Universidad de las Ciencias Informáticas
Facultad 10

"Aplicación para Generar LiveCD/DVDs de Sistemas Debian GNU/Linux."

Trabajo de Diploma para Optar por el Título de
Ingeniero en Ciencias Informáticas.

Autores: Lucia C. Domínguez Delgado
Keiver Hernández Fernández

Tutor: Ing. Ramón Paumier Samón

Ciudad de la Habana 30 de Abril del 2009
"Año del 50 Aniversario de la Revolución"

DECLARACIÓN DE AUTORÍA

Declaramos ser autores de la presente tesis y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter no exclusivo.

Para que así conste firmo la presente a los _____ días del mes de _____ del año _____.

_____ _____

Lucia C. Domínguez Delgado Keiver Hernández Fernández

_____

Ing. Ramón Paumier Samón

DATOS DEL CONTACTO

Ing. Ramón Paumier Samón

Profesor graduado de Ingeniero en Ciencias Informáticas en el año 2007 como parte de la primera graduación de la UCI. Ha impartido asignaturas como Cálculo Diferencial e Integral I y II, así como Introducción a la Programación y Programación 1. Posee categoría docente de Adiestrado.

Ha cursado Postgrados de: Ideología y Política de la Revolución, Ciencia Tecnología y Sociedad, Fundamentos de Docencia Universitaria y Metodología de la investigación. Ha presentado ponencias en eventos y forma parte del grupo de investigaciones de Migración a Software Libre de la UCI y del Grupo Técnico Nacional. Es miembro del proyecto Unicornios.

Miembro organizador del Taller de Software Libre de UCIENCIA 2008. Como parte de sus investigaciones, elaboró una Mini-Guía de Migración a SWL para los organismos del Ministerio de la Informática y las Comunicaciones. Ha trabajado en proyectos internacionales relacionados con la migración a Software Libre. En estos momentos trabaja en la elaboración de la Guía Cubana de Software Libre.

AGRADECIMIENTOS

Keiver Hernández Fernández

Les agradezco a todos mis amigos que han estado cerca cuando los he necesitado. A mis profesores y compañeros que me han ayudado a la largo de la carrera.

A mi familia que siempre se ocupó de mí y me ayudó a salir adelante. Especialmente a mis tías, tíos y primos que siempre han estado ahí para mí. Gracias.

Agradezco también a todas las personas que han contribuido de alguna forma a graduar al primer Ingeniero en Ciencias Informática de Las Veguitas.

Lucia C. Domínguez Delgado

A mis padres por traerme a este mundo, a todas aquellas personas que han contribuido de una forma u otra en mi madurez como persona y como profesional.

En especial a mi tía abuela Aida que ayudo de forma directa e indirectamente en la toma de decisiones en mi vida, que sin su ayuda no hubiese sido posible este logro.

A mis hermanos que siempre me apoyaron y me dieron su cariño.

A mi sobrino por ser la luz de mi camino.

A mis padrinos Margarita y Irán que siempre me dieron su bendición.

A toda mi familia y mis amigos que estuvieron atentos en toda la evolución de mi carrera.

DEDICATORIA

Keiver Hernández Fernández

Dedico este logro a mis padres Nilda Fernández Cardoso y Gabriel Hernández Benítez por estar siempre dándolo todo por mí. Sin ellos no hubiese logrado llegar tan lejos; son los mejores padres del mundo. A mi abuela adorada Esperanza Fernández Cardoso por siempre quererme tanto.

Lucia C. Domínguez Delgado

Le dedico este triunfo a mi mamá Lourdes Caridad Delgado Espino que siempre me dio su cariño y apoyo. Que a pesar de la distancia siempre estuvo a mi lado.

A mi papa Osvaldo Orlando Domínguez Gallardo por contribuir a mi formación desde pequeña.

A mis hermanos Pabli, Odett, Osvi, Ale y mi sobrino Dienys.

A todos mis familiares que siempre siguieron de cerca mi desarrollo.

A mis amigos Dayanis, Yadira Santos, Yadira Bagarotti, Yunia, Araibis, Ariel, Osmani Pallí, Carlos, Yosniel, Yunior, Jesús Ignacio, Manuel, Yoel, Keiver, Yoandy Rodriguez, mis compañeros de la Universidad y a todos los que han estado para apoyarme.

A Gloria, Marina, Luis, Aily, Liz y Lisbet en fin a todos los que han aportado su granito de arena para hacer realidad mi sueño.

RESUMEN

En la actualidad es muy común que determinada empresa o institución necesite probar una aplicación o distribución de Linux; una de las formas más usadas es la creación de un LiveCD/DVD que contenga dicha aplicación o sistema. Luego el LiveCD/DVD obtenido puede usarse en cualquier ordenador sin necesidad de instalar en los discos duros absolutamente nada. Otro de los usos ampliamente extendidos de los LiveCD/DVD es que se pueden usar como herramientas de rescate en caso de que se ocurra algún error en el sistema de un ordenador determinado. Teniendo en cuenta estos aspectos y con el fin de que se pueda usar una herramienta gráfica para generar LiveCD/DVDs, es que se desarrolla el presente trabajo.

En la presente investigación se hace un estudio de las diferentes herramientas para la creación de LiveCD/DVD que existen en la actualidad. También se hace un análisis del funcionamiento y características fundamentales de cada uno de estos sistemas y las herramientas empleadas para su desarrollo. Como parte del trabajo se lleva a cabo la implementación de una aplicación gráfica para la generación de LiveCD/DVDs a partir de sistemas Debian GNU/Linux de forma fácil al usuario. La aplicación desarrollada copia el sistema original y lo comprime usando un sistema de ficheros squashfs el cual permite grandes radios de compresión. El resultado final es una imagen ISO lista para ser quemada en un CD o un DVD.

PALABRAS CLAVES
LiveCD, LiveDVD, remasterización, Linux, Kernel.

TABLA DE CONTENIDO

INTRODUCCIÓN

Actualmente se vive en un período histórico conocido como la “era de la información” en el que existe una revolución tecnológica que tiene como elementos centrales la tecnología de la información y de las comunicaciones. Un papel fundamental juega en este entorno el desarrollo de software. El cual esta liderado por dos corrientes; el desarrollo cerrado o privativo y el desarrollo libre.

El Movimiento de software libre comenzó en 1983 cuando Richard Stallman anunció el proyecto GNU. La meta del movimiento es dar libertades a los usuarios sobre los programas que usan.

Según la Free Software Foundation(FSF), el software libre se refiere a la libertad de los usuarios para ejecutar, copiar, distribuir, estudiar, cambiar y mejorar el software; de modo más preciso, se refiere a cuatro libertades de los usuarios del software: la libertad de usar el programa, con cualquier propósito; de estudiar el funcionamiento del programa, y adaptarlo a las necesidades; de distribuir copias, con lo que puede ayudar a otros; de mejorar el programa y hacer públicas las mejoras, de modo que toda la comunidad se beneficie.[1]

Con el objetivo de convertir al país en una potencia en el desarrollo de software se crea la Universidad de la Ciencias Informáticas (UCI), como expresara el compañero Fidel Castro Ruz ¨*Nuestra aspiración de contar con centros de excelencia en la educación superior dio lugar al surgimiento de la Universidad de las Ciencias Informáticas, primera institución de su tipo surgida en la Batalla de Ideas,…obteniendo rápidamente significativos logros en la enseñanza y la actividad productiva.*¨ [2]

Esta universidad está estructurada en diferentes facultades, donde cada una se orienta a un perfil determinado, pero que tienen como principal objetivo la formación de profesionales leales a la Revolución, especialistas en informática, teniendo como principio del proceso docente educativo "La formación desde la producción".

La Facultad 10 o Facultad de Software Libre de la UCI tiene como tarea principal la formación continua de profesionales integrales y revolucionarios especializados en el campo del software libre, desarrollar investigaciones en esta área e impulsar el uso del software libre en nuestra comunidad universitaria así como en nuestro país. Es precisamente en el seno de esta facultad que nace la Comunidad de Software Libre de la UCI.

Como todas las pequeñas comunidades de software libre que tributan a la Comunidad de Software Libre a nivel mundial, la de la UCI desarrolla nuevas aplicaciones o colabora con otros proyectos que se están desarrollando en otras comunidades.

Las distribuciones de software libre son un elemento fundamental en este movimiento, variadas son las herramientas y sistemas que se usan para darle funcionalidad a las mismas, así como para desarrollarlas. Es muy común que determinada empresa o institución necesite una distribución con características específicas para su uso, como probar aplicaciones o sistemas sin necesidad de instalar dicha distribución en la computadora y además instalar dicha distribución en una computadora que no tenga acceso a la red para descargar las aplicaciones necesarias.

Teniendo en cuenta lo anteriormente planteado, se llega a la **situación problémica**, que consiste en la no existencia de una aplicación gráfica para automatizar el proceso de creación de un LiveCD/DVD a partir de un sistema Debian GNU/Linux preinstalado y modificado. Pues es necesario usar dicho software en el Polo de Software Libre de la UCI para el proceso de migración que se lleva a cabo actualmente.

El uso de la aplicación propuesta facilitaría los procesos de prueba de una aplicación o sistema en particular. Al ser los LiveCD/DVD una de las vías principales que se usa para probar y distribuir las mismas. Otra de las ventajas que tendría el sistema propuesto es que actualmente el proceso de creación de un LiveCD/DVD es considerado como una tarea compleja y que necesita un amplio conocimiento sobre sistemas Linux. La aplicación propuesta tendrá una interfaz gráfica agradable al usuario, haciendo el proceso de creación de un LiveCD/DVD fácil y rápido.

Surge como **problema científico** ¿Cómo implementar una aplicación gráfica capaz de crear desde un sistema ya preinstalado de Debian GNU/Linux, un sistema autoarrancable desde un CD o DVD?

El **objeto de estudio** son los programas existentes para la creación de LiveCD/DVD, por lo que el **campo de acción** son las herramientas de software libre usadas con este fin.

El **objetivo general** del presente trabajo es implementar una aplicación gráfica en entornos GNU/Linux, para crear un LiveCD/DVD a partir de un sistema ya preinstalado y modificado.

Para cumplir con el objetivo propuesto se han definido los siguientes **objetivos específicos**:

- Sistematizar las diferentes herramientas para hacer LiveCD/DVD de Debian GNU/Linux y otras distribuciones importantes existentes en Cuba y el mundo.
- Diseñar una interfaz gráfica para el sistema amigable al usuario.
- Desarrollar los scripts necesarios para hacer el sistema compatible con Debian GNU/Linux.

El desarrollo de las distintas tareas se ordena teniendo como punto central la aplicación. Se pretende que esta y todas las aplicaciones que se implementen puedan ser usadas por otros programas para beneficios de todos. Las herramientas a utilizar son todas libres, punto este que permitirá su redistribución, modificación y la propia utilización fuera de cualquier conflicto legal que pudiera surgir.

El desarrollo de dicha investigación contempla **tareas** relacionadas con:

- Sistematizar las características, ventajas y desventajas de los sistemas para crear LiveCD/DVD en Debian GNU/Linux y otras distribuciones.
- Diseñar una interfaz gráfica para el sistema.
- Crear un conjunto de scripts para hacer un LiveCD/DVD para optimizar el sistema a crear.
- Realizar todas las pruebas necesarias para eliminar los posibles errores que se puedan presentar en el ciclo de desarrollo de la aplicación.

La presente investigación está estructurada en 3 capítulos. A continuación se muestra una breve descripción de cada uno de los capítulos.

Capítulo 1: Fundamentación Teórica. En este capítulo se introducirán conceptos fundamentales como el de remasterización, y creación de un LiveCD/DVD, incluirá los tipos de aplicaciones usadas en el mundo para este fin, así como las tecnologías utilizadas para la realización de los mismos. Además se tratará el proceso general que siguen la mayoría de estos sistemas y sus características. Se incluyen además las tecnologías que se usarán para desarrollar la aplicación propuesta.

Capítulo 2: Análisis y Diseño: Se realizará el desarrollo ágil de la aplicación propuesta, y se explicará toda la dinámica del proyecto en forma de historias de usuarios, prototipos de interfaz de usuario y algunos modelos auxiliares además del plan de releases para las entregas.

Capítulo 3: Implementación y Pruebas: Finalmente en este capítulo se explicará el proceso de implementación y se plasmarán los casos de pruebas a las que fue sometida la aplicación en cada una de las iteraciones. Se exponen los resultados obtenidos y se muestran las funcionalidades alcanzadas en el período de desarrollo.

CAPÍTULO I. FUNDAMENTACIÓN TEÓRICA

Se sistematizarán los sistemas de remasterización en las diferentes distribuciones de GNU/Linux principalmente las derivadas de Debian GNU/Linux. Se mostrarán las diferentes tecnologías, los lenguajes y las herramientas que se utilizarán para el desarrollo de la aplicación propuesta así como otras que se consideren necesarias analizar.

1.1 Sistemas de remasterización de distribuciones GNU/Linux.

1.1.1 LiveCD/DVD y Remasterización.

A menudo organizaciones o instituciones necesitan crear una distribución de GNU/Linux que satisfaga sus necesidades y complementen su trabajo. Generalmente cuando se crea una nueva distribución uno de los métodos más difundidos para distribuir dicho sistema es la creación de un LiveCD/DVD de la misma, para que los usuarios sean capaces de probar dicha distribución sin tener que instalarla en su computadora. Es un método que permite la evaluación rápida del nuevo sistema.

Un LiveCD o LiveDVD, traducido en ocasiones como CD vivo o CD autónomo, es un sistema operativo almacenado en un medio extraíble, tradicionalmente un CD o un DVD (de ahí sus nombres), que puede ejecutarse desde este sin necesidad de instalarlo en el disco duro de una computadora, para lo cual usa la memoria RAM como disco duro virtual y el propio medio como sistema de ficheros.

El término remasterización es un vocablo informático usado para describir el proceso de creación de un LiveCD/DVD a partir de un sistema pre instalado y modificado a las necesidades del usuario. En la remasterización se instalan nuevos paquetes a la distribución a usar o se eliminan otros que no se desean en el sistema autoarrancable a obtener.

Para la creación de LiveCD/DVD existen actualmente varias herramientas, la mayoría de ellas a nivel de consola. Casi todas siguen una secuencia de pasos para construir un sistema autónomo.

1.1.2 Proceso de arranque de un LiveCD/DVD

Un LiveCD/DVD contiene una distribución de Linux como la que se instala en un disco duro normal. Sin embargo, no basta con copiar la instalación desde el disco duro a un CD/DVD para obtener un sistema

autoarrancable. ¿Por qué? Porque existen pequeñas diferencias entre un Live CD/DVD y un disco duro normal.

Un CD o un DVD es un medio de solo lectura y Linux necesita tener acceso de escritura a algunas partes del sistema para poder operar apropiadamente, como por ejemplo los directorios temporales de procesos y dispositivos, hay muchas formas para solucionar este problema. Todos ellos son usando la memoria RAM del sistema. Algunos de estos métodos permiten el acceso de escritura sólo a los directorios y archivos básicos, y por lo tanto, no permiten al usuario modificar el sistema o instalar nuevos paquetes en el Live CD/DVD.

Uno de los métodos antes mencionados es **unionfs**, el cual permite tener acceso de escritura en todo el sistema, haciendo posible la instalación de paquetes y la modificación de archivos del sistema. Esto se realiza mediante la fusión de parte de la memoria RAM con el sistema de archivos de sólo lectura del LiveCD/DVD y parecer un sistema de archivos que tienen acceso de lectura y escritura.

Para que el sistema quepa dentro del medio, el mismo debe estar comprimido; generalmente se usa **squashfs**. Por lo que de la misma forma el tipo de compresión usado debe ser detectado al inicio del proceso de arranque. Así el Kernel usado para el arranque del LiveCD/DVD debe tener los módulos para detectar el sistema de compresión usado.

Estas consideraciones exigen una preparación especial en el momento de arranque, algunas de las cuales debe realizarse incluso antes de montar el sistema de ficheros.

Linux presenta un mecanismo que permite que estos pasos se hagan en el momento de arranque incluso antes de que el sistema de archivos raíz este montado. Se llama el sistema de ficheros inicial o **initramfs**. Este mecanismo se utiliza también en el proceso de arranque de instalaciones en discos duros, ya que añade flexibilidad al proceso.

El gestor de arranque carga el Kernel y el initramfs en la memoria y se inicia el Kernel. Este entonces desempaqueta el initramfs y lo monta como sistema de ficheros inicial, a continuación busca el programa **init** en este sistema, una vez encontrado, lo ejecuta y comienza el proceso de arranque. Este script de inicio se encarga de encontrar el verdadero sistema de ficheros y montarlo. También es responsable por cualquier operación especial que se quiera hacer en el momento del arranque.

En esta investigación no escribiremos manualmente el **initramfs** (aunque puede hacerse) pues existen paquetes ya preparados con estos scripts como **live-initramfs** en los repositorios de Debian GNU/Linux. [3]

1.1.3 Funcionamiento de los sistemas de creación de LiveCD/DVD.

La mayor parte de estas aplicaciones parten de un sistema pre instalado en una partición de la computadora, la cual debe ocupar menos de 2GB de espacio pues la imagen comprimida que se crea posteriormente debe ser menor de 700 MB, en caso que se quiera crear un LiveDVD (4.7 GB) el tamaño de la partición puede ser mucho mayor.

Una vez hecho los cambios necesarios en los ficheros de configuración y haber personalizado todo lo que se estimara necesario, se instalan algunos paquetes necesarios para que el sistema sea autoarrancable, como los módulos necesarios del Kernel para reconocer el sistema de ficheros usado para comprimir el sistema del LiveCD/DVD y los scripts del **initramfs.**

Al estar listos los requisitos anteriormente mencionado se prosigue a crea una imagen comprimida de dicho sistema, aunque se puede hacer un LiveCD/DVD sin comprimir el sistema de ficheros usando cualquier otro que no lleve compresión, como por ejemplo ext2; siempre teniendo en cuenta el tamaño del mismo. Siguiente a esto se prepara el gestor de arranque que tendrá el LiveCD/DVD, que pueden ser Syslinux, Grub u otro y por último se crea la imagen con sistema de ficheros ISO 9660, lista para quemarla a un CD o DVD.

Otros sistemas para crear LiveCD/DVD lo que hacen es conectarse a los repositorios oficiales de las distribuciones como Debian o Ubuntu y descargan paquetes que pueden ser previamente seleccionados por el usuario. Este método no es muy útil si se quiere tener un sistema con una configuración personal. Una de las aplicaciones más usadas para este fin es **debootstrap**.

1.2 Sistemas para la creación de LiveCD/DVD que se usan actualmente.

Con el objetivo de automatizar el proceso de creación de LiveCD/DVD se han creado en todo el mundo aplicaciones que contribuyan a este fin. Algunos de estos sistemas se han adaptado a las necesidades de las distribuciones para las que fueron creadas, otras son de uso más genérico.

En la siguiente tabla se muestran algunos de estos sistemas:

Nombre	Descripción	Distribución	Licencia
dfsbuild	Genera un CD de instalación de Debian desde cero (Debian From Scratch). Se conecta al repositorio más cercano y descarga los paquetes necesarios para la creación de un CD de instalación de Debian.	Debian GNU/Linux	GPL2
bootcd	Genera un CD autoarrancable del sistema instalado en la computadora, es posible copiar el sistema del CD/DVD para la computadora. Soporta una gran variedad de gestores de arranque, como lilo, grub y syslinux.	Debian GNU/Linux	GPL2
livecd-tools	Es una herramienta para generar LiveCDs en sistemas basados en Fedora incluyendo soporte para las distribuciones como RHEL, CentOS y otras.	Fedora	GPL2
live-magic	Es una interfaz de usuario para crear LiveCD/DVD. Es un frontend para live-helper. Por lo que ofrece una serie de funcionalidades que esta aplicación provee.	Debian GNU/Linux	GPL2
live-helper	Es un conjunto de scripts para realizar imágenes de sistemas Debian. La idea detrás de live-helper es un entorno de trabajo que utiliza un directorio de configuración para automatizar completamente y personalizar todos los aspectos de la realización de una imagen Live.	Debian GNU/Linux	GPL2

remastersys	Es una herramienta que permite hacer copias de seguridad del sistema ya instalado. Incluyendo las configuraciones personales en un LiveCD/DVD	Ubuntu Linux	GPL2
Linux Live Scripts	Son un conjunto de scripts que permiten crear un LiveCD/DVD propios de un sistema ya instalado. El sistema creado es arrancable desde un CD-ROM o un dispositivo de disco, por ejemplo, USB Flash Drive, USB Pen Drive, cámara conectada al puerto USB, y así sucesivamente. Se usa Linux Live scripts también para arrancar Linux desde un iPod.	All	GPL
UCK	Ubuntu Customization Kit (UCK), permite crear un LiveCD personalizado de Ubuntu y alguno de sus derivados.	Ubuntu Linux	GPL
Reconstructor	Usa el Live CD original de Ubuntu como base y permite modificar las pantallas de arranque e inicio (boot screens), la configuración de gnome y el software que se quiere incluir en el LiveCD/DVD.	Ubuntu Linux	GPL
nova-livecdkit	Usado para generar el LiveCD de Nova, la distribución cubana. Desarrollada en bash y entre otras funcionalidades incluye la creación de perfiles para trabajar en varias secciones de trabajo a la vez.	Nova Linux	GPL

Tabla 1. Lista de algunas herramientas usadas para generar LiveCD/DVDs.

Debido a que estos sistemas están desarrollados para distintas distribuciones, usan distintas tecnologías y aplicaciones, a continuación se explicarán las características y funcionalidades de ellos.

1.2.1 dfsbuild

Debian desde cero (DFS en inglés) crea un LiveCD/DVD que está diseñado para proporcionar un núcleo y un ambiente de rescate completos. El ambiente de rescate contiene herramientas para trabajar con el sistema de ficheros, editores, un ambiente de desarrollo C, entre otras.

Es muy configurable y se pueden crear LiveCD/DVD con un núcleo personalizado, e incluso incluir un conjunto de paquetes totalmente diferente.

Dfsbuild trabaja obteniendo los paquetes del repositorio Debian más cercano y con esos paquetes generará una imagen ISO que contendrá un sistema Debian arrancable. También puede poner en la imagen todos los paquetes de Debian y ficheros que necesita debootstrap. Además, la imagen generada podrá ser usada para instalar en un ordenador el sistema base de Debian.

Tiene un archivo de configuración donde esta las principales directivas a configurar. Entre ellas repositorio a usar, desde donde dfsbuild descargará los paquetes incluidos en el LiveCD/DVD (por defecto testing), así como el Kernel a usar y la lista de paquetes a incluir en el LiveCD/DVD. [4]

Ventajas y Desventajas:

- Libre, licencia GPL.
- Se puede elegir la rama de Debian para hacer el LiveCD/DVD
- Soporta la creación de LiveCD/DVD para varias arquitecturas (amd64, i386, alpha, powerpc).
- Puede guardar contenido de archivos de configuración.
- La inclusión de terceros paquetes es complicada.
- Cuando ocurre un error, es posible continuar desde este punto, pero no es funcional la mayoría de las veces.

1.2.2 bootcd

Otra de las herramientas analizadas es **bootcd**, el cual construye un LiveCD funcional de un sistema Debian GNU/Linux con el comando bootcdwrite. Además puede crear una imagen ISO a través de NFS en un sistema remoto.

Cuando se ejecuta el sistema desde el CD no es necesario ningún disco. Todos los cambios se harán en RAM de la PC. Para la reutilización de los cambio en el próximo arranque puede guardar la configuración personal en un disquete con el comando bootcdflopcp. Si el arranque de la unidad de CD no es compatible, arrancar desde disquetes es también posible.

Si bootcd encuentra algún problema avisará y permitirá ignorarlo, continuar o cancelar. El problema más

habitual es que la imagen creada no quepa en un CD o un DVD, para que el sistema quepa en un CD tendrá que ser realmente minimalista. [5]

Principales Características:

- Soporta como gestores de arranque: Lilo, Gub, Syslinux.
- Es posible hacer copias de seguridad.
- Disponible para Debian Etch.
- No comprime con sistemas eficientes de compresión como squashfs.
- El fichero de configuración es bastante pobre.
- El proceso es totalmente cerrado, la interacción con el usuario es mínima.

Esta aplicación tiene muchas características que no la hacen ideal para la generación fácil de un LiveCD/DVD, las fundamentales son que no comprime el sistema a usar y la pobre configuración de sus opciones.

1.2.3 LiveCD-Tools

Esta es otra de las herramientas para la generación de LiveCD's en Fedora y sus derivados, incluidos los sistemas basados en las distribuciones como RHEL, CentOS y otros.

Para crear LiveCD/DVD con esta herramienta, lo que se necesita es un conjunto de archivos de configuración, además una lista de paquetes y la descripción del tipo de configuración del sistema de arranque.

Estos ficheros de configuración están empaquetados como RMPs y se guardan en el repositorio. La distribución de Fedora y sus repositorios, pasan la versión anterior a la versión actual, manteniendo la actualización de los archivos de configuración.

Existen tres paquetes encargados de hacer distintos tipos de LiveCD/DVD. Los paquetes siguen un régimen de sucesión que ayuda en la creación de las versiones derivadas de Fedora, a continuación los mismos:

1. Fedora-livecd es un LiveCD mínimo sin interfaz de usuario.
2. Fedora-livecd-gnome se basa en fedora-base e incluye un entorno de escritorio GNOME.
3. Fedora-livecd-tools se basa en la parte superior del paquete fedora-gnome, con un entorno

GNOME con otras aplicaciones y funcionalidades. [6]

Principales Características

- Posibilidad de instalar software mientras este ejecutándose el LiveCD/DVD. Usando UnionFs.
- Puede hacer LiveCD/DVD para unidades USB y discos duros.
- Utiliza SELinux para el cumplimiento de las funciones de Seguridad.
- Capacidad de crear imágenes para CD-ROM y DVD.
- Hace un LiveCD/DVD fácil de utilizar con sus paquetes y sus repositorios definidos.
- Persistencia de los datos de la sesión de trabajo del usuario y todas las configuraciones en el momento de la creación del LiveCD/DVD.

Sin dudas una herramienta que tiene grandes potencialidades para la generación de sistemas autónomos, pero el ser solamente para Fedora le resta en cuanto a uso en otras distribuciones.

1.2.4 Live-magic

Live-Magic es una herramienta que crea LiveCD/DVD de Debian, imágenes netboot, entre otras, todo esto lo hace con la ayuda de live-helper. Este tiene una interfaz con características simples para la construcción de los LiveCD/DVD. [7]

Principales Características

- Live-helper genera además de ISOs para CD, DVD de arranque, imágenes para dispositivos USB, USB Thumb, y también imágenes netboot.
- La interfaz permite la creación de un LiveCD/DVD de forma fácil al usuario.
- Es difícil realizar la inclusión de paquetes personalizado.

Esta no es más que una interfaz gráfica que permite generar un LiveCD/DVD usando live-helper como aplicación base. Muchas veces en las pruebas que se le realizaron esta fallaba o se interrumpía el proceso de creación, además de no tener gran cantidad de opciones de configuración.

1.2.5 Live-Helper

Es una herramienta de creación de LiveCD/DVD en Debian GNU/Linux, se trata de un CD estándar similar a los instaladores que permite arrancar un ordenador pero a diferencia de los instaladores su finalidad es dar una utilización totalmente funcional al ordenador sin necesidad de tocar en absoluto la configuración original del sistema instalado en el ordenador, si lo hubiera. [8]

Principales Características:

- Es una aplicación que trabaja por comandos en consola.
- La arquitectura y rama de esta herramienta crea por defecto la imagen ISO a partir de los repositorios de la rama estable o el repositorio que desee el usuario.
- Selecciona el repositorio más cercano, como Live-helper que trae sus repositorios por defecto, esta opción te permite ponerle los repositorios que desee en el LiveCD/DVD. Permite la inclusión de un repositorio local evitando lo engorroso de la descarga de paquetes.
- Funciona en equipos de bajo rendimiento permitiendo el desarrollo de un LiveCD/DVD personalizado con entorno de escritorio de modo texto, funcionando con 48 MB de RAM y Swap.
- Permite agregar o eliminar paquetes mediante la ejecución del LiveCD/DVD.
- Permite la modificación del sistema de arranque.
- Permite la edición de la estructura de directorios del ISO y reconstruirlo a partir del sistema binario.
- Agregar un instalador al LiveCD/DVD, ofreciéndole al usuario la opción de poder seleccionar si arranca el LiveCD/DVD o el instalador del mismo.

Esta herramienta es la oficial usada por el equipo de desarrollo de Debian GNU/Linux para la generación de sus medios autoarrancables y teniendo en cuenta que es una de las más completas no cumple con objetivos perseguidos por los autores de esta investigación.

1.2.6 Remastersys

Remastersys es un programa libre para la distribución Ubuntu Linux usado para modificar los LiveCD/DVD de Ubuntu o de sus derivados, y permite además hacer una copia de seguridad de todo un sistema, incluyendo los datos de usuario en un LiveCD/DVD instalable.

Permite crear dos tipos de imágenes:

1. Una imagen completa del disco duro incluyendo los datos de la carpeta /home
2. Una imagen completa del disco duro pero sin incluir los datos de la carpeta /home.

Esta última se usa para redistribuir el sistema.

Inicialmente fue creado para ser capaz de hacer copias de seguridad o crear una copia de una distribución Ubuntu o derivado de esta. La inspiración para hacer esta aplicación procedía de mklivecd, script que usa Mandrila, además de "remasterme" script que se encuentra en PCLinuxOS.
La imagen ISO creada también puede instalarse en una memoria USB, o para la creación de un LiveUSB.
Dispone de una versión de línea de comandos y una versión en interfaz gráfica. En la actualidad se trabaja con Ubuntu, Linux Mint y Klikit-Linux y posiblemente más distribuciones basadas en Ubuntu. [9]

1.2.7 Linux Live Scripts

Linux Live no son más que un conjunto de scripts de consola desarrollados en Bash que permite generar un LiveCD de una distribución de Linux ya previamente instalada. Crea un sistema autoarrancable desde dispositivos USB Flash Drive, USB Pen Drive, cámara conectada al puerto USB o un iPod. [10]

Principales Características:

- Utiliza los cambios en los parámetros de arranque para especificar persistencia en los ficheros y directorios. Esto funciona gracias a la superposición posixovl (un sistema de ficheros de usuario, ya incluido).
- Utiliza LZMA para agregar o eliminar módulos adicionales para el sistema de ficheros raíz durante la creación del LiveCD/DVD, mientras que su sistema operativo se está ejecutando.
- Configura el sector de arranque para ejecutar Linux desde LiveCD. Si se combinan con la persistencia de los cambios, LiveCD/DVD se comportará como instalado, todos los cambios son almacenados transparente para el medio de arranque (incluso en un USB).

1.2.8 UCK

UCK (Ubuntu Customization Kit) es un kit de herramientas para Ubuntu, Kubuntu, Xubuntu y Edubuntu que sirve para crear LiveCD "mejorados" a partir de las imágenes ISO de los CD oficiales de instalación de Ubuntu.

Crear LiveCD arrancable predefinidos basados en las lenguas originales de Ubuntu o Kubuntu LiveCD utilizando un asistente con una interfaz gráfica.
Permite crear un LiveCD con características especiales también mediante scripts. Es posible personalizar el sistema de ficheros raíz instalando y eliminando paquetes así como el initrd, añadiendo y eliminado módulos al Kernel de Linux. [11]

Al estar desarrollada para Ubuntu Linux y estar especializada en la modificación de los LiveCD/DVD en vez de en la creación de los mismo no es una herramienta ayude mucho en la investigación.

1.2.9 Reconstructor

Reconstructor utiliza los LiveCD oficiales de Ubuntu Linux como base y permite la personalización de las pantallas de inicio (usplash), la configuración de gnome, y el software (también puede usar el entorno chroot para hacer otros cambios antes de crear el LiveCD/DVD). Mantiene la sólida base de Ubuntu y sólo permite la personalización. Reconstructor está escrito en Python y está licenciado bajo la GNU General Public License (GPL). [12]

Principales Características:

- Posibilidad de añadir nuevo software a la recopilación.
- Capacidad para permitir y añadir todos los repositorios personalizados.
- Genera un ISO para grabar en un CD o DVD, con el objetivo de que funcione en cualquier PC.
- Capacidad para utilizar programas gráficos dentro de chroot (experimental).

Como la aplicación anteriormente analizada, esta se especializa en la modificación de los LiveCD/DVD en vez de en la creación de los mismo tampoco es una herramienta ayude mucho en la investigación.

1.2.10 nova-livecdkit

La herramienta LiveCD Kit esta desarrollada en el lenguaje de programación Bash brindando la facilidad de poder ser mantenida por cualquier desarrollador de forma muy sencilla, es altamente flexible, y sirve para generar un LiveCD al gusto del usuario, la parte más difícil es a la hora de configurar las variables de entorno ya que se debe poseer un mínimo de conocimiento del sistema operativo que se vaya a usar.

Permite mantener un control de versiones de cada LiveCD utilizando un nombre de perfil, una fecha y una versión así como mantener varios perfiles de forma que el usuario puede mantener varios proyectos a la vez. Además posibilita hacer backups completos y restaurarlos en caso de que se vaya a hacer algo de lo que no se este seguro.

Brinda una estimación del tamaño máximo que puede ocupar su sistema para no exceder el tamaño estándar de un ISO de 700 MB. El sistema posee un menú intuitivo que permite realizar la operación solicitada. Al ser multiplataforma, es decir que puede generarse un LiveCD desde cualquier plataforma GNU/Linux. Otra característica del sistema es que permite hacer chroot al sistema original o al sistema objeto. Para eliminar o adiciona paquetes así como para modificar archivos de configuración. [13]

Después de haber analizado todas estas herramientas llegamos a la conclusión de que la naturaleza de las herramientas existentes es muy variada, por lo que la aplicación que se desarrollará tendrá muchas de las características de estos sistemas ya analizados.

Dentro de las herramientas que más se aproximan a las necesidades iniciales se encuentra entre otros live-helper, una aplicación que se encuentra en los repositorios de Debian GNU/Linux y que desafortunadamente solo a nivel de consola brinda una gran cantidad de opciones de configuración y funcionalidades.

Otra de las aplicaciones analizadas que se acerca bastante a la idea inicial es live-magic, una aplicación escrita en python y GTK (*Gimp Tool Kit*) que brinda una interfaz de usuario para la aplicación live-helper bastante agradable al usuario pero que aun no brinda todas las opciones de configuración básicas que se requieren.

Debido a lo antes expuesto, se creará una herramienta que herede gran parte de las funcionalidades de las aplicaciones analizadas y que al final sea posible la creación de un LiveCD/DVD de forma rápida y sencilla.

1.3 Herramientas, lenguajes y tecnologías a utilizar.

Para el desarrollo de todo sistema informático es de vital importancia la selección de las herramientas, lenguajes y tecnologías a utilizar, paso que garantizará, de realizarse correctamente, un óptimo desempeño del sistema. Con el objetivo de implementar esta aplicación, a la cual se refiere el presente trabajo, la selección se realizó teniendo en cuenta la infraestructura tecnológica de la UCI y valorando que el sistema a desarrollar, estaría orientado a funcionar sobre el sistema operativo GNU/Linux.

1.3.1 Librerías GTK

En sus comienzos GTK fue desarrollada como un conjunto de herramientas para el Gimp. El GTK es una librería construida sobre GDK (el conjunto de herramientas de dibujo del Gimp), que a su vez no es más que una implementación de las funciones de Xlib. Conviene aclarar que en estos momentos GTK está siendo utilizado en muchos otros proyectos libres aparte del Gimp.

GTK es un API (*Application Programming Interface*) orientado a objetos. Aunque está completamente escrita en C, soporta la idea de clases y funciones de respuesta (es decir punteros a funciones).

Existe un tercer componente llamado glib que proporciona un sustituto a algunas llamadas que podríamos denominar estándar. También incorpora funciones adicionales para manejar listas enlazadas, etc. Las funciones sustituto son usadas para aumentar la portabilidad de GTK, ya que algunas de las funciones incluidas no se encuentran disponibles en otros entornos Unix, como es el caso de g_strerror(). Otras simplemente son versiones mejoradas de las que proporciona libc. Por ejemplo g_malloc() proporciona métodos de depuración que no se encuentran la versión de libc. [14]

Las librerías GTK son muy poderosas para el diseño de aplicaciones gráficas tanto en sistemas Linux como en Microsoft Windows. El tiempo entre versiones de las librerías es muy grande. Por otro lado tienen gran cantidad de dependencia y para hacer algo muy sencillo se tiene que escribir mucho código. En el desarrollo de esta aplicación se tomarán también en cuenta las librerías Qt por su sencillez y buena documentación.

1.3.2 Librerías Qt

Inicialmente Qt apareció como biblioteca desarrollada por Trolltech (en aquel momento "Quasar Technologies") en 1992 siguiendo un desarrollo basado en el código abierto, pero no libre.

Se usó activamente en el desarrollo del escritorio KDE (entre 1996 y 1998), con un notable éxito y rápida expansión. Esto fomentó el uso de Qt en programas cerrados para el escritorio, situación vista por el proyecto GNU como amenaza para el software libre. Para contrarrestar la situación se plantearon dos ambiciosas iniciativas: por un lado el equipo de GNU en 1997 inició el desarrollo del entorno de escritorio GNOME con GTK+ para GNU/Linux. Por otro lado intentaron hacer una biblioteca compatible con Qt pero totalmente libre, llamada Harmony.

En noviembre de 1998, se anunció el cambio de licencia de Qt que, a pesar de todo, no contaba con el beneplácito de la Free Software Foundation. El 4 de septiembre de 2000, Trolltech comenzó a ofrecer las bibliotecas Qt bajo la licencia GNU versión 2.1 y el 18 de enero de 2008, Trolltech anunció que también ofrecerá Qt bajo la licencia GNU v3.

Qt cuenta actualmente con un sistema de triple licencia: GNU v2 y GNU v3 para el desarrollo de software de código abierto (open source) y software libre, y otra de pago para el desarrollo de aplicaciones con cualquier licencia.

Hoy la última versión de las bibliotecas es la 4.x, y además de las múltiples mejoras, son también liberadas bajo licencia GNU para Windows y Mac. [15]

1.3.2.1 Potencialidades de Qt 4.

Las librerías de Qt están implementadas usando el lenguaje de programación C++ de forma nativa y además existen bindings para C, Python (PyQt), Java (Qt Jambi), Perl (PerlQt), Gambas (gb.qt), Ruby (QtRuby), PHP (PHP-Qt) y Mono (Qyoto) entre otros.

El API de la biblioteca cuenta con métodos para acceder a bases de datos mediante SQL, así como uso de XML y una multitud de otros para el manejo de ficheros, además de estructuras de datos tradicionales.

Entre las principales funcionalidades de Qt se encentran la posibilidad de hacer aplicaciones en modo línea

de comando sin interfaz gráfica y además se añaden las siguientes nuevas tecnologías:

Tulip, un nuevo conjunto de plantillas contenedoras de clases más ligeras, seguras y fáciles de usar que los contenedores anteriores.
Interview, una arquitectura de representación de elementos tanto en forma de árbol, como de tablas y lista.
Arthur, el nuevo motor de representación gráfica se integrará con dibujado por pixel y vectores. Además QuickDraw bajo Mac OS X, Xlib en X11, GDI en Windows, así como PostScript y OpenGL.
Scribe, nuevo motor de representación de texto que soporta Unicode, texto enriquecido y la posibilidad de texto rodeando y formas irregulares.
Mainwindow, Una nueva arquitectura basada en acciones de menús, ventanas, barras de herramientas y widgets anclados en cualquier parte de la ventana (arriba, debajo, izquierda y derecha).

Después de haber visto las diferentes características y potencialidades que tienen las librerías Qt se ha decidido llevar a cabo el desarrollo de la aplicación propuesta usando las mismas. Pues brindan todas las herramientas que se podrían necesitar.

1.3.3 Kdevelop

El Proyecto KDevelop surgió en 1998 con el fin de desarrollar un IDE (Entorno de desarrollo integrado) para sistemas Linux y otros sistemas UNIX, orientado al uso bajo el entorno gráfico KDE, aunque también funciona con otros entornos, como Gnome, es totalmente gráfico y combinado con el QtDesigner permite realizar aplicaciones con una gran apariencia en poco tiempo. Está disponible bajo la licencia GPL y funciona con distintos lenguajes de programación como C, C++, Java, Ada, SQL, Python, Perl y Pascal, así como guiones para el intérprete de comandos Bash. [16]

El propio Kdevelop incluye los tutoriales y documentación de QT y KDE, de modo que basta con seguirlos para aprender a programar sobre ese entorno desde cero sin necesidad de documentación añadida.

Principales Características:

KDevelop usa por defecto el editor de texto Kate. Las características que se mencionan a continuación son específicas del entorno de desarrollo:

- Editor de código fuente con destacado de sintaxis e indentado automático (Kate).
- Gestión de diferentes tipos de proyectos, como Automake, qmake (para proyectos basados en la biblioteca Qt y Ant (para proyectos basados en Java).
- Navegador entre clases de la aplicación.
- Front-end para gcc, el conjunto de compiladores de GNU.
- Front-end para el depurador de GNU.
- Asistentes para generar y actualizar las definiciones de las clases y el framework de la aplicación.
- Completado automático del código en C y C++.
- Compatibilidad nativa con Doxygen.
- Permite control de versiones.

KDevelop es el entorno de desarrollo por defecto en el entorno de escritorio K, pero después de bastante tiempo de desarrollo se ha convertido en un IDE poderoso pero a la vez lento en máquinas de pocas prestaciones que son las que predominan en Cuba, por lo que el proceso de configuración puede llegar a ser complicado y la compilación lenta en ocasiones. Teniendo en cuenta que la aplicación requerida no será muy grande se necesita un IDE sencillo y rápido como lo es Qt Creator.

1.3.4 Code::Blocks

Code::Blocks, es una herramienta de entorno de desarrollo integrado para el desarrollo de programas en lenguaje C++. Está basado en el conjunto de librerías WxWidgets, lo que le permite correr libremente en diversos sistemas operativos, y es de licencia GPL. [17]

Soporte de Compiladores:

Debido a que en sí Code::Blocks es sólo la interfaz del entorno de desarrollo, puede enlazarse a una variedad de compiladores para poder desarrollar su trabajo. Por defecto, Code::Blocks buscará una serie de compiladores y configurará los que encuentre. En particular, usa MinGW.

Otros compiladores soportados:

- Microsoft Visual Studio Toolkit (una extensión de compilador de C++ de Microsoft).
- GCC, en sus versiones para Microsoft (ya sea MinGW o Cygwin) y Linux.
- Digital Mars Compiler.
- Intel C++ Compiler.

Todos estos compiladores pueden ser detectados automáticamente si están ya instalados al iniciar Code::Blocks.

Características de Edición:

Entre otras, Code::Blocks soporta varias de las características ya consideradas "clásicas" y que sirven de apoyo a la programación, como el coloreo de sintaxis, el auto código (generar plantillas de código para proyectos), el auto completado, etc.

Compatibilidad con Bibliotecas:

Code::Blocks trae integradas plantillas para generar varias clases de programas, ya sea la clásica aplicación de consola, bibliotecas estáticas o dinámicas, o proyectos completos enlazados con populares bibliotecas como OpenGL y SDL; sin embargo, Code::Blocks integra sólo las plantillas, las bibliotecas deben instalarse por separado.

Otras de las características de Code::Blocks:

- Importación de proyectos Visual C++ y Dev-C++.
- Soporte para packs del Dev-C++.
- Inclusión y generación de plug-ins.
- Vista de Árbol de Proyectos.
- Estadísticas y Resumen de código.
- Generación de XML para proyectos.
- Exportación a formatos XML, RTF y de OpenOffice.org.

Este IDE es uno de los más completos para el desarrollo de aplicaciones en C++ sobre Linux. Pero como en

el desarrollo de la presente aplicación se está optando por las librerías Qt no se usará este pues no soporta las mismas.

1.3.5 Qt Creator

Qt Creator es un IDE (Integrated Development Environment) desarrollado por Nokia para el desarrollo de aplicaciones basadas en el conjunto de librerías Qt. La cual es multiplataforma, actualmente existen versiones para Linux, Microsoft Windows XP, Microsoft Windows Vista y Mac OS.
Su objetivo no es remplazar Visual Studio o Eclipse, sino ser un IDE rápido, simple y usable para el desarrollo de aplicaciones para Linux, Windows y Mac OS así como embebidas en dispositivos portátiles como teléfonos y PDAs.

Proporciona un conjunto de características para aumentar la productividad de los desarrolladores experimentados de Qt, y para ayudar a los nuevos usuarios a poner en marcha sus proyectos usando Qt más rápidamente.[18]

Principales Funcionalidades:

- Asistente para la generación de proyectos.
- Pantalla de bienvenida para iniciar rápidamente los últimos proyectos o sesiones.
- Integrado sistema de ayuda sensible al contexto sobre Qt.
- Incluidos módulos de desarrollo como Phonon, XML, OpenGL, entre otros.
- Avanzado editor de código C++.
- Completamiento de código muy preciso.
- Potente herramienta para navegar entre los archivos y clases.
- Integrado Qt Designer para la creación de interfaces gráficas.
- Interfaz gráfica para GDB (Depurador Simbólico de GNU).
- Añadido el conocimiento de la estructura de las clases Qt para acelerar la depuración.
- Construcción de proyectos Qt integrados a qmake. Herramienta de construcción multiplataforma.
- Es posible la gestión de proyectos dentro de Qt Creator.

Debido a las características de este IDE, creemos que es el más idóneo para el desarrollo de la aplicación propuesta. Entre ellas el poderoso sistema de ayuda que incluye con la documentación de las librerías Qt, así como su velocidad y sencillez.

1.3.6 Qt Designer

Es una herramienta muy potente que permite diseñar de forma muy sencilla y rápida ventanas de dialogo con las librerías Qt. Esta herramienta es una aplicación mediante la cual se puede realizar el diseño de aplicaciones GUI de forma gráfica y muy intuitiva. [19]

Principales Características:

- Dispone de una paleta de widgets muy completa, que incluyen los widgets más comunes de las librerías QT. Si además se han instalado las librerías para el desarrollo de aplicaciones KDE, tendremos widgets adicionales.
- El funcionamiento es de estilo Selecciona y dibuja, es decir, basta con seleccionar un tipo de widget en la paleta y luego al ponernos sobre el formulario se dibuja con la geometría que se quiera.
- Las propiedades de un widget cualquiera se pueden cambiar fácilmente en tiempo de diseño con el panel de propiedades.

Signal y Slot:

Los slots y signals (señales) son un mecanismo de comunicación entre objetos, esta es la principal característica de Qt y es el rasgo que hace distintas las librerías Qt del resto de herramientas para la elaboración de GUI, es un mecanismo de comunicación seguro, flexible y totalmente orientado a objetos y por supuesto implementado en C++.

En la programación con GUI se busca que los cambios producidos en un objeto sean comunicados a otros objetos, por ejemplo cuando se hace clic en un botón para que se cierre una ventana, lo que se hace es posibilitar la comunicación entre los dos objetos.

Otras herramientas de diseño de GUI llevan a cabo la comunicación entre objetos usando los llamados callbacks. Un callback es un puntero a una función, con este mecanismo si se quiere procesar una determinada función cada vez que ocurre un evento en un objeto, lo que se hace es pasar un puntero a otra función (el callback) a la función deseada y será esta la que se encargue de llamar al callback en el momento apropiado.

Este tipo de comunicación tiene el inconveniente de no ser totalmente seguro puesto que no se sabe si se

llamará al callback con los argumentos apropiados y además la función que llama al callback debe saber exactamente a que callback llamar, además es un sistema inflexible y no esta orientado a objetos.[20]

1.3.7 Mksquashfs

Como parte de las aplicaciones que se usarán por la herramienta que se desarrollará se encuentra mksquashfs, la cual es una aplicación de línea de comando usada para crear sistemas de ficheros comprimidos con squashfs, un poderoso sistema de ficheros de solo lectura muy útil para almacenar grandes cantidades de datos. Es un sistema de ficheros estable que ha sido probado en varias arquitecturas como PowerPC, i586, Sparc y ARM.

Esta aplicación también puede ser usada para modificar ficheros comprimidos con squashfs, de la misma forma la descompresión es posible, pero con otra herramienta llamada unsquashfs.

Mksquashfs tiene un gran número de opciones para modificar su sistema de ficheros comprimidos de forma rápida, pero la mayoría son para cambiar opciones avanzadas que raramente hace falta cambiar. Con solo un comando en la consola se obtiene el sistema comprimido. [21]

Esta será usada por la herramienta propuesta para comprimir el sistema original a usar en el LiveCD/DVD.

1.3.8 Genisoimage

Otra de las herramientas a usar por la aplicación a desarrollar es genisoimage en cual es un programa para generar imágenes de sistemas de ficheros híbridos ISO9660/Joliet/HFS, que pueden grabarse en un CD o un DVD usando el programa wodim. Genisoimage permite realizar CDs de arranque usando El Torito - extensión a la especificación ISO 9660, además de CDs que funcionen en el sistema de archivos HFS de Macintosh.

Si las opciones de línea de comandos Joliet o HFS híbrido se especifican, genisoimage creará el sistema de archivos de metadatos adicionales necesarios para Joliet o HFS. En caso contrario se generará un sistema de archivos ISO9660 puro.

Esta aplicación tiene una gran cantidad de opciones de configuración lo que lo hace una herramienta

potente y robusta. La misma es una continuación de la aplicación mkisofs, por lo que hereda todas sus características y funcionalidades.

En distribuciones como Debian, esta aplicación forma parte del paquete cdrkit, el cual incluye un conjunto de herramientas para el trabajo con medios extraíbles. [22]

Esta aplicación será usada para crear la imagen ISO arrancable que contendrá el LiveCD/DVD.

1.3.9 cp

Cp es un comando básico de sistemas Linux que se usa para copiar archivos de forma local, o sea en la misma computadora. A diferencia de rsync que puede copiar ficheros entre diferentes ordenadores.

Entre las variadas opciones de configuración que tiene dicha aplicación se encuentran algunas de mayor importancia que permiten copiar un sistema se ficheros solamente o preservar las características de los ficheros que copia.

Otra de sus funcionalidades principales es que permite preguntar al usuario antes de sobre escribir un fichero determinado. Así como seguir o no los enlaces simbólicos.
Para el desarrollo de esta aplicación no se utilizará cp pues no tiene la opción de excluir ficheros y directorio de los datos que se van a copiar. Funcionalidad esta que el programa rsync presenta. Por lo que será el indicado. [23]

1.3.10 Rsync

Rsync es una aplicación para sistemas Unix que ofrece transmisión eficiente de datos incrementales comprimidos y cifrados. Mediante una técnica de delta encoding, permite sincronizar archivos y directorios entre dos máquinas de una red o entre dos ubicaciones en una misma máquina, minimizando el volumen de datos transferidos.

La aplicación rsync provee otras funciones que asisten en la transferencia. Estas incluyen compresión y descompresión de los datos bloque por bloque, utilizando zlib, y soporte para protocolos de cifrado, tal como SSH. Adicionalmente puede utilizarse una aplicación de tunneling para asegurar los datos.

Los principales usos de rsync incluyen espejado (mirroring) o Respaldo de múltiples clientes Unix dentro de un servidor Unix central. Habitualmente se lo ejecuta mediante herramientas de scheduling como cron, para automatizar procesos de sincronización.

Además de archivos, el algoritmo permite copiar directorios, aún recursivamente, así como vínculos, dispositivos, grupos y permisos. No requiere por defecto privilegios de root para su uso.

Esta última característica es el que se usa en esta investigación para aprovechar la capacidad de excluir ficheros y directorios durante la copia de archivos. Característica esta que lo hace idóneo para su uso en la aplicación a desarrollar. [24]

1.3.11 unionfs

Una de las tecnologías que se usarán para el correcto funcionamiento de los LiveCD/DVD se encuentra unionfs, el cual es un servicio para sistemas de archivos de Linux que permite montar un sistema de archivos formado por la unión de otros sistemas de archivos de Linux. Permite que archivos y directorios de sistemas de archivos distintos, conocidos como ramas, se superpongan de forma transparente, formando un único sistema de archivos. Los contenidos de directorios que tienen la misma ruta en las ramas que se combinan aparecerán juntos en un único directorio en el nuevo sistema de archivos virtual.

Unionfs se utiliza principalmente en LiveCD o sistemas sin disco, donde el directorio raíz de sólo lectura se combina con un sistema de archivos tmpfs (que reside en memoria y es modificable). De este modo, todos los archivos de la raíz de sólo lectura pueden ser modificados, y la modificación se mantiene en memoria.

En los LiveCDs se utiliza para superponer un sistema de archivos modificable sobre un medio de sólo lectura. En Knoppix por ejemplo la unión del sistema de archivos en el CD-ROM o DVD puede hacerse con el sistema de archivos contenido en una imagen llamada knoppix.img en una memoria USB que tenga prioridad sobre el sistema de archivos de sólo lectura.

El usuario ve un sistema de archivos en el que es posible añadir y cambiar archivos en cualquier sitio. Lo que ocurre en el nivel físico es que si un archivo pertenece a la rama de sólo lectura es reemplazada con una nueva versión que se almacena en el archivo de imagen. Dado que el sistema de archivos de unión combina las dos ramas de forma transparente, el usuario simplemente ve la nueva versión. [25]

1.3.12 aufs

Aufs (Another UnionFS) es una versión alternativa de unionfs, un servicio para sistemas de archivos Linux que permite montar un sistema de ficheros formado por la unión de otros sistemas de archivos. Esta siendo desarrollado por Junjiro Okajima desde principios del 2006.

Esta tecnología es una re-implementación completa de unionfs, con el ánimo de mejorar la estabilidad, la confiabilidad y el rendimiento. Este también incluye algunos nuevos conceptos como el balance entre ramas escribibles entre otras muchas mejoras. Algunas de estas nuevas ideas han sido implementadas por unionfs en su versión 2.x.

Aufs ha sido escogido como reemplazo para unionfs en la popular distribución autónoma Knoppix desde finales del 2006. Así como en la distribución Slax desde su versión 6.

Debido a su estabilidad y rendimiento se decide usar para el desarrollo de la aplicación propuesta aufs en ves de unionfs por sus características. Además existen paquetes precompilados para Debian GNU/Linux con módulos para el Kernel a usar en el LiveCD/DVD.

1.3.13 Squashfs

Squashfs es otra de las tecnologías que se usarán para el correcto funcionamiento de los LiveCD/DVD generados, el cual es un sistema de archivos comprimido de sólo lectura para Linux. Squashfs comprime archivos, inodos y directorios, y soporta tamaños de bloque de hasta 1024 KB para mayor compresión. Squashfs es software libre (licenciado como GPL).

Squashfs está pensado para ser usado como sistema de archivos genérico de sólo lectura en dispositivos de bloques/sistemas de memoria limitados (por ejemplo, sistemas integrados), donde se requiere poca sobrecarga. La versión estándar de Squashfs utiliza compresión mediante gzip, aunque existe también otro proyecto que dota de compresión LZMA a Squashfs.

Squashfs se utiliza en las versiones en Live CD de Debian, Finnix, Gentoo Linux, Ubuntu y Mandriva, y en sistemas integrados como firmwares de dispositivos. A menudo se combina con un sistema de archivos de unión con otros sistemas de archivos, como UnionFS o aufs, para proveer un entorno de lectura-escritura

para LiveCD/DVDs. De este modo se combinan las ventajas de la alta velocidad de compresión de SquashFS con la posibilidad de alterar la distribución mientras se ejecuta ésta desde un LiveCD/DVD. Distribuciones como Slax, Debian Live, Mandriva One y Nova Linux usan esta combinación. [26]

1.3.14 UML

Con el objetivo de crear los diagramas necesarios para el modelado de la aplicación se usará UML, el cual es un lenguaje de modelado visual que se usa para especificar, visualizar, construir y documentar artefactos de un sistema de software. Se usa para entender, diseñar, configurar, mantener y controlar la información sobre los sistemas a construir.

UML capta la información sobre la estructura estática y el comportamiento dinámico de un sistema. Un sistema se modela como una colección de objetos discretos que interactúan para realizar un trabajo que finalmente beneficia a un usuario externo.

Este lenguaje de modelado pretende unificar la experiencia pasada sobre técnicas de modelado e incorporar las mejores prácticas actuales en un acercamiento estándar.

UML no es un lenguaje de programación. Las herramientas pueden ofrecer generadores de código de UML para una gran variedad de lenguaje de programación, así como construir modelos por ingeniería inversa a partir de programas existentes.

Este es un lenguaje de propósito general para el modelado orientado a objetos. UML es también un lenguaje de modelamiento visual que permite una abstracción del sistema y sus componentes.

Existían diversos métodos y técnicas Orientadas a Objetos, con muchos aspectos en común pero utilizando distintas notaciones, se presentaban inconvenientes para el aprendizaje, aplicación, construcción y uso de herramientas, etc., además de pugnas entre enfoques, lo que generó la creación del UML como estándar para el modelamiento de sistemas de software principalmente, pero con posibilidades de ser aplicado a todo tipo de proyectos. [27]

1.3.15 Bash

Los scripts que se usarán para optimizar el sistema a crear serán desarrolados en **Bash**, que es un intérprete de comandos para Unix, escrito para el proyecto GNU. Su nombre es el acrónimo de bourne-again shell (otro shell bourne) — haciendo un juego de palabras (born-again significa renacimiento en Ingles) sobre el Bourne shell (sh), que fue uno de los primeros intérpretes importantes de Unix.

Hacia 1978 el intérprete Bourne era el intérprete distribuido con el Unix Version 7. Stephen Bourne, por entonces investigador de los Laboratorios Bell, escribió el intérprete Bourne original. Brian Fox escribió el intérprete bash en 1987. En 1990, Chet Ramey se convirtió en su principal desarrollador. Bash es el intérprete predeterminado en la mayoría de sistemas GNU/Linux, además de Mac OS X Tiger, y puede ejecutarse en la mayoría de los sistemas operativos tipo Unix. También se ha llevado a Microsoft Windows por el proyecto Cygwin.

La sintaxis de órdenes de bash es un superconjunto de la sintaxis del intérprete Bourne. La especificación definitiva de la sintaxis de órdenes de bash, puede encontrarse en el bash Reference Manual distribuido por el proyecto GNU. [28]

La mayoría de los shell scripts (guiones de órdenes) Bourne pueden ejecutarse por bash sin ningún cambio, con la excepción de aquellos scripts de shell Bourne que hacen referencia a variables especiales de Bourne o que utilizan una orden interna de Bourne. La sintaxis de órdenes de bash incluye ideas tomadas desde el Korn Shell (ksh) y el C Shell (csh), como la edición de la línea de órdenes, el historial de órdenes, la pila de directorios, las variables $RANDOM y $PPID, y la sintaxis de substitución de órdenes POSIX: $(...). Cuando se utiliza como un intérprete de órdenes interactivo, bash proporciona auto completado de nombres de programas, nombres de archivos, nombres de variables, etc., cuando el usuario pulsa la tecla TAB. [29]

1.3.16 C/C++

En 1970, Ken Thompson, inmerso en el desarrollo de UNIX en los laboratorios Bell, creó el lenguaje B. Se trataba de una versión de BCPL (Basic Combined Programming Language), para una máquina y sistema específico (DEC PDP-7 y UNIX), el cual se adaptó a su funcionamiento particular y necesidades.

Por lo tanto, en 1971, Dennis Ritchie, con su equipo de desarrollo de los laboratorios Bell, inició el

desarrollo de un compilador llamado B que, entre otras cosas, fuera capaz de generar código ejecutable directamente. El "Nuevo B" luego llamado C introducía además algunos conceptos nuevos al lenguaje como los tipos de datos.

C es un lenguaje muy eficiente puesto que es posible utilizar sus características de bajo nivel para realizar implementaciones óptimas. A pesar de su bajo nivel es el lenguaje más portado en existencia, habiendo compiladores para casi todos los sistemas conocidos. Proporciona facilidades para realizar programas modulares así como utilizar código o bibliotecas existentes. [30]

Entre las características de C estas son algunas de las más importantes:

- Un núcleo del lenguaje simple, con funcionalidades añadidas importantes, como funciones matemáticas y de manejo de ficheros, proporcionadas por bibliotecas.
- Es un lenguaje muy flexible que permite programar con múltiples estilos. Uno de los más empleados es el estructurado no llevado al extremo (permitiendo ciertas licencias rupturistas).
- Un sistema de tipos que impide operaciones sin sentido.
- Usa un lenguaje de preprocesado, el preprocesador de C, para tareas como definir macros e incluir múltiples ficheros de código fuente.
- Acceso a memoria de bajo nivel mediante el uso de punteros.
- Un conjunto reducido de palabras clave.
- Los parámetros se pasan por valor. El paso por referencia se puede simular pasando explícitamente el valor de los punteros.
- Punteros a funciones y variables estática, que permiten una forma rudimentaria de encapsulado y polimorfismo.
- Tipos de datos agregados (struct) que permiten que datos relacionados se combinen y se manipulen como un todo.

C++ es un lenguaje imperativo orientado a objetos derivado del C. En realidad un conjunto de C, que nació para añadirle cualidades y características de las que carecía. El resultado es que como su ancestro, sigue muy ligado al hardware subyacente, manteniendo una considerable potencia para programación a bajo nivel, pero se la han añadido elementos que le permiten también un estilo de programación con alto nivel de abstracción. [31]

C++ tiene ciertas características respecto a otros lenguajes de programación. Las más notables son:

Programación Orientada a Objetos:

La posibilidad de la programación orientada a objetos le permite al programador diseñar las aplicaciones desde un punto de vista comunicativo entre objetos más que una secuencia estática de código. Además de posibilitar la reusabilidad de código de un modo más lógico y productivo.

Portabilidad:

Es posible compilar el mismo código C++ en cualquier tipo de computadora o sistema operativo, sin que ocurra algún cambio. C++ es el lenguaje de programación más usado y portado de todo el mundo.

Menos Código:

El código escrito en C++ es mucho más corto que el mismo escrito en otro lenguaje, pues el uso de caracteres especiales es mucho más usado que las palabras claves. Haciendo que el programador ahorre espacio y tiempo.

Programación Modular:

El cuerpo de una aplicación en C++ puede estar separado en múltiples ficheros de código, que son compilados de forma separada y luego enlazarlos. Esto ahora tiempo pues no es necesario compilar todos los ficheros cuando se hace cambios solo en un archivo. Además esta característica permite que se pueda enlazar código C++ con otros como por ejemplo código ensamblador o C.

Compatibilidad con C:

C++ es compatible con el lenguaje C, cualquier código escrito en C puede ser fácilmente incluido en un programa C++ sin hacer ningún cambio.

Velocidad:

El resultado del proceso de compilación es muy eficiente, debido a su dualidad para el trabajo a bajo y alto nivel, y a la reducción del tamaño del código en si.

Teniendo en cuenta todas las potencialidades de C++ será el indicado para desarrollar la aplicación propuesta. De la misma forma se usará Bash para crear los scripts de mantenimiento que se usarán para optimizar el sistema.

1.3.17 XP

Extreme Programming (Programación Extrema), es una metodología ágil de desarrollo que incorpora un conjunto de buenas prácticas conocidas en programación que tienen por objetivo:

- Aplicar un modelo de espiral evolutivo donde el cliente es parte activa de proceso de desarrollo.
- Ser imperativo en la despersonalización del código, es por eso que se aplica programación de pares.
- Obtener un código limpio y legible, sin el temor de reprogramar si es necesario.

Se dio a conocer como metodología en el 2001, en un proyecto desarrollado por Kent Beck en DaimlerChrysler, después de haber trabajado varios años con Ward Cunningham en busca de una nueva aproximación al problema del desarrollo de software, que hiciera las cosas más simples de lo acostumbrado por los métodos existentes. Es una metodología centrada en potenciar las relaciones interpersonales como clave para el éxito en el desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores, y propiciando un buen clima de trabajo.

De todas las metodologías ágiles, ésta es la que ha recibido más atención, pero esta popularidad se ha vuelto un problema, pues ha acaparado la atención fuera de las otras metodologías y sus valiosas ideas. XP se basa en realimentación continua entre el cliente y el equipo de desarrollo, comunicación fluida entre todos los participantes, simplicidad en las soluciones implementadas y coraje para enfrentar los cambios. XP se define como especialmente adecuada para proyectos con requisitos imprecisos y muy cambiantes, y donde existe un alto riesgo técnico. Los principios y prácticas son de sentido común pero llevadas al extremo, de ahí proviene su nombre.

A continuación presentaremos las características esenciales de XP organizadas en los tres apartados siguientes: *historias de usuario, roles, proceso y prácticas*. [32]

Historias de Usuario

Son las técnicas utilizadas para especificar los requisitos del software. Se trata de tarjetas de papel en las cuales el cliente describe brevemente las características que el sistema debe poseer, sean requisitos funcionales o no funcionales. El tratamiento de las historias de usuario es muy dinámico y flexible. Cada historia de usuario es lo suficientemente comprensible y delimitada para que los programadores puedan implementarla en unas semanas.

Los **roles** que XP propone de acuerdo con la propuesta original de Beck son:

***Programador**:* El programador escribe las pruebas unitarias y produce el código del sistema.

***Cliente**:* Escribe las historias de usuario y las pruebas funcionales para validar su implementación. Además, asigna la prioridad a las historias de usuario y decide cuáles se implementan en cada iteración centrándose en aportar mayor valor al negocio.

Encargado de pruebas (Tester): Ayuda al cliente a escribir las pruebas funcionales. Ejecuta las pruebas regularmente, difunde los resultados en el equipo y es responsable de las herramientas de soporte para pruebas.

Encargado de seguimiento (Tracker): Proporciona realimentación al equipo. Verifica el grado de acierto entre las estimaciones realizadas y el tiempo real dedicado, para mejorar futuras estimaciones. Realiza el seguimiento del progreso de cada iteración.

Entrenador (Coach): Es responsable del proceso global. Debe proveer guías al equipo de forma que se apliquen las prácticas XP y se siga el proceso correctamente.

***Consultor**:* Es un miembro externo del equipo con un conocimiento específico en algún tema necesario para el proyecto, en el que puedan surgir problemas.

Gestor (Big boss): Es el vínculo entre clientes y programadores, ayuda a que el equipo trabaje efectivamente creando las condiciones adecuadas. Su labor esencial es de coordinación.

El ciclo de vida en XP es iterativo basado en seis fases fundamentales:

1. ***Exploración**:* Los clientes plantean a grandes rasgos las historias de usuario que son de interés para la primera entrega del producto. Al mismo tiempo el equipo de desarrollo se familiariza con las herramientas, tecnologías y prácticas que se utilizarán en el proyecto, a partir de lo cual se exploran las posibilidades de la arquitectura del sistema construyendo un prototipo.
2. ***Planificación de la Entrega**:* El cliente establece la prioridad de cada historia de usuario, y a su vez, los programadores realizan una estimación del esfuerzo necesario de cada una de ellas. Se toman acuerdos sobre el contenido de la primera entrega y se determina un cronograma en conjunto con el cliente.
3. ***Iteraciones**:* Incluye varias iteraciones sobre el sistema antes de ser entregado. En la primera iteración se puede intentar establecer una arquitectura del sistema que pueda ser utilizada durante el resto del proyecto. Al final de la última iteración el sistema estará listo para entrar en producción.
4. ***Producción**:* Esta fase requiere de pruebas adicionales y revisiones de rendimiento antes de que el

sistema sea trasladado al entorno del cliente. Al mismo tiempo, se deben tomar decisiones sobre la inclusión de nuevas características a la versión actual, debido a cambios durante esta fase.

5. ***Mantenimiento**:* Mientras la primera versión se encuentra en producción, el proyecto XP debe mantener el sistema en funcionamiento al mismo tiempo que desarrolla nuevas iteraciones. Para realizar esto se requiere de tareas de soporte para el cliente.
6. ***Muerte del Proyecto:*** Ocurre cuando el cliente no tiene más historias para ser incluidas en el sistema, siendo satisfechas todas sus necesidades. Se genera la documentación final del sistema y no se realizan más cambios en la arquitectura. La muerte del proyecto también ocurre cuando el sistema no genera los beneficios esperados por el cliente o cuando no hay presupuesto para mantenerlo.

El proceso XP es un ciclo de desarrollo que consiste (a grandes rasgos) en los siguientes pasos:

1. El cliente define el valor de negocio a implementar.
2. El programador estima el esfuerzo necesario para su implementación.
3. El cliente selecciona qué construir, de acuerdo con sus prioridades y las restricciones de tiempo.
4. El programador construye ese valor de negocio.
5. Vuelve al paso 1.

- Las **prácticas** que XP propone son:

1. ***El juego de la planificación:*** Hay una comunicación frecuente el cliente y los programadores. El equipo técnico realiza una estimación del esfuerzo requerido para la implementación de las historias de usuario y los clientes deciden sobre el ámbito y tiempo de las entregas y de cada iteración.
2. ***Entregas pequeñas:*** Producir rápidamente versiones del sistema que sean operativas, aunque no cuenten con toda la funcionalidad del sistema. Esta versión ya constituye un resultado de valor para el negocio. Una entrega no debería tardar más 3 meses.
3. ***Metáfora:*** El sistema es definido mediante una metáfora o un conjunto de metáforas compartidas por el cliente y el equipo de desarrollo. Una metáfora es una historia compartida que describe cómo debería funcionar el sistema (conjunto de nombres que actúen como vocabulario para hablar sobre el dominio del problema, ayudando a la nomenclatura de clases y métodos del sistema).
4. ***Diseño simple:*** Se debe diseñar la solución más simple que pueda funcionar y ser implementada en

un momento determinado del proyecto.

5. ***Pruebas:*** La producción de código está dirigida por las pruebas unitarias. Éstas son establecidas por el cliente antes de escribirse el código y son ejecutadas constantemente ante cada modificación del sistema.
6. ***Refactorización (Refactoring):*** Es una actividad constante de reestructuración del código con el objetivo de remover duplicación de código, mejorar su legibilidad, simplificarlo y hacerlo más flexible para facilitar los posteriores cambios. Se mejora la estructura interna del código sin alterar su comportamiento externo.
7. ***Programación en parejas****:* Toda la producción de código debe realizarse con trabajo en parejas de programadores. Esto conlleva ventajas implícitas (menor tasa de errores, mejor diseño, mayor satisfacción de los programadores).
8. ***Propiedad colectiva del código:*** Cualquier programador puede cambiar cualquier parte del código en cualquier momento.
9. Integración continua: Cada pieza de código es integrada en el sistema una vez que esté lista. Así, el sistema puede llegar a ser integrado y construido varias veces en un mismo día.
10. ***40 horas por semana:*** Se debe trabajar un máximo de 40 horas por semana. No se trabajan horas extras en dos semanas seguidas. Si esto ocurre, probablemente está ocurriendo un problema que debe corregirse. El trabajo extra desmotiva al equipo.
11. ***Cliente in-situ:*** El cliente tiene que estar presente y disponible todo el tiempo para el equipo. Éste es uno de los principales factores de éxito del proyecto XP. El cliente conduce constantemente el trabajo hacia lo que aportará mayor valor de negocio y los programadores pueden resolver de manera inmediata cualquier duda asociada. La comunicación oral es más efectiva que la escrita.
12. ***Estándares de programación:*** XP enfatiza que la comunicación de los programadores es a través del código, con lo cual es indispensable que se sigan ciertos estándares de programación para mantener el código legible.

El mayor beneficio de las prácticas se consigue con su aplicación conjunta y equilibrada puesto que se apoyan unas en otras.

La mayoría de las prácticas propuestas por XP no son novedosas sino que en alguna forma ya habían sido propuestas en ingeniería del software e incluso demostrado su valor. El mérito de XP es integrarlas de una forma efectiva y complementarlas con otras ideas desde la perspectiva del negocio, los valores humanos y el trabajo en equipo.

Durante el desarrollo de las fases propuestas en XP se obtienen un conjunto de artefactos:

Artefactos Definidos en XP
Historias de usuarios
Tarjetas de historias
Arquitectura del sistema
Listas de tareas
Gráficos visibles
Plan de Entrega
Plan de la iteración
Velocidad del proyecto

Tabla 2. Artefactos de XP

Sin embargo ninguno de estos artefactos tienen en cuenta el desarrollo del proceso investigativo a la hora de desarrollar un software, por lo que se puede afirmar que XP no cuenta con elementos que favorezcan el desarrollo de esta actividad durante su utilización. [33]

1.3.18 SCRUM

Scrum es una forma de gestionar proyectos de software. No es una metodología de análisis ni diseño, sino una metodología de gestión del trabajo. Surge a partir de un artículo en 1986 de la Harvard Business Review titulado “The New New Product Development Game” de Takeuchi y Nonaka, desarrollada por Ken Schwaber, Jeff Sutherland y Mike Beedle y en el año 1995 es formalizado este proceso. Define un proceso empírico, iterativo e incremental de desarrollo que se fundamenta en la aceptación de la naturaleza caótica del desarrollo de software, y la utilización de prácticas tendientes a manejar la impredictibilidad y el riesgo a niveles aceptables. No está concebido como método independiente, sino que se promueve como complemento de otras metodologías, incluyendo XP, MSF o RUP.

Como método, enfatiza valores y prácticas de gestión de manera que puede ser considerado como un conjunto de patrones organizacionales, sin pronunciarse sobre requerimientos, implementación y demás cuestiones técnicas; de allí su deliberada insuficiencia y su complementariedad. [34]
Sus principales características se pueden resumir en dos:

1. El desarrollo de software se realiza mediante iteraciones, denominadas sprints, con una duración de 30 días, el cual es un incremento ejecutable que se muestra al cliente.
2. Reuniones a lo largo proyecto, destacando una reunión diaria de 15 minutos del equipo de desarrollo para coordinación e integración.

El ciclo de vida de Scrum consta de las siguientes fases:

1. ***Pre-Juego: Planeamiento:*** El propósito es establecer la visión, definir expectativas y asegurarse la financiación. Si se está desarrollando un nuevo sistema, esta fase consiste en la conceptualización y el análisis. Si el sistema ya existe, esta fase consistirá en el análisis limitado.
2. ***Pre-Juego: Montaje (Staging):*** El propósito es identificar más requerimientos y priorizar las tareas para la primera iteración. Se diseñan cómo los requisitos de la reserva serán puestos en ejecución. Esta fase incluye la modificación de la arquitectura del sistema y el diseño de alto nivel.
3. ***Juego o desarrollo:*** El propósito es implementar un sistema listo para entrega en una serie de iteraciones de treinta días llamadas “corridas” (sprints).
4. ***Pos-Juego: Liberación:*** El propósito es el despliegue operacional. Las actividades, documentación, entrenamiento, mercadeo y venta.

Propone dentro de su ciclo de vida la utilización de los artefactos que se muestran a continuación:

Artefactos definidos en SCRUM
Documento visión.
Documento presupuesto.
Documento arquitectura del sistema.
Pila del producto o Lista de Retraso de Producto
Incidentes.
Incremento.
Manual de Usuario.
Casos de Uso
Diseño del sistema
Pila del sprint (Sprint Backlog o Lista de Retraso de Sprint)
Diagramas de flujo de datos.

Tabla 3. Artefactos de SCRUM

Luego de analizar estos artefactos, se pudo comprobar que ninguno de ellos relaciona el desarrollo del

proceso investigativo. Esta metodología enfatiza la forma en que se gestionarán las actividades en el proceso de desarrollo del software, sin especificar el nivel de documentación que debe tener un producto software, por lo que no propone artefactos que tengan en cuenta el desarrollo de esta actividad.

1.3.19 SXP

Es una metodología compuesta por las metodologías SCRUM y XP que ofrece una estrategia tecnológica, a partir de la introducción de procedimientos ágiles que permitan actualizar los procesos de software para el mejoramiento de la actividad productiva fomentando el desarrollo de la creatividad, aumentando el nivel de preocupación y responsabilidad de los miembros del equipo, ayudando al líder del proyecto a tener un mejor control del mismo. SCRUM es una forma de gestionar un equipo de manera que trabaje de forma eficiente y de tener siempre medidos los progresos, de forma que sepamos por dónde andamos.

XP más bien es una metodología encaminada para el desarrollo; consiste en una programación rápida o extrema, cuya particularidad es tener como parte del equipo, al usuario final, pues es uno de los requisitos para llegar el éxito del proyecto. [35]

SXP consta de 4 fases principales:
Planificación-Definición: Se establece la visión, se fijan las expectativas y se realiza el aseguramiento del financiamiento del proyecto;
Desarrollo: Se realiza la implementación del sistema hasta que este listo para ser entregado;
Entrega y Puesta en marcha: Se entrega el software y se comienza a usar.
Mantenimiento: Se realiza el soporte para el cliente.

De cada una de estas fases se realizan numerosas actividades tales como el levantamiento de requisitos, la priorización de la Lista de Reserva del Producto, definición de las Historias de Usuario, diseño, implementación, pruebas, entre otras; de donde se generan artefactos para documentar todo el proceso. Las entregas son frecuentes, y existe una refactorización continua, lo que nos permite mejorar el diseño cada vez que se le añada una nueva funcionalidad.

SXP esta especialmente indicada para proyectos de pequeños equipos de trabajo, rápido cambio de requisitos o requisitos imprecisos, muy cambiantes, donde existe un alto riesgo técnico y se orienta a una entrega rápida de resultados y una alta flexibilidad. Ayuda a que trabajen todos juntos, en la misma

dirección, con un objetivo claro, permitiendo además seguir de forma clara el avance de las tareas a realizar, de forma que los jefes pueden ver día a día cómo progresa el trabajo. A continuación se muestran los artefactos que se generan con esta metodología de software.

Artefactos Generados en SXP	
1	Concepción del Sistema
2	Estado del Arte
3	Modelo de Historias de Usuarios del Negocio
4	LRP(Lista de Reservas de Producto)
5	Historia de Usuario
6	Lista de Riesgo
7	Modelo de Diseño
8	Tarea de Ingeniería
9	Plan de Releases
10	Estándar de Programación
11	Caso de Prueba de Aceptación
12	Manual de Usuario
13	Manual de Identidad
14	Manual de Desarrollo
15	Gestión de Cambios

Tabla 4. Artefactos generados en SXP.

Teniendo en cuenta las características de las distintas metodologías analizadas, para el desarrollo de la presente aplicación se decidió usar SXP como metodología de desarrollo de software ágil.

En el capitulo recién concluido se sistematizaron y evaluaron las principales aplicaciones para crear LiveCD/DVDs. Así como de las diferentes herramientas y sistemas que se usarán para el desarrollo de la aplicación propuesta.

Después del análisis antes expuesto, se acuerda usar el entorno de desarrollo integrado Qt Creator por su usabilidad y robustez. Así como el conjunto de librerías Qt para el diseño gráfico de la aplicación y bash como lenguaje de script; el cual será usado para crear los scripts necesarios para automatizar el proceso de creación de un LiveCD/DVD.

CAPÍTULO II. DESARROLLO ÁGIL DE LA APLICACIÓN.

En este capítulo se realizará el desarrollo ágil de Inoue Live Creator utilizando la metodología ágil **SXP** para el desarrollo de software que se realizó en el año 2008 por la ingeniera Gladys Marsi Peñalver Romero (GLADYS M., P. R. 2008). Se explicará toda la dinámica del proyecto en forma de historias de usuarios, prototipos de interfaz de usuario y algunos modelos auxiliares.

2.1 Roles del Proyecto

Rol	Responsabilidad	Nombre
Gerente	Controla el progreso de desarrollo.	Keiver Hernández Fernández
Cliente	Participa en las tareas de la lista de reserva del producto.	UCI, Comunidad de Software Libre
Programador	Escribir código y pruebas unitarias.	Keiver Hernández Fernández Lucia C. Domínguez Delgado
Analista	Escribir las Historias de Usuario y las pruebas funcionales.	Lucia C. Domínguez Delgado
Diseñador	Diseñar el sistema y los prototipos de interfaces.	Keiver Hernández Fernández
Probador	Ejecutar pruebas regulares.	Lucia C. Domínguez Delgado
Arquitecto	Estructurar y diseñar en conjunto con el Analista y el diseñador.	Keiver Hernández Fernández

2.2 Historias de Usuario y Prototipos de Interfaz

<table>
<tr><th colspan="2">Historia de Usuario</th></tr>
<tr><td>Número:
UH-ILC-01</td><td>Nombre Historia de Usuario:
Programar Scripts Auxiliares</td></tr>
<tr><td colspan="2">Modificación de Historia de Usuario Número:</td></tr>
<tr><td>Usuario:

Keiver Hernández Fernández

Lucia C. Domínguez Delgado</td><td>Iteración Asignada: 1</td></tr>
<tr><td>Prioridad en Negocio: Alta

(Alta/ Media/ Baja)</td><td>Puntos Estimados: 1</td></tr>
<tr><td>Riesgo en Desarrollo: Alta

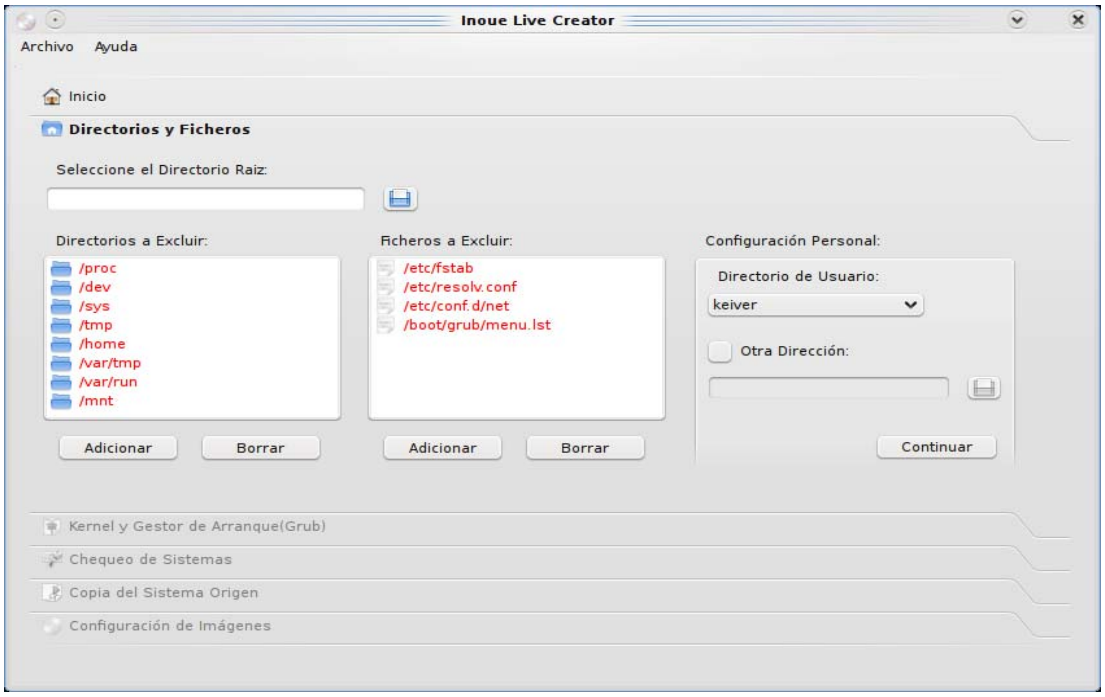
(Alta/ Media/ Baja)</td><td>Puntos Reales: 1</td></tr>
<tr><td colspan="2">Descripción: Se programan los scripts para optimizar las configuraciones y los permisos de la aplicación. Estos scripts deben permitir montar las particiones /dev y /proc, ejecutar tareas de mantenimiento como la creación y eliminación de ficheros entre otras.</td></tr>
<tr><td colspan="2">Observaciones: Los scripts se programarán usando Bash. Los mismos permitirán optimizar el sistema del LiveCD/DVD.</td></tr>
</table>

Tarea de Ingeniería	
Número Tarea: 1	**Número Historia de Usuario:** UH-ILC-01
Nombre Tarea: Crear script para optimizar el sistema.	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 03/01/09	**Fecha Fin:** 04/01/09
Programador Responsable: Keiver Hernández Fernández Lucia C. Domínguez Delgado	
Descripción: Este script permite modificar el sistema para que sea más óptimo. Entre las tareas que debe realizar se encuentra montar las particiones necesarias y ejecutar tareas de mantenimiento como la creación y eliminación de ficheros entre otras.	

Tarea de Ingeniería	
Número Tarea: 2	**Número Historia de Usuario:** UH-ILC-01
Nombre Tarea: Crear scripts que copia los datos personales del usuarios seleccionado.	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 05/01/09	**Fecha Fin:** 06/01/09
Programador Responsable: Keiver Hernández Fernández Lucia C. Domínguez Delgado	
Descripción: Este script permite copiar los datos personales del usuario seleccionado para el LiveCD/DVD. Los datos se copian a /etc/skel.	

Tarea de Ingeniería	
Número Tarea: 3	**Número Historia de Usuario:** UH-ILC-01
Nombre Tarea: Crear script para desmontar las particiones usadas por el script chroot.sh.	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 06/01/09	**Fecha Fin:** 07/01/09
Programador Responsable: Keiver Hernández Fernández Lucia C. Domínguez Delgado	
Descripción: Este script permite desmontar las particiones usadas por el script chroot.sh.	

Historia de Usuario	
Número: UH-ILC-02	**Nombre Historia de Usuario:** Obtener parámetros de configuración.
Modificación de Historia de Usuario Número:	
Usuario: Keiver Hernández Fernández Lucia C. Domínguez Delgado	**Iteración Asignada:** 1
Prioridad en Negocio: Alta (Alta/ Media/ Baja)	**Puntos Estimados:** 1
Riesgo en Desarrollo: Alta (Alta/ Media/ Baja)	**Puntos Reales:** 1
Descripción: Se obtiene los parámetros para las configuraciones de la imagen a generar. Entre ellos la dirección del sistema a usar, la lista de directorios y ficheros que se excluirán del LiveCD/DVD y el usuario del cual se copiarán los datos personales.	



Tarea de Ingeniería	
Número Tarea: 1	**Número Historia de Usuario:** UH-ILC-02
Nombre Tarea: Tomar nombre y ruta del proyecto.	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 08/01/09	**Fecha Fin:** 08/01/09
Programador Responsable: Keiver Hernández Fernández Lucia C. Domínguez Delgado	
Descripción: Obtener el nombre y la dirección donde se creará el nuevo proyecto.	

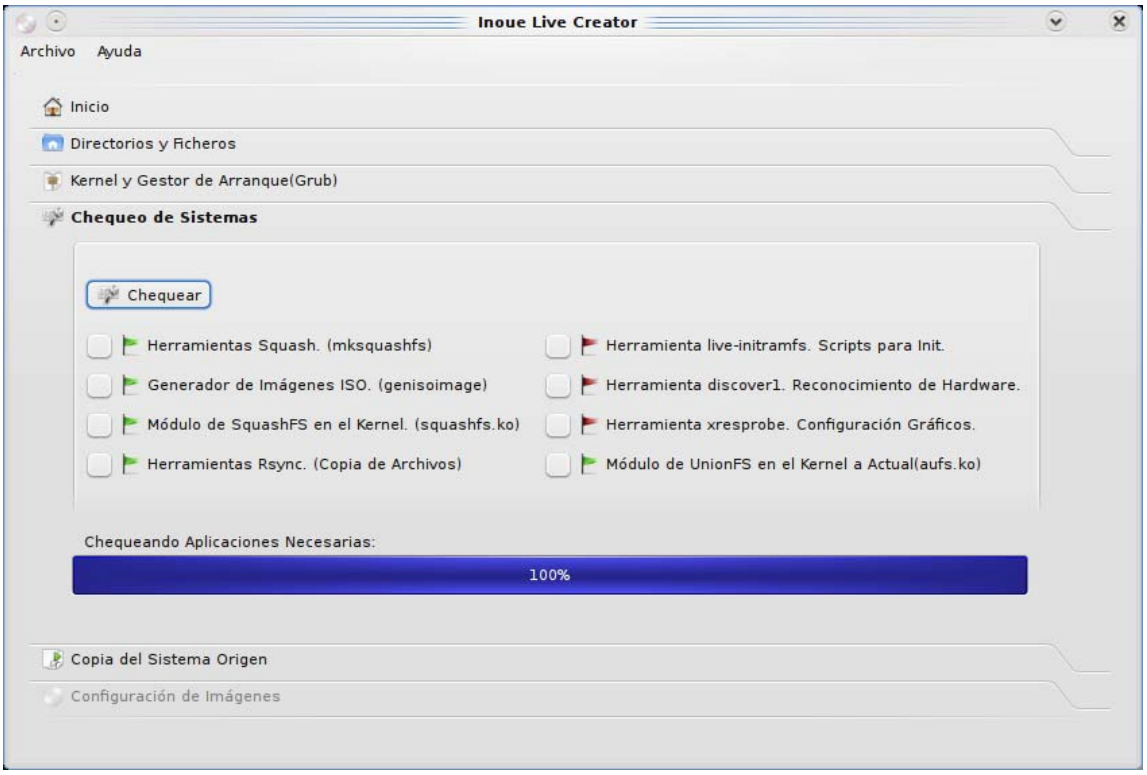
Tarea de Ingeniería	
Número Tarea: 2	**Número Historia de Usuario:** UH-ILC-02
Nombre Tarea: Tomar la ruta del sistema origen	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 09/01/09	**Fecha Fin:** 09/01/09
Programador Responsable: Keiver Hernández Fernández Lucia C. Domínguez Delgado	
Descripción: Obtener la dirección del sistema origen entrada por el usuario.	

Tarea de Ingeniería	
Número Tarea: 3	**Número Historia de Usuario:** UH-ILC-02
Nombre Tarea: Obtener lista de directorios y ficheros a excluir.	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 12/01/09	**Fecha Fin:** 12/01/09
Programador Responsable: Keiver Hernández Fernández Lucia C. Domínguez Delgado	
Descripción: Obtener la lista de los directorios y ficheros que se excluirán en el momento de crear el LiveCD/DVD.	

Tarea de Ingeniería	
Número Tarea: 4	**Número Historia de Usuario:** UH-ILC-02
Nombre Tarea: Obtener valores de configuración del gestor de arranque.	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 13/01/09	**Fecha Fin:** 14/01/09
Programador Responsable: Keiver Hernández Fernández Lucia C. Domínguez Delgado	
Descripción: Obtener los valores para el gestor de arranque (Grub) como la imagen Splash y el título del LiveCD/DVD.	

Tarea de Ingeniería	
Número Tarea: 5	**Número Historia de Usuario:** UH-ILC-02
Nombre Tarea: Obtener ruta del Kernel y la imagen initrd a usar.	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 16/01/09	**Fecha Fin:** 17/01/09
Programador Responsable: Keiver Hernández Fernández Lucia C. Domínguez Delgado	
Descripción: Obtener las direcciones del Kernel y el Initrd para el LiveCD/DVD. Es posible seleccionar un kernel que esté instalado en el sistema.	

Historia de Usuario	
Número: UH-ILC-03	**Nombre Historia de Usuario:** Chequear Sistema
Modificación de Historia de Usuario Número:	
Usuario: Keiver Hernández Fernández Lucia C. Domínguez Delgado	**Iteración Asignada:** 2
Prioridad en Negocio: Alta (Alta/ Media/ Baja)	**Puntos Estimados:** 1
Riesgo en Desarrollo: Alta (Alta/ Media/ Baja)	**Puntos Reales:** 1
Descripción: Chequea que las herramientas necesarias para que funcione el sistema estén instaladas. Entre ellas: rsync, xresprobe, discover y live-initramfs.	



Tarea de Ingeniería	
Número Tarea: 1	**Número Historia de Usuario:** UH-ILC-03
Nombre Tarea: Chequear los módulos necesario para el Kernel.	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 20/01/09	**Fecha Fin:** 21/01/09
Programador Responsable: Keiver Hernández Fernández Lucia C. Domínguez Delgado	
Descripción: Verifica que existan los módulos necesarios para el Kernel. Se chequean squashfs y aufs.	

Tarea de Ingeniería	
Número Tarea: 2	**Número Historia de Usuario:** UH-ILC-03
Nombre Tarea: Chequear que este instalada la herramienta genisoimage.	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 22/01/09	**Fecha Fin:** 23/01/09
Programador Responsable: Keiver Hernández Fernández Lucia C. Domínguez Delgado	
Descripción: Verifica que este instalada la herramienta genisoimage. Necesaria para el correcto funcionamiento de la aplicación.	

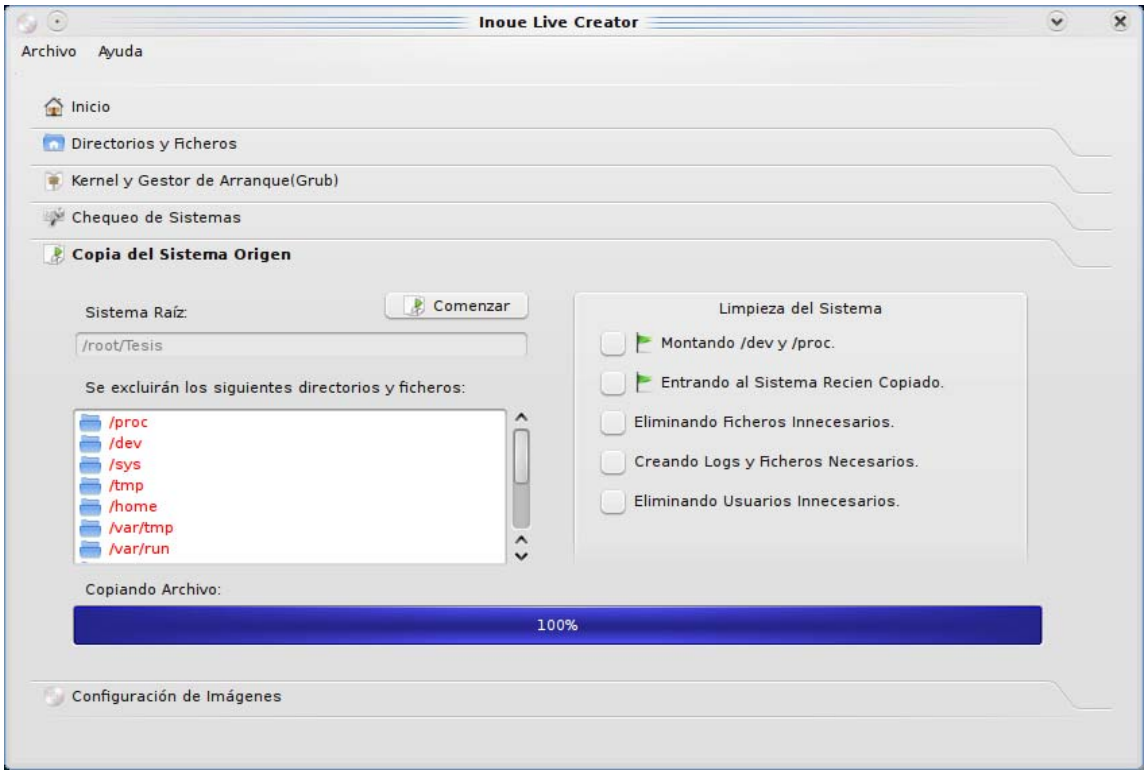
Tarea de Ingeniería	
Número Tarea: 3	**Número Historia de Usuario:** UH-ILC-03
Nombre Tarea: Chequear herramienta live-initramfs.	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 26/01/09	**Fecha Fin:** 27/01/09
Programador Responsable: Keiver Hernández Fernández Lucia C. Domínguez Delgado	
Descripción: Verifica que exista la herramienta live-initramfs. Necesaria para el correcto funcionamiento de la aplicación.	

Tarea de Ingeniería	
Número Tarea: 4	**Número Historia de Usuario:** UH-ILC-03
Nombre Tarea: Chequear la herramienta discover.	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 28/01/09	**Fecha Fin:** 28/01/09
Programador Responsable: Keiver Hernández Fernández Lucia C. Domínguez Delgado	
Descripción: Verifica que exista la herramienta discover. Necesaria para el correcto funcionamiento de la aplicación.	

Tarea de Ingeniería	
Número Tarea: 5	**Número Historia de Usuario:** UH-ILC-03
Nombre Tarea: Chequear la herramienta xresprobe.	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 29/01/09	**Fecha Fin:** 30/01/09
Programador Responsable: Keiver Hernández Fernández Lucia C. Domínguez Delgado	
Descripción: Verifica que exista la herramienta xresprobe. Necesaria para el correcto funcionamiento de la aplicación.	

Tarea de Ingeniería	
Número Tarea: 6	**Número Historia de Usuario:** UH-ILC-03
Nombre Tarea: Chequear la herramienta rsync.	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 29/01/09	**Fecha Fin:** 30/01/09
Programador Responsable: Keiver Hernández Fernández Lucia C. Domínguez Delgado	
Descripción: Verifica que exista la herramienta rsync. Necesaria para el correcto funcionamiento de la aplicación.	

Historia de Usuario	
Número: UH-ILC-04	**Nombre Historia de Usuario:** Copia del sistema origen.
Modificación de Historia de Usuario Número:	
Usuario: Keiver Hernández Fernández Lucia C. Domínguez Delgado	**Iteración Asignada:** 2
Prioridad en Negocio: Alta (Alta/ Media/ Baja)	**Puntos Estimados:** 1
Riesgo en Desarrollo: Alta (Alta/ Media/ Baja)	**Puntos Reales:** 1
Descripción: Se hace una copia del sistema del cual se creará el LiveCD/DVD. Usando rsync, de la cual se excluirán los ficheros y directorios previamente seleccionados.	

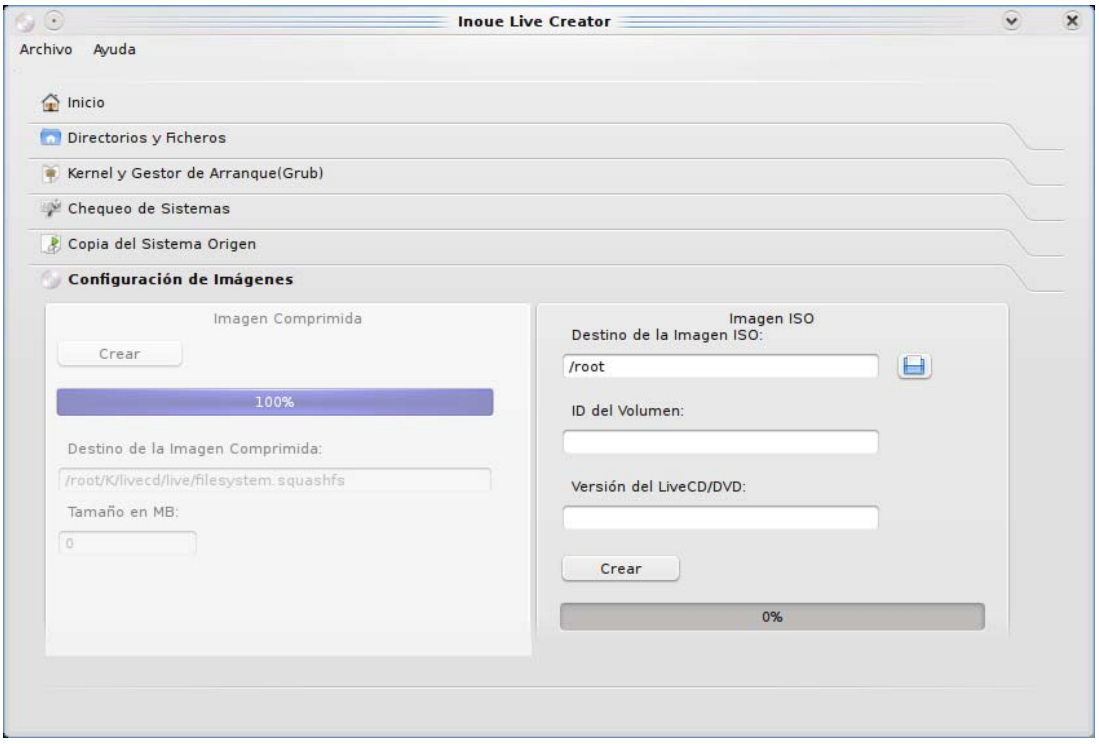


Tarea de Ingeniería	
Número Tarea: 1	**Número Historia de Usuario:** UH-ILC-04
Nombre Tarea: Definir expresiones regulares para obtener el por ciento de copia de un fichero.	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 02/02/09	**Fecha Fin:** 02/02/09
Programador Responsable: Keiver Hernández Fernández Lucia C. Domínguez Delgado	
Descripción: Se determina la expresión regular para el por ciento de copia del sistema. Necesario para mostrarlo en la interfaz gráfica.	

Tarea de Ingeniería	
Número Tarea: 2	**Número Historia de Usuario:** UH-ILC-04
Nombre Tarea: Definir expresiones regulares para obtener el nombre del fichero que se está copiando.	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 03/02/09	**Fecha Fin:** 03/02/09
Programador Responsable: Keiver Hernández Fernández Lucia C. Domínguez Delgado	
Descripción: Se determina la expresión regular para obtener el nombre del fichero que se está copiando. Necesario para mostrarlo en la interfaz gráfica.	

Tarea de Ingeniería	
Número Tarea: 3	**Número Historia de Usuario:** UH-ILC-04
Nombre Tarea: Realizar la copia del sistema original.	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 03/02/09	**Fecha Fin:** 03/02/09
Programador Responsable: Keiver Hernández Fernández Lucia C. Domínguez Delgado	
Descripción: Se copia el sistema original previamente seleccionado usando la herramienta rsync.	

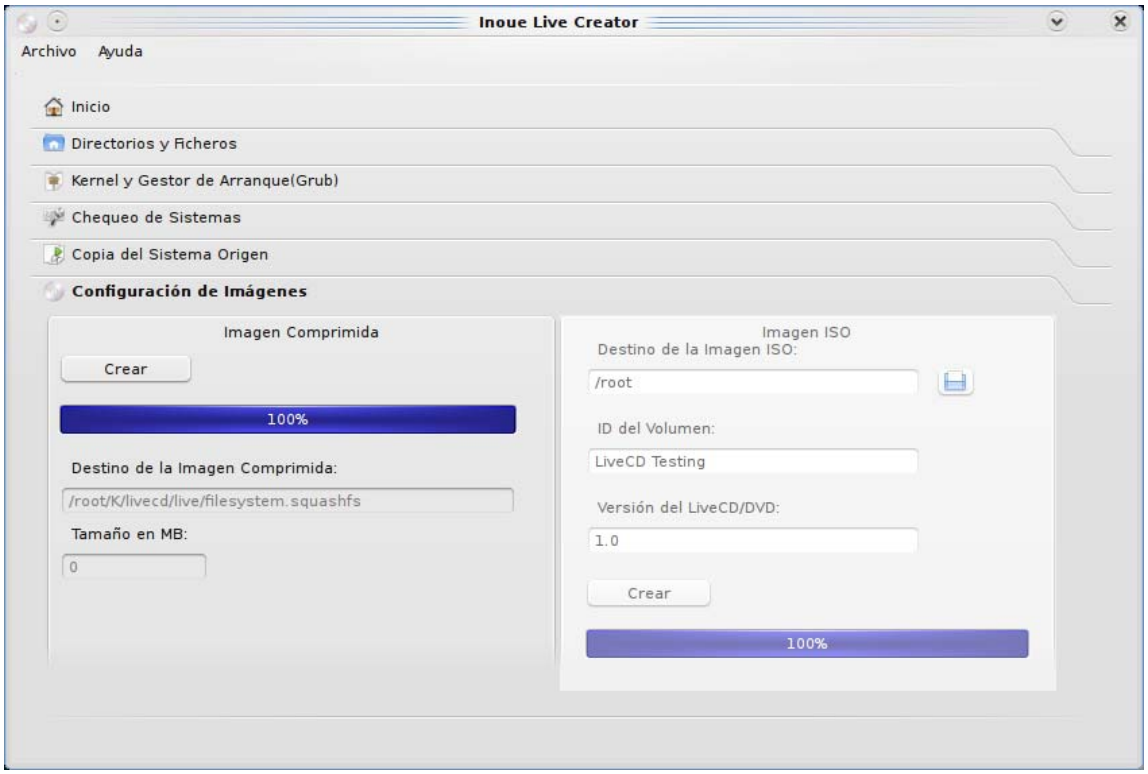
Historia de Usuario	
Número: UH-ILC-05	**Nombre Historia de Usuario:** Crear imagen Comprimida.
Modificación de Historia de Usuario Número:	
Usuario: Keiver Hernández Fernández Lucia C. Domínguez Delgado	**Iteración Asignada:** 3
Prioridad en Negocio: Alta (Alta/ Media/ Baja)	**Puntos Estimados:** 1
Riesgo en Desarrollo: Alta (Alta/ Media/ Baja)	**Puntos Reales:** 1
Descripción: Comprime el sistema creando una imagen comprimida con squashfs. Usando la aplicación mksquashfs.	



Tarea de Ingeniería	
Número Tarea: 1	**Número Historia de Usuario:** UH-ILC-05
Nombre Tarea: Crear expresión regular para obtener el por ciento de creada de la imagen	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 04/02/09	**Fecha Fin:** 04/02/09
Programador Responsable: Keiver Hernández Fernández Lucia C. Domínguez Delgado	
Descripción: Se determina la expresión regular para obtener el por ciento de la creación de la imagen. Necesaria para mostrarlo en la interfaz gráfica.	

Tarea de Ingeniería	
Número Tarea: 2	**Número Historia de Usuario:** UH-ILC-05
Nombre Tarea: Calcula el tamaño de la imagen comprimida.	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 05/02/09	**Fecha Fin:** 05/02/09
Programador Responsable: Keiver Hernández Fernández Lucia C. Domínguez Delgado	
Descripción: Se calcula el tamaño de la imagen comprimida. Necesaria para mostrarlo en la interfaz gráfica.	

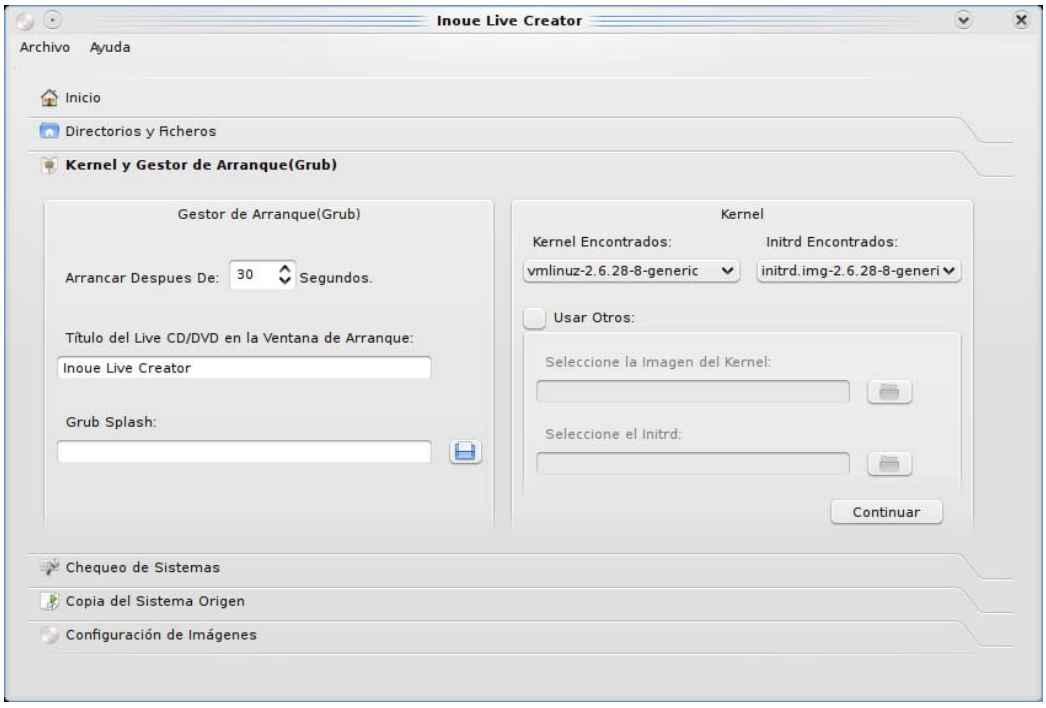
Historia de Usuario	
Número: UH-ILC-06	**Nombre Historia de Usuario:** Crear imagen ISO.
Modificación de Historia de Usuario Número:	
Usuario: Keiver Hernández Fernández Lucia C. Domínguez Delgado	**Iteración Asignada:** 3
Prioridad en Negocio: Alta (Alta/ Media/ Baja)	**Puntos Estimados:** 1
Riesgo en Desarrollo: Alta (Alta/ Media/ Baja)	**Puntos Reales:** 1
Descripción: Crea una imagen ISO lista para usarse a partir del sistema comprimido. Se calcula su tamaño y se aconseja al usuario si debe usar un CD o un DVD.	



Tarea de Ingeniería	
Número Tarea: 1	**Número Historia de Usuario:** UH-ILC-06
Nombre Tarea: Obtener valores de configuración de la imagen ISO.	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 06/02/09	**Fecha Fin:** 06/02/09
Programador Responsable: Keiver Hernández Fernández Lucia C. Domínguez Delgado	
Descripción: Obtiene los valores para la imagen ISO. Entre ellos, la versión del LiveCD/DVD, el ID del volumen y la dirección donde se guardará la imagen ISO.	

Tarea de Ingeniería	
Número Tarea: 2	**Número Historia de Usuario:** UH-ILC-06
Nombre Tarea: Crear expresión regular para obtener el por ciento de creada de la imagen ISO.	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 08/02/09	**Fecha Fin:** 08/02/09
Programador Responsable: Keiver Hernández Fernández Lucia C. Domínguez Delgado	
Descripción: Obtiene el por ciento de creada de la imagen ISO. Necesaria para mostrarlo en la interfaz gráfica.	

Historia de Usuario	
Número: UH-ILC-07	**Nombre Historia de Usuario:** Gestión de la Configuración.
Modificación de Historia de Usuario Número:	
Usuario: Keiver Hernández Fernández Lucia C. Domínguez Delgado	**Iteración Asignada:** 1
Prioridad en Negocio: Alta (Alta/ Media/ Baja)	**Puntos Estimados:** 1
Riesgo en Desarrollo: Alta (Alta/ Media/ Baja)	**Puntos Reales:** 1
Descripción: Se gestionan los parámetros para la configuración a utilizar en la creación del proyecto. Entre ellos la verificación del usuario con que se ejecuta la aplicación, la gestión del kernel y del script management.sh.	



Tarea de Ingeniería	
Número Tarea: 1	**Número Historia de Usuario:** UH-ILC-07
Nombre Tarea: Copia del script management.sh necesario para chroot.	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 09/02/09	**Fecha Fin:** 11/02/09
Programador Responsable: Keiver Hernández Fernández Lucia C. Domínguez Delgado	
Descripción: Copia el script necesario para el chroot, este script es el que hace las tareas de mantenimiento dentro del sistema recién copiado.	

Tarea de Ingeniería	
Número Tarea: 2	**Número Historia de Usuario:** UH-ILC-07
Nombre Tarea: Gestionar la selección del Kernel del sistema.	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 12/02/09	**Fecha Fin:** 13/02/09
Programador Responsable: Keiver Hernández Fernández Lucia C. Domínguez Delgado	
Descripción: Gestiona todo lo relacionado con el Kernel a utilizar por el LiveCD/DVD. Se muestra una lista de los Kernels disponibles en el sistema, y es posible seleccionar uno personalizado.	

Tarea de Ingeniería	
Número Tarea: 3	**Número Historia de Usuario:** UH-ILC-07
Nombre Tarea: Gestionar los permisos de la aplicación.	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 16/02/09	**Fecha Fin:** 20/02/09
Programador Responsable: Keiver Hernández Fernández Lucia C. Domínguez Delgado	
Descripción: Gestiona los permisos para ejecutar la aplicación como root. Al inicio de la aplicación se chequea el usuario que ejecuta la misma, si es root, se ejecuta normalmente, de lo contrario se muestra un mensaje de información y se cierra la aplicación.	

2.3 Diagrama de Componentes

Los diagramas de componentes describen los elementos físicos del sistema y sus relaciones. Muestran las opciones de realización incluyendo códigos fuente, binarios y ejecutables. Los componentes representan todos los tipos de elementos de software que entran en la fabricación de aplicaciones informáticas. Pueden ser simples archivos, paquetes, bibliotecas cargadas dinámicamente, entre otros.

A continuación se presente el diagrama de componentes para el sistema propuesto.

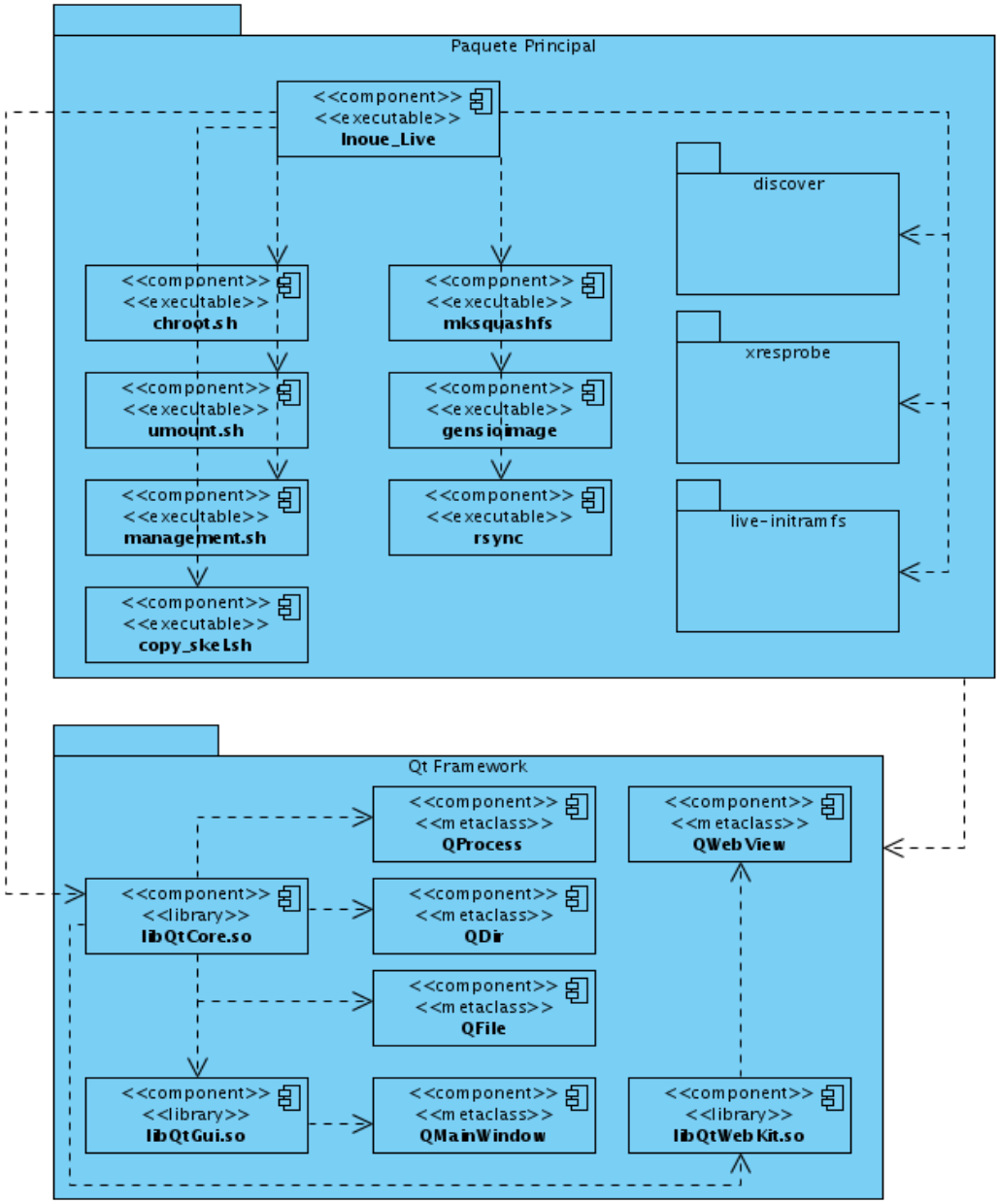


Imagen 1. Diagrama de componentes.

En este diagrama de componentes perteneciente a la aplicación Inoue Live Creator, se muestran los principales elementos físicos así como sus relaciones, entre ellos los scripts de optimización del sistema y las clases y librerías de Qt.

El componente principal es el ejecutable **Inoue_Live** que usa un conjunto de scripts necesarios para que el sistema funcione correctamente. Ellos son **chroot.sh**, **management.sh** y **copy_skel.sh**.

Se ve representado también, la interacción de la aplicación con ejecutables como **mksquashfs**, **genisoimage** y **rsync** que son aplicaciones que se usan para comprimir el sistema, crear la imagen ISO y realizar la copia del sistema respectivamente. De la misma forma se ve el uso de paquetes como **discover**, **xresprobe** y **live-initramfs** usados en el sistema cuando se está ejecutando el LiveCD/DVD garantizando un correcto funcionamiento del mismo.

En la parte inferior del diagrama se representan las principales clases y bibliotecas usadas del conjunto de librerías Qt que son la base de la aplicación. Esto último se ve representado en la relación existente entre el ejecutable principal y la librería **libQtCore.so**.

2.4 Plan de Releases

A continuación se define el plan de releases e iteraciones para realizar las entregas intermedias y la entrega final. Tiene como entrada la relación de Historias de Usuario previamente definidas. Para colocar una historia en cada iteración se tiene en cuenta la prioridad que definió el cliente para dicha Historia. Como resultado de este proceso se llegó a la siguiente planificación:

Release	Historias de Usuario	Tiempo Estimado(Semanas)
1	1,2,7	6
2	3,4	5
3	5,6	5

En el trabajo de este capítulo se llevó a cabo la planificación del proyecto expresando esta en forma de historias de usuario, tareas de ingeniería y diagramas auxiliares que facilitan la comprensión del sistema. La planificación de los releases se hizo teniendo en cuenta las principales funcionalidades que la aplicación debe tener. De la misma forma se construyó el diagrama de componentes, el cual ayuda en el entendimiento de la estructura del sistema.

CAPÍTULO III. IMPLEMENTACIÓN Y PRUEBAS

A continuación se tratarán temas relacionados con el proceso de implementación y los casos de pruebas o pruebas de aceptación a las que fue sometida la aplicación Inoue Live Creator en cada una de las iteraciones, el cumplimiento de estos casos de pruebas fue el hito para avanzar hacia la próxima iteración. Se expone además una relación de las funcionalidades con las que cuenta el sistema hasta la fecha.

3.1 Propuesta del sistema a implementar

Una vez analizadas las principales herramientas usadas a nivel mundial para la generación automática de LiveCD/DVDs se llega a la conclusión de que ninguna es una solución factible para generar un LiveCD/DVD usando una interfaz gráfica de forma fácil y configurable. Por lo que se decide comenzar la implementación de la aplicación Inoue Live Creator la cual hereda gran parte de las características y funcionalidades de estos.

En el proceso de implementación de la aplicación se debe utilizar una capa de presentación y una capa lógica de negocio. En la primera estará la interfaz que el usuario observará y que se implementará en Qt, mientras que la capa lógica de negocio contendrá todo el núcleo de la aplicación, principales clases y funcionalidades.

La interfaz gráfica de la aplicación estará compuesta por una barra de menú que dará acceso a funciones básicas de aplicación como la opción de salir y el acceso a la ayuda. El funcionamiento general de la misma está basado en las distintas opciones de configuración que el usuario deberá ir completando para ir realizando los pasos necesarios para generar el LiveCD/DVD.

Primeramente la aplicación solicitará el nombre del proyecto el cual será el nombre del LiveCD/DVD que se quiera generar. Así como la ruta donde se guardarán los ficheros necesarios para que la aplicación genere el LiveCD/DVD de forma correcta.

Seguidamente se pasará a la sección donde se configurarán los directorios principales del LiveCD/DVD a generar. Se seleccionará el directorio raíz que no es más que la dirección de donde está el sistema a usar en el LiveCD/DVD. Es aquí también donde se deben seleccionar los directorios y ficheros a excluir en el proceso. Así como la dirección de la carpeta personal con la configuración del usuario que se usará en el LiveCD/DVD.

Una vez configurados los directorios se procederá a la selección del kernel y la imagen initrd a usar en el

LiveCD/DVD, se listarán los disponible en el sistema y además se dará la posibilidad de seleccionar uno personalido.

Finalmente se copiará el sistema previamente seleccionado y se comprimirá usando un sistema squashfs y se generará la imagen ISO lista para grabar en un medio extraíble.

El sistema debe permitir chequear las aplicaciones necesarias para el correcto funcionamiento del LiveCD/DVD, informándole al usuarios si están o no instaladas.

3.2 Estándar de código

Para el desarrollo de la aplicación se utilizó un estándar de código que permite el entendimiento de todo el código escrito. Este estándar describe como están compuestos los nombres de las clases, los métodos, las variables, los comentarios y la estructura del código en general.

Parámetros de métodos

Recomendación:

Los nombres comienzan con el prefijo "p" y continúan siguiendo el esquema de las variables miembro de clases (inicia con minúscula y continua junto comenzando palabras con mayúscula).

Justificación:

De esta forma es posible diferenciar claramente los argumentos de un método/función de las variables locales y miembros de clases (y otros tipos de variables). Cuando se hace referencia al valor de inicialización de una variable miembro de la clase, en caso de que se esté dentro de un constructor, el nombre de la variable debería ser el mismo que el de la variable miembro.

Ejemplo:

```
class CCLase
{
  double mivariable;
  CCLase(double pmiVariable)
        {
              miVariable = pmiVariable;
        }
}
```

Variables locales

Recomendación:

Los nombres comienzan con minúscula y continúan siguiendo el esquema de las variables miembro de

clases (inicia con minúscula y continua junto comenzando palabras con mayúscula).

Justificación:

Es importante saber a partir del nombre el alcance de la variable.

Ejemplo:

```
void unaFuncionQueNoHaceNada()
{
        int  unaVariableParaMetodoX = 5;
}
```

Clases

Recomendación:

Los nombres de clases comienzan con mayúsculas y con la letra "C", con las palabras que la forman en minúsculas y separadas por mayúsculas, de la misma forma que las variables locales.

Justificación:

Al igual que las variables locales, los nombres de clase abundan en el código por lo que es práctico utilizar nombres intuitivos para ellas. La marca distintiva en este caso es la mayúscula inicial. Este formato es clásico y compatible con estándares de notación para lenguajes como C++.

Ejemplo:

```
class CLive: public QWebKit
{
        void Creat(...);
}
```

Métodos miembros de clases

Recomendación:

Los métodos miembros de clases siguen la notación idéntica que para las variables locales, es decir, comenzando con minúscula y separando las palabras con mayúsculas.

Justificación:

Al igual que las variables miembro de clase, estos nombres son muy utilizados y es clave una lectura rápida y clara de su significado. Sin embargo, al contrario de las variables miembros que pueden coexistir con otras clases de variable, las clases de función existentes no son muchas.

Ejemplo:

```
class Ejemplo
```

```
{
  public:
  metodoPublico();
}
```

Nombres de archivos

Recomendación:

Cada archivo debe en lo posible albergar la interfase/implementación de solo una clase.

El nombre del archivo debe ser el mismo que el de la clase (nombre en minuscula), con la extensión .h para encabezados y .cpp para implementación.

Justificación:

Es importante, a la hora de hacer cambios o depurar código, ubicar las clases rápidamente en los archivos en que estén definidas. La forma más rápida e intuitiva es que cada clase esté definida en su propio archivo, y que dicho archivo lleve el nombre de la clase de forma de saber inmediatamente de que clase se trata al moverse en los directorios en donde se encuentran los fuentes del programa.

Ejemplo:

inoue_live.h

inoue_live.cpp

Documentación de funciones y métodos

Previo a la declaración de cada función o método de clase debe incluirse un bloque de comentario. Dicho bloque debe describir brevemente la funcionalidad del método, si éste es concreto, o el contrato general, si éste es virtual puro. Es imprescindible incluir comentarios sobre cada parámetro y valor de retorno de la función / método en cuestión.

Ejemplo:

```
//Crea la estructura de directorios del proyecto.
void CProjectManagement::createProjectSkel()
{
(…)
}
```

3.3 Arquitectura de la aplicación

La aplicación propuesta será desarrollada usando Qt, que es una biblioteca multiplataforma para desarrollar interfaces gráficas de usuario.

Dentro de las principales clases usadas en el desarrollo de la aplicación, se encuentran QProcess y QObject. QProcess es una clase usada para gestionar la ejecución de procesos dentro de las aplicaciones desarrolladas con Qt. Entre otras funcionalidades esta clase permite, ejecutar un programa y a la vez adicionar parámetros al mismo.

Una vez ejecutado un proceso con dicha clase, es posible leer desde la salida estándar o la salida de error; este parámetro es configurable una vez inicializado el objeto de tipo QProcess. Esta clase permite además poner variables de entorno independientemente de la plataforma en que se esté ejecutando la aplicación.

Otra de las funcionalidades fundamentales de QProcess, es que puede ejecutar proceso como hijos de otros procesos.

QObject es la clase base de todos los objetos de Qt y es el centro del modelo de objetos de Qt. La característica fundamental de este poderoso mecanismo es el sistema de comunicación entre objetos llamado signals y slots. No son más que las señales que emiten los mismos y las funciones que se ejecutan al ser llamadas estas.

Como parte de la arquitectura básica de la aplicación, se implementan las clases CLive, CProjectManagement e Inoue_Live encargadas de gestionar los objetos principales de la misma. CLive es la clase base en la que se encuentra representado un LiveCD/DVD con todos sus atributos y funciones básicas. CProjectManagement es la clase encargada de contener las principales funcionalidades que debe tener el sistema para su correcto funcionamiento. En ella se encuentran funciones importantes como las responsables de copiar y comprimir el sistema así como de generar el ISO final. La clase Inoue_Live es una clase que hereda de QMainWindow y es la responsable de gestionar los elementos gráficos.

3.4 Listado de Historias de Usuarios a probar

Para comenzar con las pruebas que se le harán al sistema, se listan a continuación las Historias de Usuarios a las cuales se le harán dichas pruebas. Luego se definirá el cronograma para la realización de las mismas.

Historias de Usuarios a probar:

1. *UH-ILC-01, Programar Scripts Auxiliares.*

En este caso de prueba se probará el correcto funcionamiento de los scripts para la optimización del sistema una vez copiado.

2. *UH-ILC-02, Obtener parámetros de configuración.*

La obtención de los parámetros de configuración es una de las tareas más importantes en el correcto funcionamientos de la aplicación, es este caso de prueba se chequea que la entrada de los datos se haga de forma correcta.

3. *UH-ILC-03, Chequear Sistema.*

Para el correcto funcionamiento de la aplicación propuesta es necesario que se encuentren instaladas algunas herramientas que son claves en este sentido. En este caso de prueba se chequea que la verificación de estas herramientas se haga correctamente.

4. *UH-ILC-04, Copia del Sistema Original.*

Para evitar que se dañe el sistema original en el proceso de creación del LiveCD/DVD, Inoue Live Creator hace una copia del mismo para luego hacer algunas tareas de optimización, en este caso de pruebas se chequea que se haga de correctamente la copia del sistema.

5. *UH-ILC-05, Crear Imagen Comprimida.*

Una vez copiado el sistema se comprime este usando un sistema de archivos squashfs. En este caso de pruebas se chequea que este proceso se haga correctamente.

6. *UH-ILC-06, Crear Imagen ISO.*

Al comprimirse el sistema, el paso final en la generación del LiveCD/DVD es la creación del la imagen ISO, no es más que una imagen de CD o DVD autoarrancable. En esta sección se chequeará que este paso se haga satisfactoriamente.

7. *UH-ILC-07, Gestión de la Configuración.*

Debido a que la aplicación propuesta necesita utilizar permisos de administración entre otras características, en este caso de pruebas se chequea la gestión de la configuración entre otros aspectos.

3.5 Cronograma de realización de pruebas.

No	Tarea	Fecha	Responsable	Participantes	Observaciones
1	*Programar Scripts Auxiliares.*	07/01/09	Keiver Hernández Fernández	Lucia C. Domínguez Delgado, Ramon Paumier	Prueba Satisfactoria.
2	*Obtener parámetros de configuración.*	17/01/09	Lucia C. Domínguez Delgado	Keiver Hernández Fernández, Ramon Paumier	Prueba Satisfactoria.
3	*Chequear Sistema.*	30/01/09	Keiver Hernández Fernández	Lucia C. Domínguez Delgado, Ramon Paumier	Prueba Satisfactoria.
4	*Copia del Sistema Original.*	03/02/09	Keiver Hernández Fernández	Lucia C. Domínguez Delgado, Ramon Paumier	Prueba Satisfactoria.
5	*Crear Imagen Comprimida.*	05/02/09	Lucia C. Domínguez Delgado	Keiver Hernández Fernández, Ramon Paumier	Prueba Satisfactoria.
6	*Crear Imagen ISO.*	08/02/09	Lucia C. Domínguez Delgado	Keiver Hernández Fernández, Ramon Paumier	Prueba Satisfactoria.
7	*Gestión de la Configuración.*	20/02/09	Keiver Hernández Fernández	Lucia C. Domínguez Delgado, Ramon Paumier	Prueba Satisfactoria.

3.6 Casos de Prueba

En SXP los casos de prueba se escriben antes que el código y los desarrolladores escriben pruebas unitarias. Los clientes especifican pruebas funcionales, los cambios se integran en el código base varias veces por día. Todos lo casos de prueba se deben pasar antes y después de la integración, se dispone de una maquina para la integración y se realizan pruebas funcionales en donde participa el cliente.

Durante el desarrollo de Inoue Live Creator, se diseñaron un conjunto de casos de pruebas a las que fue sometido el sistema para comprobar su funcionamiento según las Historias de Usuarios.

Se definieron casos de pruebas para todas las Historias de Usuarios, a continuación se presentan estos casos de pruebas con sus respectivas descripciones.

3.6.1 Caso de Prueba Historia de Usuario: UH-ILC-01

Esta sección cubre el conjunto de pruebas funcionales relacionadas con la historia de usuario: *Programar Scripts Auxiliares*.

En esta historia de usuario se intenta probar la valides y funcionalidad de los scripts auxiliares para el correcto funcionamiento de la aplicación Inoue Live Creator.

Primeramente se comprobará la valides del script **chroot.sh** el cual se encarga de montar las particiones necesarias para el correcto funcionamiento del comando chroot en el sistema del cual se creará el LiveCD/DVD.

Luego se analizará el script **management.sh** que se encarga de realizar tareas de optimización en el sistema recién copiado el cual se comprimirá posteriormente. Finalmente se comprobará que se desmonten las particiones previamente montadas con el comando chroot.sh, usando el script **umount.sh**.

Caso de Prueba de Aceptación	
Código Caso de Prueba: 1	**Número Historia de Usuario:** UH-ILC-01
Descripción de la Prueba: Al concluir la copia del sistema original, se pasa a optimizar el sistema. Primeramente se montan las particiones en los directorios previamente definidos. Luego se ejecuta el script management.sh encargado de optimizar el sistema recién copiado y se hacen las siguientes tareas: 1-Montar el /dev y /proc 2-Eliminar ficheros innecesarios. 3-Crear Logs y ficheros necesarios. 4-Eliminar usuarios que no son del sistema.	
Condiciones de Ejecución: Que la aplicación se este ejecutando como root. Y además que se copie un sistema de archivos válidos. De lo contrario la llamada a chroot falla.	
Entrada / Pasos de ejecución: Se llama al script chroot el cual monta las particiones y ejecuta el script management.sh en el nuevo sistema.	
Resultado Esperado: Que se ejecute el script management.sh de forma correcta y haga todas las tareas definidas dentro del mismo.	
Evaluación de la Prueba: Prueba Satisfactoria.	

Caso de Prueba de Aceptación	
Código Caso de Prueba: 2	**Número Historia de Usuario:** UH-ILC-01
Descripción de la Prueba: Una ves copiado el sistema original y optimizado con los scripts de mantenimiento. Se pasa a copiar la información personal del usuario previamente seleccionado con el objetivo de que las configuraciones de este estén en el LiveCD/DVD a crear. Para hacer esto se debe seleccionar a cual usuario se le copiarán los datos pues no siempre se quiere los datos de root, que es el usuario que puede usar la aplicación. En su lugar se puede seleccionar otro usuario.	
Condiciones de Ejecución: Que se halla seleccionado previamente un usuario.	
Entrada / Pasos de ejecución: Se toma el nombre de usuario o la dirección de su carpeta personal y se copian los datos a la carpeta /etc/skel del sistema a usar en el LiveCD/DVD.	
Resultado Esperado: La copia satisfactoria de los datos del usuario seleccionado.	
Evaluación de la Prueba: Prueba Satisfactoria.	

3.6.2 Caso de Prueba Historia de Usuario: UH-ILC-02

Esta sección cubre el conjunto de pruebas funcionales relacionadas con la historia de usuario: *Obtener parámetros de configuración.*

En esta historia de usuario lo que se verifica es la obtención de los parámetros de configuración permitiéndole a la aplicación Inoue Live Creator generar el LiveCD/DVD correctamente.

Primeramente se comprobará la valides del Directorio Raíz del cual se copiaran los datos para el LiveCD/DVD.

Luego se cargarán los ficheros **exclude_dirs.exc** y **exclude_files.exc** que tienen dentro los directorios y ficheros predeterminados que no hacen falta para la creación del LiveCD/DVD. Permitiendo adicionar directorios y ficheros o borrar algunos de los mismos.

Se seleccionará el nombre del usuario del cual se copiarán los ficheros de configuración personal. Se probará también la obtención del nombre y la dirección donde se creará un nuevo proyecto. Finalmente se chequearán las direcciones del Kernel y el initrd para el LiveCD/DVD.

Caso de Prueba de Aceptación	
Código Caso de Prueba: 1	**Número Historia de Usuario:** UH-ILC-02
Descripción de la Prueba: Primeramente se comprobará la valides del Directorio Raíz del cual se copiaran los datos para el LiveCD/DVD.	
Condiciones de Ejecución: Que la aplicación se este ejecutando como root. Y además que se especifique una dirección de sistema de archivos válida.	
Entrada / Pasos de ejecución: Se selecciona el sistema del cual se realizará el LiveCD/DVD. Se comprueba que se haya especificado al menos una dirección.	
Resultado Esperado: Si se seleccionó una dirección, la ejecución del programa continua normalmente, de lo contrario se muestra un mensaje de error indicando que se debe seleccionar el sistema a copiar.	
Evaluación de la Prueba: Prueba Satisfactoria.	

Caso de Prueba de Aceptación	
Código Caso de Prueba: 2	**Número Historia de Usuario:** UH-ILC-02
Descripción de la Prueba: Se cargarán los ficheros **exclude_dirs.exc** y **exclude_files.exc** que tienen dentro los directorios y ficheros predeterminados que no hacen falta para la creación del LiveCD/DVD. Permitiendo adicionar directorios y ficheros o borrar algunos de los mismos.	
Condiciones de Ejecución: Que existan los ficheros exclude_dirs.exc y exclude_files.exc.	
Entrada / Pasos de ejecución: Se cargan los ficheros exclude_dirs.exc y exclude_files.exc los cuales contienen los directorios y ficheros a excluir. Luego se muestran las respectivas listas en la interfaz gráfica.	
Resultado Esperado: Se carguen y se muestren satisfactoriamente las listas de directorios y ficheros a excluir.	
Evaluación de la Prueba: Prueba Satisfactoria.	

Caso de Prueba de Aceptación	
Código Caso de Prueba: 3	**Número Historia de Usuario:** UH-ILC-02
Descripción de la Prueba: Se seleccionará el nombre del usuario del cual se copiarán los ficheros de configuración personal.	
Condiciones de Ejecución: En caso que se vaya a especificar un usuario que cuya carpeta personal no se encuentre en /home, se debe especificar la ruta a la carpeta del nuevo usuario.	
Entrada / Pasos de ejecución: Se selecciona el usuario del cual se copiarán los archivos de configuración personal, seleccionando uno de los usuarios de la lista los cuales corresponden a las carpetas personales que existen /home. O se selecciona una carpeta personal que se encuentre en una ruta específica.	
Resultado Esperado: Se seleccione de cualquiera de las dos vías la carpeta del usuario.	
Evaluación de la Prueba: Prueba Satisfactoria.	

Caso de Prueba de Aceptación	
Código Caso de Prueba: 4	**Número Historia de Usuario:** UH-ILC-02
Descripción de la Prueba: Se seleccionará la dirección y el nombre del nuevo proyecto para generar un LiveCD/DVD.	
Condiciones de Ejecución:	
Entrada / Pasos de ejecución: Se entra el nombre del proyecto y se selecciona la dirección de este, luego si no se especificó alguno de estos datos, se muestra un mensaje de error, alertando al usuario de que debe llenar todos los campos, incluido que la dirección del proyecto sea una ruta válida.	
Resultado Esperado: Que los campos del nombre y la dirección del nuevo proyecto no estén vacías y además que sea una dirección válida.	
Evaluación de la Prueba: Prueba Satisfactoria.	

Caso de Prueba de Aceptación	
Código Caso de Prueba: 5	**Número Historia de Usuario:** UH-ILC-02
Descripción de la Prueba: Se chequearán las direcciones del Kernel y el initrd para el LiveCD/DVD.	
Condiciones de Ejecución:	
Entrada / Pasos de ejecución: Se selecciona uno de los Kernels e initrd disponibles en listas llenadas con los datos de la carpeta /boot. O se selecciona un Kernel e initrd de una ruta definida por el usuario. Estas últimas deben ser válidas.	
Resultado Esperado: Que se seleccione un Kernel y un initrd desde las listas que muestra la aplicación o se seleccione un Kernel e initrd personalizados y que además sean direcciones válidas.	
Evaluación de la Prueba: Prueba Satisfactoria.	

3.6.3 Caso de Prueba Historia de Usuario: UH-ILC-03

Esta sección cubre el conjunto de pruebas funcionales relacionadas con la historia de usuario: *Chequear Sistema.*

En esta historia de usuario lo que se verifica es el chequeo del sistema, permitiéndole a la aplicación Inoue Live Creator generar el LiveCD/DVD correctamente.

Primeramente se comprobará que existan los módulos del Kernel necesarios para la generación del LiveCD/DVD.

Luego se verificará que existan un conjunto de herramientas que debe tener instaladas el sistema original como el sistema del LiveCD/DVD.

Caso de Prueba de Aceptación	
Código Caso de Prueba: 1	**Número Historia de Usuario:** UH-ILC-03
Descripción de la Prueba: Se comprobarán los módulos del Kernel a usar que son necesarios para la correcta generación del LiveCD/DVD.	
Condiciones de Ejecución: Que el Kernel que se está ejecutando en el sistema sea el mismo que se usará en la creación del LiveCD/DVD.	
Entrada / Pasos de ejecución: Se comprueba la existencia de los módulos siguientes necesarios para el correcto funcionamiento del LiveCD/DVD: • **squasfs**(Modulo de SquashFS, usado para la compresión del sistema.) • **aufs**(Union File System(Unión de Sistemas de Ficheros), usado para poder modificar el LiveCD/DVD cuando se esté ejecutando.)	
Resultado Esperado: Si los módulos están instalados se muestra en la interfaz gráfica como instalada con una pequeña bandera verde. De lo contrario con una bandera roja.	
Evaluación de la Prueba: Prueba Satisfactoria.	

Caso de Prueba de Aceptación	
Código Caso de Prueba: 2	**Número Historia de Usuario:** UH-ILC-03
Descripción de la Prueba: Se comprobará que este instalada la herramienta genisoimage.	
Condiciones de Ejecución: Que se esté ejecutando la aplicación en un sistema Debian GNU/Linux o derivado.	
Entrada / Pasos de ejecución: Se comprueba que la aplicación esté instalada en el sistema, lo cual se hace chequeando la dirección del ejecutable teniendo en cuenta el empaquetamiento oficial de la distribución Debian GNU/Linux.	
Resultado Esperado: Si la aplicación está instalada se muestra en la interfaz gráfica como instalada con una pequeña bandera verde. De lo contrario con una bandera roja.	
Evaluación de la Prueba: Prueba Satisfactoria.	

Caso de Prueba de Aceptación	
Código Caso de Prueba: 3	**Número Historia de Usuario:** UH-ILC-03
Descripción de la Prueba: Se comprobará que este instalada la herramienta live-initramfs.	
Condiciones de Ejecución: Que se esté ejecutando la aplicación en un sistema Debian GNU/Linux o derivado.	
Entrada / Pasos de ejecución: Se comprueba que la aplicación esté instalada en el sistema, lo cual se hace chequeando la dirección del ejecutable teniendo en cuenta el empaquetamiento oficial de la distribución Debian GNU/Linux.	
Resultado Esperado: Si la aplicación está instalada se muestra en la interfaz gráfica como instalada con una pequeña bandera verde. De lo contrario con una bandera roja.	
Evaluación de la Prueba: Prueba Satisfactoria.	

Caso de Prueba de Aceptación	
Código Caso de Prueba: 4	**Número Historia de Usuario:** UH-ILC-03
Descripción de la Prueba: Se comprobará que este instalada la herramienta discover.	
Condiciones de Ejecución: Que se esté ejecutando la aplicación en un sistema Debian GNU/Linux o derivado.	
Entrada / Pasos de ejecución: Se comprueba que la aplicación esté instalada en el sistema, lo cual se hace chequeando la dirección del ejecutable teniendo en cuenta el empaquetamiento oficial de la distribución Debian GNU/Linux.	
Resultado Esperado: Si la aplicación está instalada se muestra en la interfaz gráfica como instalada con una pequeña bandera verde. De lo contrario con una bandera roja.	
Evaluación de la Prueba: Prueba Satisfactoria.	

Caso de Prueba de Aceptación	
Código Caso de Prueba: 5	**Número Historia de Usuario:** UH-ILC-03
Descripción de la Prueba: Se comprobará que este instalada la herramienta xresprobe.	
Condiciones de Ejecución: Que se esté ejecutando la aplicación en un sistema Debian GNU/Linux o derivado.	
Entrada / Pasos de ejecución: Se comprueba que la aplicación esté instalada en el sistema, lo cual se hace chequeando la dirección del ejecutable teniendo en cuenta el empaquetamiento oficial de la distribución Debian GNU/Linux.	
Resultado Esperado: Si la aplicación está instalada se muestra en la interfaz gráfica como instalada con una pequeña bandera verde. De lo contrario con una bandera roja.	
Evaluación de la Prueba: Prueba Satisfactoria.	

Caso de Prueba de Aceptación	
Código Caso de Prueba: 6	**Número Historia de Usuario:** UH-ILC-03
Descripción de la Prueba: Se comprobará que este instalada la herramienta rsync.	
Condiciones de Ejecución: Que se esté ejecutando la aplicación en un sistema Debian GNU/Linux o derivado.	
Entrada / Pasos de ejecución: Se comprueba que la aplicación esté instalada en el sistema, lo cual se hace chequeando la dirección del ejecutable teniendo en cuenta el empaquetamiento oficial de la distribución Debian GNU/Linux.	
Resultado Esperado: Si la aplicación está instalada se muestra en la interfaz gráfica como instalada con una pequeña bandera verde. De lo contrario con una bandera roja.	
Evaluación de la Prueba: Prueba Satisfactoria.	

3.6.4 Caso de Prueba Historia de Usuario: UH-ILC-04

Esta sección cubre el conjunto de pruebas funcionales relacionadas con la historia de usuario: *Copia del Sistema Original.*

En esta historia de usuario lo que se verifica es la copia del sistema original, permitiéndole a la aplicación Inoue Live Creator generar el LiveCD/DVD correctamente.

Se hará una copia del sistema del cual se creará el LiveCD/DVD, para esto debe estar instalada la herramienta rsync.

De la misma forma se probarán las expresiones regulares para obtener los valores de por ciento y el nombre del fichero que se está copiando respectivamente.

Caso de Prueba de Aceptación	
Código Caso de Prueba: 1	**Número Historia de Usuario:** UH-ILC-04
Descripción de la Prueba: Se copia el sistema original teniendo en cuenta los directorios y ficheros a excluir de la copia, especificados previamente. La copia se realiza usando la herramienta rsync.	
Condiciones de Ejecución: Que ya se haya especificado el sistema a copiar.	
Entrada / Pasos de ejecución: Se ejecuta la aplicación rsync pasándole como parámetro los ficheros y directorios que se definieron como no necesarios para el LiveCD/DVD así como las direcciones del sistema a copiar entre otros valores.	
Resultado Esperado: La copia del sistema raíz realizada correctamente.	
Evaluación de la Prueba: Prueba Satisfactoria.	

Caso de Prueba de Aceptación	
Código Caso de Prueba: 2	**Número Historia de Usuario:** UH-ILC-04
Descripción de la Prueba: Al ejecutarse la herramienta rsync, se comienza con la copia del sistema, produciendo una salida que se usará para saber el por ciento de ejecución de este proceso. En esta prueba se chequeará la expresión regular usada para obtener este valor.	
Condiciones de Ejecución: Que este ejecutándose el proceso rsync.	
Entrada / Pasos de ejecución: Al copiarse el sistema, la aplicación rsync produce una salida que se usará para saber el por ciento de ejecución de este proceso. Primeramente se lee la salida estándar del proceso y luego se usa esta expresión regular usada para encontrar patrones dentro de la cadena leída. Este número es el que se pasa a la barra de progreso de copia.	
Resultado Esperado: Que funcione correctamente la expresión regular usada para mostrar el por ciento de copia del sistema.	
Evaluación de la Prueba: Prueba Satisfactoria.	

Caso de Prueba de Aceptación	
Código Caso de Prueba: 3	**Número Historia de Usuario:** UH-ILC-04
Descripción de la Prueba: Al ejecutarse la herramienta rsync, se comienza con la copia del sistema, produciendo una salida que se usará para saber el por ciento de ejecución de este proceso. En esta prueba se chequeará la expresión regular usada para obtener el nombre del fichero que se está copiando.	
Condiciones de Ejecución: Que este ejecutándose el proceso rsync.	
Entrada / Pasos de ejecución: Al copiarse el sistema, la aplicación rsync produce una salida que se usará para saber el por ciento de ejecución de este proceso. Primeramente se lee la salida estándar del proceso y luego se usa esta expresión regular usada para encontrar patrones dentro de la cadena leída. Al obtenerse el nombre del fichero se muestra como el que se está copiando.	
Resultado Esperado: Que funcione correctamente la expresión regular usada para obtener el nombre del fichero que se esté copiando en ese momento.	
Evaluación de la Prueba: Prueba Satisfactoria.	

3.6.5 Caso de Prueba Historia de Usuario: UH-ILC-05

Esta sección cubre el conjunto de pruebas funcionales relacionadas con la historia de usuario: *Crear Imagen Comprimida.*

En esta historia de usuario lo que se verifica es la creación de la imagen comprimida, permitiéndole a la aplicación Inoue Live Creator generar el LiveCD/DVD correctamente.

Primeramente se comprimirá el sistema original previamente copiado. Para hacer esto se usará la herramienta **mksquashfs**, la cual debe estar instalada. Luego se probará la expresión regular usada para obtener el por ciento de progreso de creación de la imagen.

Finalmente se calculará el tamaño de la imagen comprimida para mostrarla al usuario y pueda ver el tamaño que va llevando el LiveCD/DVD.

Caso de Prueba de Aceptación	
Código Caso de Prueba: 1	**Número Historia de Usuario:** UH-ILC-05
Descripción de la Prueba: Se comprimirá el sistema original previamente copiado.	
Condiciones de Ejecución: Que ya se haya copiado el sistema raíz.	
Entrada / Pasos de ejecución: Se comprime el sistema original previamente copiado. Para hacer esto se usará la herramienta **mksquashfs**, la cual debe estar instalada. En este paso se ejecuta la herramienta mksquashfs la cual recibe la dirección del directorio a comprimir y la dirección donde se guardará la imagen comprimida.	
Resultado Esperado: La compresión del sistema se realiza de forma correcta.	
Evaluación de la Prueba: Prueba Satisfactoria.	

Caso de Prueba de Aceptación	
Código Caso de Prueba: 2	**Número Historia de Usuario:** UH-ILC-05
Descripción de la Prueba: Al ejecutarse la herramienta mksquashfs, se comienza con la compresión del sistema, produciendo una salida que se usará para saber el por ciento de ejecución de este proceso. En esta prueba se chequeará la expresión regular usada para obtener por ciento de la compresión del sistema.	
Condiciones de Ejecución: Que este ejecutándose el proceso mksquashfs.	
Entrada / Pasos de ejecución: Al comprimirse el sistema, la aplicación mksquashfs produce una salida que se usará para saber el por ciento de ejecución de este proceso. Primeramente se lee la salida estándar del proceso y luego se usa esta expresión regular usada para encontrar patrones dentro de la cadena leída. Al obtenerse el por ciento de la comprensión se le pasa a la barra de progreso en la interfaz.	
Resultado Esperado: Que funcione correctamente la expresión regular usada para obtener el por ciento de la comprensión.	
Evaluación de la Prueba: Prueba Satisfactoria.	

Caso de Prueba de Aceptación	
Código Caso de Prueba: 3	**Número Historia de Usuario:** UH-ILC-05
Descripción de la Prueba: Se calcula el tamaño de la imagen comprimida para mostrarlo en la interfaz gráfica.	
Condiciones de Ejecución: Que ya se haya comprimido el sistema.	
Entrada / Pasos de ejecución: Al terminar de comprimirse el sistema original, se calcula el tamaño del archivo generado y se muestra en la interfaz.	
Resultado Esperado: Se calcule el tamaño y se muestre el mismo de forma correcta.	
Evaluación de la Prueba: Prueba Satisfactoria.	

3.6.6 Caso de Prueba Historia de Usuario: UH-ILC-06

Esta sección cubre el conjunto de pruebas funcionales relacionadas con la historia de usuario: *Crear Imagen ISO.*

En esta historia de usuario lo que se verifica es la creación de la imagen ISO, permitiéndole a la aplicación Inoue Live Creator generar el LiveCD/DVD correctamente.

Después de comprimido el sistema original se pasará a crear la imagen con sistema de fichero ISO 9660. Para hacer esto se usará la herramienta **genisoimage**, la cual debe estar instalada. Luego se probará la expresión regular usada para obtener el por ciento de progreso de creación de la imagen.

Finalmente se probará la entrada de los datos para la creación de la imagen ISO, entre los que se encuentran el ID del volumen y la versión del LiveCD/DVD.

Caso de Prueba de Aceptación	
Código Caso de Prueba: 1	**Número Historia de Usuario:** UH-ILC-06
Descripción de la Prueba: Se creará la imagen ISO previamente comprimida.	
Condiciones de Ejecución: Que ya se haya comprimido el sistema original.	
Entrada / Pasos de ejecución: Se creará la imagen con sistema de fichero ISO 9660. Para hacer esto se usará la herramienta **genisoimage**, la cual debe estar instalada. En este paso se ejecuta la herramienta genisoimage la cual recibe la dirección del directorio donde está preparada la estructura del LiveCD/DVD y la dirección donde se guardará ISO a crear.	
Resultado Esperado: La generación del archivo .iso de forma correcta.	
Evaluación de la Prueba: Prueba Satisfactoria.	

Caso de Prueba de Aceptación	
Código Caso de Prueba: 2	**Número Historia de Usuario:** UH-ILC-06
Descripción de la Prueba: Al ejecutarse la herramienta genisoimage, se comienza con la generación de la imagen ISO, produciendo una salida que se usará para saber el por ciento de ejecución de este proceso. En esta prueba se chequeará la expresión regular usada para obtener por ciento de la creación del archivo ISO.	
Condiciones de Ejecución: Que este ejecutándose el proceso genisoimage.	
Entrada / Pasos de ejecución: Al ejecutarse la aplicación genisoimage, esta produce una salida que se usará para saber el por ciento de creación de este proceso. Primeramente se lee la salida estándar del proceso y luego se usa esta expresión regular usada para encontrar patrones dentro de la cadena leída. Al obtenerse el por ciento de la creación, se le pasa a la barra de progreso en la interfaz.	
Resultado Esperado: Que funcione correctamente la expresión regular usada para obtener el por ciento de creación de la imagen ISO.	
Evaluación de la Prueba: Prueba Satisfactoria.	

Caso de Prueba de Aceptación	
Código Caso de Prueba: 3	**Número Historia de Usuario:** UH-ILC-06
Descripción de la Prueba: Se probará la entrada de los datos para la creación de la imagen ISO, entre los que se encuentran el ID del volumen y la versión del LiveCD/DVD.	
Condiciones de Ejecución: Que ya se haya comprimido el sistema.	
Entrada / Pasos de ejecución: Al terminar de comprimirse el sistema original, se llenan los datos necesarios para la generación de la imagen ISO, para esto se chequea que se entre el ID de Volumen y la versión del LiveCD/DVD. Si el usuario no entra algún dato necesario, se muestra un mensaje de error.	
Resultado Esperado: Se entren los datos correctamente.	
Evaluación de la Prueba: Prueba Satisfactoria.	

3.6.7 Caso de Prueba Historia de Usuario: UH-ILC-07

Esta sección cubre el conjunto de pruebas funcionales relacionadas con la historia de usuario: *Gestión de la Configuración.*

En esta historia de usuario lo que se verifica es la gestión de la configuración, permitiéndole a la aplicación Inoue Live Creator generar el LiveCD/DVD correctamente.

En este caso de pruebas se chequeará la copia del script management.sh encargado de ejecutar la optimización del sistema original después de copiado.

Luego se verá la gestión de la configuración del Kernel a usar por el LiveCD/DVD. Así como los permisos necesarios para la ejecución de la aplicación.

Caso de Prueba de Aceptación	
Código Caso de Prueba: 1	**Número Historia de Usuario:** UH-ILC-07
Descripción de la Prueba: Se copiará el script management.sh y se le cambian los permisos para ser ejecutado en el nuevo sistema usando chroot.	
Condiciones de Ejecución: Que ya se haya copiado el sistema original.	
Entrada / Pasos de ejecución: Se copiará el script management.sh y se le cambian los permisos para ser ejecutado en el nuevo sistema usando chroot. Entre las tareas que debe hacer este script se encuentran la de borrar archivos y usuarios innecesarios así como crear otros ficheros importantes.	
Resultado Esperado: La copia correcta del archivo management.sh al sistema recién copiado.	
Evaluación de la Prueba: Prueba Satisfactoria.	

Caso de Prueba de Aceptación	
Código Caso de Prueba: 2	**Número Historia de Usuario:** UH-ILC-07
Descripción de la Prueba: Esta prueba se hace para chequear la copia de los ficheros vmlinuz e initrd.img según el Kernel seleccionado anteriormente. Necesarios para arrancar el LiveCD/DVD.	
Condiciones de Ejecución: Que se haya seleccionado un Kernel y un initrd anteriormente.	
Entrada / Pasos de ejecución: Se copian los ficheros del Kernel y el initrd previamente seleccionados con los nombres vmlinuz e initrd.img respectivamente a la carpeta del LiveCD/DVD para ser usados en el arranque del mismo.	
Resultado Esperado: Que se copien correctamente los ficheros vmlinuz y initrd.img.	
Evaluación de la Prueba: Prueba Satisfactoria.	

Caso de Prueba de Aceptación	
Código Caso de Prueba: 3	**Número Historia de Usuario:** UH-ILC-06
Descripción de la Prueba: Se probará que la aplicación solo sea accesible por el usuario root, pues se necesitan los permisos para hacer chroot al sistema copiado a usar en el LiveCD/DVD.	
Condiciones de Ejecución: Que el usuario que inicia la aplicación sea root.	
Entrada / Pasos de ejecución: Al iniciar la aplicación si no es root el usuario que la inicia, se muestra un mensaje alertando al usuario que la aplicación debe tener permisos de administración y la aplicación termina. De lo contrario la aplicación se ejecuta de forma normal.	
Resultado Esperado: Que si no es root el usuario que ejecuta la aplicación, que se muestre un mensaje de error y termine. De lo contrario que se ejecute normalmente.	
Evaluación de la Prueba: Prueba Satisfactoria.	

3.7 Resultados Obtenidos

En este apartado se relacionan los resultados obtenidos hasta el momento por el equipo de desarrollo de Inoue Live Creator. Resaltar que como resultado de este trabajo, la aplicación está disponible en su versión 0.10. Aunque se esperan muchos más resultados en versiones posteriores.

3.7.1 Acerca del Tiempo de Desarrollo

A mediados del mes de noviembre de 2008 en se comenzaron las tareas de investigación sobre el tema. Principalmente de aplicaciones que hacían este trabajo en varias distribuciones de Linux. Aproximadamente un mes tomó aprender a usarlas y probarlas. Luego se comenzó con el análisis y el diseño de la aplicación propuesta la cual incluía las principales funcionalidades y características de los sistemas analizados.

De todas las tareas de implementación que se llevaron a cabo la que más tiempo consumió fue la creación de los scripts escritos en Bash, usados para optimizar el sistema a usar en el LiveCD/DVD, aproximadamente tomó un mes para que estuviesen listos.

Luego se comenzó a desarrollar la interfaz gráfica para la aplicación la cual fue desarrollada en un tiempo muy corto pues las bondades de las librerías Qt4 son muchas y fáciles de usar. Cerca de dos meses para tener la aplicación en estado beta en la versión 0.9.

3.7.2 Acerca de las Funcionalidades Obtenidas.

Entre las principales funcionales que posee Inoue Live Creator hasta su versión 0.10 se pueden mencionar:

1. Adición o eliminación de ficheros y directorios a excluir del LiveCD/DVD.
2. Selección del sistema raíz personalizado, no tiene porque ser el del sistema que se está usando.
3. Posibilita seleccionar el directorio personal de un usuario para usar esa configuración en el LiveCD/DVD.
4. Configuración del Boot Splash y el título del LiveCD/DVD. Así como del tiempo en que demorará en arrancar el mismo.
5. Es posible seleccionar el Kernel y el Initrd a usar en el LiveCD/DVD de manera fácil seleccionando desde una lista o se puede especificar manualmente otro cualquiera.

6. La aplicación es capaz de chequear si está instalada la herramienta rsync. Necesaria para el correcto funcionamiento de la misma.
7. La aplicación es capaz de chequear si está instalada la herramienta xresprobe. Necesaria para el correcto funcionamiento de la misma.
8. La aplicación es capaz de chequear si está instalada la herramienta live-initramfs. Necesaria para el correcto funcionamiento de la misma.
9. La aplicación es capaz de chequear si está instalada la herramienta discover. Necesaria para el correcto funcionamiento de la misma.
10. La aplicación es capaz de chequear si están instalados los módulos del Kernel necesarios para el correcto funcionamiento de la misma.
11. Es capaz de hacer chroot al sistema a usar en el LiveCD/DVD y hacer modificaciones para optimizar el sistema. Esta funcionalidad le da robustez a la aplicación pues sin tener que recompilar, solo modificando los scripts necesarios, se pueden hacer nuevas tareas de mantenimiento dentro del sistema a usar en el LiveCD/DVD.
12. Comprime el sistema usando squashfs + lzma.
13. Es capaz de mostrar el tamaño de la imagen comprimida, dándole al usuario una idea del tamaño que será el LiveCD/DVD final.
14. Genera un LiveCD/DVD autoarrancable usando Grub como gestor de arranque.

En este capítulo se presentaron algunos casos de pruebas que guiaron la calidad del sistema, y determinaron en cada momento si se estaba o no en condiciones de continuar avanzando. En el análisis de los resultados obtenidos se muestran las funcionalidades alcanzadas por el sistema en el período que se ha estado trabajando en su desarrollo.

CONCLUSIONES

En esta investigación, se ha hecho un estudio sobre las principales aplicaciones existentes a nivel mundial para la generación de LiveCD/DVDs en diferentes plataformas. De las cuales la mayoría son a nivel de consola lo que hace que el proceso de creación de un LiveCD/DVD sea una tarea muy complicada para el usuario final.

De estas aplicaciones se analizaron sus características y funcionamiento llegando a un consenso de cuales debería tener el sistema propuesto y cuales no. A partir de aquí se implementó y probó la aplicación **Inoue Live Creator** que permite generar un LiveCD/DVD de forma fácil y rápida a partir de un sistema Debian GNU/Linux instalado en el ordenador . Por lo anteriormente planteado se concluye que los objetivos trazados en este trabajo se han cumplido satisfactoriamente.

RECOMENDACIONES

Como producto final de la presente investigación, la aplicación Inoue Live Creator permite la generación de un LiveCD/DVD de forma rápida y fácil para usuarios finales. Esta presenta un gran número de opciones de configuración, entre otras características, permite seleccionar un Kernel personalizado, copiar las configuraciones personales de un usuario determinado para usar en el LiveCD/DVD, comprime el sistema para que quepa dentro de un CD o en un DVD y chequea que las aplicaciones necesaria para el correcto funcionamiento del mismo.

La aplicación tiene un manual de usuario que muestra al usuario las diferentes secciones de configuración del sistema. Por todas estas características se recomienda:

1. Que el presente trabajo se siga desarrollando para implementar nuevas funcionalidades y mejorar su calidad y robustez.
2. Que se realice un estudio de la complejidad de los algoritmos empleados para el desarrollo del producto con el objetivo de mejorar su rendimiento.
3. Que el producto sea probado a gran escala a fin de validar su funcionamiento.
4. Que se ponga la aplicación a disposición de la Comunidad de Software Libre en el GForge de la Facultad.
5. Que el trabajo sea inscrito como un producto propio de la UCI.

REFERENCIAS BIBLIOGRÁFICAS

[1] Free Software Foundation. Free Software. [Fecha de consulta: 19 de noviembre del 2008]. Disponible en <http://www.gnu.org/philosophy/free-sw.html>

[2] Fidel C. R, 2004, Discurso en la clausura del VIII Congreso de la UJC. Disponible en <http://www.cuba.cu/gobierno/discursos/2004/esp/f051204e.html>

[3] Ubuntu Forums Users. LiveCD/DVD HowTo. [Fecha de consulta: 21 de noviembre del 2008]. Disponible en: <http://www.remastersys.klikit-linux.com/capink.html>

[4] Debian GNU/Linux Packages. Dfsbuild. [Fecha de consulta: 21 de noviembre del 2008]. Disponible en: < http://packages.debian.org/es/etch/dfsbuild>

[5] Debian Administration Team. Bootcd. [Fecha de consulta: 22 de noviembre del 2008]. Disponible en: < http://www.debian-administration.org/articles/148>

[6] Fedora Project Team. LiveCD-Tools. [Fecha de consulta: 22 de noviembre del 2008]. Disponible en: < http://fedoraproject.org/wiki/FedoraLiveCD>

[7] Debian GNU/Linux Packages. Live-Magic. [Fecha de consulta: 23 de noviembre del 2008]. Disponible en: <http://packages.debian.org/lenny/live-magic>

[8] EsDebian.org Team. Live-Helper. [Fecha de consulta: 23 de noviembre del 2008]. Disponible en: < http://www.esdebian.org/wiki/live-helper>

[9] GuiaUbuntu.org Team. Remastersys. [Fecha de consulta: 23 de noviembre del 2008]. Disponible en <http://www.guia-ubuntu.org/index.php?title=Remastersys>

[10] Linux Live Team. Linux Live Scripts. [Fecha de consulta: 24 de noviembre del 2008]. Disponible en: < http://www.linux-live.org/>

[11] UCK Team. Ubuntu Customization Kit. [Fecha de consulta: 24 de noviembre del 2008]. Disponible en: < http://uck.sourceforge.net/>

[12] Reconstructor Team. Reconstructor. [Fecha de consulta: 24 de noviembre del 2008]. Disponible en: <http://reconstructor.aperantis.com/>

[13] Nova Team. LiveCD-Kit. [Fecha de consulta: 24 de noviembre del 2008]. Disponible en: < http://gforge.f10.uci.cu/projects/nova >

[14] GTK Developers. GTK. . [Fecha de consulta: 24 de noviembre del 2008]. Disponible en: < http://www.gtk.org/>

[15] Qt Software. Librerías Qt. [Fecha de consulta: 25 de noviembre del 2008]. Disponible en: < http://doc.trolltech.com/>

[16] KDevelop Team. KDevelop. [Fecha de consulta: 25 de noviembre del 2008]. Disponible en: < http://www.kdevelop.org/>

[17] DAYRON, P. R., 2008. Sistema de Clonación y Distribución de Imágenes de Sistemas Operativos. Disponible en: <http://bibliodoc.uci.cu/TD/TD_1245_08.pdf>

[18] Qt Creator Team. QtCreator. [Fecha de consulta: 26 de noviembre del 2008]. Disponible en: < http://www.qtsoftware.com/developer/qt-creator>

[19] Qt Designer Team. Qt Designer. [Fecha de consulta: 26 de noviembre del 2008]. Disponible en: < http://doc.trolltech.com/3.3/designer-manual.html>

[20]Idem [19]

[21] Tomas M. Squash LZMA. [Fecha de consulta: 28 de noviembre del 2008]. Disponible en: < http://www.squashfs-lzma.org/>

[22] Debian GNU/Linux Packages. GenIsoImage. [Fecha de consulta: 29 de noviembre del 2008]. Disponible

en: <http://packages.debian.org/unstable/otherosfs/genisoimage>

[23] Linux / Unix Command. cp. [Fecha de consulta: 29 de noviembre del 2008]. Disponible en: < http://linux.about.com/od/commands/l/blcmdl1_cp.htm>

[24] Wayne Davison. Rsync. [Fecha de consulta: 29 de noviembre del 2008]. Disponible en: < http://samba.anu.edu.au/rsync>

[25] UnionFS Team. UnionFS. [Fecha de consulta: 5 diciembre del 2008]. Disponible en: <http://www.filesystems.org/project-unionfs.html>

[26] SquashFS Team. SquashFS. [Fecha de consulta: 6 diciembre del 2008]. Disponible en: <http://squashfs.sourceforge.net/>

[27] OMG Team. UML. [Fecha de consulta: 6 diciembre del 2008]. Disponible en: <http://www.omg.org/technology/documents/formal/uml.htm>

[28] GNU.org Team. Bash Reference Manual. [Fecha de consulta: 6 diciembre del 2008]. Disponible en: <http://www.gnu.org/software/bash/manual/bashref.html>

[29] GNU.org Team. Bash. [Fecha de consulta: 6 diciembre del 2008]. Disponible en: <http://www.gnu.org/software/bash/>

[30] Juan Soulie. C/C++. [Fecha de consulta: 7 diciembre del 2008]. Disponible en: <http://www.cplusplus.com/info/description.html>

[31] Idem [30]

[32] MALAY, R. V., 2007 Introducción de procedimientos ágiles en la producción de software en la Facultad 7 de la Universidad de las Ciencias Informáticas. Disponible en: <http://bibliodoc.uci.cu/TD/TD_0693_07.pdf>

[33] Idem [32]

[34] Idem [33]

[35] GLADYS M P. R., 2008, MA-GMPR-UR2, Metodología ágil para proyectos de software libre. Disponible en: <http://gforge.f10.uci.cu/projects/magmpr/>

BIBLIOGRAFÍA

MALAY, R. V., 2007 Introducción de procedimientos ágiles en la producción de software en la Facultad 7 de la Universidad de las Ciencias Informáticas. Disponible en: <http://bibliodoc.uci.cu/TD/TD_0693_07.pdf>

Capink Ubuntu Forums User. LiveCD/DVD HowTo. [En Línea] [Fecha de consulta: 21 de noviembre del 2008]. Disponible en: <http://www.remastersys.klikit-linux.com/capink.html>

DAYRON, P. R., 2008. Sistema de Clonación y Distribución de Imágenes de Sistemas Operativos. Disponible en: <http://bibliodoc.uci.cu/TD/TD_1245_08.pdf>

Free Software Foundation. Free Software. [Fecha de consulta: 19 de noviembre del 2008]. Disponible en http://www.gnu.org/philosophy/free-sw.html

Fidel C. R, 2004, Discurso en la clausura del VIII Congreso de la UJC. Disponible en <http://www.cuba.cu/gobierno/discursos/2004/esp/f051204e.html>

Ubuntu Forums Users. LiveCD/DVD HowTo. [Fecha de consulta: 21 de noviembre del 2008]. Disponible en: <http://www.remastersys.klikit-linux.com/capink.html>

Debian GNU/Linux Packages. Dfsbuild. [Fecha de consulta: 21 de noviembre del 2008]. Disponible en: < http://packages.debian.org/es/etch/dfsbuild>

Debian Administration Team. Bootcd. [Fecha de consulta: 22 de noviembre del 2008]. Disponible en: < http://www.debian-administration.org/articles/148>

Fedora Project Team. LiveCD-Tools. [Fecha de consulta: 22 de noviembre del 2008]. Disponible en: < http://fedoraproject.org/wiki/FedoraLiveCD>

Debian GNU/Linux Packages. Live-Magic. [Fecha de consulta: 23 de noviembre del 2008]. Disponible en: <http://packages.debian.org/lenny/live-magic>

EsDebian.org Team. Live-Helper. [Fecha de consulta: 23 de noviembre del 2008]. Disponible en:
< http://www.esdebian.org/wiki/live-helper>

GuiaUbuntu.org Team. Remastersys. [Fecha de consulta: 23 de noviembre del 2008]. Disponible en
<http://www.guia-ubuntu.org/index.php?title=Remastersys>

Linux Live Team. Linux Live Scripts. [Fecha de consulta: 24 de noviembre del 2008]. Disponible en:
< http://www.linux-live.org/>

UCK Team. Ubuntu Customization Kit. [Fecha de consulta: 24 de noviembre del 2008]. Disponible en:
< http://uck.sourceforge.net/>

Reconstructor Team. Reconstructor. [Fecha de consulta: 24 de noviembre del 2008]. Disponible en:
<http://reconstructor.aperantis.com/>

Nova Team. LiveCD-Kit. [Fecha de consulta: 24 de noviembre del 2008]. Disponible en:
< http://gforge.f10.uci.cu/projects/nova >

GTK Developers. GTK. . [Fecha de consulta: 24 de noviembre del 2008]. Disponible en:
< http://www.gtk.org/>

Qt Software. Librerías Qt. [Fecha de consulta: 25 de noviembre del 2008]. Disponible en:
< http://doc.trolltech.com/>

KDevelop Team. KDevelop. [Fecha de consulta: 25 de noviembre del 2008]. Disponible en:
< http://www.kdevelop.org/>

Qt Creator Team. QtCreator. [Fecha de consulta: 26 de noviembre del 2008]. Disponible en:
< http://www.qtsoftware.com/developer/qt-creator>

Qt Designer Team. Qt Designer. [Fecha de consulta: 26 de noviembre del 2008]. Disponible en:
< http://doc.trolltech.com/3.3/designer-manual.html>

Tomas M. Squash LZMA. [Fecha de consulta: 28 de noviembre del 2008]. Disponible en:
< http://www.squashfs-lzma.org/>

Debian GNU/Linux Packages. GenIsoImage. [Fecha de consulta: 29 de noviembre del 2008]. Disponible en:
<http://packages.debian.org/unstable/otherosfs/genisoimage>

Linux / Unix Command. cp. [Fecha de consulta: 29 de noviembre del 2008]. Disponible en:
< http://linux.about.com/od/commands/l/blcmdl1_cp.htm>

Wayne Davison. Rsync. [Fecha de consulta: 29 de noviembre del 2008]. Disponible en:
< http://samba.anu.edu.au/rsync>

UnionFS Team. UnionFS. [Fecha de consulta: 5 diciembre del 2008]. Disponible en:
<http://www.filesystems.org/project-unionfs.html>

SquashFS Team. SquashFS. [Fecha de consulta: 6 diciembre del 2008]. Disponible en:
<http://squashfs.sourceforge.net/>

OMG Team. UML. [Fecha de consulta: 6 diciembre del 2008]. Disponible en:
<http://www.omg.org/technology/documents/formal/uml.htm>

GNU.org Team. Bash Reference Manual. [Fecha de consulta: 6 diciembre del 2008]. Disponible en:
<http://www.gnu.org/software/bash/manual/bashref.html>

GNU.org Team. Bash. [Fecha de consulta: 6 diciembre del 2008]. Disponible en:
<http://www.gnu.org/software/bash/>

Juan Soulie. C/C++. [Fecha de consulta: 7 diciembre del 2008]. Disponible en:
<http://www.cplusplus.com/info/description.html>

GLADYS M P. R., 2008, MA-GMPR-UR2, Metodología ágil para proyectos de software libre. Disponible en:
<http://gforge.f10.uci.cu/projects/magmpr/>

GLOSARIO DE TERMINOS

C

Callback: En programación de computadoras, un callback es una porción de código ejecutable que es pasado como un argumento a otro código. Esto permite construir código de bajo nivel como capas de abstracción que pueden ser llamadas desde una subrutina (o función) definida en una capa de mayor nivel.

F

Framework: es una estructura de soporte definida en la cual otro proyecto de software puede ser organizado y desarrollado. Típicamente, un framework puede incluir soporte de programas, bibliotecas y un lenguaje interpretado entre otros software para ayudar a desarrollar y unir los diferentes componentes de un proyecto.

FSF: La Fundación para el Software Libre (Free Software Foundation) es una organización creada en Octubre de 1985 por Richard Matthew Stallman y otros entusiastas del Software Libre con el propósito de difundir este movimiento. La Fundación para el Software Libre (FSF) está dedicada a eliminar las restricciones sobre la copia, redistribución, entendimiento, y modificación de programas de computadoras. Con este objeto, promociona el desarrollo y uso del software libre en todas las áreas de la computación, pero muy particularmente, ayudando a desarrollar el sistema operativo GNU.

G

Gcc: GCC es un compilador integrado del proyecto GNU para C, C++, Objective C y Fortran; es capaz de recibir un programa fuente en cualquiera de estos lenguajes y generar un programa ejecutable binario en el lenguaje de la máquina donde ha de correr.

GDB: GDB o GNU Debugger es el depurador estándar para el sistema operativo GNU. Es un depurador portable que se puede utilizar en varias plataformas Unix y funciona para varios lenguajes de programación como C, C++ y Fortran. GDB fue escrito por Richard Stallman en 1988. GDB es software libre distribuido bajo la licencia GPL.

GPL: La GNU General Public License (inglés: Licencia Pública General) es una licencia creada por la Free Software Foundation y orientada principalmente a los términos de distribución, modificación y uso de software. Su propósito es declarar que el software cubierto por esta licencia es Software Libre.

Grub Splash: Un Grub Splash es una imagen de 14 colores que se pone detrás del menú de selección del gestor de arranque Grub., 98

Grub: En computación, el Gestor de Arranque Unificado (GRand Unified Bootloader), es un gestor de arranque múltiple que se usa comúnmente para iniciar dos o más sistemas operativos instalados en un mismo ordenador.

GTK: GTK+ o The GIMP Toolkit es un conjunto de bibliotecas multiplataforma para desarrollar interfaces gráficas de usuario (GUI), principalmente para los entornos gráficos GNOME, XFCE y ROX aunque también se puede usar en el escritorio de Windows, MacOS y otros. Inicialmente fueron creadas para desarrollar el programa de edición de imagen GIMP, sin embargo actualmente se usan mucho por muchos otros programas en los sistemas GNU/Linux. Junto a Qt es uno de las bibliotecas más populares para X Window System.

I

IDE: Entorno de desarrollo integrado o en inglés Integrated Development Environment (IDE). Es un programa compuesto por un conjunto de herramientas para un programador desde el que se pueden editar programas, compilarlos y depurarlos.

Init: El propósito principal de init en el initramfs es preparar el montaje y el acceso al sistema de archivos raíz real. Entre otras tareas es responsable de cargar los módulos del núcleo y gestionar la configuración de la red.

Initramfs: Es un pequeño sistema de archivos que el núcleo puede cargar en un disco RAM. Proporciona un entorno Linux mínimo que habilita la ejecución de programas antes de que se monte el sistema de archivos raíz. El entorno Linux mínimo se carga en la memoria mediante las rutinas de la BIOS y no necesita requisitos específicos de hardware, únicamente una memoria suficiente. Initramfs siempre debe proporcionar un ejecutable denominado init que ejecuta el programa init actual en el sistema de archivos raíz para que se lleve a cabo el proceso de arranque.

Instalar: Incorporar a la computadora un programa o dispositivo para ser utilizado.

ISO 9660: El estándar ISO 9660 es una norma publicada inicialmente en 1986 por la ISO, que especifica el formato para el almacenaje de archivos en los soportes de tipo disco compacto. El estándar ISO 9660 define un sistema de archivos para CD-ROM. Su propósito es que tales medios sean legibles por diferentes sistemas operativos,

de diferentes proveedores y en diferentes plataformas, por ejemplo, MS-DOS, Microsoft Windows, Mac OS y UNIX.

K

Kernel: Núcleo. Parte esencial de un sistema operativo que provee los servicios más básicos del sistema. Se encarga de gestionar los recursos como el acceso seguro al hardware de la computadora. Se encarga también del multiplexado, determinando qué programa accederá a un determinado hardware si dos o más quieren usarlo al mismo tiempo.

Knoppix: Es una distribución de GNU/Linux basada en Debian y que por defecto utiliza KDE aunque en el menú de arranque se puede especificar el tipo de interface gráfica a usar (Gnome, IceWM,...). Desde la versión actual 6.0.X incorpora el escritorio LXDE. Esta distribucion está desarrollada por el consultor de GNU/Linux Klaus Knopper.

L

Libre: O Software Libre se refiere a la libertad de los usuarios para ejecutar, copiar, distribuir, estudiar, cambiar y mejorar el software.

LiveCD/DVD: Un Live CD o Live DVD, más genéricamente Live Distro, (traducido en ocasiones como CD vivo o CD autónomo), es un sistema operativo (normalmente acompañado de un conjunto de aplicaciones) almacenado en un medio extraíble, tradicionalmente un CD o un DVD (de ahí sus nombres), que puede ejecutarse desde éste sin necesidad de instalarlo en el disco duro de una computadora, para lo cual usa la memoria RAM como disco duro virtual y el propio medio como sistema de ficheros.

LiveUSB: Un Live USB es una Memoria USB que contiene un completo sistema operativo, el cual permite arrancar una computadora desde él. Los Live USBs están estrechamente relacionados con los LiveCDs, y algunas veces son usados de manera intercambiable. Tal como los LiveCDs, los Live USBs pueden ser usados para la administración de sistemas, la recuperación de datos, o para pruebas en distribuciones del sistema operativo GNU/Linux, sin modificar una instalación local, en la unidad de disco duro. Muchas de las más pequeñas distribuciones Linux también pueden ser usadas desde una memoria USB.

Logs: Un log es un registro oficial de eventos durante un periodo de tiempo en particular. Para los profesionales en seguridad informática es usado para registrar datos o información sobre quién, qué, cuándo, dónde y por qué un evento ocurre para un dispositivo en particular o aplicación. En sistemas tipo Unix los logs generalmente están almacenados en el directorio /var/logs.

LZMA: El algoritmo Lempel-Ziv-Markov, es un algoritmo usado para comprimir datos, esta siendo desarrollado desde 1998, y es usado en el formato 7z del archivador 7-Zip. Este algoritmo usa un esquema de compresión por diccionario similar al LZ77. Este algoritmo tiene un gran radio de compresión (generalmente superior que el de bzip2).

P

Phonon: Phonon es el nuevo framework multimedia de KDE 4. El objetivo de Phonon es facilitar a los programadores el uso de tecnologías multimedia en sus programas, así como asegurar que las aplicaciones que usen Phonon funcionen en diversas plataformas y arquitecturas de sonido.

Privativo: El software no libre (también llamado software propietario, software privativo, software privado, software con propietario o software de propiedad) se refiere a cualquier programa informático en el que los usuarios tienen limitadas las posibilidades de usarlo, modificarlo o redistribuirlo (con o sin modificaciones), o cuyo código fuente no está disponible o el acceso a éste se encuentra restringido.

Q

Qt: Qt es una biblioteca multiplataforma para desarrollar interfaces gráficas de usuario. La biblioteca la desarrolla la que fue su creadora, la compañía noruega Trolltech, actualmente renombrada a Qt Software, y que desde junio de 2008 es propiedad de Nokia.

R

RAM: Memoria de acceso aleatorio. Tipo de memoria donde la computadora guarda información para que pueda ser procesada más rápidamente. En la memoria RAM se almacena toda información que está siendo usada en el momento.

Root: En sistemas operativos del tipo Unix, root es el nombre convencional de la cuenta de usuario que posee todos los derechos en todos los modos (mono o multiusuario). root es también llamado superusuario. Normalmente esta es la cuenta de administrador. El usuario root puede hacer muchas cosas que un usuario común no, tales como cambiar el dueño de archivos y enlazar a puertos de numeración.

S

Scripts: En informática, un script es un guión o conjunto de instrucciones. Permiten la automatización de tareas creando pequeñas utilidades. Es muy utilizado para la administración de sistemas UNIX. Son ejecutados por un intérprete de línea de órdenes y usualmente son archivos de texto. También un script puede considerarse una

alteración o acción a una determinada plataforma. Los scripts de este programa están programados en Bash.

SELinux: SELinux (del inglés Security-Enhanced Linux, Seguridad Mejorada de Linux) es una característica de seguridad de Linux que provee una variedad de políticas de seguridad, incluyendo el estilo de acceso a los controles del Departamento de Defensa de Estados Unidos, a través del uso de módulos de Seguridad en el núcleo de Linux. No es una distribución de Linux, sino un set de modificaciones que pueden ser aplicadas a un sistema Tipo-Unix como Linux y BSD.

Squashfs: Es un sistema de ficheros de solo lectura que tienen grandes índices de compresión. Al ser usado en un LiveCD/DVD los ficheros residen comprimidos y se van descomprimiendo según se usan. Optimizando así el espacio que ocupan en el disco y memoria. Haciendo transparente su descompresión. Ésta sólo se hace cuando se va a utilizar algo, con lo que no hay que descomprimir todo el sistema a RAM, sólo lo que se esté usando.

Swap: La swap o espacio de intercambio es una zona del disco (un fichero o partición) que se usa para guardar información sobre los procesos que no han de mantenerse en memoria física. Permitiendo liberar la memoria principal para cargar otros procesos.

T

Trolltech: Qt Software (conocida anteriormente como Trolltech y antes como Quasar Technologies) es una compañía de software fundada en el año 1994 en Oslo, Noruega. Su principal actividad es proveer herramientas y bibliotecas de desarrollo de software, así como servicio experto de consulta.

U

UnionFS: Es un servicio para sistemas de archivos de Linux que permite montar un sistema de archivos formado por la unión de otros sistemas de archivos de Linux. Permite que archivos y directorios de sistemas de archivos distintos, conocidos como ramas, se superpongan de forma transparente, formando un único sistema de archivos. Los contenidos de directorios que tienen la misma ruta en las ramas que se combinan aparecerán juntos en un único directorio en el nuevo sistema de archivos virtual.

UNIX: Unix (registrado oficialmente como UNIX®) es un sistema operativo portable, multitarea y multiusuario; desarrollado, en principio, en 1969 por un grupo de empleados de los laboratorios Bell de AT&T, entre los que figuran Ken Thompson, Dennis Ritchie y Douglas McIlroy.