pantalla 25. Terminal

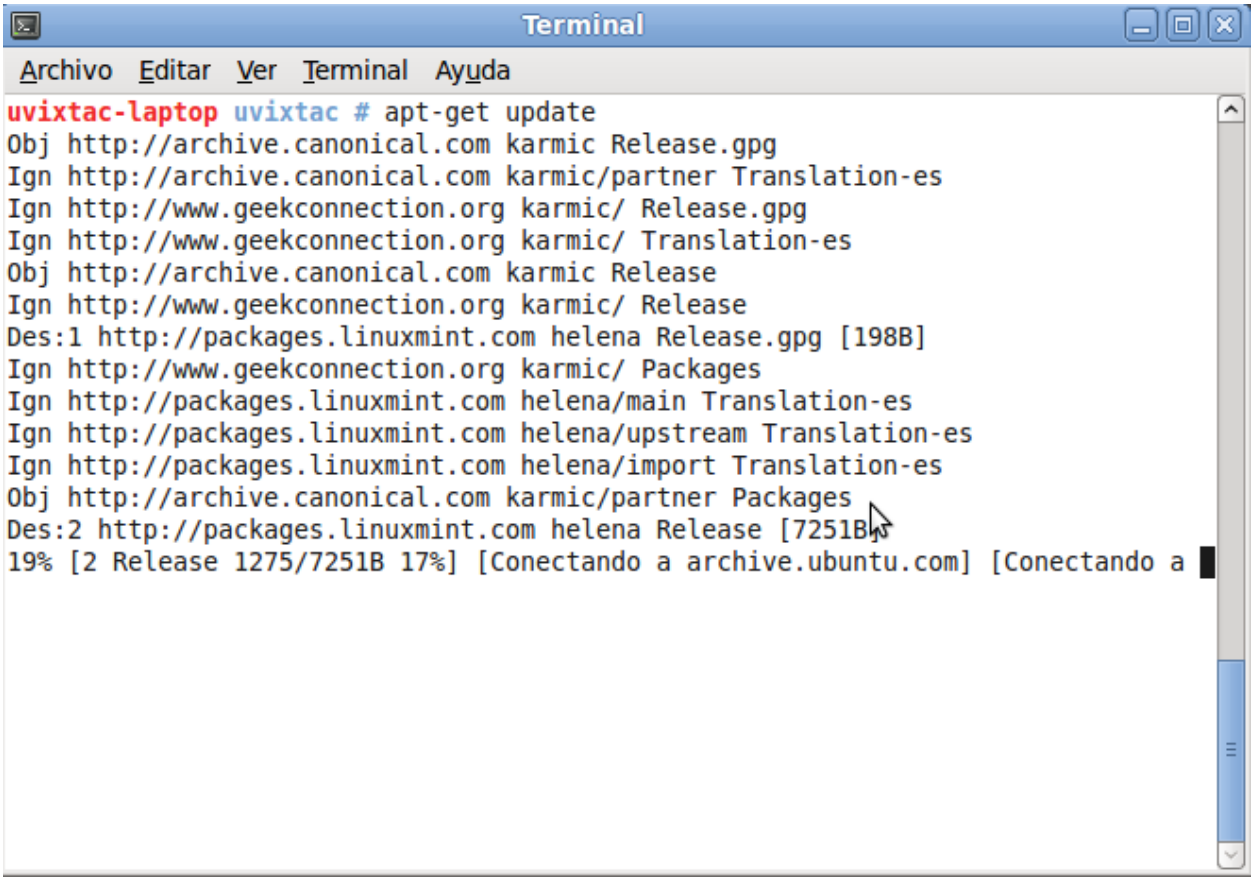
Se edita el archivo /etc/apt/sources.list, y al final, agregar la siguiente línea.

deb http://www.geekconnection.org/remastersys/repository karmic/



Pantalla 26. Repositorios

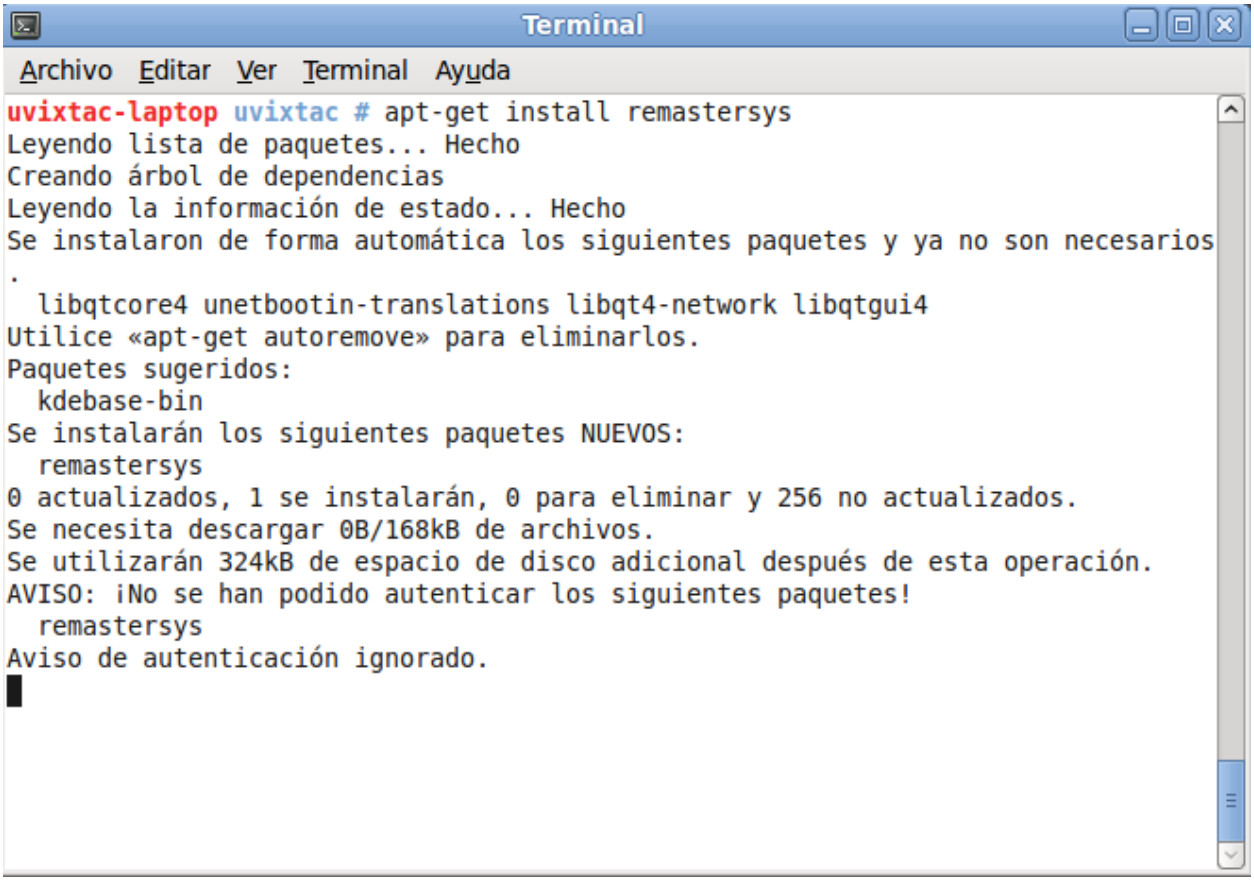
Después un “apt-get update”, para actualizar los paquetes disponibles en los diversos repositorios.



Pantalla 27. Actualización de repositorios

Una vez que se ha actualizado la lista, entonces ya se puede instalar remastersys, con el siguiente comando.

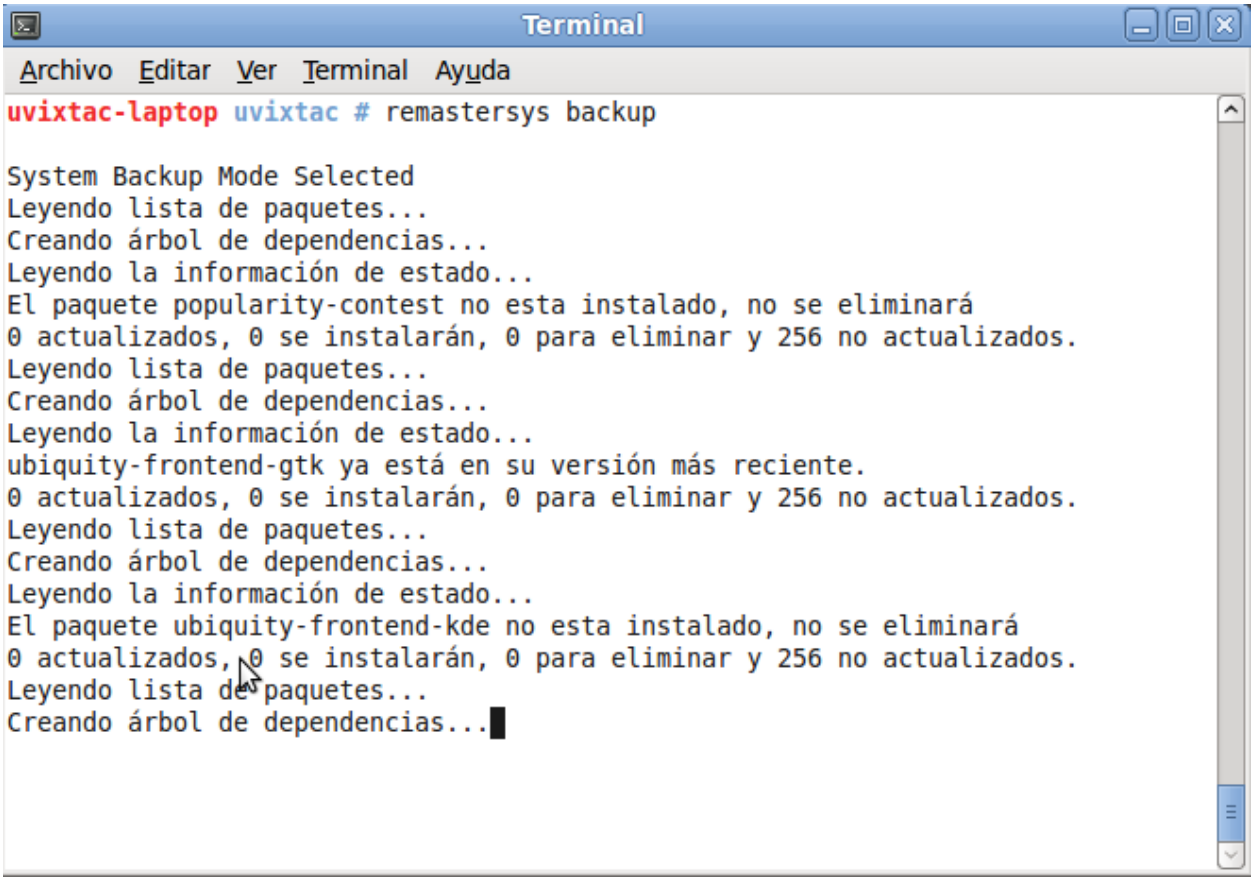
Apt-get install remastersys



Pantalla 28. Instalación de remastersys

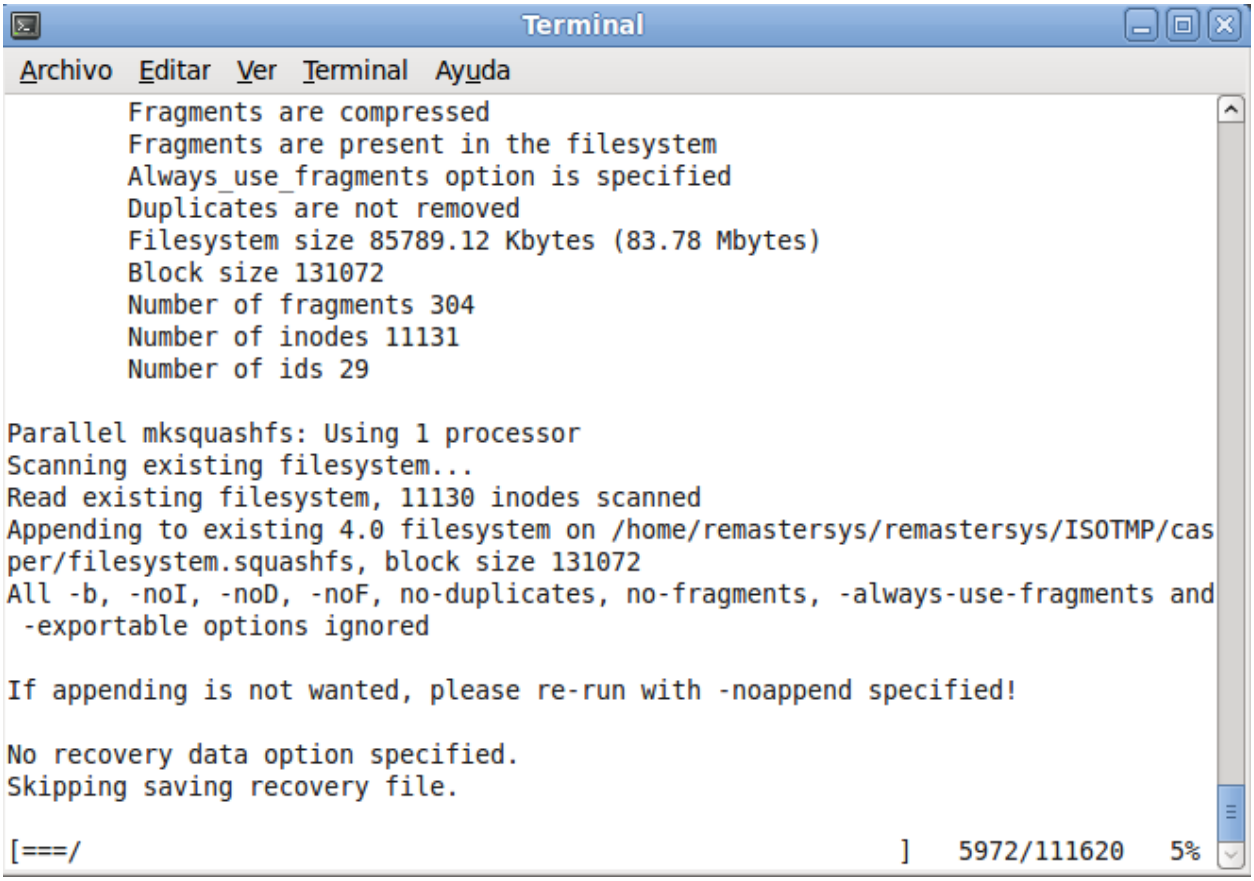
Antes de poder crear el liveUSB, remastersys, va a crear una imagen de disco en formato ISO de nuestro sistema, esto se puede lograr con el siguiente comando:

remastersys backup



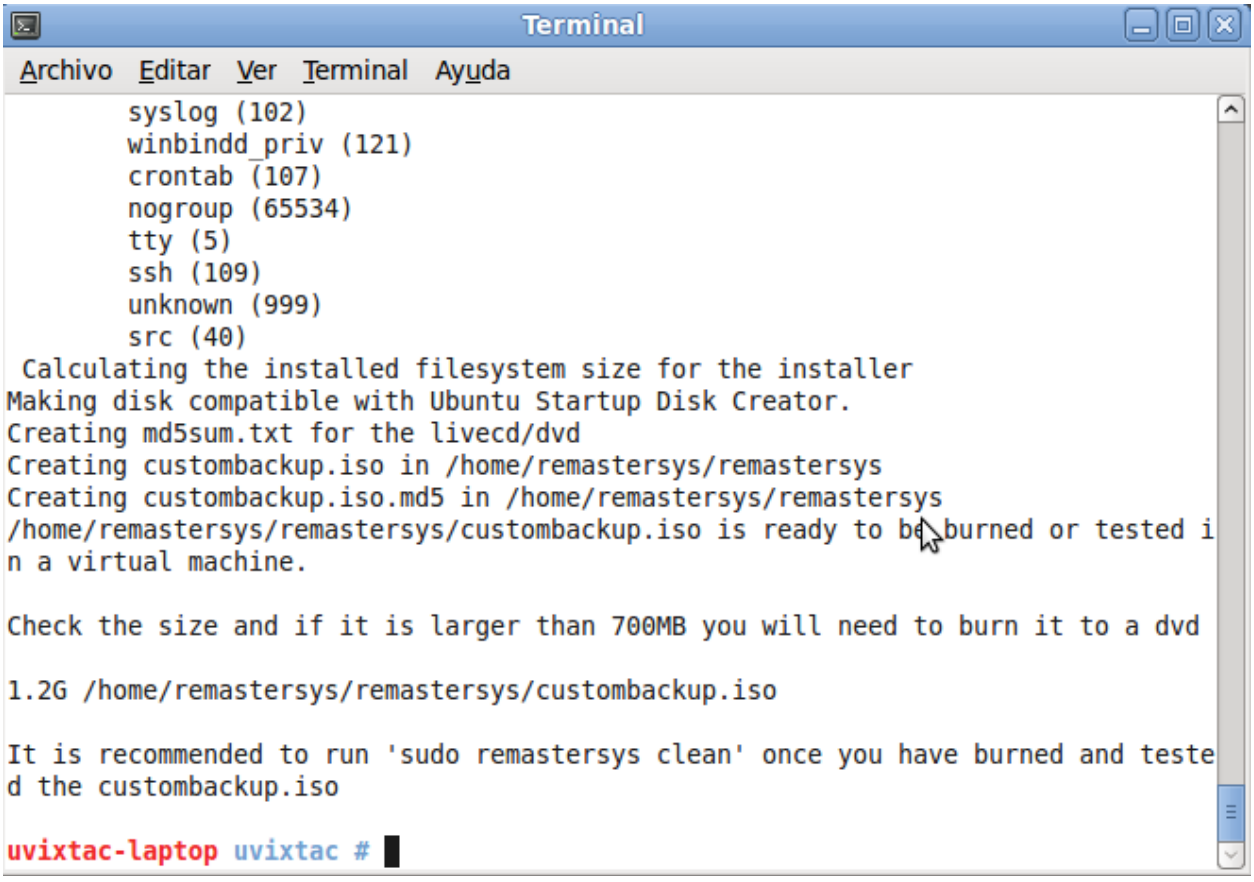
Pantalla 29. Respaldo

El proceso de creación de la imagen puede demorar dependiendo de los elementos que contenga el sistema original.



Pantalla 30. Creación de imagen

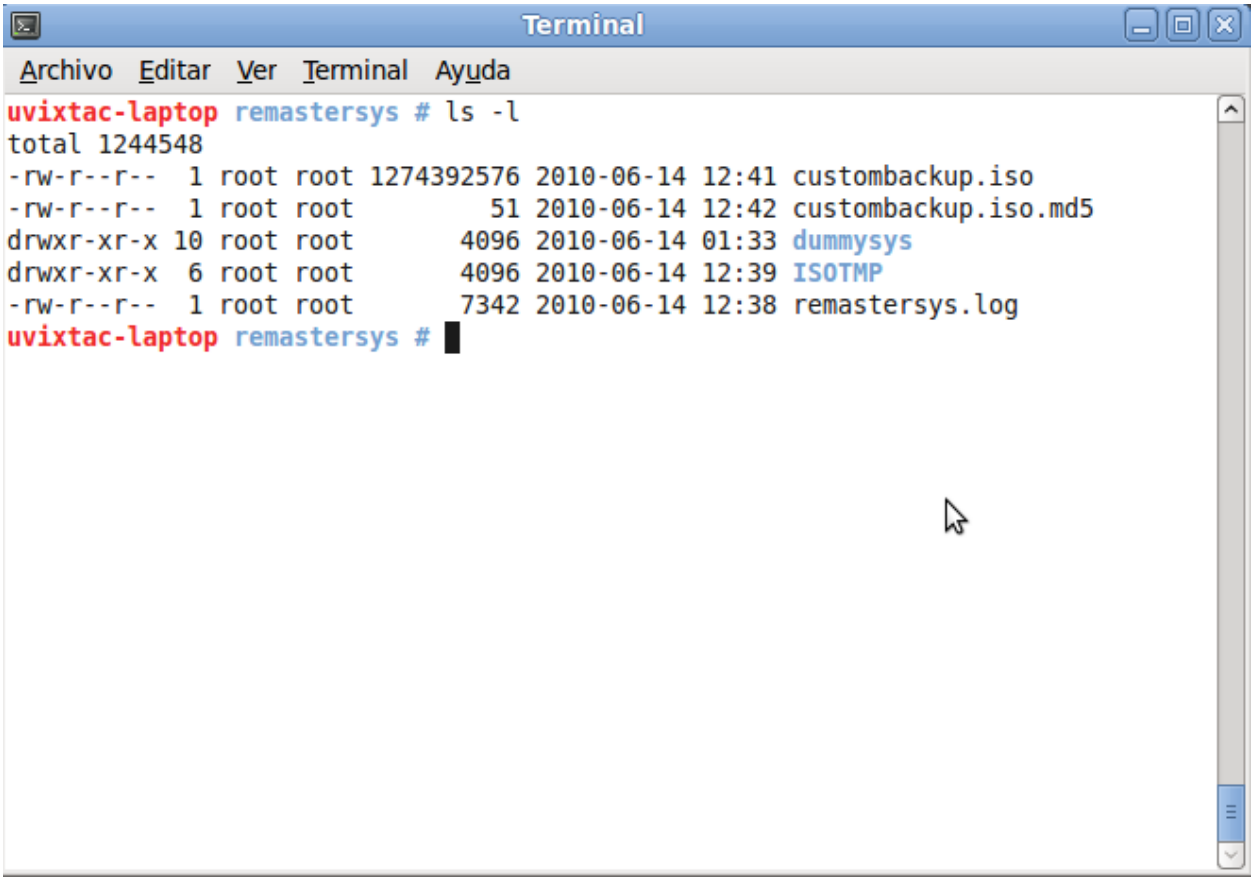
Una vez que termina, crea la imagen en la carpeta /home/remastersys/remastersys.



Pantalla 31. Imagen creada

Dicha imagen ya está lista para grabarse en disco compacto, DVD, o memoria USB.

En éste caso, el tamaño de la imagen supera el giga de información, por lo que deberá usarse una memoria USB de por lo menos 2GB.



Pantalla 32. Lista de archivos

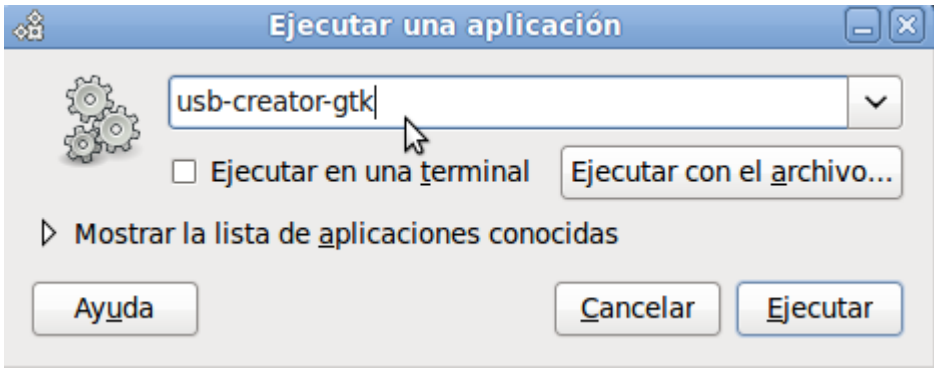
La aplicación que se va a utilizar para generar el USB autoarrancable se llama usb-creator, y se instala de la misma forma que el programa anterior, con:

apt-get install usb-creator

Y una vez que finaliza el proceso de instalación, puede ejecutarse presionando “alt+f2”, y escribiendo su nombre en el cuadro de diálogo.

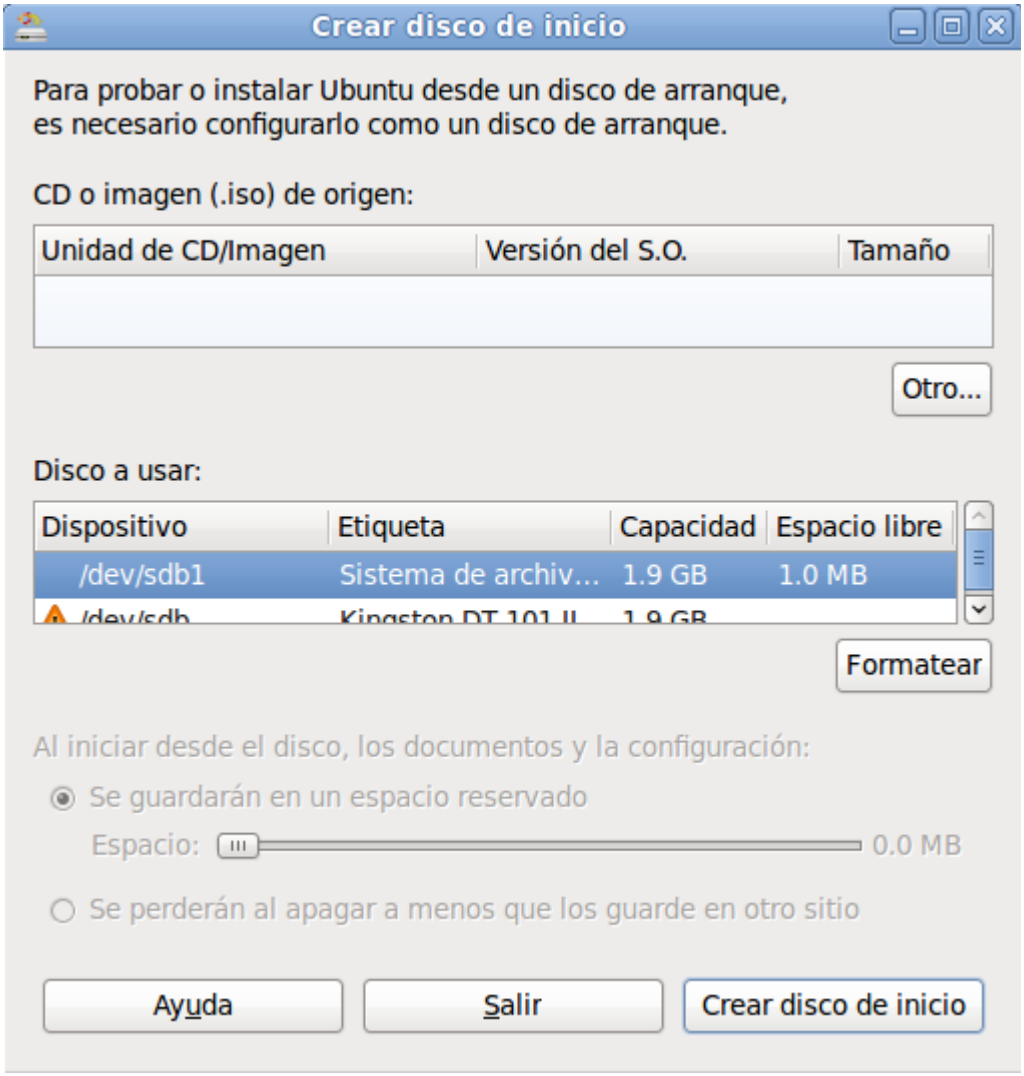
Usb-creator tiene una interfaz en gtk, por lo tanto si queremos ejecutar la aplicación en modo gráfico escribimos:

Usb-creator-gtk



Pantalla 33. Ejecutar

Lo que ejecutará el programa, mostrándonos la siguiente pantalla.

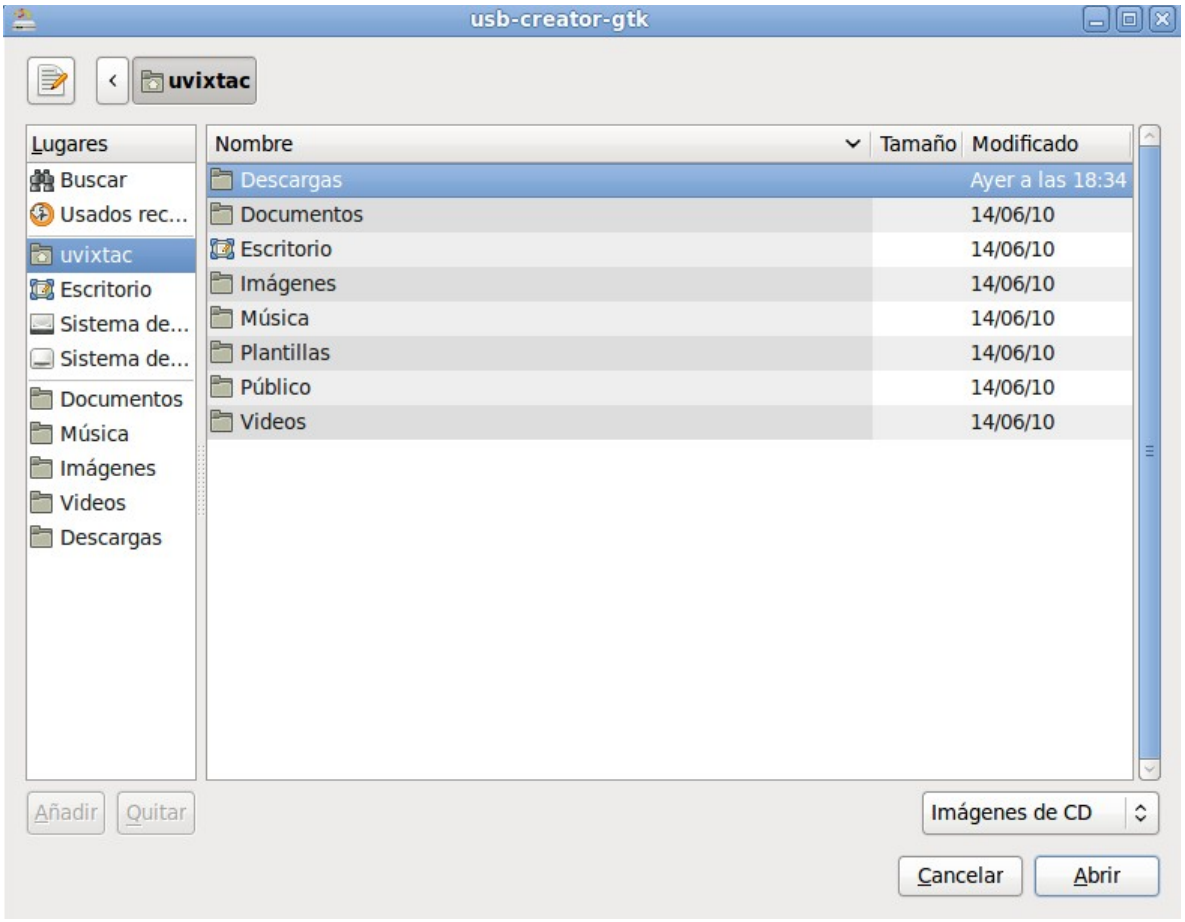


Pantalla 34. USB creator

Ya debe estar conectada la memoria, para que el programa al arrancar la detecte y la reconozca, en éste caso, es /dev/sdb1. La opción que nos interesa para crear

el liveUSB es la de Imagen de ISO que hemos creado, por lo que damos clic en el botón con la leyenda “Otro”.

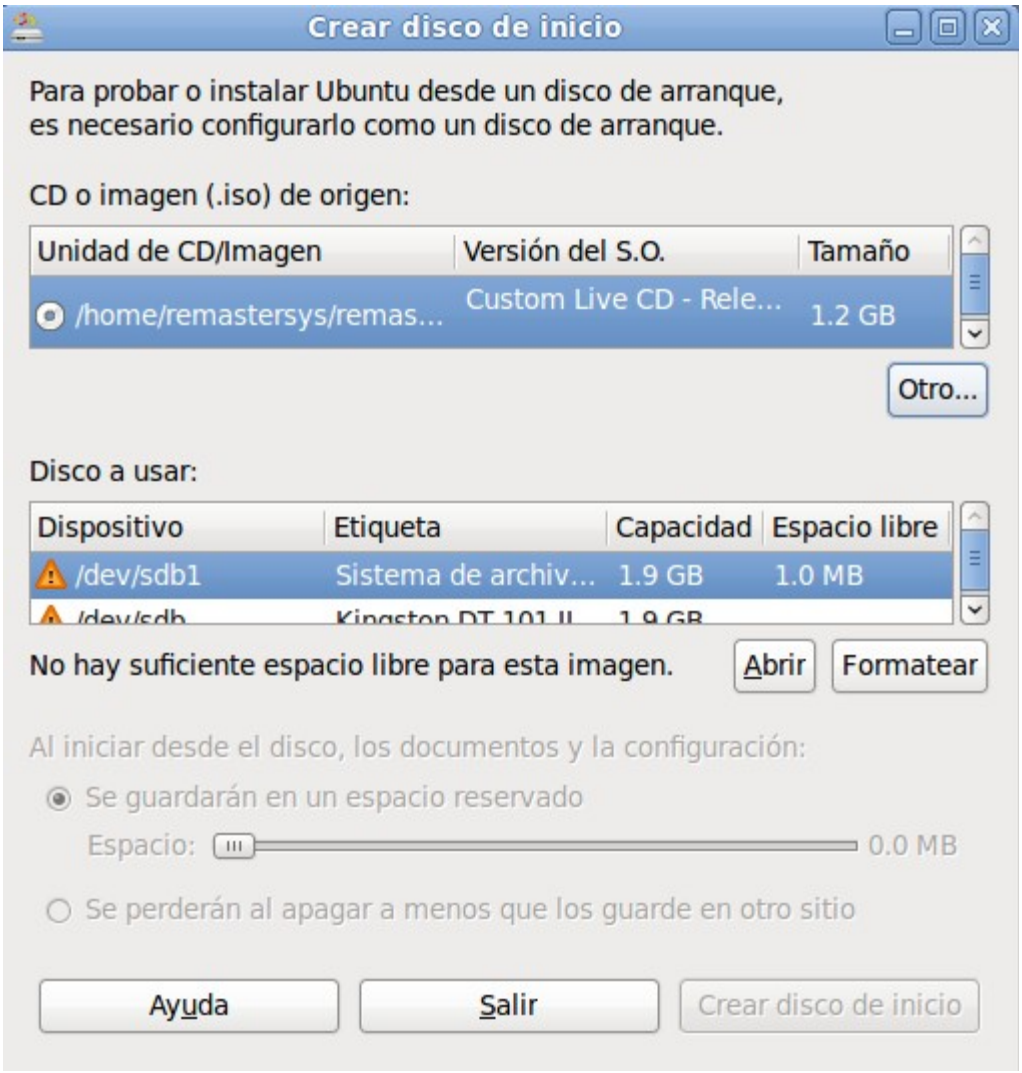
Al hacerlo se mostrará la siguiente pantalla.



Pantalla 35. Abrir imagen

Una vez seleccionada, la cargará y si la memoria está vacía o tiene espacio suficiente, se está listo para comenzar a grabar, en la siguiente pantalla se

muestra el caso de que la memoria esté llena, por lo que se indica que no hay suficiente espacio libre.

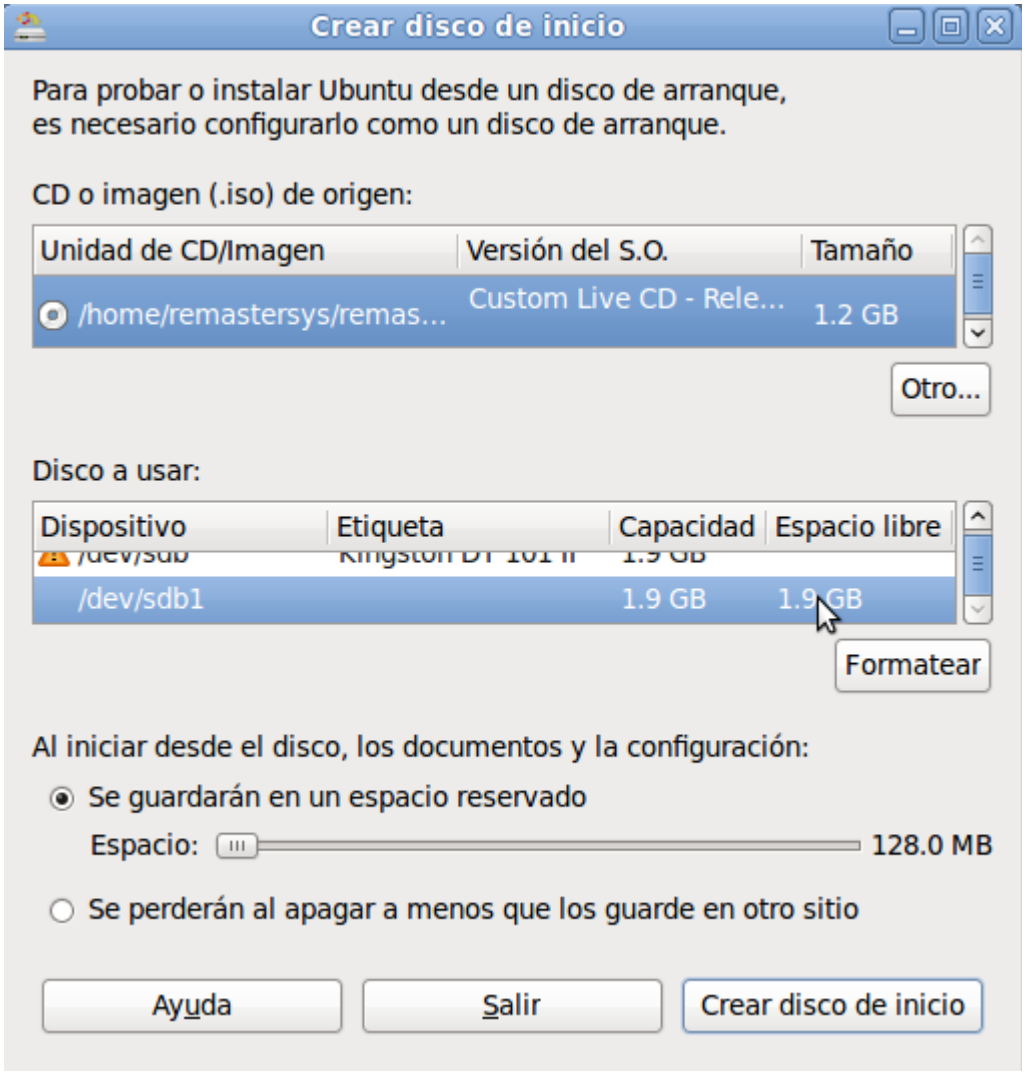


Pantalla 36. Imagen cargada

Por lo cual debemos presionar el botón de formatear, de lo contrario no podemos crear el disco de inicio.

Hay que tener cuidado ya que esa opción elimina todos los archivos existentes en la memoria, y una vez que finaliza el procedimiento nos habilita la opción de crear el disco de inicio.

Antes de proceder podemos configurar la persistencia, que no es otra cosa que permitir al usuario que guardo datos en la misma memoria, así como permitirle agregar o quitar aplicaciones, así como cambiar la configuración.



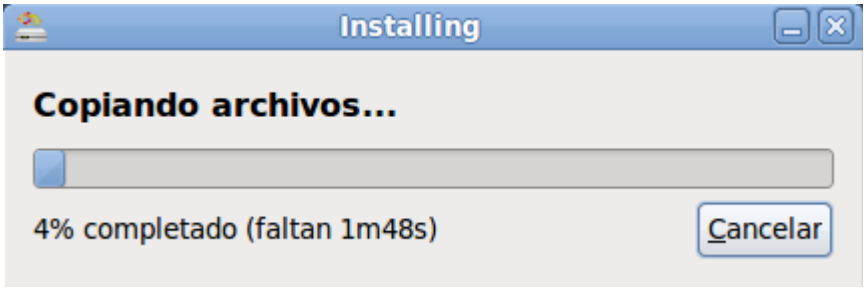
Pantalla 37. Unidad formateada.

Podemos elegir el espacio que queramos darle al usuario, de preferencia sería todo el espacio disponible ya que de lo contrario sería espacio desperdiciado.

Pantalla 38. Eligiendo persistencia.

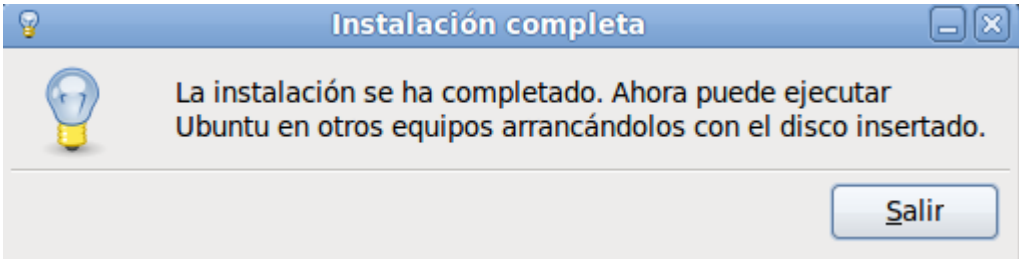
Una vez teniendo todo listo, podemos comenzar el proceso de creación al presionar el botón con la leyenda “Crear disco de inicio”

Éste proceso puede tardar dependiendo de qué tan grande es la imagen.



Pantalla 39. Creando LiveUSB

Ya que finaliza, podemos continuar trabajando en el sistema, o reiniciar para probar la liveUSB ya creada.



Pantalla 40. LiveUSB creada

Éste procedimiento puede repetirse cuantas veces sea necesario, pudiendo crear varias liveUSB. Además, se tiene la ventaja de que pueden configurarse diversos sistemas para las diferentes carreras de la universidad, lo que conllevaría a una buena personalización de las necesidades de los estudiantes.

Por otro lado, el dotar de persistencia a la memoria, permite que los alumnos puedan guardar sus documentos y personalizar su configuración.

Conclusiones

Cuando se propone implementar software libre en algún centro de cómputo de una escuela, siempre las primeras objeciones son en torno a que no es conveniente obligar a un cambio radical a los estudiantes, más cuando sus carreras no son relacionadas directamente al área computacional.

El hecho de ofrecer un dispositivo USB con una distribución modificada de Linux orientada al carácter escolar, es una manera de introducir a todos los estudiantes al mundo del software libre, que puede dar pie y ser un precedente para que tiempo más adelante se decida finalmente dar el cambio no sólo en los centros de cómputo, sino en oficinas también.

No sólo se combatirá la piratería, sino también la universidad ofrecerá soluciones alternas que permitirán al alumno desarrollar su curiosidad en un sistema que puede, y se tiene permitido, hacer modificaciones libremente, por lo que pueden nacer proyectos de alta calidad que involucren todas las áreas de la universidad, siendo entonces de verdad, universal.

Los bajos costos que se tienen actualmente en cuestión de hardware permiten que los alumnos puedan disponer de una memoria USB que la misma Universidad puede dotar. Lo que permitirá que cada año los alumnos de nuevo ingreso tengan su propio dispositivo, y ya que el sistema operativo es libre, todas las ideas para mejorarlo pueden implementarse.

Además se cuenta con el precedente de L.U.L.A, y el apoyo de la UNESCO, lo que hace al proyecto mucho más viable, pues se puede desde desarrollar software a medida para necesidades dentro de la Universidad, como utilizar software ya existente, por ejemplo un sincronizador de archivos que permita que todas las

tareas de un alumno se guarden en una carpeta y ésta automáticamente se sincronice en otra PC, como en la de su profesor. O software colaborativo en tiempo real que permita que un alumno exponga su presentación y todos sus compañeros vean su pantalla en su propia computadora.

Todo ello son tan solo unos ejemplos que se pueden agregar al proyecto LiveUSB, y que al llevarse a cabo y después de derribar la brecha tecnológica y del monopolio, haciendo a un lado las licencias y las restricciones, permitirán al estudiante una mejor experiencia en el manejo de la tecnología.

Bibliografía

ADELL, Jordi & BERNABÉ, Y. Software libre en educación, Madrid, McGraw-Hill

ALONZO, José Luís. Instalación y configuración Linux Mint 8 Helena para nuevos usuarios.
http://www.xarxatic.com/wp-content/uploads/2010/02/Tutorial-Instalación-Linux-Mint-8-Helena.pdf, 12-06-2010, 4:39

CÁCERES, Leonardo. Software Libre en las Escuelas Públicas de la Argentina.
http://www.kdeblog.com/wp-content/uploads/2009/12/Software-Libre-en-las-Escuelas-P%C3%BAblicas-de-la-Argentina.odt

CHAPARRO, Enrique A. 2002. El caso de Linux en las Universidades.
http://www.rau.edu.uy/universidad/csdi/docs/kegel.htm

FREE SOFTWARE FOUNDATION (2004). MACH
http://mirror.gnu.cype.es/software/hurd/gnumach.es.html

FREE SOFTWARE FOUNDATION (2006) HURD
http://mirror.gnu.cype.es/software/hurd/hurd.es.html

HUERTA, Rafael Guevara. 2010. Universidad Veracruzana, semblanza histórica.
http://www.uv.mx/universidad/info/semblanza.html

LORANDI Medina, Alberto Pedro. Breve historia de las distribuciones Linux.
www.uv.mx/alorandi/documents/**DistribucionesLinux**.pdf

LORANDI Medina, Alberto Pedro. ¿Tienen sentido tantas distribuciones?
www.uv.mx/alorandi/documents/**Distribuciones**Linux-ParteI.pdf

www.uv.mx/alorandi/documents/**Distribuciones**Linux-ParteII.pdf

www.uv.mx/alorandi/documents/**Distribuciones**Linux-ParteIII.pdf

LULA, Proyecto. 2010.
http://lula.unex.es/index.php?seccion=participantes

http://lula.unex.es/index.php?seccion=luladef

MARTÍNEZ, Rafael. 2009. Sobre Linux
http://www.linux-es.org/sobre_linux

MARTÍNEZ, Rafael. 2009. Sobre Linux
http://www.linux-es.org/distribuciones

PONS, Nicolás. Linux, principios básicos del uso del sistema. Ediciones Eni, 320 págs.

RODRÍGUEZ, Luís Durán. El gran libro del PC interno: programación de sistemas: hardware a fondo, Alfaomega grupo editor, S.A. 2007. México

STALLMAN, Richard M. Software libre para una sociedad libre, Traficantes de sueños Madrid (España), 2004 232 págs.

STALLMAN, Richard M. 2003. Linux y el Proyecto GNU
http://www.gnu.org/gnu/linux-and-gnu.es.html

STALLMAN, Richard M. 2009. Por qué las escuelas deberían usar exclusivamente software libre.
http://www.gnu.org/philosophy/schools.es.html

STALLMAN, Richard M. 2010. La definición de software libre
http://www.gnu.org/philosophy/free-sw.es.html

TANENBAUM Andrew S. Sistemas operativos modernos, Pearson Educación, México, 2003. 976 págs.

UNIVERSIA. Universidad Veracruzana
http://estudios.universia.net/mexico/institucion/universidad-veracruzana

UNIVERSIDAD VERACRUZANA. 2010. DES Económico – Administrativa, Campus Ixtac
http://www.uv.mx/conta_nog/quienes/historia.html