Universidad De Las Ciencias Informáticas.

Trabajo De Diploma Por El Título De Ingeniero Informático

Desarrollo Y Mantenimiento De Una Distribución De Linux

Autores:

Anielkis Herrera González

Yoandy Rodríguez Martínez

Tutor:

Héctor Rodríguez Figueredo

back cover

Agradecimientos

A la presencia de mi madre, espero que esto la hubiera hecho sentir orgullosa.

A mi padre que es mi fuerza, mi palabra, mi mano y mi apoyo.

A mis dos abuelas que siempre me hicieron creer que podía dar más de mi.

A Héctor por permitirnos las molestias en su casa.

A los muchachos del proyecto.

A la Revolución por la oportunidad de servirle.

A todos los que apoyaron de un modo u otro

Yoandy Rodríguez

A mi familia, que me ha apoyado y dado fuerzas cuando lo necesité.

A todos los que han echo realidad este proyecto y en especial a nuestro tutor, Hector, quien nos ha apoyado y servido de guía a través de los años.

A los integrantes del proyecto.

A la Revolución, que ha echo posible que hayamos crecido en un pais libre, donde podemos estudiar y trabajar por el bien de la humanidad.

Anielkis Herrera

Declaración de autoría.

Resumen

El software libre ha surgido como una alternativa ante los crecientes monopolios de la industria informática y de las comunicaciones. El sistema operativo Linux se alza actualmente como el proyecto bandera de este movimiento proporcionando, junto a la filosofía y las libertades del modelo, una plataforma para el desarrollo de soluciones personalizables.

En este trabajo se describe Gentoo como metadistribución a partir de la cual se desarrolló Nova LNX, una distribución creada con el fin de facilitar el proceso de migración hacia software libre en la Universidad de las Ciencias Informáticas. Se describe además el procedimiento para crear una plataforma de investigación y desarrollo para la infraestructura productiva, así como para el desarrollo de distribuciones de Linux. Se propone una estructura de soporte técnico a los usuarios internos o externos que pueda tener el producto y crear una cultura de uso de aplicaciones libres favorable al propósito de la nación de combatir la dependencia tecnológica.

Índice de Contenido

Índice de ilustraciones

Introducción.

Linux es un sistema operativo desarrollado por una comunidad libre que en los últimos años se ha alzado como una alternativa ante los sistemas propietarios imperantes en el mercado. En la actualidad, varios países ,incluido el nuestro, han iniciado un proceso de cambio que implica adoptar de dicho sistema y aplicaciones que se ejecuten en su entorno y la paulatina eliminación de software propietario.

Este proceso de migración de software incluye un análisis de las soluciones existentes y la búsqueda de aplicaciones equivalentes en el área de software libre minimizando la pérdida de características y comodidad para el usuario.

Este trabajo tiene como objeto de estudio las distribuciones de Linux y su factibilidad para el proceso de migración de software en la Universidad de las Ciencias Informáticas, centrándose en los métodos de personalización y desarrollo de nuevas soluciones para las mismas.

Los objetivos trazados en el presente son:

- Adaptar una distribución existente a las necesidades específicas de la migración en la Universidad.
- Desarrollar un esquema de desarrollo para dicha distribución.

Para el cumplimiento de dichos objetivos se llevo a cabo un estudio comparativo de las distribuciones consideradas líderes en el mundo del software libre, teniendo en cuenta su capacidades de personalización y optimización en nuevos entornos.

El trabajo esta estructurado en tres capítulos:

- Capítulo uno donde se trata la fundamentación teórica y se discuten los antecedentes.
- Capitulo dos, donde se hace un estudio de la distribución seleccionada como base.
- Capitulo tres, donde se describe el proceso de creación de Nova LNX.

Fundamentación Teórica.

Linux es el esfuerzo conjunto de miles de personas que contribuyen voluntariamente al desarrollo de una plataforma libre. Este esfuerzo ha conllevado a que en los últimos 5 años el sistema operativo y las aplicaciones que lo soportan experimentaran un crecimiento sin precedentes.

Este capítulo es el resultado de la investigación previa al comienzo del proyecto Nova en la cual se analizó el contexto en que se desarrolla software libre en la actualidad. Se hace énfasis en la capacidad de Linux para sustituir a soluciones no libres como sistema operativo de escritorio y las características del proceso de migración, así como una breve análisis de el entorno actual en el que se desarrollan distribuciones de Linux

Software Libre.

Software libre es el software que, una vez obtenido, puede ser usado, copiado, estudiado, modificado y redistribuido libremente. El software libre suele estar disponible gratuitamente en Internet, o a precio del coste de la distribución a través de otros medios; sin embargo no es obligatorio que sea así y, aunque conserve su carácter de libre, puede ser vendido comercialmente. Análogamente, el software gratuito (denominado usualmente Freeware) incluye en algunas ocasiones el código fuente; sin embargo, este tipo de software no es libre en el mismo sentido que el software libre, al menos que se garanticen los derechos de modificación y redistribución de dichas versiones modificadas del programa. No debemos confundir software libre con software de dominio público. Este último es aquel por el que no es necesario solicitar ninguna licencia y cuyos derechos de explotación son para toda la humanidad porque pertenece a todos por igual. Cualquiera puede hacer uso de él, siempre con fines legales y consignando su autoría original. Este software sería aquel cuyo autor lo dona a la humanidad o cuyos derechos de autor han expirado tras transcurrir 70 años de la muerte de su autor. Si un autor condiciona su uso bajo una licencia, por muy débil que sea, ya no es dominio público. En resumen, el software de dominio público es la pura definición de la libertad de usufructo de una propiedad intelectual que tiene la humanidad porque así lo ha decidido su

autor o la ley tras 70 años de la muerte de éste.

De acuerdo con tal definición, el software es "libre" si garantiza las siguientes libertades:

- "libertad 0", ejecutar el programa con cualquier propósito (privado, educativo, público, comercial, etc.)
- "libertad 1", estudiar y modificar el programa (para lo cuál es necesario poder acceder al código fuente)
- "libertad 2", copiar el programa de manera que se pueda ayudar al vecino o a cualquiera
- "libertad 3", mejorar el programa, y hacer públicas las mejoras, de forma que se beneficie toda la comunidad.

Es importante señalar que las libertades 1 y 3 obligan a que se tenga acceso al código fuente. La "libertad 2" hace referencia a la libertad de modificar y redistribuir el software libremente licenciado bajo algún tipo de licencia de software libre que beneficie a la comunidad. Algunos teóricos usan este punto cuarto (libertad 3) para justificar parcialmente las limitaciones impuestas por la licencia GNU GPL frente a otras licencias de software libre, sin embargo el sentido original es más libre, abierto y menos restrictivo que el que se le otorga.

La licencia GPL posibilita la modificación, redistribución del software, pero únicamente bajo esa misma licencia. Y añade, que si se reutiliza en un mismo programa código "A" licenciado bajo licencia GNU GPL y código "B" licenciado bajo otro tipo de licencia libre, el código final "C", independientemente de la cantidad y calidad de cada código "A" y "B" debe de estar bajo la licencia GNU GPL.. Debido a esta cláusula es denominada a veces una licencia “contaminante”.

En la práctica esto hace que las licencias de software libre se dividan en dos grandes grupos, aquellas que pueden ser mezcladas con código licenciado bajo GNU GPL (y que inevitablemente desaparecerán en el proceso, al ser el código resultante licenciado bajo esta) y las que no lo permiten al incluir mayores u otros requisitos que no contempla ni admite la GNU GPL y que por lo tanto no puede ser enlazadas ni mezcladas con código gobernado por la licencia GNU GPL.

Esta situación de incompatibilidad, que podría ser resuelta en la próxima versión 3.0 de la licencia GNU GPL (en desarrollo), causa en estos momentos graves prejuicios a la comunidad de programadores de

software libre, que muchas veces no pueden reutilizar o mezclar códigos de dos licencias distintas, pese a que las libertades teóricamente lo deberían permitir. En el sitio web de la FSF (Free Software Foundation) hay una lista de licencias que cumplen las condiciones impuestas por la GNU GPL .

El término software no libre se emplea para referirse al software distribuido bajo una licencia de software más restrictiva que no garantiza estas cuatro libertades. Las leyes de la propiedad intelectual reservan la mayoría de los derechos de modificación, duplicación y redistribución para el dueño del copyright; el software dispuesto bajo una licencia de software libre rescinde específicamente la mayoría de estos derechos reservados.

Ventajas del software Libre

Escrutinio Publico:

a) Al ser muchos las personas que tienen acceso al código fuente, eso lleva a un proceso de corrección de errores muy dinámico, no hace falta esperar que el proveedor del software saque una nueva versión.

Independencia del proveedor:

a) Al disponer del código fuente, cualquier persona puede continuar ofreciendo soporte, desarrollo u otro tipo de servicios para el software.

b) No existe dependencia del proveedor para el soporte del sistema.

Manejo de la Lengua:

a) Traducción: cualquier persona capacitada puede traducir y adaptar un software libre a cualquier lengua.

b) Corrección ortográfica y gramatical: una vez traducido el software libre puede presentar errores de este tipo, los cuales pueden ser subsanados con mayor rapidez por una persona capacitada.

Mayor seguridad y privacidad:

a) Los sistemas de almacenamiento y recuperación de la información son públicos. Cualquier persona puede ver y entender como se almacenan los datos en un determinado formato o sistema.

b) Existe una mayor dificultad para introducir código malicioso como puede ser: espía (Ejemplo capturador de teclas), de control remoto (Ejemplo, Troyano), de entrada al sistema (Ejemplo. puerta trasera), etc.

Garantía de continuidad:

a) el software libre puede seguir siendo usado aun después de que haya desaparecido la persona que lo elaboro, dado que cualquier técnico informático puede continuar desarrollándolo, mejorándolo o adaptándolo.

b) Ahorro en costos: en cuanto a este tópico debemos distinguir cuatro grandes costos: de adquisición, de implantación (este a su vez se compone de costos de migración y de instalación), de soporte o mantenimiento, y de interoperabilidad. El software libre principalmente disminuye el costo de adquisición ya que al otorgar la libertad de distribuir copias la puedo ejercer con la compra de una sola licencia y no con tantas como computadoras posea (como sucede en la mayoría de los casos de software propietario). Cabe aclarar que también hay una disminución significativa en el costo de soporte, no ocurriendo lo mismo con los costos de implantación y de interoperabilidad.

Desventajas del software Libre

Si observamos la situación actual, es decir la existencia mayoritaria de Software Propietario, tenemos:

Dificultad en el intercambio de archivos:

- Esto se da mayormente en los documentos de texto (generalmente creados con Microsoft Word), ya que si los queremos abrir con un Software Libre (Ejemplo Open Office o LaTeX) nos da error o se pierden datos. Pero esta claro que si Microsoft Word creara sus documentos con un formato abierto (o publico) esto no sucedería.

Mayores costos de implantación e interoperabilidad:

- Dado que el software constituye "algo nuevo", ello supone afrontar un costo de aprendizaje, de instalación, de migración, de interoperabilidad, etc., cuya cuantía puede verse disminuida por: mayor facilidad en las instalaciones y/o en el uso, uso de emuladores (Ejemplo. Si el usuario utiliza Microsoft Windows, la solución seria instalar alguna distribución de GNU/Linux y luego un emulador de Windows, como Wine, VMWare. Terminal X, Win4Lin). Vale aclarar que el costo de migración esta referido al software, ya que en lo que hace a Hardware generalmente el Software Libre no posee mayores requerimientos que el Software Propietario.

Diversidad:

- La diversidad de distribuciones, métodos de empaquetamiento, licencias de uso, herramientas con un mismo fin, etc., pueden crear confusión en cierto número de personas.* Hay quienes ven esto como una fortaleza porque se pueden encontrar desde distribuciones especializadas en sistemas embebidos con muchas limitantes de almacenamiento y dispositivos periféricos de uso especializado hasta distribuciones optimizadas para su uso en servidores de alto rendimiento con varios procesadores y gran capacidad de almacenamiento; pasando por las distribuciones diseñadas para su uso en computadoras de escritorio y entre las cuales se encuentran las diseñadas para el usuario neófito que son muy fáciles de instalar y utilizar y las diseñadas para el usuario avanzado con todas las herramienta necesarias para explotar el software libre en todo su potencial. Cabe notar que la posibilidad de crear distribuciones completamente a la medida para atacar situaciones

muy específicas es una ventaja que muy pocas marcas de software propietario pueden ofrecer y que Microsoft ha sido completamente incapaz de hacer.

¿Qué es Linux?

El término Linux estrictamente se refiere al núcleo Linux, pero es más comúnmente utilizado para describir al sistema operativo tipo Unix (que implementa el estándar POSIX), que utiliza primordialmente filosofía y metodologías libres (también conocido como GNU/Linux) y que está formado mediante la combinación del núcleo Linux con las bibliotecas y herramientas del proyecto GNU y de muchos otros proyectos/grupos de software (libre o no libre). El núcleo no es parte oficial del proyecto GNU (el cual posee su propio núcleo en desarrollo, llamado Hurd), pero es distribuido bajo los términos de la licencia GPL (GNU General Public License).

La expresión también es utilizada para referirse a las distribuciones Linux, colecciones de software que suelen contener grandes cantidades de paquetes además del núcleo. El software que suelen incluir consta de una enorme variedad de aplicaciones, como: entornos gráficos, suites ofimáticas, servidores web, servidores de correo, servidores FTP, etcétera. Coloquialmente se aplica el término Linux a éstas, aunque en estricto rigor sea incorrecto, dado que la distribución es la forma más simple y popular para obtener un sistema Linux.

La marca Linux (Número de serie: 1916230) pertenece a Linus Torvalds y se define como "un sistema operativo para computadoras que facilita su uso y operación".

Desde su lanzamiento, Linux ha incrementado su popularidad en el mercado de servidores. Su gran flexibilidad ha permitido que sea utilizado en un rango muy amplio de sistemas de cómputo y arquitecturas: computadoras personales, supercomputadoras, dispositivos portátiles, etc.

Los sistemas Linux funcionan sobre más de 20 plataformas diferentes de hardware; entre ellas las más comunes son las de los sistemas compatibles con PCs x86 y x86-64, computadoras Macintosh, PowerPC, Sparc y MIPS.i.

Distribución de Linux

Una distribución es un conjunto de aplicaciones reunidas por un grupo, empresa o persona para permitir instalar fácilmente un sistema Linux. Es un 'sabor' de Linux. En general se destacan por las herramientas para configuración y sistemas de paquetes de software a instalar.

Existen numerosas distribuciones Linux (también conocidas como "distros"), ensambladas por individuos, empresas y otros organismos. Cada distribución puede incluir cualquier número de software adicional, incluyendo software que facilite la instalación del sistema. La base del software incluido con cada distribución incluye el núcleo Linux, al que suelen adicionarse también varios paquetes de software.

Las herramientas que suelen incluirse en las distribución de este sistema operativo se obtienen de diversas fuentes, incluyendo de manera importante proyectos de código abierto o libre, como el GNU y el BSD. Debido a que las herramientas que en primera instancia volvieron funcional al núcleo de Linux provienen de un proyecto anterior a Linux, Richard Stallman (fundador del proyecto GNU) pide a los usuarios que se refieran a dicho sistema como GNU/Linux. A pesar de esto, la mayoría de los usuarios continúan llamando al sistema simplemente "Linux" y las razones expuestas por Richard Stallman son eterno motivo de discusión. La mayoría de los sistemas Linux incluyen también herramientas procedentes de BSD.

Usualmente se utiliza la plataforma XFree86 o la Xorg para sostener interfaces gráficas (esta última es un fork de XFree86, surgido a raíz del cambio de licencia que este proyecto sufrió en la versión 4.4 y que lo hacía incompatible con la GPL).

Principales distribuciones en la actualidad.

En la actualizad existen cientos de distribuciones de Linux de usos que van desde sistemas embebidos hasta grandes sistemas de clustering y aunque todas comparten el kernel como característica, su modelo de funcionamiento, herramientas y grupo al que está enfocado convierte a cada una de ellas en un producto diferente del resto.

Aspectos que diferenciación entre distribuciones.

Comparar distribuciones de Linux es harto complicado. Todas incluyen versiones similares del kernel, una misma base de herramientas GNU o BSD, versiones compatibles de X Windows, etc. Por lo tanto usaremos como método de comparación no el contenido de la distribución sino

Instalación.

Se define como instalación al proceso de crear una imagen funcional en disco duro u otro medio de almacenamiento del sistema a partir de los medios proporcionados por el distribuidor o fabricante. La instalación es ejecutada normalmente una sola vez por PC pero tiene como característica especial ser el primer contacto del usuario con la distribución y ayuda a en gran parte a obtener una idea de la usabilidad de la misma.

Configuración.

Es el acto relacionado con la puesta a punto del sistema una vez instalado. Ya sea el reconocimiento de nuevo hardware, el acceso a Internet, la puesta en marcha de los servicios de notificación o el sistema de hibernación de la PC. No incluir herramientas que faciliten estas acciones puede hacer el sistema inoperable por usuarios inexpertos. Las facilidades de configuración son uno de los puntos en que más se diferencias las distribuciones.

Manejo de paquetes.

Aunque varias distribuciones comparten un mismo formato de paquetes el modo en que se adicionan o actualizan puede variar visiblemente entre una y otra. El soporte de repositorios globales desde los que se puede obtener todo el software necesario se esta haciendo cada día más popular y necesario.

Soporte técnico.

La capacidad de obtener ayuda ante cualquier eventualidad del sistema es crucial para los usuarios tanto experimentados como recién iniciados. Esta ayuda puede venir en forma de soporte

especializado por el vendedor, documentación on-line, o respuestas en foros o canales IRC.

Fedora

El proyecto Fedora fue creado en el 2003 como una continuación a las versiones “community” de Red Hat Linux . Su ciclo de producción es de 8 meses al final de los cuales liberan una nueva versión del Fedora Core con actualizaciones para el kernel, el sistema de ventanas y otros componentes esenciales del sistema. Aunque su propósito es servir tanto para entornos de escritorio como de servidores es preferente usarla en los primeros, función que cumple perfectamente debido a la gran cantidad de software de ese propósito que incorpora en sus repositorios. Fedora usa por defecto la rama 2.6 del kernel con todas las mejoras del proyecto freedesktop.org y es compatible con la **LSB**.

Instalación.

El proceso de instalación es manejado por Anaconda, un programa de instalación creado por Red Hat utilizando **Python** y **C**. El programa presenta una interfaz en forma de asistentes (wizards) amigable aún para los usuarios no familiarizados con el entorno. Permite fácilmente la configuración del software básico (periféricos, tarjetas de red, tarjetas PCMCIA) y opciones para sistemas de levantamiento dual (**dual boot**) en caso de que existan en otra partición del disco duro.

Configuración

Fedora carece de un panel de control centralizado para la configuración del entorno una vez instalado, lo cual suele ser un poco difícil para los usuarios principiantes; a pesar de esto incluye un conjunto de herramientas aisladas que cumplen con todas las tareas de puesta a punto del sistema incluyendo soporte para tecnologías propietarias como Microsoft Active Directory, redes LAN basadas en Microsoft Windows, servicios de bases de datos Oracle, etc.

Manejo de paquetes

Al igual que Red Hat, Fedora usa el formato **RPM**(Red Hat Package Management) La actualización

o incorporación de nuevo software en el sistema puede hacerse utilizando una herramienta **GUI** o desde terminal utilizando **YUM**. Esta herramienta también permite el manejo de repositorios en Internet para la localización automática de actualizaciones y dependencias.

Soporte técnico

Existe una gran cantidad de libros publicados acerca de Fedora y Red Hat, siendo esta la primera fuente de búsqueda de ayuda junto con la Web del fabricante. La comunidad se ha agrupado en sitios no oficiales como http://www.fedorafaq.org

Gentoo

El proyecto Gentoo fue iniciado por Daniel Robbins a mediados del año 2000 pero su salida al público ocurrió en el 2002. Su ciclo de producción podría calcularse en cerca de 4 meses aunque en realidad depende de si existen suficientes cambios como para desechar el release anterior. La filosofía detrás de Gentoo es muy parecida a la del proyecto ***LFS*** (Linux From Scratch) que permite que el usuario cree su sistema "a la medida" compilando todo el software desde su base teniendo en cuenta opciones conocidas como ***USE*** que indican que características se quieren o no incluir en el sistema. Gentoo soporta todas el software y los estándares de freedesktop.org pero no es compatible con la **LSB**.

Instalación

El proceso de instalación es bastante complicado y engorroso aún para usuarios experimentados ya que normalmente hay que hacerlo todo manualmente sin ayuda de ningún tipo de software . Para suplir esa dificultad, todo el proceso y las vías para la posterior configuración y puesta en marcha del sistema están ampliamente documentados en el sito oficial del proyecto o en la ***Wiki*** de la comunidad.

Configuración

Gentoo carece casi por completo de herramientas ***GUI*** oficiales para la configuración del sistema

por lo que se basa mayormente en ***CLI*** o en las proporcionadas por los entornos de escritorio .

Manejo de paquetes

La administración de paquetes se basa en un sistema conocido como ***Portage*** capaz de manejar tanto paquetes binarios como código fuente y que hace uso de repositorios en Internet. La principal herramienta del portage es el comando **emerge**, el cual se ejecuta en terminales de texto, las versiones **GUI** están en proceso de desarrollo.

Soporte técnico.

Aunque los creadores de la distribución no tienen un servicio de soporte técnico oficial, mantienen una serie de documentos on-line con la solución a los problemas más comunes del proceso de instalación y configuración. Existe junto a esto un foro de usuarios, un canal de IRC donde puede obtener ayuda de expertos e incluso de responsables de proyecto y una **Wiki** colaborativa donde los usuarios escriben soluciones a distintos problemas a modo de "recetas".

Mandriva

Surge de la fusión de Mandrake Linux y Conectiva, ambas compañías y distribuciones de gran éxito en el mundo del software libre. Históricamente se ha caracterizado por una interfaz gráfica amigable desde el proceso de instalación hasta el escritorio inicial. Mandriva Linux, forma parte del grupo LSB (Linux Standard Base), viene con varios 12306 paquetes de software (versión 2006), incluyendo juegos, programas de oficina, servidores y utilidades de Internet., A diferencia de otras distribuciones, no se basa en un único entorno de escritorio. Así, proporciona apoyo tanto a KDE (QT) como a GNOME (GTK), apoyando tanto el desarrollo de programas QT (Kat, buscador integrado en KDE) como GTK (las herramientas de de administración de Mandriva están escritas en GTK).

Instalación.

El instalador de Mandriva Linux es, probablemente, el más amigable de entre las diferentes

distribuciones de Linux. El instalador está traducido a más de 70 idiomas. Y presenta una interfaz de tanta calidad como la de instaladores de otras distribuciones (Red Hat y SuSE por ejemplo) o sistemas propietarios (Microsoft Windows)

Configuración.

El Mandriva Control Center es la herramienta de configuración por defecto. Presenta una interfaz estilo panel de control por categorías desde donde se puede acceder a cualquier punto del sistema. La herramienta permite configurar procesos de arranque, usuarios, firewall, hardware, servicios avanzados, acceso a redes basadas en Microsoft Windows, etc.

Manejo de paquetes.

Mandriva usa un formato particular de ***RPM*** no compatible con Red Hat y en sus ultimas versiones también incorpora una herramienta para el trabajo con repositorios conocida como urpmi. Las software y sus fuentes de instalación pueden configurarse a través del Mandriva Control Center.

SuSE

SUSE Linux es una de las más conocidas distribuciones Linux existentes a nivel mundial. Entre las principales virtudes de esta distribución se encuentra el que sea una de las más sencillas de instalar y administrar, ya que cuenta con varios asistentes gráficos para completar diversas tareas.

Su nombre "SuSE" es el acrónimo del alemán "Software- und Systementwicklung", el cual formaba parte del nombre original de la compañía y que se podría traducir como "desarrollo de software y sistemas". En la actualidad SuSE Linux fue adquirida por Novell que sigue produciendo la llamada Community Edition

YaST

Acrónimo de Yet another Setup Tool, cuya traducción aproximada es "Otra aplicación de configuración más", es una aplicación disponible en sistemas SuSE Linux para la administración del sistema. En las últimas versiones viene a ser instalador/administrador, su historia se remonta desde los inicios de la distribución SuSE. Recientemente se ha cambiado la licencia de YaST a GPL

Entre sus funciones, se encuentran:

1. Instalación de la distribución.
2. Gestión de usuarios y grupos
3. Políticas de seguridad
4. Instalar/desinstalar software
5. Configuración de hardware genérico (tarjeta de sonido, ratones, joysticks, tarjetas de televisión, particiones, impresoras, escáneres...)
6. Generar discos de arranque.
7. Cargar discos de controladores del fabricante (lee la mayoría de ficheros .inf de windows)

Su diseño incluye la posibilidad de adicionarle módulos o plugins para extender su funcionamiento.

Soporte técnico.

Novell da soporte comercial para la los que adquieran SuSE Linux y existen libros que detallan el funcionamiento de la distribución. Los usuarios se agrupan principalmente en foros o listas de correo electrónico donde discuten todo lo relacionado con el sistema.

Ubuntu

Es una distribución basada en Debian (aunque no exista compatibilidad ABI entre ellas) y está disponible en 3 arquitecturas: Intel x86, AMD64, PowerPC. Sus desarrolladores de Ubuntu se basan en gran medida en el trabajo de las comunidades de Debian y GNOME para liberar una versión

estable cada 6 meses y mantenerla actualizada en materia de seguridad hasta 18 meses después de su lanzamiento. El escritorio oficial es GNOME aunque existe una variante llamada Kubuntu que incluye KDE, ambas ramas sincronizan sus versiones con los últimos lanzamientos.

También hay un proyecto para lanzar en el futuro una versión de Ubuntu basada en el entorno de escritorio XFCE, cuyo nombre será Xubuntu.

El sistema incluye funciones avanzadas de seguridad y entre sus políticas se encuentra el no activar procesos latentes por omisión al momento de instalarse. Por eso mismo, no hay un firewall predeterminado, ya que no existen servicios que puedan atentar a la seguridad del sistema.

Para labores/tareas administrativas incluye una herramienta llamada sudo (similar al Mac OS X), con la que se evita el uso del usuario root.

Existe un proceso de mejora en cuanto a la accesibilidad y la internacionalización, de modo que el software esté disponible para tanta gente como sea posible. En la versión 5.04, el UTF-8 es la codificación de caracteres por defecto.

Todos los lanzamientos de Ubuntu se proporcionan sin costo alguno. Las liberaciones incluyen un CD instalable y un Live CD que muestra las características principales del sistema e incluye versiones de aplicaciones libres portadas para plataforma Win32 de modo que se puedan ejecutar sobre Microsoft Windows. Los CDs de la distribución se envían de forma gratuita a cualquier persona que los solicite. También es posible descargar las imágenes ISO de los discos por transferencia directa o bajo la tecnología Bittorrent.

Ubuntu no cobra honorarios por la suscripción de las mejoras de la "Edición Empresarial".

Instalación

Aunque no consta de un instalador gráfico el proceso de instalar Ubuntu no es difícil. El software de instalación incluye detección de hardware básico y tarjetas de redes inalámbricas así pero no muestra muchas opciones acerca de la selección de paquetes a instalar.

Configuración

La carencia de un panel de control centralizado y estándar entre las tres subdistribuciones es un aspecto en contra a pesar de que se incluyen herramientas de administración de sistema pensadas para usuarios recién iniciados.

Manejo de paquetes

Para adicionar o actualizar el software en Ubuntu debemos tener acceso a un repositorio ya sean en Internet o una LAN local. El formato de los repositorios y de los paquetes es muy parecido al de Debian su distribución base pero se ha reportado incompatibilidad binaria entre ambas. La herramienta de instalación Synaptic, una aplicación desarrollada usando GTK (la subdistribución Kubuntu incluye Kpackage un gestor desarrollado para KDE).

Soporte técnico

La principal fuente de soporte está en la comunidad y en sitios on-line aunque existe un servicio de soporte por el fabricante. Se incluye una herramienta que genera información útil acerca de problemas de hardware.

Lenguajes y herramientas utilizadas.

En el desarrollo de la distribución y sus distintas aplicaciones se hizo uso de diferentes lenguajes, herramientas y plataformas cuyas licencias son compatibles con la definición de software libre o de código abierto.

Python

Python es un lenguaje interpretado orientado objetos creado por Guido van Rossum en 1990. Ofrece una plataforma sólida para la integración con otros lenguajes y puede ser perfectamente adaptado a cualquier tipo de entorno siendo esto clave en el desarrollo de proyectos con requisitos variables en el tiempo.

Las principales ventajas del lenguaje son:

1. **Totalmente auditable:** Debido a que es un lenguaje script, el código fuente es distribuido con la aplicación, permitiendo la revisión total de este en busca de fallos de seguridad o errores. Estos pueden ser subsanados sin necesidad de redistribuir la aplicación en su totalidad.
2. **Integración:** Python facilita el desarrollo de Web services a través de sus módulos estándar, puede comunicarse con objetos *COM* y *CORBA*, utilizar bibliotecas desarrolladas en C o C++ e incluso Java (utilizando *Jython*).
3. **Soporte a Internet:** Los módulos base incluyen implementaciones de todos los protocolos estándares de Internet así como la capacidad de trabajar con los formatos de trasmisión de datos más comunes.
4. **Portabilidad:** A pesar de ser principalmente un lenguaje script, Python soporta la generación de *bytecode* independiente de la plataforma e incluso puede ser embebido dentro de otras aplicaciones para interpretar scripts.
5. **Desarrollo Web**: Es posible la creación de simples guiones utilizando *CGI* hasta sistemas completos basados en *Zope* o *Plone* y el soporte nativo para XML.
6. **Pruebas de software:** Python incluye su propia suite de pruebas de unidad la cual puede ser integrada incluso a proyectos que un lenguaje diferente.
7. **Reusabilidad del código:** Las aplicaciones pueden ser utilizadas como módulos por otras aplicaciones garantizando un alto nivel de reutilización del código sin necesidad de ningún otro proceso.
8. **Trabajo con texto y expresiones regulares:** Python posee
9. **Desarrollo de aplicaciones de desktop:** Es posible desarrollar software para desktop utilizando enlaces a las bibliotecas mas conocidas como son wxWindows, GTK y Qt así como hacer uso de las facilidades de los entornos de escritorio..
10. **Desarrollo de aplicaciones embebidas:** Debido a las facilidades para embeber el

interprete en otra aplicación y sus posibilidades multiplataforma algunos fabricantes como Nokia han desarrollado versiones minimalistas de Python que se ejecutan en entornos embebidos. Un ejemplo de esto es el kit de desarrollo para el Nokia 3770.

11. **Acceso a fuentes de datos:** Existen módulos del lenguaje capaces de acceder a distintos tipos de fuentes de datos o gestores de bases de datos. Actualmente se soporta incluso la serialización a XML o un tipo nativo conocido como pickle Los más populares incluyen a *ZopeDB* y *sqlite*.
12. **Aplicaciones científicas:** La biblioteca numérica y de tratamiento de imágenes conforman un kit poderoso para el desarrollo de aplicaciones de corte científico.

C / C++

C es un lenguaje de programación creado en 1969 por Ken Thompson y Dennis M. Ritchie en los Laboratorios Bell. Se trata de un lenguaje de medio nivel pero con muchas características de bajo nivel. Dispone de las estructuras típicas de los lenguajes de alto nivel pero, a su vez de construcciones del lenguaje que permiten un control a muy bajo nivel como la posibilidad de mezclar código en ensamblador o acceder directamente a memoria o dispositivos periféricos. Destaca su gran riqueza de operadores y expresiones.

Esto fue tomado en consideración por Bjarne Stroustrup, a finales de los 80 para diseñar C++, una extensión orientada a objetos de C muy utilizada en la industria de software.

Ambos lenguajes tienen una muy estrecha relación la familia de sistemas basados en el estándar POSIX entre los que se encuentra Linux y constituyen la base de casi la totalidad de componentes y bibliotecas disponibles en el sistema.

GLADE

Glade es un diseñador de interfaz gráfica de usuario (GUI) para aplicaciones basadas en GTK o GNOME. Es también el nombre del framework de desarrollo que se encarga de convertir los archivos XML generados por el diseñador ya sea en el código fuente que genera dicha interfaz o en la interfaz

en sí al ser invocado en tiempo de ejecución por un programa.

El uso de Glade y libglade permite que un mismo diseño de GUI pueda ser aprovechado por distintos programas o lenguajes.

UML

Según la definición dada por la enciclopedia libre on-line UML “es el lenguaje de modelado de sistemas de software más conocido en la actualidad; aún cuando todavía no es un estándar oficial, está apoyado en gran manera por el OMG (Object Management Group). Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema de software. El UML ofrece un estándar para escribir un plano del sistema, incluyendo aspectos conceptuales tales como procesos de negocios y funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y componentes de software reutilizables”.

El UML cuenta con varios tipos de modelos, los cuales muestran diferentes aspectos de las entidades representadas y son conocidos como artefactos. UML no representa por si solo una metodología de desarrollo de software

XMI

XMI o XML Metadata Interchange (XML de Intercambio de Metadatos) es una especificación para el Intercambio de Diagramas.

La especificación para el intercambio de diagramas fue escrita para proveer una manera de compartir modelos UML entre diferentes herramientas de modelado. En versiones anteriores de UML se utilizaba un esquema XML para capturar los elementos utilizados en el diagrama; pero este esquema no decía nada acerca de cómo el modelo debía graficarse.

Para solucionar este problema la nueva Especificación para el Intercambio de Diagramas fue desarrollada mediante un nuevo esquema XML que permite construir una representación SVG (Scalable Vector Graphics). Esta especificación es soportada por un diferentes de herramientas para modelado UML.

XMI es utilizado en el proyecto de modo que el desarrollo de diagramas UML no dependa de una herramienta en específico sino de la especificación.

Subversion

Subversion es un sistema de control de versiones desarrollado por Tigris (http://www.tigris.org), una comunidad libre de programadores centrada en la creación de herramientas open source para ingenieros de software. Su objetivo principal es sustituir al sistema dominante en los entornos Unix conocido como CVS.

Las principales ventajas de usar subversion como sistema de control de versiones son:

1. Soporte transaccional, las actualizaciones al repositorio se ejecutan o se cancelan como un todo.
2. No se necesitan comandos especiales para el manejo de directorios.
3. Soporte para versiones en archivos binarios.
4. Permite asignar metadatos y propiedades a los archivos bajo control de versiones. Estos metadatos son incluidos también dentro de dicho control
5. La integración con el servidor Apache permite que los repositorios puedan quedar accesibles por los protocolos http y https.

Metodología de desarrollo de software

Como metodología de desarrollo de software se optó por la programación extrema una de las llamadas técnicas ágiles

XP (eXtreme Programing) nace como nueva disciplina de desarrollo de software hace aproximadamente unos seis años, y ha causado un gran revuelo entre el colectivo de programadores del mundo. Kent Beck, su autor, es un programador que ha trabajado en múltiples empresas y que actualmente lo hace como

programador en la conocida empresa automovilística Daimler Chrysler. Con sus teorías ha conseguido el respaldo de gran parte de la industria del software y el rechazo de otra parte.

La programación extrema se basa en la simplicidad, la comunicación y el reciclado continuo de código.

El objetivo primordial de XP es la satisfacción del cliente. Esta metodología trata de dar al cliente el software que él necesita y cuando lo necesita. Por tanto, se centra en minimizar el tiempo de respuesta a las necesidad de este incluso cuando los cambios sean al final de ciclo de la programación.

XP potencia al máximo el trabajo en grupo. Tanto los jefes de proyecto, los clientes y desarrolladores, son parte del equipo y están involucrados en el desarrollo del software.

Este modo de desarrollo colaborativo y abierto es una de las premisas indicadas por Eric Raymond en su libro “La catedral y el bazar” [RAY00] para el desarrollo de software de código abierto por lo cual XP y las metodologías ágiles en general han tenido gran aceptación en el mundo de los desarrolladores libres.

Problemas del desarrollo de software

En la actualidad existen varios problemas que afectan el desarrollo de software.

- Retrasos en la planificación: llegada la fecha de entregar el software éste no esta disponible.
- Sistemas deteriorados: el software se ha creado pero después de un par de año el coste de su mantenimiento es tan complicado que definitivamente se abandona su producción.
- Tasa de defectos: el software se pone en producción pero los defectos son tantos que nadie lo usa.
- Requisitos mal comprendidos: el software no resuelve los requisitos planificados inicialmente.
- Cambios de negocio: el problema que resolvía nuestro software ha cambiado y nuestro software no se ha adaptado.
- Falsa riqueza: el software hace muchas cosas técnicamente muy interesantes y divertidas, pero no resuelven el problema del cliente.

- Cambios de personal: después de unos años de trabajo los programadores comienzan a abandonar el proyecto

Con XP se intenta evitar el surgimiento de cualquiera de esos problemas en nuestro proyecto.

Metodología de trabajo según XP

Los puntos fundamentales en la metodología de XP están centrados acelerar el proceso de desarrollo manteniendo una relación aceptable con la calidad y el costo del proyecto.

Planificación:

XP plantea la planificación como un permanente dialogo entre las partes la empresarial (deseable) y la técnica (posible). Las personas del negocio necesitan determinar:

- **Ámbito**: ¿ Qué es lo que el software debe de resolver para que este genere valor ?
- **Prioridad**: ¿ Qué debe ser hecho en primer lugar ?
- **Composición de versiones**: ¿ Cuánto es necesario hacer para saber si el negocio va mejor con software que sin el ?. En cuanto el software aporte algo al negocio se debe de tener lista las primeras versiones.
- **Fechas de versiones**: ¿ Cuáles son las fechas en la presencia del software o parte del mismo pudiese marcar la diferencia ?

El personal del negocio no puede tomar en vació estas decisiones, y el personal técnico tomará las decisiones técnicas que proporcionan la metería prima para las decisiones del negocio. Esto conlleva a :

- **Estimaciones**: ¿ Cuanto tiempo lleva implementar una característica ?
- **Consecuencias**: Informar sobre las consecuencias de la toma de decisiones por parte del negocio. Por ejemplo el cambiar las bases de datos a Oracle.

- **Procesos**: ¿ Cómo se organiza el trabajo y el equipo ?
- **Programación detallada**: Dentro de una versión ¿ Qué problemas se resolverán primero ?

Cada versión debe de ser tan pequeña como fuera posible, conteniendo los requisitos de negocios más importantes, las versiones tiene que tener sentido como un todo.

Es mucho mejor planificar para 1 mes o 2 que para seis meses y un año, las compañías que entregan software muy voluminoso no son capaces de hacerlo con mucha frecuencia.

Diseño:

- **Diseño sencillo.**

El diseño adecuado par el software es aquel que:

1. Funciona con todas las pruebas.
2. No tiene lógica duplicada.
3. Manifiesta cada intención importante para los programadores
4. Tiene el menor número de clases y métodos.

El diseño se debe hacer lo más simple posible sin y manteniendo las reglas anteriores.

Desarrollo:.

- **Recodificación**.

Al implementar nuevas características en un producto aparece la necesidad de hacerlo del modo más simple posible. Esto conlleva a un proceso de análisis con el objetivo de simplificar el software sin perder funcionalidad este proceso se le denomina recodificar o refactorizar (refactoring). Esto a veces puede llevar a hacer mas trabajo del necesario, pero a la vez se estará preparando el sistema para que en un futuro acepte nuevos cambios y pueda albergar nuevas características.

- **Programación por parejas.**

Todo el código de producción lo escriben dos personas frente al ordenador, cada miembro de la

pareja juega un papel activo en todo momento. Esto es especialmente ventajoso cuando uno de los miembros se esta iniciando en un área de trabajo dominada por el otro.

- **Propiedad colectiva.**

Cualquiera que crea que puede aportar valor al código en cualquier parcela puede hacerlo, ningún miembro del equipo es propietario del código. Si alguien quiere hacer cambios en el código puede hacerlo. Si hacemos el código propietario,creamos dependencia del autor y esto se aleja a la totalidad del equipo de la comprensión del problema. XP propone un propiedad colectiva sobre el código nadie conoce cada parte igual de bien pero todos conoce algo sobre cada parte, esto nos preparará para la sustitución no traumática de cada miembro del equipo.

Este punto determina además el uso de software para desarrollo colaborativo (herramientas de control de versiones, editores colaborativos, etc)

- **Integración continúa.**

El código se debe integrar periódicamente (algunos plantean incluso que una vez al día), y realizar las pruebas sobre la totalidad del sistema. Esto puede ser delegado a un equipo especial de programadores o asignado siguiendo la técnica de las parejas

- **Limite de horas semanales.**

Esta regla plantea que la cantidad de horas semanales dedicadas a la implementación no deben exceder las 40, es decir, que el aumento de horas extras (trabajo los fines de semana, etc) denota fallas en la planificación inicial del trabajo.

- **Presencia del cliente.**

Un cliente real debe sentarse con el equipo de programadores, estar disponible para responder a sus preguntas, resolver discusiones y fijar las prioridades. Esto garantiza que cualquier duda sobre el negocio será solucionada de inmediato y minimiza la posibilidad de que aparezcan cambios de última hora.

- **Estándares de codificación.**

Debido a lo abierto del modelo y la posibilidad de cambios en los roles del equipo es necesario

establecer guías y patrones en cuanto al código y la documentación adjunta a este. Esto garantiza la comprensión por todo el equipo de cualquiera de los módulos y elimina los errores de interpretación debido a estilos de programación distintos.

Pruebas:

- **Carácter obligatorio de las pruebas.**

No debe existir ninguna característica en el programa que no haya sido probada, los programadores escriben pruebas para chequear el correcto funcionamiento del programa, los clientes realizan pruebas funcionales. El resultado un programa mas seguro que conforme pasa el tiempo es capaz de aceptar nuevos cambios. XP favorece el modelo de pruebas unitarias desarrollado e integración continua desarrollado por Beck y Fowler en [PEX00]

Gentoo Linux

Este capítulo trata acerca de las características especificas de Gentoo Linux como metadistribución .

Gentoo Linux

Gentoo Linux es un producto que gana adeptos aceleradamente en todo el mundo. Es una distribución especial que aunque no esta pensada para usuarios noveles puede adaptarse a las necesidades específicas de cualquiera.

Gracias a una tecnología denominada Portage, Gentoo Linux puede convertirse en un servidor seguro ideal, estación de trabajo para desarrollo, escritorio profesional, sistema para juegos, solución encastrada o para cualquier otro uso -- lo que usted necesite que sea. Dada su casi ilimitada adaptabilidad, denominamos a Gentoo Linux como una **meta**distribución.

Como plantea su contrato social: “Gentoo, por sí mismo, es una colección de conocimientos libre. Conocimiento en este contexto puede ser definido como documentación y metadatos relacionados con conceptos o dominios relevantes para sistemas operativos y sus componentes, como en Software Libre contribuido por varios desarrolladores del proyecto Gentoo. El sistema operativo Gentoo, se deriva del conocimiento como concepto base descrito anteriormente. Un sistema operativo Gentoo debe satisfacer el requisito de ser autosuficiente. En otras palabras, el sistema operativo debe poderse construir desde el principio con las herramientas mencionadas. Si un producto asociado con el proyecto oficial Gentoo no satisface estos requisitos, el producto no será cualificado como sistema operativo Gentoo. “

Gentoo está siendo desarrollado activamente. Toda la distribución utiliza un estilo de desarrollo de alta velocidad: los parches de los paquetes son rápidamente integrados en el árbol principal, la documentación es actualizada a diario, con frecuencia se añaden características a portage, las versiones se suceden rápidamente y se publican unas cuatro versiones oficiales al año.

Estructura de Gentoo Linux

El sistema Gentoo Linux lo conforman varias aplicaciones con usos específicos, por ejemplo:

- Portage, la herramienta de gestión de software de Gentoo Linux.
- Eselect, herramienta de configuración y administración de Gentoo Linux.
- Genkernel, compilador automático para el kernel.
- Gentoolkit, útiles de administración de Gentoo Linux.

Portage

Portage es probablemente la más importante innovación de Gentoo en la gestión de software. Con su gran flexibilidad y una gran cantidad de características es frecuentemente apreciado como la mejor herramienta de gestión de software disponible para Linux.

Comprender el funcionamiento de Portage es crucial para el desarrollo e incluso el trabajo diario con Gentoo Linux por lo cual fue necesario realizar un estudio acerca del mismo como parte de esta investigación dirigido hacia los aspectos mas importantes de su configuración y mantenimiento.

Estructura

Portage esta estructurado en diferentes archivos que conforman el perfil y las configuraciones locales. Existen además una serie de directorios que es necesario especificar para el cumplimiento de ciertas operaciones. A continuación se muestra un resumen de los más importantes.

Archivos de configuración del perfil:

Los archivos de perfil radican en el directorio */usr/portage/profiles* y definen las características generales de instalación para las distintas variantes de un sistema Gentoo cada uno de ellos esta organizado de la forma:

<nombre general>/<plataforma de hardware>/<numero de liberación>

Cada perfil incluye o hereda los siguientes archivos de configuración:

1. **make.defaults:** Incluye una lista de variables de configuración por defecto para el perfil, estas variables
2. **packages:** Listado de paquetes necesarios para recrear el perfil desde cero. Dichos paquetes no deben tener restricciones.
3. **packages.build:** Lista de paquetes(uno por linea) necesarios para construir un stage1, solo usable por desarrolladores de stages.
4. **packages.provided:** Lista de paquetes que portage asume que ya están instalados en el sistema. Portage solo intentará actualizar uno de estos paquetes si algún otro necesita actualizarlo.

   El formato de este archivo es el siguiente:

   - los comentarios comienzan con **#**
   - sólo un paquete por linea
   - no son permitidos operadores relacionales
   - el paquete debe incluir la versión

   Por ejemplo, si se mantiene una versión propia del kernel, versión 2.6.14, entonces se debe especificar al portage que "*sys-kernel/gentoo-sources-2.6.14*" ya está instalado:

   # este paquete está instalado

   sys-kernel/gentoo-sources-2.6.14

5. **use.defaults:** Este archivo contiene las relaciones entre valores de USE y paquetes que se utilizan para esa funcionalidad.

   El formato de este archivo es el siguiente:

   - los comentarios comienzan con **#**

- solo una bandera USE por línea, con una lista de paquetes que la proveen.

  Por ejemplo:

  # media-libs/libsdl activará "sdl"

  sdl media-libs/libsdl

  # activar tcltk sólo si están dev-lang/tcl y dev-lang/tk

  tcltk dev-lang/tcl dev-lang/tk

6. **use.mask:** Listado de valores para la variable USE bloqueadas para el perfil, esto es utilizado para bloquear USEs que son específicas para alguna arquitectura.

   El formato de este archivo es el siguiente:

   - los comentarios comienzan con ***#***
   - solo una bandera USE por línea

7. **virtuals:** Este archivo controla que paquetes pueden proveer un virtual. Por ejemplo, si un paquete necesita enviar un e-mail, necesitará virtual/mta. En caso de ausencia de alguno de los que proveen este virtual(como qmail, sendmail, postfix, etc.), portage leerá este archivo para ver que paquete instalar. En este caso, Gentoo utiliza net-mail/ssmtp como predeterminado(como se define en el archivo virtuals).

   El formato de este archivo es el siguiente:

   - los comentarios comienzan con ***#***
   - solo un virtual y un paquete por linea

   Por ejemplo:

   # usar net-mail/ssmtp como el mta predeterminado

   virtual/mta net-mail/ssmtp

   # usar app-dicts/aspell-en como el diccionario predeterminado

   virtual/aspell-dict app-dicts/aspell-en

8. **parent:** Camino relativo hacia el directorio padre del perfil.
9. **deprecated:** La existencia de este archivo marca a un profile como deprecated, el cual no continuará siendo soportado por Gentoo. La primera línea contiene el profile al cual deben cambiar los usuarios, y las demás son instrucciones de cómo deben realizar el cambio.

    Por ejemplo:

    default-linux/x86/2005.0

    # emerge -n '>=sys-apps/portage-2.0.51'

    # rm -f /etc/make.profile

    # ln -s /usr/portage/profiles/default-linux/x86/2005.0 /etc/make.profile

Para que el usuario pueda especificar paquetes, variables e USEs, sobreescribiendo las del perfil que utiliza, existe el directorio /etc/portage, el cual puede contener, entre otros, los siguientes archivos:

1. **packages.mask:** Listado de paquetes a bloquear. Útil cuando alguna versión de un paquete no funciona bien.

    El formato de este archivo es el siguiente:

    - los comentarios comienzan con **#**
    - sólo un paquete por linea

    Por ejemplo:

    Si del paquete de controladores de tarjetas NVIDIA solo funcionan bien las versiones anteriores a la 1.0.4496.

    # bloquear versión 1.0.4496 del controlador nvidia

    # y superiores

    >=media-video/nvidia-kernel-1.0.4496

    >=media-video/nvidia-glx-1.0.4496

2. **packages.unmask:** Listado de paquetes bloqueados y que aún así, se desea instalar.
3. **packages.keywords:** Listado de paquetes, que puede incluir versión o no, que aún siendo inestables se desea instalar.
4. **package.use:** Listado de banderas USE activadas o desactivadas para paquetes específicos.

   El formato de este archivo es el siguiente:

   - los comentarios comienzan con **#**
   - sólo un paquete por linea, seguido por las USEs separadas por espacios

   Por ejemplo:

   # activar docs para GTK 2.x

   =x11-libs/gtk+-2* doc

   # deshabilitar soporte de mysql a QT

   x11-libs/qt -mysql

5. **mirrors:** Cuando portage encuentra una URL de la forma ***mirror://***, busca en este archivo el mirror correspondiente. Si el mirror no es encontrado aquí, portage busca en el archivo /usr/portage/profiles/thirdpartymirrors. Este archivo puede contener mirrors de tipo “local”. Este lista de mirrors es revisada antes que la variable GENTOO_MIRRORS y es utilizada aún si el paquete tiene restricciones de tipo RESTRICT="nomirror".

   El formato de este archivo es el siguiente:

   - los comentarios comienzan con **#**
   - el tipo de mirror, seguido de una lista de equipos

   Por ejemplo:

   # mirror local privado usado por una compañía

   local ftp://192.168.0.3/mirrors/gentoo http://192.168.0.4/distfiles

 # personas en Japón que desean utilizar el mirror japonés

sourceforge http://keihanna.dl.sourceforge.net/sourceforge

 # personas en Taiwan que desean utilizar el mirror local

gnu ftp://ftp.nctu.edu.tw/UNIX/gnu/

6. **categories:** Contiene la lista de las categorías que puede contener el árbol del portage, el PORTDIR_OVERLAY, o el directorio de binarios. En este archivo, en el directorio /etc/portage se puede añadir categorías personales.

   El formato de este archivo es el siguiente:

   - cada línea contiene una categoría.

   Por ejemplo:

   app-hackers

   media-other

En la liberación 2006.0 de Gentoo se incluyen las siguientes opciones de perfil:

- **base:** Es un perfil abstracto que actúa como plantilla para el resto de los perfiles. Todo perfil valido debe “heredar” de este. No establece un tipo de kernel o biblioteca de C por defecto
- **default-darwin**: Es el perfil por defecto para las versiones de Gentoo Darwin, la versión del sistema basada en el kernel libre desarrollado por Apple © y compatible con el sistema propietario MacOSX © . Establece como kernel al Darwin y define las restricciones para construir aplicaciones.
- **default-linux:** Este es el perfil estándar de Gentoo Linux y describe todos los requerimientos para construir un sistema genérico basado en el kernel y las bibliotecas estándares de

Linux.

- **embeded:** Dedicado a sistemas empotrados o integrados.
- **hardened:** Utilizado para construir sistemas utilizando una versión especial de la biblioteca GNU para C conocida como hardened(endurecida o protegida) orientada a sistemas que necesitan requerimientos especiales de seguridad en el código.
- **selinux:** Provee una base para desarrollar sistemas que utilicen el conjunto de reglas y parches de seguridad del proyecto SELinux.
- **uclibc:** Dedicado a sistemas empotrados o integrados. Este perfil cambia la glibc estándar por uclibc.

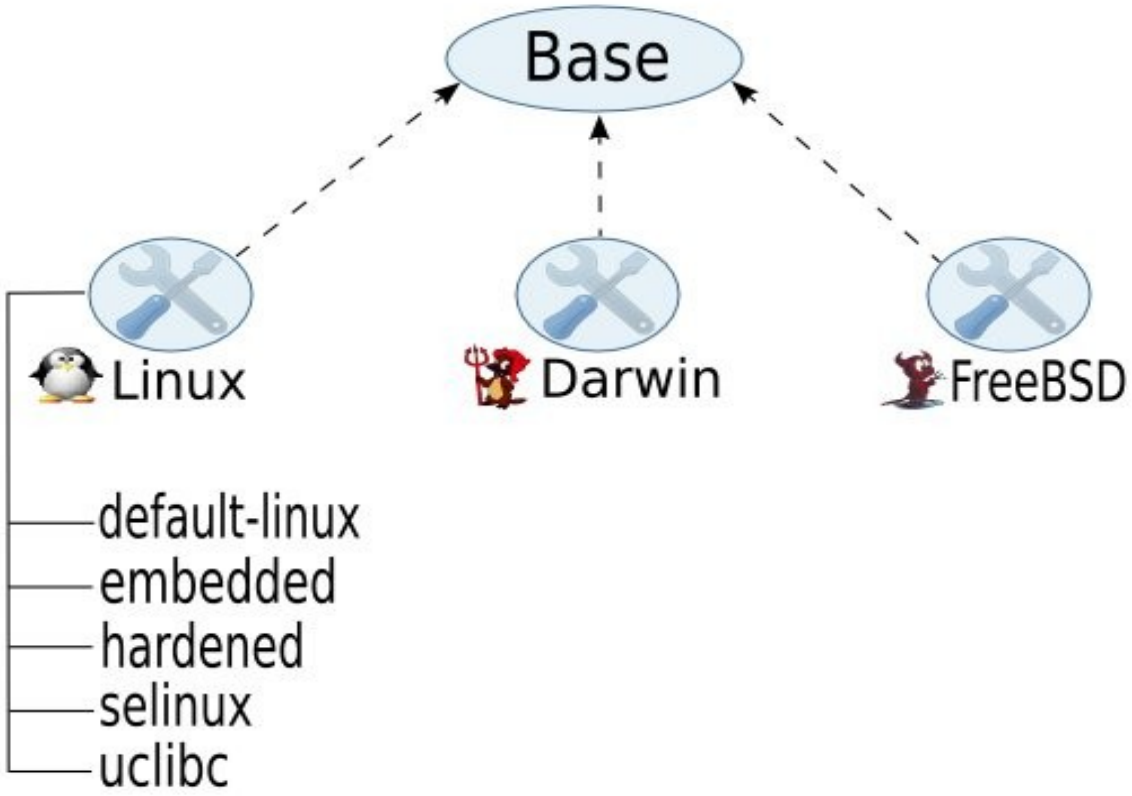

Ilustración 1: Sistema de perfiles del portage

El archivo make.conf

Este archivo contiene variables que permiten personalizar el comportamiento del Portage.

Algunas de las variables que pueden o no existir en el make.conf son:

- **ACCEPT_KEYWORDS =** [lista de palabras claves separadas por espacios]: define las palabras claves a admitir para los paquetes.

  por ejemplo:

  ACCEPT_KEYWORDS="x86"

- **AUTOCLEAN =** *[yes,no]*: borrar automáticamente los archivos desactualizados.
- **CFLAGS** y **CXXFLAGS**: incluyen las optimizaciones a pasar al compilador.

  por ejemplo:

  CFLAGS="-O2 -march=pentium4 -fomit-frame-pointer"

  CXXFLAGS="${CFLAGS}"

- **CHOST**: define la familia de arquitectura que se va a utilizar.

  por ejemplo:

  CHOST="i686-pc-linux-gnu" o ***CHOST="i386-pc-linux-gnu"***

- **PORTDIR =** [entrada]: directorio donde está ubicado el árbol del portage.
- **DISTDIR =** [entrada]: directorio donde se ubicarán los fuentes.
- **PKGDIR =** [entrada]: directorio donde se ubicarán los binarios.
- **FEATURES =** "[listado de acciones separadas por espacios]": define acciones al portage.
- **GENTOO_MIRRORS =** "[URLs separadas por espacios]": mirrors de Gentoo de los cuales se descargarán los fuentes de las aplicaciones;
- **MAKEOPTS**: define la cantidad de hilos de compilación que permite desarrollar al compilador, casi siempre es la cantidad de procesadores + 1.

por ejemplo:

MAKEOPTS="-j2"

- **PORTAGE_BINHOST =** "[URLs separadas por espacios]": listado de URLs a servidores de binarios.
- **PORTDIR_OVERLAY =** [entradas separadas por espacios]: directorios que contienen árboles de portage con ebuilds personalizados.
- **SYNC =** "[URL]": dirección al árbol de portage compartido por rsync.

  por ejemplo:

  SYNC="rsync://server/gentoo-portage"

- **USE =** "[valores separados por espacios]": listado de la valores de la variable USE, utilizados para definir el comportamiento del Portage para las aplicaciones y hardware a utilizar.

Eselect

Eselect es una herramienta de configuración y administración de Gentoo Linux. El eselect es un framework modular para herramientas de configuración.

Consiste en:

- Un programa principal llamado eselect.
- Varios módulos (archivos *.eselect) encargados de tareas específicas.
- Varias librerías que ayudan a tener un entorno cómodo y facilitan la creación de nuevos módulos.

Uso del comando

La sinopsis del comando es:

eselect <opciones globales> <módulos> <acciones> <opciones>

Opciones globales:

--no-color, --no-colour - deshabilita los colores en la salida.

help - Muestra la ayuda del comando.

list-modules - Lista los módulos disponibles.

usage - Muestra una ayuda mínima del comando.

version - Muestra la versión del eselect.

Módulos más utilizados:

bashcomp - Administra el autocompletamiento de comandos.

binutils - Administra las versiones de sys-devel/binutils.

blas - Administra las implementaciones de BLAS instaladas.

env - Administra las variables de entorno incluidas en /etc/env.d/

kernel - Administra el vinculo /usr/src/linux a las versiones del kernel instaladas.

lapack - Administra las implementaciones LAPACK instaladas.

mailer - Administrar los archivos de control de distintos servidores de correo /etc/mail.

profile - Administrar el profile utilizado cambiando el vinculo dinámico /etc/make.profile.

rc - Administra los scripts de los servicios y los runlevels.

Acciones:

- help - Muestra la ayuda del módulo.
- usage - Muestra de cómo utilizar el módulo.
- version - Muestra la versión del módulo y .otra información importante.
- list - Muestra las opciones disponibles.
- show - Muestra la(s) configuración(es) activas.
- set - Selecciona una de las opciones mostradas con *list*.
- enable - Activa una de las opciones del módulo.
- disable - Desactiva una de las opciones del módulo.
- update - Selecciona una de las opciones mostradas con *list*, pero automáticamente selecciona opciones en dependencia de los parámetros.
- scan - Muestra información acerca del sistema y la almacena para uso posterior.

Genkernel

Genkernel está diseñado para permitir a los usuarios, que no suelen compilar un núcleo propio, utilizar una configuración similar a la del CD de Instalación Gentoo que auto-detecta su hardware. El genkernel compila automáticamente los módulos y el núcleo, los empaqueta y pone en /boot,

permitiendo que usuarios inexpertos construyan el núcleo de su sistema.

Parámetros de Configuración

- **--menuconfig**: Activa el comando make menuconfig (que invoca a la utilidad de configuración interactiva del núcleo basada en menús de pantalla) antes de compilar el núcleo.
- **--gconfig**: Provee la utilidad de configuración del núcleo que depende de las bibliotecas GTK+-. La ventaja de esta opción es que la mayoría de los usuarios encuentran que usar esta herramienta hace más fácil y clara la configuración del núcleo ya que se basa en el sistema X-Window. Su desventaja es que **necesita** el sistema X-Window para usarla, así que no funcionará en la línea de comandos.
- **--xconfig**: Provee la utilidad de configuración del núcleo que depende de las bibliotecas QT. La ventaja de esta opción es que la mayoría de los usuarios encuentran que usar esta herramienta hace más fácil y clara la configuración del núcleo ya que se base en el sistema X-Window. Su desventaja es que **necesita** el sistema X-Window para usarla, así que no funcionará en la línea de comandos.
- **--save-config**: Guarda el archivo de configuración del núcleo a un archivo en el directorio /etc/kernels/ para uso posterior.
- **--no-save-config**: No guarda el archivo de configuración del núcleo.

Parámetros de Compilación

- **--kerneldir=/ruta/a/las/fuentes/**: Especifica una ubicación alternativa a las fuentes del núcleo en vez de la ubicación por defecto */usr/src/linux/*.
- **--kernel-config=/ruta/al/archivo/de/configuración**: Especifica qué archivo alternativo de configuración del núcleo será usado en vez del archivo por defecto */ruta/a/las/fuentes/.config*.

- **--module-prefix=/ruta/al/directorio/de/prefijo**: Especifica un prefijo al directorio donde serán instalados los módulos del núcleo (la ruta por defecto es el directorio */lib/modules/*).
- **--clean**: Activa el comando *make clean* antes de compilar su núcleo. El comando *make clean* elimina todos los archivos objeto y dependencias del árbol de fuentes del núcleo.
- **--no-clean**: Desactiva el comando *make clean* antes de compilar su núcleo.
- **--mrproper**: Activa el comando *make mrproper* antes de compilar su núcleo. Tal como el comando *make clean*, *make mrproper* elimina todos los archivos objeto y dependencias del árbol de fuentes del núcleo. Sin embargo, los archivos de configuración anteriores (en */ruta/a/las/fuentes/.config* o */ruta/a/las/fuentes/config.old*) **también** serán borrados del árbol de fuentes del núcleo.
- **--no-mrproper**: Desactiva el comando *make mrproper* antes de compilar su núcleo.
- **--oldconfig**: Ejecuta el comando *make oldconfig*, el cual intenta reunir información de configuración de la arquitectura del sistema a partir del script genérico ubicado en */usr/share/genkernel/*. Este es un proceso no-interactivo; no se requiere el usuario dé información. También, si se usa *--oldconfig* junto con *--clean*, este último parámetro es negado lo que resulta en la activación del parámetro *--no-clean*.
- **--callback="echo Hola"**: Realiza una llamada a los argumentos especificados (*echo Hola*, en este caso) luego de que se hayan construido el núcleo y los módulos relevantes, pero antes de construir la imagen de initrd. Esto puede ser útil si quiere instalar módulos externos en la imagen initrd invocándolos a través de la llamada *--callback=* y redefiniendo el grupo de módulos de genkernel.
- **--install**: Activa el comando make install, que instala la nueva imagen del núcleo, el archivo de configuración, la imagen initrd y el mapa de símbolos del sistema en la partición *boot* ya montada. Así mismo se instalarán los módulos compilados.
- **--no-install**: Desactiva el comando make install.
- **--no-initrdmodules**: Esta opción no copia ningún módulo al initrd creado por genkernel.

- **--genzimage**: Crea la imagen initrd previo a la imagen del núcleo. (Este hack actualmente sólo es válido en los sistemas Pegasos PPC).

Parámetros del Compilador

Genkernel soporta los siguientes parámetros que pueden ser pasados a aplicaciones relevantes mientras está siendo ensamblado el núcleo. Dichos parámetros hacen efecto sobre el compilador usado para el proceso de compilación del núcleo aunque a un nivel mucho más bajo.

- **--kernel-cc=algúnCompilador**: Especifica qué compilador utilizar para compilar el núcleo.
- **--kernel-ld=algúnEnlazador**: Define el enlazador que debe ser utilizado durante el proceso de compilación del núcleo.
- **--kernel-as=algúnEnsamblador**: Especifica qué ensamblador utilizar para ensamblar el núcleo.
- **--kernel-make=algúnMake**: Define una alternativa para GNU Make a utilizar durante la compilación del núcleo.
- **--utils-cc=algúnCompilador**: Especifica que compilador utilizar para compilar las utilidades auxiliares.
- **--utils-ld=algúnEnlazador**: Define el enlazador que debe ser utilizado durante el proceso de compilación de las utilidades auxiliares.
- **--utils-as=algúnEnsamblador**: Especifica qué ensamblador utilizar para ensamblar las utilidades auxiliares.
- **--utils-make=algúnMake**: Define una alternativa para GNU Make a usar durante la compilación de las utilidades auxiliares.
- **--makeopts=-jX**: Define el número de hebras concurrentes que puede implementar la utilidad *make* mientras está siendo compilado el núcleo (y sus utilitarios). La variable **'X'** es un número que se obtiene al sumar uno (1) al número de procesadores usados por el sistema. Así, para un sistema que tenga una CPU, el parámetro apropiado es -j2; un sistema

con 2 procesadores usará el parámetro -j3 y así sucesivamente. (Un sistema con un sólo procesador que soporte la tecnología Hyper-Threading™ (HT) puede usar el parámetro -j3 toda vez que se haya activado el soporte de Procesamiento Múltiple Simétrico (SMP) en el núcleo).

Parámetros de Depuración

El uso de parámetros de depuración durante el proceso de compilación del núcleo controla la cantidad de información que se reporta, así como la presentación de dicha información.

- **--debuglevel=NivelDeVerbosidad**: Controla el nivel de verbosidad de la información que entrega genkernel. La variable NivelDeVerbosidad es un entero cuyo valor está entre 0 y 5. El nivel '0' representa la mínima cantidad de información de salida mientras que '5' entrega tanta información como es posible acerca de las actividades de genkernel durante el proceso de compilación del núcleo.
- **--debugfile=/ruta/al/archivo/de/salida**: Ignora el valor ajustado por el parámetro *--debuglevel* y envía **todos** los datos de depuración generados por genkernel al archivo de salida especificado, el cual se ubica por defecto en */var/log/genkernel.log*.
- **--color**: Activa la salida en colores de la información de depuración (generada por genkernel) usando secuencias de escape.
- **--no-color**: Desactiva la salida en colores de la información de depuración (generada por genkernel) usando secuencias de escape.

Parámetros de Inicialización

Algunos de estos parámetros son meramente para efectos de estética mientras que otros pueden ser esenciales para la activación de algunas características en el sistema.

- **--bootsplash**: Activa el soporte para *bootsplash* en el initrd que genkernel va a construir. Esta característica está soportada en un número limitado de arquitecturas y se requiere de

un núcleo que también le dé soporte.

- **--no-bootsplash**: Desactiva] el soporte para *bootsplash*.
- **--no-gensplash**: Activa el soporte para *gensplash* en el initrd que genkernel va a construir. Para anular el tema por defecto usado por *gensplash* use el parámetro ***--gensplash=TemaPreferido*** (donde *TemaPreferido* es el título de uno de los directorios ubicados dentro del directorio */etc/splash/*.
- **--no-gensplash**: Desactiva el soporte para *gensplash* en el initrd que genkernel va a construir.
- **--gensplash-res=ResoluciónPreferida**: Este parámetro le permite seleccionar las resoluciones de pantalla de arranque soportadas en el initrd durante el inicio del sistema.
- **--do-keymap-auto**: Fuerza la selección del mapa de teclado durante la secuencia de arranque.
- **--lvm2**: Incluye soporte de almacenamiento vía LVM2 a partir de binarios estáticos, si están disponibles en el sistema. Los binarios (estáticos) de LVM2 se compilan en caso de no estar presentes.
- **--evms2**: Incluye suporte de almacenamiento usando el EVMS2, si está disponible.
- **--dmraid**: Incluye suporte para DMRAID; la utilidad que crea mapeos RAID usando el subsistema de mapeo de dispositivos del núcleo. DMRAID descubre, activa, desactiva y muestra las propiedades de conjuntos de software RAID (por ejemplo, ATARAID) y particiones DOS contenidas.
- **--linuxrc=/ruta/a/su/linuxrc**: Especifica la ruta al archivo creado por el usuario y denominado *linuxrc* — un script inicializado durante la etapa de inicio del núcleo previo al proceso de arranque. (La ubicación por defecto del script *linuxrc* debería ser el directorio */usr/share/genkernel/*) Este script le permite iniciar el arranque de un pequeño y modular núcleo y cargar los drivers necesarios (como módulos) por el sistema.
- **--cachedir=/ruta/a/un/directorio/alternativo/**: Anula la ubicación por defecto del caché

usado mientras se compila el núcleo.

- **--tempdir=/ruta/al/nuevo/directorio/temporal**: Especifica la ubicación del directorio temporal usado por genkernel mientras compila el núcleo.
- **--unionfs**: Incluye soporte de unionfs en la imagen del initrd.

Parámetros misceláneos

- **-mountboot**: Detecta si el directorio /boot/ necesita o no ser montado en una partición separada. Revisa el archivo */etc/fstab* para ver cómo montar la partición boot en un sistema de archivos (si es necesario).
- **--kernname=Sobrenombre**: Le permite modificar el nombre del núcleo y las imágenes initrd en el directorio */boot/*, de manera que las imágenes producidas por el núcleo sean kernel-Sobrenombre-versión e initramfs-Sobrenombre-versión.

Acciones posibles

Una acción le dice a genkernel qué construir. Actualmente, se reconocen las siguientes acciones:

- **initrd:** Construye solamente la imagen initrd
- **bzImage:** Construye solamente la imagen del núcleo
- **kernel:** Construye solamente la imagen del núcleo y los módulos
- **all:** Construye todas las etapas — initrd, imagen del núcleo y módulos

La última acción, all, es la que se recomienda para la mayoría de los usuarios pues construye las etapas requeridas para tener un núcleo funcional.

Gentoolkit

El paquete gentoolkit incluye herramientas de administración de Gentoo, entre ellas están: eclean,

equery, euse, revdep-rebuild y glsa-check .

Eclean

Eclean es una herramienta para la limpieza de los fuentes y los binarios de Gentoo. Permite que los directorios de fuentes y binarios no crezcan infinitamente, manteniendo los archivos que aún puedan hacer falta.

Opciones:

-C, --nocolor, desactivar la salida en colores.

-d, --destructive, solo mantener el mínimo para una reinstalación.

-e, --exclude-file=<path>, excluye los archivos listados ***packages*** y ***distfiles***, ubicados en ***path***.

-i, --interactive, pedir confirmación antes de borrar los archivos.

-n, --package-names, proteger todas las versiones, solo si se utiliza --destructive

-p, --pretend, solo mostrar los archivos a borrar

-q, --quiet, no mostrar ninguna salida.

-t, --time-limit=<time>, no borrar los archivos que han sido modificados después de ***time***.

-h, --help, mostrar la ayuda.

-V, --version, mostrar la versión de eclean.

Equery

Equery es una utilidad muy flexible que permite obtener información sobre los paquetes, tal como los archivos que contiene, las USE con que se compiló, el md5 de cada uno de los archivos que

pertenecen al paquete especificado, entre otras.

Sinopsis:

equery <opciones-globales> command <opciones-locales>

*donde **<opciones-globales>** es uno de*

-q, --quiet - muestra la salida mínima.

-C, --nocolor – desactiva los colores en la salida.

-h, --help - muestra la ayuda.

-V, --version - muestra información sobre la versión.

-N, --no-pipe - desactiva la detección de pipes.

*donde **command** es uno de*

belongs(b) <opciones-locales> archivos... - muestra los paquetes a los que pertenece el archivo

check(k) paquete - chequea el MD5 y tiempo de los archivos pertenecientes al paquete.

depends(d) <opciones-locales> paquete – lista todas los paquetes que dependen del especificado.

depgraph(g) <opciones-locales> paquete – muestra el árbol de dependencias del paquete especificado.

files(f) <opciones-locales> paquete – muestra los archivos pertenecientes al paquete .

hasuse(h) <opciones-locales> useflag – muestra los paquetes que tienen la USE especificada.

list(l) <opciones-locales> paquete – lista los paquetes que coinciden con el especificado.

size(s) <opciones-locales> paquete – muestra el tamaño del paquete especificado.

uses(u) <opciones-locales> paquete – muestra las USE con que se compiló el paquete.

which(w) paquete – muestra la dirección completa del ebuild del paquete.

Revdep-rebuild

revdep-rebuild revisa las librerías y binarios del sistema en busca de inconsistencia entre sus dependencias.

-X | --package-names instala la última versión, no la que estaba instalada

--library=NAME solo revisa los paquetes que dependen de la librería ***NAME***.

NAME puede ser el nombre de la librería o una expresión regular básica

-nc | --no-color desactiva los colores en la salida.

-i | --ignore ignora los archivos temporales anteriores

-q | --quiet muestra la salida mínima.

-vv | --extra-verbose mostrar mas información.

Además de estas, también se pueden utilizar las opciones de emerge.

Euse

El comando euse permite editar la variable ***USE*** en el *make.conf*, obtener información detallada de algunos de sus valores, el status, etc.

-E, --enable Activa valores de la variable ***USE*** en el make.conf. Acepta uno o varios valores separados por espacios.

-D, --disable Desactiva valores de la variable ***USE*** en el make.conf. Acepta uno o varios valores separados por espacios.

-P, --prune Elimina todos los valores de la variable ***USE***, activos o no, en el make.conf.

-i, --info Muestra información detallada sobre el valor USE especificado.

La salida tiene el siguiente formato:

[- cD] alpha – indica la arquitectura ...

[-] moznocompose (net-www/mozilla):.br Desabilita la construcción con mozilla composer

Los indicadores en la primera columna son:

- **+** si está activada, **-** si no lo está.
- **E** si está activada en el entorno, **e** si no está activada en el entorno, o nada si no está afectada por el entorno.
- **C** si está activada en el make.conf, **c** si no está activada en el make.conf o nada si no está afectada por el make.conf.
- **D** si está activada en make.defaults, **d** si no está activada en el make.defaults, o nada si no está afectada por el make.defaults.
- **G** si está activada en make.globals, **g** si no está activada en make.globals, o nada si no está afectada por el make.globals.

-a, --active muestra todas las USEs que están activas.

-h, --help muestra una ayuda, listando las opciones para USE.

-v, --version muestra información sobre la versión.

glsa-check

GLSA(*Gentoo Linux Security Advises*) es la solución de Gentoo Linux a las actualizaciones de

seguridad**.** Esta solución consiste en un script de python(***glsa-check***) el cual revisa entre varios archivos xml que se encuentran en el directorio metadata/glsa en el árbol del portage.

Cada uno de los archivos contiene la información de los problemas de seguridad que han ido apareciendo, y el nombre esta formado por un id de glsa y la extensión .xml, por ejemplo:

```
<?xml version="1.0" encoding="utf-8"?>
<?xml-stylesheet href="/xsl/glsa.xsl" type="text/xsl"?>
<?xml-stylesheet href="/xsl/guide.xsl" type="text/xsl"?>
<!DOCTYPE glsa SYSTEM "http://www.gentoo.org/dtd/glsa.dtd">

<glsa id="200411-24">
  <title>BNC: Buffer overflow vulnerability</title>
  <synopsis>
    BNC contains a buffer overflow vulnerability that may lead to Denial of
Service and execution of arbitrary code.
  </synopsis>
  <product type="ebuild">BNC</product>
  <announced>November 16, 2004</announced>
  <revised>November 16, 2004: 01</revised>
  <bug>70674</bug>
  <access>remote</access>
  <affected>
    <package name="net-irc/bnc" auto="yes" arch="*">
      <unaffected range="ge">2.9.1</unaffected>
```

<vulnerable range="lt">2.9.1</vulnerable>

</package>

</affected>

<background>

BNC (BouNCe) is an IRC proxy server.

</background>

<description>

Leon Juranic discovered that BNC fails to do proper bounds checking when checking server response.

</description>

<impact type="high">

 An attacker could exploit this to cause a Denial of Service and potentially execute arbitary code with the permissions of the user running BNC.

</impact>

<workaround>

 There is no known workaround at this time.

</workaround>

<resolution>

 All BNC users should upgrade to the latest version:

```
<code>
# emerge --sync
# emerge --ask --oneshot --verbose ">=net-irc/bnc-2.9.1"</code>
</resolution>
<references>
<uri link="http://gotbnc.com/changes.html">BNC ChangeLog</uri>
<uri link="http://security.lss.hr/en/index.php?page=details&amp;ID=LSS-2004-11-03">LSS-2004-11-03</uri>
</references>
<metadata tag="requester" timestamp="Thu, 11 Nov 2004 20:17:39 +0000"> lewk
</metadata>
<metadata tag="submitter" timestamp="Thu, 11 Nov 2004 21:49:41 +0000"> jaervosz
</metadata>
<metadata tag="bugReady" timestamp="Fri, 12 Nov 2004 23:44:26 +0000"> jaervosz
</metadata>
</glsa>
```

Contiene el nombre, la fecha en que se anunció y se revisó, una breve descripción, el paquete afectado, qué versiones son afectadas y cuales no, el tipo de impacto y cómo se puede solucionar la vulnerabilidad.

Sinopsis del comando:

glsa-check <option> [glsa-list]

Las distintas opciones que tenemos son:

- -l,--list lista todas las GLSAs no aplicadas.
- -d, --dump, --print muestra toda la información sobre la GLSA dada.
- -t, --test prueba si el sistema es afectado por la GLSA dada.
- -p, --pretend muestra los el modo de corregir la vulnerabilidad.
- -f, --fix intenta corregir la GLSA (aún experimental).
- -i, --inject inyecta la GLSA en las aplicadas.
- -n, --nocolor deshabilita los colores en la salida.
- -h, --help muestra el mensaje de ayuda.
- -V, --version información sobre esta herramienta.
- -v, --verbose muestra mas mensajes, modo verboso.
- -c, --cve muestra los IDs de las GLSAs en modo de lista.

Aspectos del desarrollo de Nova LNX

Metas de desarrollo.

- Desarrollar una distribución de Linux adaptada a las necesidades de migración de servicios y aplicaciones de nuestra universidad.
- Proveer una estructura de desarrollo de versiones “a la medida” según varíen las necesidades originales de la migración.

Necesidad de una distribución local.

A pesar de la variedad de distribuciones de Linux en existencia, pocas de ellas implican un proceso real de desarrollo u optimización sino que consisten en el ensamblado de paquetes de una distribución base en forma de solución o muestra y la mayoría no permite optimizaciones o personalizaciones para entornos específicos o sistemas dedicados. Una distribución desarrollada en nuestra Universidad no solo podría ser orientada y optimizada según nuestras necesidades de migración y desarrollo de software sino que también proporcionaría una plataforma para la investigación.

Tecnologías.

En el desarrollo de la distribución se han utilizado tecnologías que encapsulan lo mejor de las técnicas de programación sobre Linux y en general en vías de asegurar la mayor estabilidad y portabilidad del código.

- **XML y XSLT** para la representación de datos y su transformación a diferentes formatos. Actúa

además como una plataforma intermedia entre los datos representados y los archivos de configuración.

- **Expresiones regulares** en combinación con algunas herramientas como sed o utilizando las bibliotecas desarrolladas por la GNU. Se utilizan principalmente en el manejo de fuentes de datos en texto plano.
- **FUSE** o sistema de archivos en espacio usuario, una combinación de módulos del kernel y bibliotecas que permiten definir programas capaces de simular el funcionamiento de sistemas de archivos convencionales con la ventaja de que esta disponible para usuarios sin privilegios de administración. Implementaciones comunes son GmailFS y FlickrFS, dos sistemas de archivos que se comunican con servicios Web.
- **device-mapper** es una nueva infraestructura del kernel de Linux, versión 2.6, que provee una vía de crear interfaces virtuales a dispositivos, las cuales pueden efectuar distintas acciones sobre los dispositivos reales, tales como concatenación, copia, cifrado, etc.
- **dm-crypt** es una vía para que el device-mapper realice una encripción transparente de algún dispositivo, utilizando la API de encripción del kernel. El usuario solo debe especificar cual de los algoritmos de cifrado simétrico se utilizará, una clave(de cualquier tamaño permitido), y una identificación, la cual será el nombre del dispositivo en /dev/mapper. La escritura y la lectura al dispositivo serán cifradas automáticamente..
- **LUKS** está surgiendo como un estándar de Linux para encripción de particiones. Al proveer un estándar de formato de disco, no solo facilita la compatibilidad entre distribuciones, sino también una administración segura de las claves de múltiples usuarios. En contraste con las otras soluciones existentes, LUKS almacena toda la información necesaria en el inicio de la partición, permitiendo al usuario transportar o mover de lugar sus datos sin problema alguno.
- **distcc** es un programa para distribuir la construcción de código C, C++, Objective C y Objective C++ a través de una red de computadoras. distcc está diseñado para generar el mismo resultado que el proceso de compilación clásico, es fácil de instalar y configurar y normalmente acelera el proceso de compilación de un 30 a un 40%. Los sistemas hospederos no necesitan compartir el

mismo filesystem, tener sincronización de tiempo o incluso las mismas versiones de las bibliotecas o archivos de encabezado siempre y cuando el compilador lo permita.

Arquitectura de Nova LNX.

Nova LNX hereda de Gentoo los componentes base de su arquitectura. El portage, el sistema de selección y la filosofía de desarrollo han sido asimilados y transformados en herramientas con una mejor integración con el escritorio creando front-ends que puedan ser manejados visualmente a través de GUI.

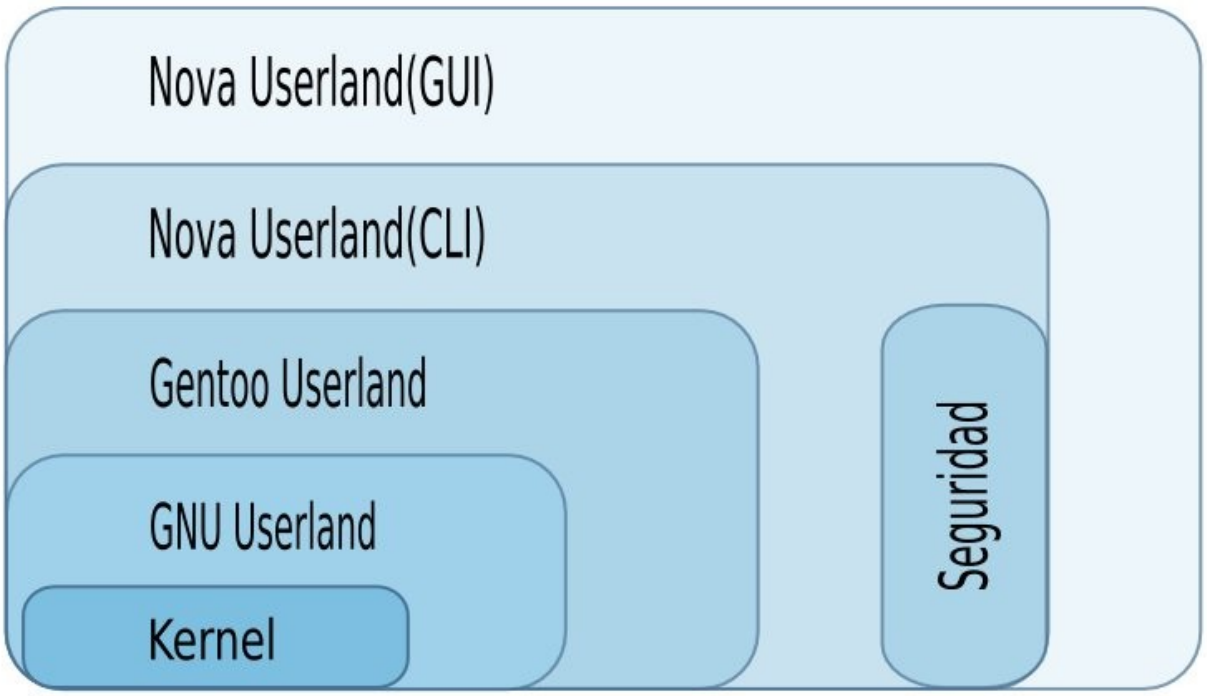

Ilustración 2: Esquema de arquitectura

Kernel.

El kernel y sus interfaces con el sistema son consideradas como la primera capa de componentes de Nova y actúa como base para varias de las optimizaciones y tecnologías presentes en el sistema especialmente el modelo de seguridad.

El hecho de que el kernel forme parte de la arquitectura crea una dependencia que lejos de ser dañina resulta provechosa para tareas de bechmarking y pruebas del sistema así como en el proceso de corrección de errores.

Base de usuario GNU.

El proyecto GNU incluye una serie de componentes estándar importados íntegramente en Nova LNX. Estos componentes descritos por el estándar POSIX son la base de toda distribución de Linux y definen en su conjunto

Base de usuario Gentoo.

Gentoo Linux incluye un conjunto de herramientas tratadas en el capítulo II para el manejo básico de paquetes y configuración. Debido a que son parte fundamental del sistema y a al hecho de que alterar el funcionamiento estándar de cualquiera de ellas llevaría a la incompatibilidad parcial o total con la distribución base. La base de usuario incluye además una serie de permisos y políticas de usuarios y grupos en los que se basa la seguridad al nivel más básico.

Nova CLI.

Esta es la capa responsable por interactuar directamente con cualquiera de los servicios proporcionado por cualquiera de las capas anteriores que no sea accesible fácilmente por el usuario inexperto. Consiste principalmente en scripts sustitutos o envoltorios para comandos ya existentes así como interfaces que actúen como intermediarias para algunas herramientas GUI. Esta capa incluye además los mecanismos necesarios para el control de privilegios de usuario basado en el estricto sistema de permisos que forma parte de Gentoo Linux.

Nova GUI.

Siendo la fuente de interacción principal con el usuario, esta capa actúa como front-end a varios procesos de configuración y mantenimiento del sistema. Entre las herramientas actualmente desarrolladas se encuentra el instalador del sistema, software para la administración de discos duros, instalador de aplicaciones, administrador de recursos compartidos.

Esquema de seguridad

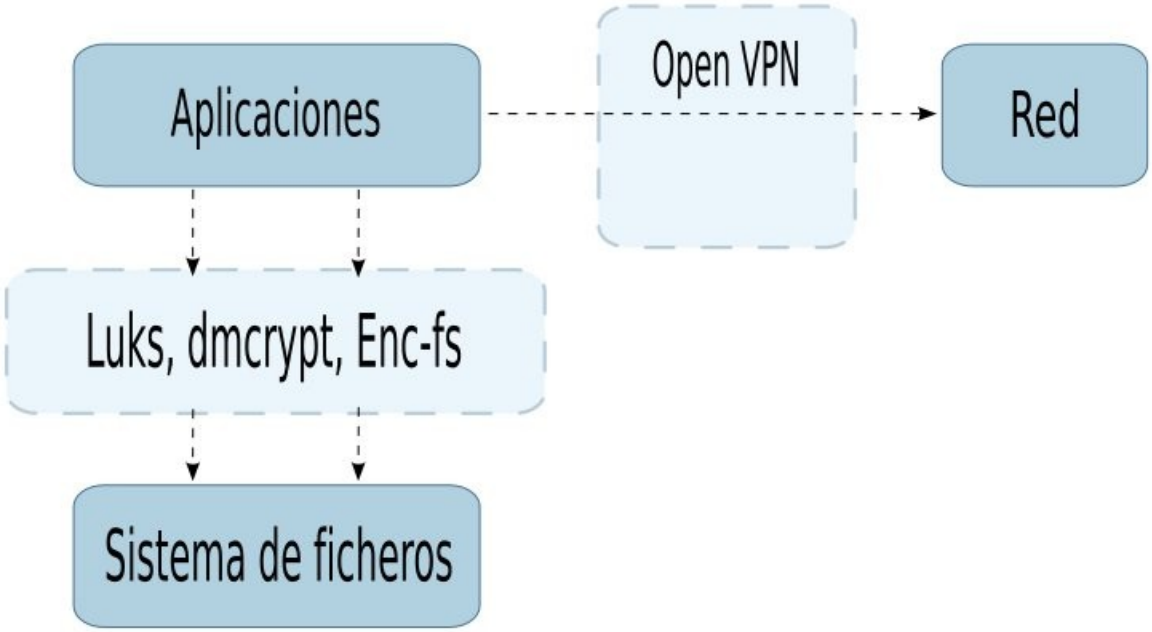

Ilustración 3: Esquema de seguridad

Protección de dispositivos:

La protección de datos en dispositivos tanto externos como fijos es una de las preocupaciones principales de la seguridad informática. Utilizando una implementación de LUKS y una tecnología para manejo dinámico de las rutinas de acceso a los dispositivos basada en device-mapper es posible codificar toda la información presente de forma que los datos necesarios para decodificarla se mantengan con el dispositivo convirtiéndolo en un medio seguro. Esta tecnología puede ser aplicada a cualquier dispositivo siempre y cuando exista una implementación de LUKS en el sistema

huésped.

Protección parcial de filesystem:

Con la utilización de la tecnología FUSE es posible crear interfaces virtuales entre el modelo VFS(Virtual Filesystem) del kernel y la representación real de datos en el dispositivo. Esta tecnología ha tenido bastante éxito proporcionando acceso a través del sistema de archivos a servicios como Gmail y Flickr. En el esquema de seguridad de seguridad propuesto en Nova, esta tecnología es utilizada para encriptar directorios o archivos específicos ya sea en un sistema de archivos encriptado o en texto plano.

Transmisión protegida de datos:

La transmisión de datos a través de la red es potencialmente insegura en entornos abiertos. Haciendo uso de la tecnología de redes privadas virtuales (OpenVPN) integrada en Nova LNX es posible asegurar la protección de estos datos estableciendo un canal de comunicación seguro entre dos o más estaciones de trabajo utilizando los dispositivos de túneles del kernel y las bibliotecas OpenSSL.

Desarrollo de utilidades.

La configuración y administración de un sistema Linux es usualmente un proceso complicado que implica el conocimiento acerca de varios archivos o procedimientos que normalmente escapan del interés del usuario común. El directorio */etc/* , donde radica los archivos de configuración de cada programa instalado en el sistema, cuenta con aproximadamente 2190 archivos en una sistema típico, de estos cerca de 2179 están relacionados con los parámetros de funcionamiento de algún programa. Estos archivos a pesar de estar en mayormente en texto plano, cuentan una gran variedad de formatos y reglas que en caso de violarse podrían alterar la estabilidad del sistema. En el caso de Gentoo Linux existen varios scripts que se ejecutan automáticamente al instalar un nuevo software que eliminan una pequeña parte de esta barrera, pero aún así el usuario tiene que recordar nombres y modos de invocación para cada uno.

La solución Nova LNX a este problema consiste en la introducción de software utilitario que englobe los aspectos de configuración, instalación de aplicaciones y mantenimiento del sistema, abstrayendo al usuario de la interacción con las CLI. Estas utilidades además se integran al modelo de seguridad de seguridad mediante el uso de Quantum, un framework escrito en python aún en proceso de desarrollo, y las facilidades de reuso de código que proporciona este lenguaje.

Actualmente se han desarrollado herramientas para la instalación del sistema, la configuración de carpetas compartidas utilizando el protocolo SMB y la instalación de nuevos paquetes. En el Anexo III se incluyen imágenes de dichas herramientas.

Documentación y ayuda.

La documentación, tanto técnica como de soporte, es parte del valor agregado de cualquier producto informático. En esa linea de pensamiento, la generación y mantenimiento de documentos, tutoriales y el sistema de ayuda es considerado como una parte más del desarrollo de distribuciones. Esta ayuda puede ser distribuida en formas que van desde videos especializados hasta simple página del manual del sistema.

En el proceso de construcción y mantenimiento de Nova LNX la documentación juega un papel fundamental, no sólo es el método de comunicar las posibilidades y especificaciones del producto a los usuarios sino que además actúa como un medio de comunicación entre desarrolladores. La documentación en Nova LNX esta dividida en dos grupos:

Documentación de Desarrollo.

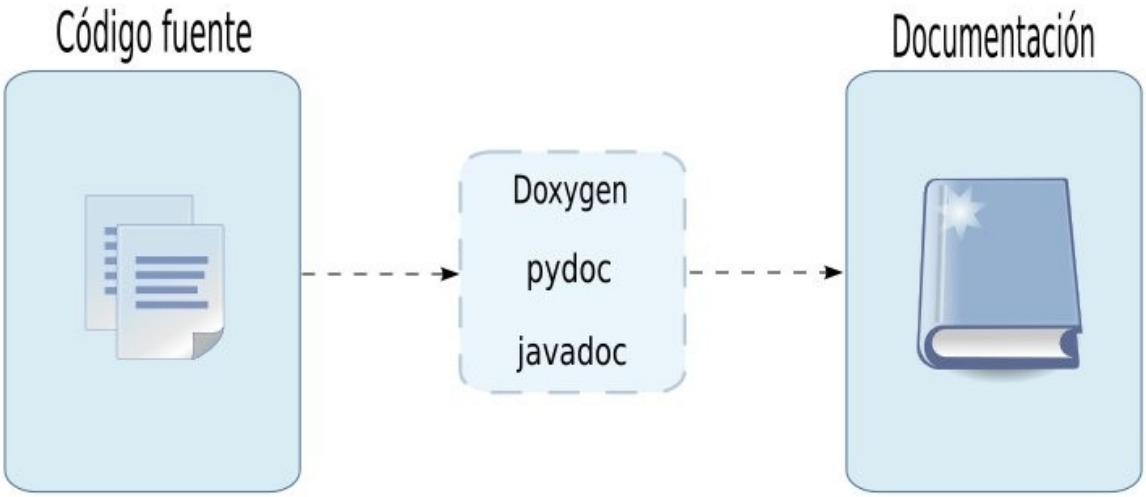

Ilustración 4: Generación de documentación para desarrolladores

La documentación técnica es fundamentalmente para el uso de los desarrolladores y se genera a partir diagramas UML o código fuente siendo esta última la vía en uso actual. Los comentarios del desarrollador del software son la materia prima de este proceso.

Herramientas utilizadas en la generación de documentos.

- **Javadoc:** Es el sistema estándar para crear documentación técnica en Java. La documentación es generada a partir del código fuente y comentarios especiales conocidos como "javadoc comments". La documentación es generada en formato HTML e incluye información acerca de paquetes, clases, métodos y relaciones de herencia.

- **Doxygen:** Doxygen es un sistema de documentación para C,C++,Java,Objective-C, Python e IDL. Sus funciones principales son:

    1. Generar documentos a partir de un grupo de archivos de código fuente comentados según las especificaciones de la herramienta. Los formatos soportados para la generación de documentos son: HTML,PDF,

LATEX,Postscript y el formato del manual de Unix. Al igual que con javadoc la documentación es extraída a partir de los fuentes lo cual ayuda a mantener la consistencia de esta.

2. Generar una representación de la estructura de código fuente que no esté documentado, lo cual resulta útil en casos donde no es conveniente comenta el código pero es necesario mantener un mapa documentado del mismo. Es posible además visualizar las relaciones entre elementos en el código a partir de gráficos de dependencia, herencia y colaboración creados automáticamente.

- **Pydoc:** Es un sistema parecido a Javadoc capaz de generar documentación a partir de los comentarios escritos en scripts de Python. A diferencia de las demás herramientas, no es necesario guardar el resultados del proceso debido a que el código fuente siempre va a estar disponible en el script. Pydoc incluye además un servidor Http ligero que permite acceder a la documentación del sistema o un módulo específico de este con la ayuda de un navegador como si se tratara de un sitio web.

Guías

Documentación de Usuario.

La documentación de usuario es la forma primaria de asistencia en el sistema y comprende un rango amplio de sistemas y formatos de documentos, los más distribuidos en sistemas Unix son las páginas del manual y el sistema textinfo, ambos proporcionados en Linux por las herramientas GNU.

Básicamente un sistema de ayuda consiste en herramientas de generación de hipertexto y un "lector"(help reader) o software dedicado a mostrar el contenido. Estos documentos deben presentar la ayuda en modo conciso y fácil de comprender por el usuario, sin asumir que existe conocimiento técnico previo, en caso de una herramienta o un procedimiento se deben incluir ejemplos de como utilizar los mismos . Aunque cualquier formato es posible, existe una tendencia a los sistemas de

hipertexto por la facilidades de presentación y navegación de los documentos.

GNOME incluye un sistema desarrollado por el GNOME Documentation Project (GDP) para presentar los documentos de ayuda al usuario, la forma de uso de este y sus especificaciones se detallan en las GNOME Documentation Guidelines[GDP00].

Ilustración 5: Sistema de ayuda al usuario

Este sistema consta de tres partes:

-

Releng(Release engineering)

El término *releng* (o release engineering) se refiere al proceso de creación de el producto o los productos finales de una distribución de Linux

Debido a las características del *releng* en Nova LNX de se hizo necesaria la creación de una herramienta capaz de ejecutar los procesos sin interacción humana. Aunque en la implementación de la misma no se practicó el concepto de programación orientada a objetos debido a limitaciones del lenguaje (scripts de shell) este fue aplicado en la etapa de diseño creando abstracciones modelan la funcionalidad de cada uno de los módulos de implementación.

Descripción del proceso *releng.*

El proceso de *releng* esta compuesto por un grupo de subprocesos encargados de una parte específica del producto. Cada proceso de *releng* puede ser lanzado sin intervención de un operador (Ejemplo: por una tarea del cron u otro proceso) y su resultado ,en caso de que no ocurran errores externos depende únicamente de la configuración del entorno y no de las características del sistema base.

Sandbox

Los sandbox o cajas de arena son mecanismos de protección usuales en procesos que pueden dañar el entorno donde se ejecutan. Un ejemplo conocido de sandbox es la que ejecuta la máquina virtual de Java (JVM o Java Virtual Machine). Aunque los modos de implementación pueden variar las premisas son similares en todos los modelos: Todo proceso, programa o sección que se ejecute dentro del sandbox esta restringida a usar solamente los recursos que se proporcionen en el mismo. Encaso de que existan intentos de acceso el sandbox puede ejecutar acciones que van desde la terminación del proceso o ignorar dicha petición (denegación de servicio) o la asignación virtual del recurso, aunque este último enfoque no siempre es posible.

Los sistemas operativos implementan diferentes modos de crear entornos protegidos que funcionan a modo de sandbox limitando acceso a directorios, memoria o dispositivos externos. Esto puede ser implementado como una serie de políticas, permisos y listas de control de acceso (ACL o Access Control List) sobre dichos recursos.

Linux incluye herramientas especializadas para el desarrollo de sandbox como chroot o jail que logran el objetivo de restringir a todo un grupo de procesos lanzados desde un mismo shell. Este modo de sandboxing no incluye protección de memoria sino que se limita a restringir el acceso a archivos.

La figura siguiente muestra el resultado de una llamada chroot que restringe a uno o varios procesos a un sistema de archivos virtual localizado en un directorio del sistema de archivo primario.

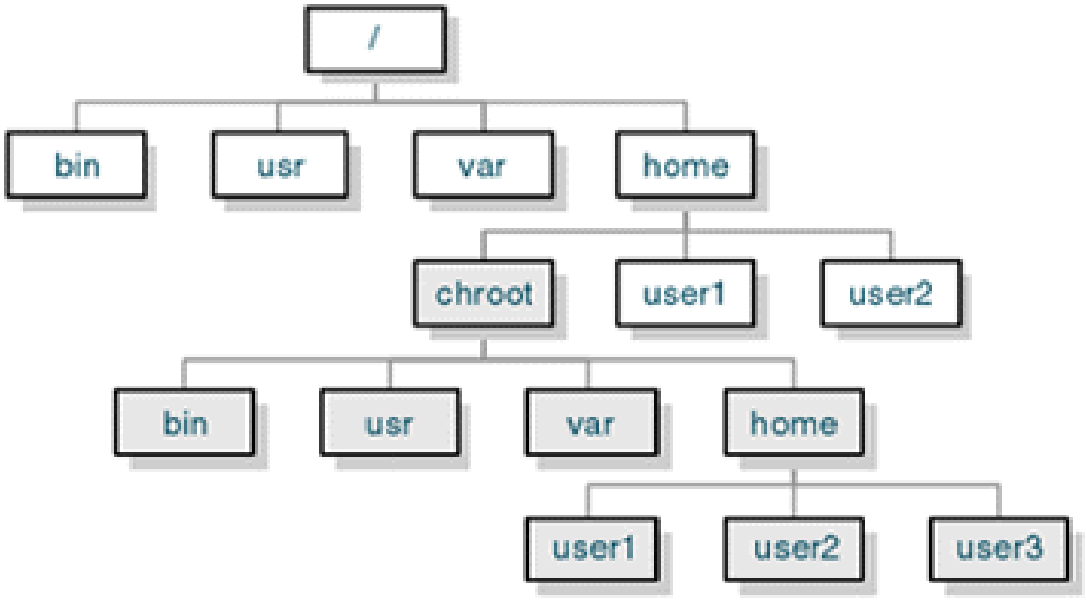

Ilustración 6: Sandbox

Objetivos:

- Inicializar un entorno protegido en el directorio seleccionado de modo que el resto del sistema de archivos quede protegido.

Requisitos:

- El proceso que inicialice el sandbox debe tener los permisos necesarios para lanzar la llamada chroot en el archivo
- El directorio donde se va a ejecutar el sandbox debe tener al menos un subsistema de archivos suficiente para ejecutar al shell como un proceso contenido.

Dependencias:

- Este estado depende de aplicaciones externas como jail o chroot

Componentes

- Ninguno

Funcionamiento:

Crea un sistema virtual donde es posible realizar procesos que pudieran dañar al filesystem

Etapa 4 (stage 4)

Al igual que el concepto de etapa en Gentoo la etapa cuatro define una etapa de instalación del sistema en este caso superior a la etapa 3 (stage3). Esta consiste en una imagen de sistema de archivos que contiene toda la base de Gentoo Linux y los cambios y configuraciones de userland para Nova LNX.

Este estado (como dice su nombre) define la creación de una etapa 4 funcional y es un estado interno al sistema de liberación al cual solo acceden otros estados.

Objetivos:

- El objetivo de este proceso es la automatizar la creación de un entorno base protegido y usable por los demás procesos. El cual contiene un sistema básico, auto configurable y que permita la posterior instalación se los paquetes que necesite el usuario.

Requisitos:

- Una imagen previa de stage3.
- Portage y configuración de perfiles
- Árbol y configuración de componentes
- Optimizaciones
- Preferencias locales (en caso de que el próximo estado lo necesite)
- Enlaces y puntos de montaje necesarios.
- Ejecución de un sandboxing.

Depende de:

- Sandbox.
- Compilador de C y entornos de ejecución de python y perl.
- Otras aplicaciones estándares del sistema (shell, editor sed, mount, etc)

Componentes

- *nova-base/components*
- *nova-base/layout*

Funcionamiento:

El estado utiliza el stage-3 de la instalación estándar de Gentoo para crear un entorno base en el cual se van a instalar o crear los componentes necesarios. Una vez posicionados los contenidos del stage-3 se procede a crear un sistema de ejecución virtual creando enlaces hacia los directorios /proc(imagen de procesos y configuración del kernel actual) y /dev(entradas virtuales de los dispositivos del sistema) Ese entorno queda protegido después de la inicialización de un sandbox para que en caso de que ocurra alguna alteración del proceso el sistema hospedero no quede dañado por la eliminación o escritura de archivos. Al ser creadas las condiciones se pasa a inicializar todos los archivos de configuración necesarios según las opciones definidas en el entorno y finalmente a la instalación y creación de los componentes base del estado.

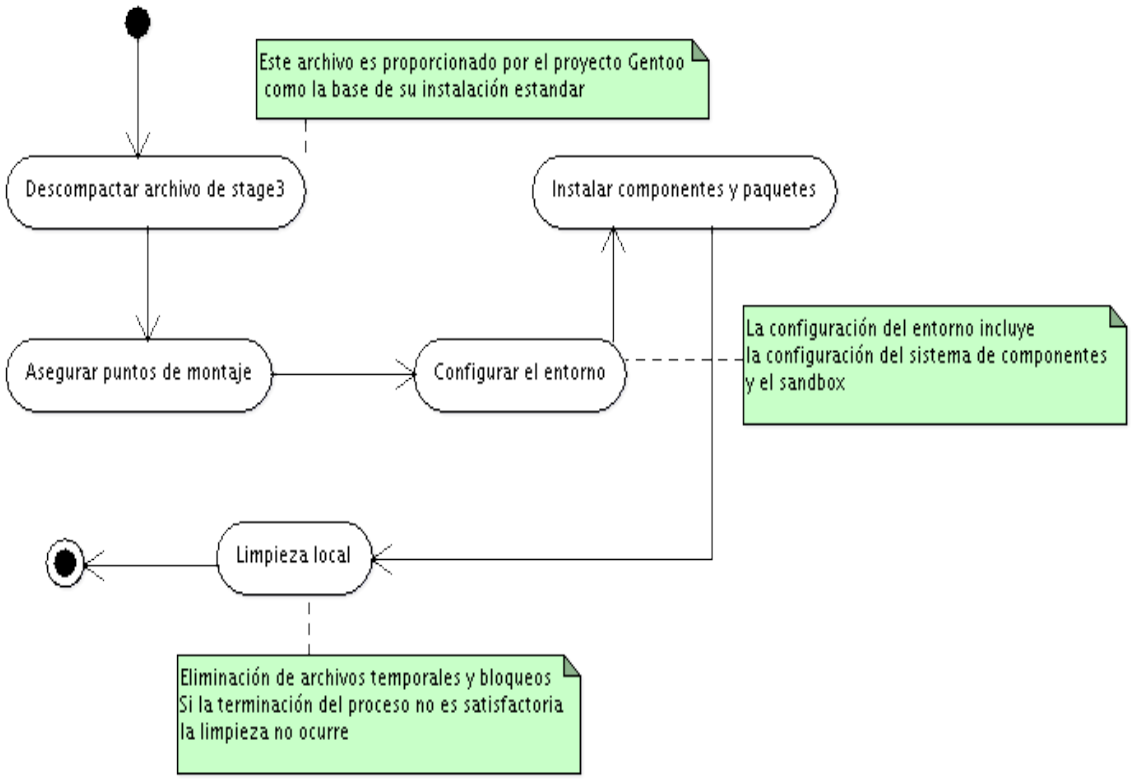

Ilustración 7: Diagrama de actividad para el proceso stage4

LiveCD

Un LiveCD es básicamente un sistema que se ejecuta en un entorno restringido el medio físico es de solo lectura (a diferencia de otras tecnologías como el LiveUSB donde el soporte es de lectura escritura) y que se apoya principalmente en tecnologías del BIOS que permiten elegir al dispositivo CDROM como primario para el encendido del PC.

Debido a que este tipo de sistema no puede crear archivos temporales en el medio físico se utilizan puntos de montaje creados con cualquiera de los sistemas de archivos en memoria de Linux (ramfs o tmpfs) siendo el último el más utilizado al no tener restricciones acerca de la cantidad de memoria a la que puede acceder.

Objetivos:

- El objetivo de este proceso es automatizar la creación de sistemas LiveCD

Requisito:

- Optimizaciones y componentes para sistemas livecd.
- Utilizar un baselayout especialmente creado para sistemas livecd.

Componentes:

- *nova-live/baselayout*
- *nova-live/desktop-light*
- *nova-live/autoboot*

Depende de:

- Etapa 4.
- Aplicaciones externas.

Funcionamiento:

El proceso de creación de un LiveCD incluye en sí la creación de una stage4 a la cual se le agregan componentes y paquetes con cambios especiales que le permiten ejecutarse con las restricciones de escritura.

Esto es configurable en los paquetes por medio de la USE “livecd” la cual es una indicación del tipo de sistema objetivo, los componentes para sistemas livecd se encuentran el la categoría *nova-live.* Este proceso permite reutilizar una stage4 previamente creada.

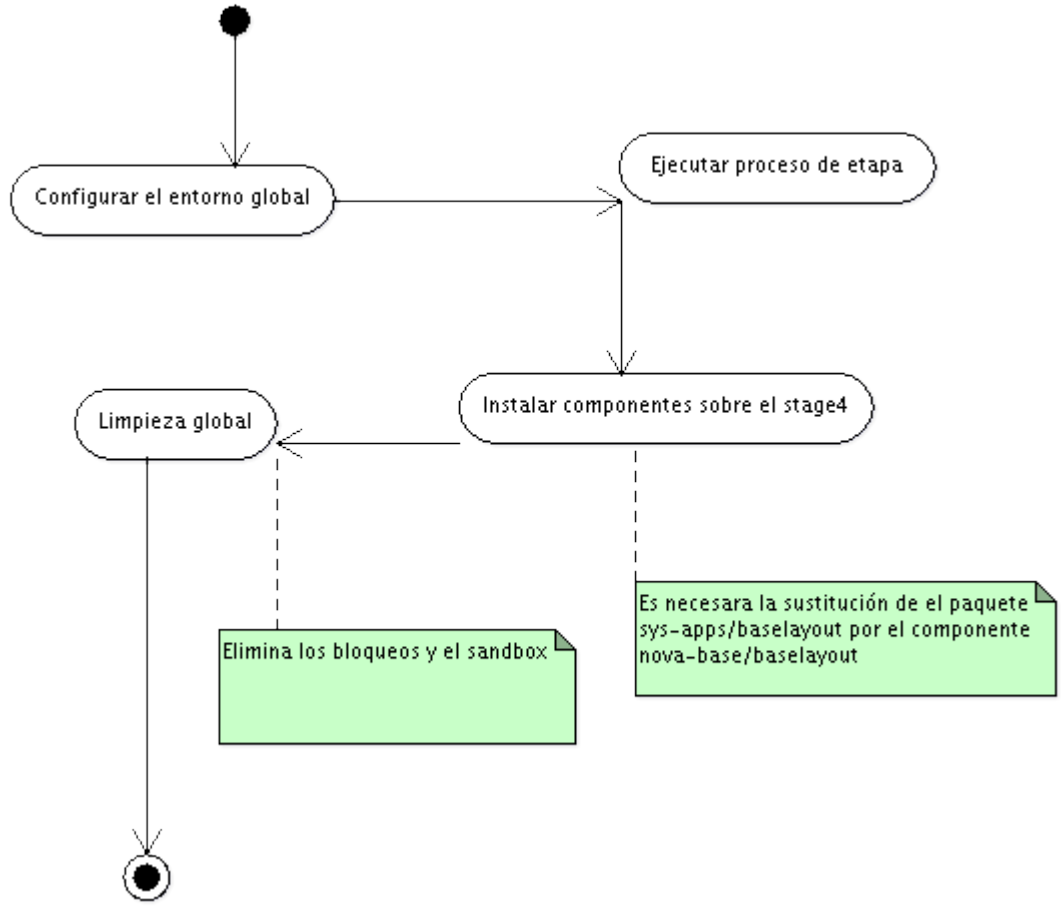

Ilustración 8: Diagrama de actividad para el proceso LiveCD

Base instalación

La plataforma de instalación es una extensión de la Live lo que simplemente no crea un disco en vivo del sistema final sino que crea un pseudo sistema cuya única función es iniciar el instalador y contener una imagen base que será instalada y configurada a gusto del usuario.

Objetivos:

- Crear un sistema live mínimo que contenga

Requisitos:

- Un release dedicado al sistema instalable y otro dedicado al pseudo sistema de instalación.

Depende de:

- Plataforma de bases.

- Aplicaciones externas.

Funcionamiento:

Este proceso funciona en dos etapas, una encargada de conformar el la imagen de instalación y la otra responsable por ensamblar esa imagen con el filesystem preparado para ejecutarse desde algún dispositivo portables (CDROMS y próximamente dispositivos USB)

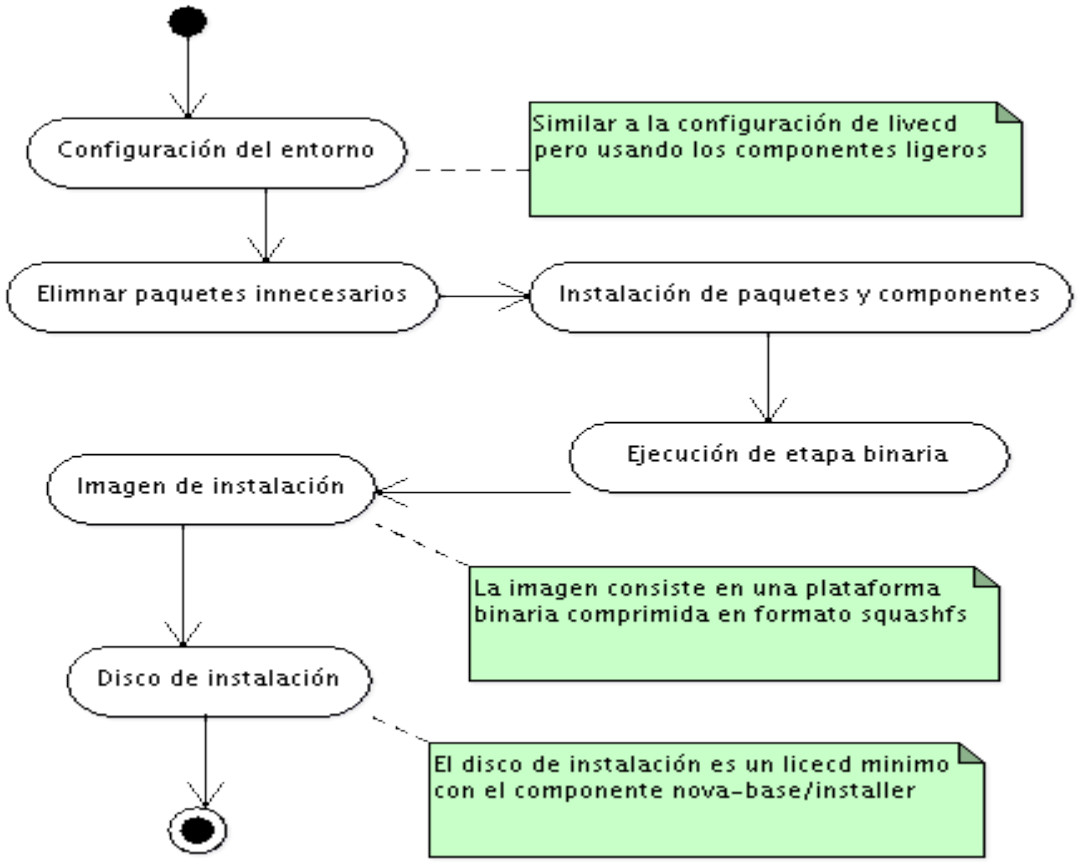

Ilustración 9: Diagrama de actividad para el proceso Base Instalación

Base binaria.

A diferencia de los otros estados la base binaria no tiene como resultado la creación de un sistema o equivalentes virtuales de este sino que se centra en crear componentes

Objetivos:

- Crear la base para los repositorios binarios de paquetes y componentes.

Requisitos:

- Stage4 creado para una plataforma de instalación

Depende de:

- Aplicaciones externas.

Funcionamiento:

La base binaria actúa creando paquetes que pueden ser usados como parte de un repositorio ya sea local o en una

Aspectos de diseño.

A continuación se presentan los aspectos principales del diseño de la herramienta de liberación(Aquila) desarrollada en Nova LNX

Modelo conceptual.

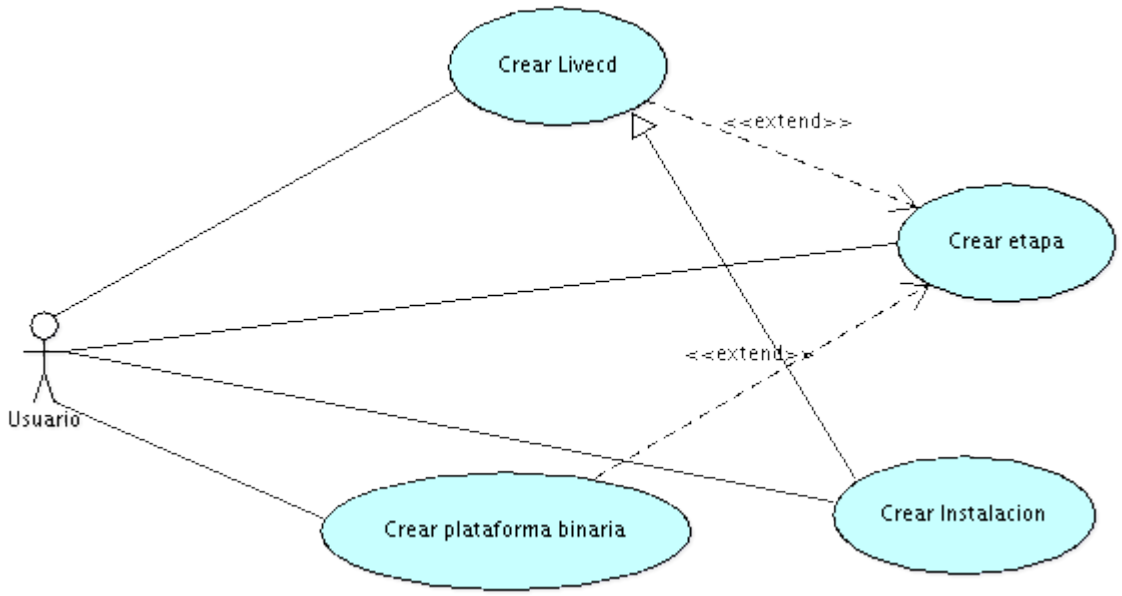

Ilustración 10: Modelo conceptual del sistema releng

En este modelo cada caso de uso corresponde a un proceso

Representación de los módulos

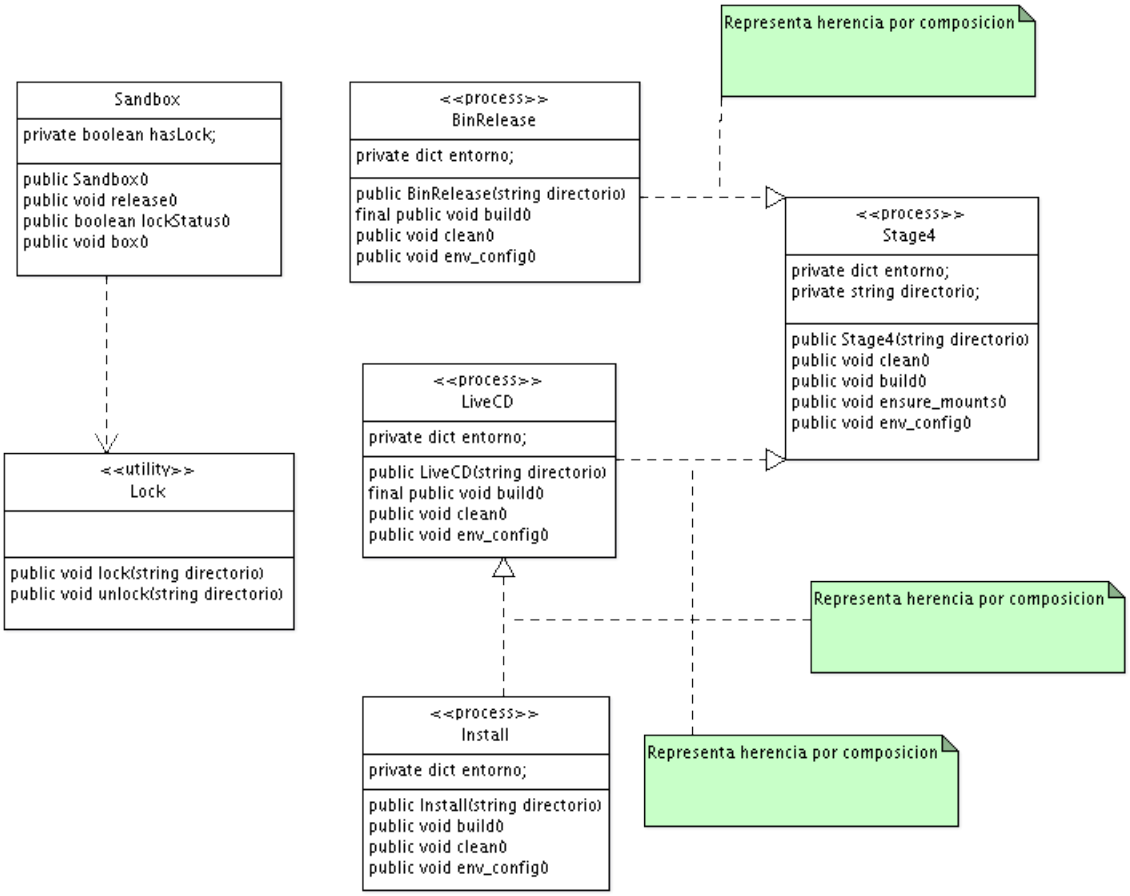

Ilustración 11: Representación de los módulos en un diagrama de clases.

Descripción de las clases.

Clase Sandbox	
Estereotipo	Clase
Atributos	***Tipo***
hasLock	boolean
Responsabilidades	
Nombre	**Descripción**
box():void	Aplica un sandbox al directorio actual
lockStatus():boolean	Devuelve el estado de los cerrojos del

Clase Sandbox	
	sandbox
release():void	Libera el sandbox y los cerrojos

Clase Lock	
Estereotipo	Utilidad
Atributos	***Tipo***
No aplica	No aplica
Responsabilidades	
Nombre	**Descripción**
lock():void	Aplica un sandbox al directorio actual
lockStatus():boolean	Devuelve el estado de los cerrojos del sandbox

Clase Stage4	
Estereotipo	Proceso
Atributos	***Tipo***
entorno	diccionario
directorio	string
Responsabilidades	
Nombre	**Descripción**
build():void	Inicia el proceso de construcción del stage4
clean():void	Libera todos los recursos asignados por este proceso y sus hijos.
ensure_mounts():void	Crea los enlaces o puntos de montajes necesarios en el sistema de archivo virtual después que este fue descompactado en un directorio
env_config():void	Configura el entorno para la creación de

Clase Stage4	
	paquetes y componentes.

Clase Livecd	
Estereotipo	Proceso
Atributos	***Tipo***
entorno	directorio
Responsabilidades	
Nombre	**Descripción**
build():void	Construye la imagen de sistema livecd invocando primero a los procesos necesarios.
clean():boolean	Libera todos los recursos asignados por este proceso y sus hijos.
env_config():void	Genera una configuración especial para el proceso y sus hijos. Esta incluye optimizaciones para sistemas live, archivos ejecutables más pequeños,etc
Dependencias	
Stage4	Es necesario para construir el sistema al que se le agregarán componentes y paquetes para soporte livecd.

Clase BinRelease	
Estereotipo	Proceso
Atributos	***Tipo***
entorno	diccionario
Responsabilidades	
Nombre	**Descripción**
build():void	Construye el conjunto de paquetes y

Clase BinRelease	
	componentes binarios a partir de un stage4 intermedio. El resultado es un grupo de componentes y paquetes los cuales es posible insertar en un repositorio.
clean():boolean	Libera todos los recursos asignados por este proceso y sus hijos.
env_config():void	Configura el entorno de la etapa cuatro para que conserve el resultado de la construcción de los paquetes y los componentes
Dependencias	
Stage4	Provee la base donde se ejecutará este proceso.
Entorno distribuido(opcional)	Facilita la compilación de los componentes y paquetes

Clase Install	
Estereotipo	Proceso
Atributos	***Tipo***
entorno	diccionario
Responsabilidades	
Nombre	**Descripción**
build():void	Crea la imagen de instalación. Inicia a LiveCD y a stage4 como procesos hijos.
clean():boolean	Libera todos los recursos asignados por este proceso y sus hijos.
env_config():void	Se encarga de configurar los hijos de acuerdo a las necesidades específicas del proceso

Benchmarking.

Benchmarking es la tarea de medir la velocidad con la que un ordenador ejecuta procesos, de forma que se puedan realizar comparaciones entre diferentes combinaciones de programas/componentes. Esta definición no tiene en cuenta la sencillez de uso, estética o ergonomía o cualquier otro tipo de juicio subjetivo. A pesar de la variedad de pruebas y herramientas que existen para calcular el rendimiento de un sistema es necesario definir las áreas de interés sobre las que es necesario ejecutarlas. En el caso de Nova LNX las pruebas críticas esta relacionadas mayormente con las mejoras necesarias para los sistemas de escritorio y las implicaciones del modelo de seguridad sobre el rendimiento de los procesos de lectura y escritura sobre dispositivos de almacenamiento. Otras pruebas necesarias incluyen un análisis total de rendimiento y manejo de recursos por el sistema.

Las herramientas utilizadas en el benchmarking son:

- **Bonnie:** Según sus desarrolladores “una utilidad para ejecutar pruebas utilizando un archivo de tamaño conocido”[BON00]. Bonnie ejecuta una serie de operaciones de lectura y escritura sobre el sistema de archivos reportando la cantidad de bytes procesados por segundo de CPU así como el % de carga de CPU.
- **Whetstone** : Fue una de las primeras herramientas creada con el propósito de medir el rendimiento total del sistema midiendo las posibilidades de cálculo y carga del CPU. Permite además analizar el rendimiento de la biblioteca estándar del sistema y el planificador de tareas del kernel.
- **Stress**: Como su nombre lo indica realiza una serie de pruebas llevando al máximo la carga del sistema. Permite analizar rendimiento del filesystem, CPU, planificador de tareas y manejador de memoria.

Mantenimiento.

El mantenimiento de un producto es parte fundamental de su ciclo de producción. En el caso Nova el mantenimiento y creación de nuevas versiones es efectuado a partir de los reportes de error de Gentoo

Linux y la retroalimentación de los usuarios. Las facilidades del portage de señalar paquetes como inestables o potencialmente peligrosos permite organizar el trabajo de los beta-testers y generar parches al código o personalización defectuosa.

El servicio de atención al usuario incluye:

- Actualizaciones del portage.
- Avisos utilizando el sistema GLSA.
- Actualizaciones del repositorio e inclusión de nuevo software.

Conclusiones.

El estudio acerca de las posibilidades de uso y adaptación de alguna de las metadistribuciones líderes llevó a la creación de una nueva distribución basada en Gentoo Linux adaptada a las necesidades de la Universidad para su migración a software libre, logrando un sistema estable, cómodo, flexible y fácil de mantener; con un esquema de desarrollo robusto y versátil basado en los resultados de explotación del mismo.

Este sistema se ha estado utilizando en aulas, laboratorios de producción y de cursos y la biblioteca de nuestra Universidad. Además, recientemente, se escogió como sistema a utilizar en la migración del Ministerio de las Fuerzas Armadas Revolucionarias a software libre, instalándose Nova LNX en las primeras 300 estaciones de trabajo en el mes de Marzo del actual año.

Recomendaciones.

Se recomienda un estudio más avanzado acerca del tema de la compatibilidad binaria entre distribuciones y las posibilidades de utilizar paquetes binarios de Nova o Gentoo en otros sistemas Linux.

Es recomendado un análisis acerca de la factibilidad de modificar Nova LNX de modo que sea compatible con la LSB

Bibliografía.

Referencias bibliográficas.

- [RAY00] *Eric Raymond. "La catedral y el bazaar", http://www.sindominio.net/biblioweb/telematica/cat*
- [PEX00] *Martin Fowler et al. "Planning extreme programming", Addison Wesley, 2000.*
- [BON00] *"Introducing Bonnie", http://www.textuality.com/bonnie/intro.html*
- [GDP00] *GNOME Documentation Project"The GNOME Handbook of Writing Software Documentation"*

Material consultado.

- *Rafeeq Ur Rehman y Christopher Paul. The Linux Development Platform, Prentice Hall.*
- *Enciclopedia Libre Wikipedia. http://www.wikipedia.org*
- *GNOME Documentation Project, GNOME Documentation Style Guide http://developer.gnome.org/documents/style-guide/*
- *Gentoo Documentation Project,*
- *Free Software Foundation, Unix Manual Pages*
- *Free Software Foundation, Textinfo Documentation System.*

Anexo I: Tablas comparativas de las distribuciones.

Las tablas incluyen datos de Debian, distribución en la que originalmente Ubuntu estuvo basada pero que actualmente no posee compatibilidad binaria.

Datos generales.

	Empresa	Precio (€)	Licencia	Público	País
Debian GNU/Linux	Debian Project	Gratis	cualquier DFSG	Desktop, Workstation, Server	Mundial
Fedora Core	Fedora Project	Gratis	GPL	Workstation, Server, Público	EEUU
Gentoo	Gentoo Foundation	Gratis	GPL	Workstation, Server, Público	Mundial
Mandriva Linux	Mandriva	Gratis (Download edition)	GPL	Desktop, Workstation, Server	Mundial
Slackware Linux	Patrick Volkerding	Gratis	GPL	Workstation, Server, Público	EEUU
SUSE Linux	Novell	Descarga gratuita disponible Ed. Profesional: 74,54	GPL	Workstation, Server, Público	Alemania

Técnica

	Kernel	Sistemas de ficheros soportados	Arquitectura	Herramienta de Actualización online	API principal e Idioma para Aplicaciones CLI
Debian GNU/Linux	Linux 2.4/2.6	ext2, JFS, XFS, FAT, NTFS, ISO 9660, UDF, NFS, ReiserFS	x86, IA64, PPC, SPARC, SPARC64, Alpha, MIPS, ARM, PA-RISC, Mac/VME 68k, S/390	APT, Synaptic	pre-LSB con C, otros (estándar POSIX)
Gentoo	Linux 2.6	ext2, JFS, XFS, FAT, NTFS, ISO 9660, UDF, NFS, ReiserFS	Alpha, hppa, PPC,AMD64, x86, ARM, PPC64, SPARC, S/390, MIPS, m68k, IA64	Porthole(gráfica), emerge	pre-LSB con C, otros (estándar POSIX)
Fedora Core	Linux 2.6	ext2, ReiserFS, FAT, ISO 9660, UDF, NFS	x86, x86-64, i386, PowerPC	up2date, yum, APT (limitado)	LSB con C, otros (estándar POSIX)
Mandriva Linux	Linux 2.6	ext2, JFS, XFS, FAT, NTFS, ISO 9660, UDF, NFS, ReiserFS	x86 (i586), x86-64, PPC	urpmi	LSB con C, otros (estándar POSIX)
Slackware Linux	Linux 2.4	JFS, XFS, FAT, NTFS, ISO 9660, UDF, NFS	x86, IA64, S/390	Swaret, Slapt-get, algunos no oficiales	LSB con C, otros (estándar POSIX)
SUSE Linux	Linux 2.6	ext2, ext3, JFS, XFS, FAT, NTFS, ISO 9660, UDF, NFS, Reiser4	x86, IA64, x86-64, PPC	YaST2	LSB con C, otros (estándar POSIX)

Aspectos misceláneos.

	Paquetes	¿Instalación Gráfica?	Explorador de ficheros	Navegador Web	Entorno Gráfico Principal	Paquete de Ofimática
Debian GNU/Linux	18000	No	Nautilus/Konqueror	Konqueror/Mozilla Firefox	GNOME	OpenOffice.org/KOffice/GNOME Office
Gentoo	10700	Experimental	Elección del usuario	Elección del usuario	Elección del usuario	Elección del usuario
Fedora Core	5000	Si	Nautilus	Epiphany	GNOME	OpenOffice.org/KOffice/GNOME Office
Mandriva Linux	4000	Si	Konqueror	Konqueror/Mozilla Firefox	KDE	OpenOffice.org
Slackware Linux	?	Si	**N/A**	**N/A**	**N/A**	**N/A**
SUSE Linux	12500	Si	Nautilus	Mozilla Firefox	Gnome - KDE	OpenOffice.org

Anexo II: Herramientas de configuración

Galaxia, instalador de Nova LNX

Ilustración 12: Galaxia, pantalla de inicio

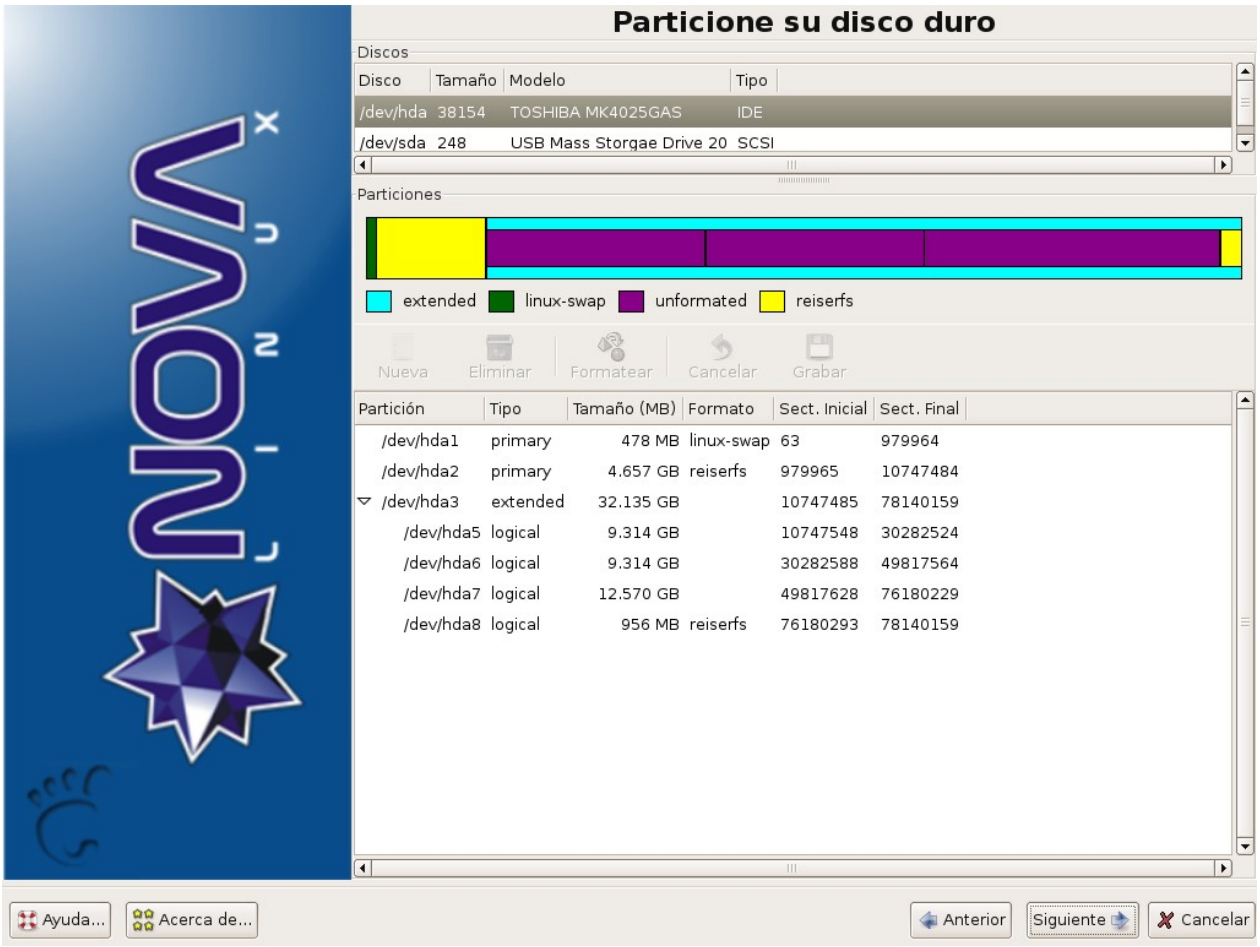

Ilustración 13: Galaxia, selección de particiones

Yukiyu, instalador de paquetes de Nova.

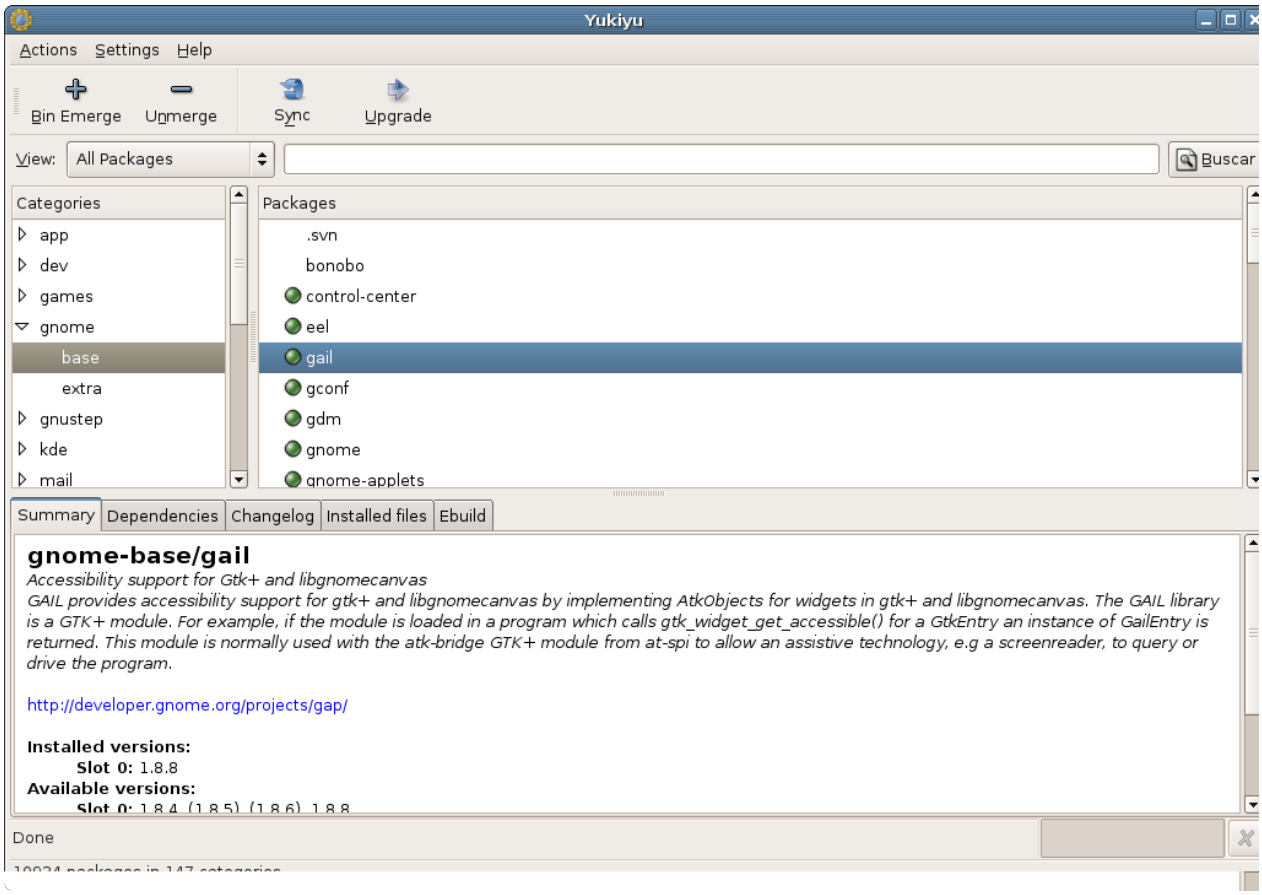

Ilustración 14: Yukiyu, resumen de gnome-base/gail

Nsmb, configuración de carpetas compartidas.

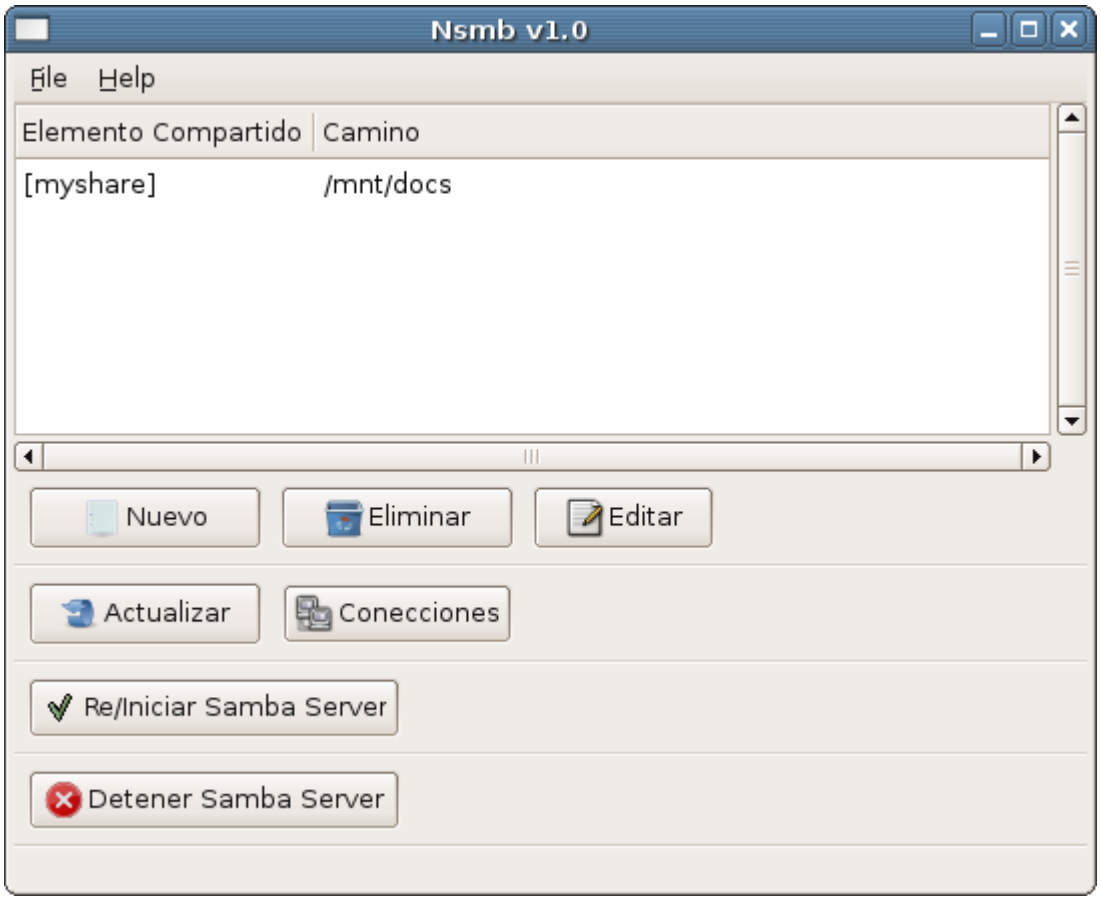

Ilustración 15: Nsmb, pantalla de inicio mostrando una carpeta compartida

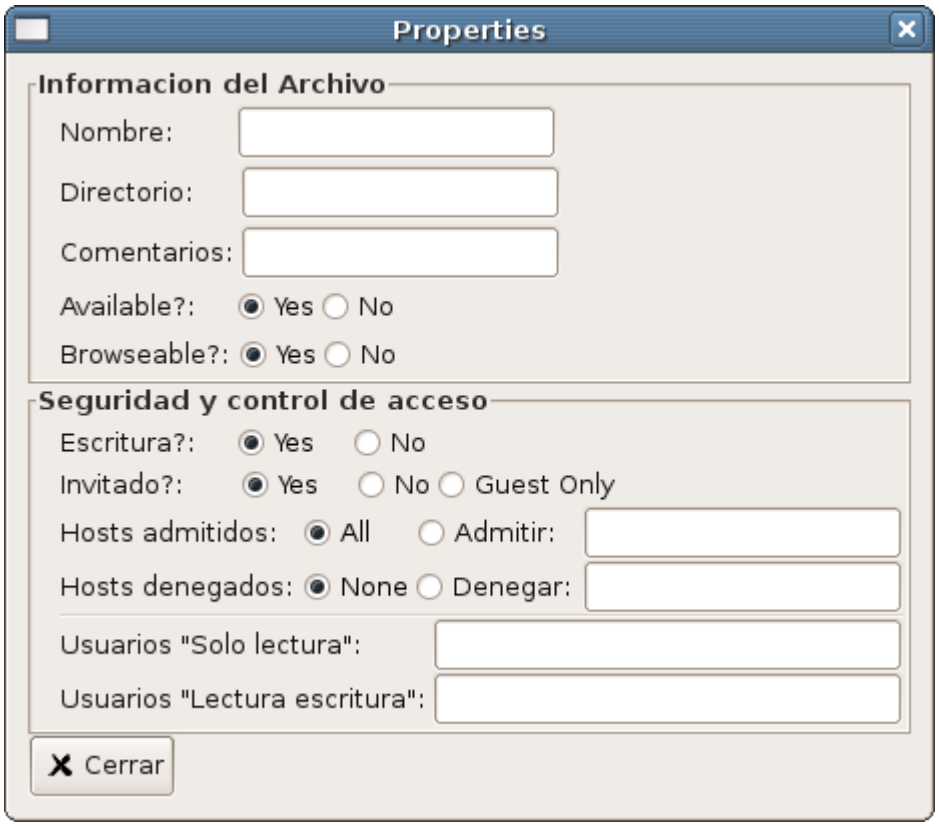

Ilustración 16: Nsmb, propiedades de archivo compartido

Anexo III Licencias.

GNU Free Documentation License. Version 1.2, November 2002

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other
functional and useful document "free" in the sense of freedom: to
assure everyone the effective freedom to copy and redistribute it,
with or without modifying it, either commercially or noncommercially.
Secondarily, this License preserves for the author and publisher a way
to get credit for their work, while not being considered responsible
for modifications made by others.

This License is a kind of "copyleft", which means that derivative

works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of

transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which

states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy

a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.

C. State on the Title page the name of the publisher of the Modified Version, as the publisher.

D. Preserve all the copyright notices of the Document.

E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.

H. Include an unaltered copy of this License.

I. Preserve the section Entitled "History", Preserve its Title, and add

to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.

N. Do not retitle any existing section to be Entitled "Endorsements"

or to conflict in title with any Invariant Section.

O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties--for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of,

you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number.

Make the same adjustment to the section titles in the list of
Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History"
in the various original documents, forming one section Entitled
"History"; likewise combine any sections Entitled "Acknowledgements",
and any sections Entitled "Dedications". You must delete all sections
Entitled "Endorsements".

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents
released under this License, and replace the individual copies of this
License in the various documents with a single copy that is included in
the collection, provided that you follow the rules of this License for
verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute
it individually under this License, provided you insert a copy of this
License into the extracted document, and follow this License in all
other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to

copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

> Copyright (c) YEAR YOUR NAME.
> Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the "with...Texts." line with this:

> with the Invariant Sections being LIST THEIR TITLES, with the Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.

If you have Invariant Sections without Cover Texts, or some other

combination of the three, merge those two alternatives to suit the
situation.

If your document contains nontrivial examples of program code, we
recommend releasing these examples in parallel under your choice of
free software license, such as the GNU General Public License,
to permit their use in free software.

The GNU General Public License (GPL). Version 2, June 1991

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the

Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made

generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE

PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

 to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

One line to give the program's name and a brief idea of what it does.
Copyright (C) <year> <name of author>

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU

General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) year name of author Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'. This is free software, and you are welcome to redistribute it under certain conditions; type `show c' for details.

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than `show w' and `show c'; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program `Gnomovision' (which makes passes at compilers) written by James Hacker.

signature of Ty Coon, 1 April 1989
Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Library General Public License instead of this License.

PSF LICENSE AGREEMENT FOR PYTHON 2.3

1. This LICENSE AGREEMENT is between the Python Software Foundation ("PSF"), and the Individual or Organization ("Licensee") accessing and otherwise using Python 2.3 software in source or binary form and its associated documentation.

2. Subject to the terms and conditions of this License Agreement, PSF hereby grants Licensee a nonexclusive, royalty-free, world-wide license to reproduce, analyze, test, perform and/or display publicly, prepare derivative works, distribute, and otherwise use Python 2.3 alone or in any derivative version, provided, however, that PSF's License Agreement and PSF's notice of copyright, i.e., "Copyright (c) 2001, 2002, 2003 Python Software Foundation; All Rights Reserved" are retained in Python 2.3 alone or in any derivative version prepared by Licensee.

3. In the event Licensee prepares a derivative work that is based on or incorporates Python 2.3 or any part thereof, and wants to make the derivative work available to others as provided herein, then Licensee hereby agrees to include in any such work a brief summary of the changes made to Python 2.3.

4. PSF is making Python 2.3 available to Licensee on an "AS IS" basis. PSF MAKES NO REPRESENTATIONS OR WARRANTIES, EXPRESS OR

IMPLIED. BY WAY OF EXAMPLE, BUT NOT LIMITATION, PSF MAKES NO AND DISCLAIMS ANY REPRESENTATION OR WARRANTY OF MERCHANTABILITY OR FITNESS FOR ANY PARTICULAR PURPOSE OR THAT THE USE OF PYTHON 2.3 WILL NOT INFRINGE ANY THIRD PARTY RIGHTS.

5. PSF SHALL NOT BE LIABLE TO LICENSEE OR ANY OTHER USERS OF PYTHON 2.3 FOR ANY INCIDENTAL, SPECIAL, OR CONSEQUENTIAL DAMAGES OR LOSS AS A RESULT OF MODIFYING, DISTRIBUTING, OR OTHERWISE USING PYTHON 2.3, OR ANY DERIVATIVE THEREOF, EVEN IF ADVISED OF THE POSSIBILITY THEREOF.

6. This License Agreement will automatically terminate upon a material breach of its terms and conditions.

7. Nothing in this License Agreement shall be deemed to create any relationship of agency, partnership, or joint venture between PSF and Licensee. This License Agreement does not grant permission to use PSF trademarks or trade name in a trademark sense to endorse or promote products or services of Licensee, or any third party.

8. By copying, installing or otherwise using Python 2.3, Licensee agrees to be bound by the terms and conditions of this License Agreement.