Universitat Oberta de Catalunya
Ingeniería Técnica en Informática de Sistemas
Trabajo Final de Carrera

UOCbuntu:

Distribución Ubuntu orientada a estudiantes de ingeniería informática

Memoria

Luis Sánchez Nogales
Consultor: Miquel A. Senar Rosell
Junio 2012

Índice de contenido

1 Introducción

1.1 Descripción

El presente proyecto trata la creación de una distribución *live persistente* a partir de la distribución Ubuntu 12.04 LTS. Esta distribución, denominada **UOCbuntu**, incluye las herramientas *open source* necesarias para satisfacer las necesidades de los estudiantes de las ingenierías en informática de la Universitat Oberta de Catalunya.

1.2 Objetivos

El objetivo principal es ofrecer al estudiante una plataforma completa para el estudio de las asignaturas de ingeniería informática. Se añade la posibilidad del modo *live persistente* para utilizar UOCbuntu a partir de una instalación en una memoria usb de forma que se almacenen los cambios realizados por el usuario.

La creación de la nueva distribución se realiza a partir de otra distribución liveCD existente para proporcionar soporte en cuanto a nuevas actualizaciones. Esta distribución se modifica de forma que el software incluido se limite a satisfacer las necesidades de los estudiantes, con lo que el punto central son las herramientas utilizadas a lo largo de los estudios de ETIS, aprovechando la apuesta que hace la UOC por el software libre.

Si bien durante los estudios es necesaria la utilización de software propietario, el desarrollo de este TFC también abarca la búsqueda, prueba e inclusión de aquellas herramientas equivalentes con licencia pública. También se incluyen otras opciones de software relacionado con las asignaturas de la ingeniería y, aunque no son utilizadas actualmente en la docencia, pueden resultar de gran utilidad para el estudio. Por último, la opción de utilizar software propietario o de código no libre queda descartada. UOCbuntu sólo incluye software creado bajo cualquier tipo de licencia pública.

1.3 Motivaciones

A lo largo de los estudios de ETIS son muchas las herramientas utilizadas y muchas las aplicaciones y las finalidades con las que se utilizan. La UOC proporciona a

los estudiantes el software necesario, ya sea para utilizar sobre soporte propietario como para su uso sobre plataformas libres. Esto da lugar a la obligación por parte de la UOC de dotar a los alumnos al inicio de cada semestre con todas estas herramientas, licencias y diferentes versiones, cuando podemos encontrar una solución que englobe todas estas opciones: la creación de una distribución basada en GNU/Linux propia para la UOC.

A partir de las distribuciones basadas en el sistema operativo GNU/Linux tenemos la oportunidad de crear una distribución propia con la ventaja del soporte existente para dichas distribuciones. Algunos ejemplos son: Ubuntu, openSUSE, Debian, Linux Mint, Fedora y Mandriva entre otras. Esto nos permitirá modificar una de estas distribuciones para adaptarla a las necesidades de los alumnos de informática de la UOC, para que con un único producto los estudiantes dispongan de lo necesario para llevar a cabo sus estudios. Además, se añade la capacidad de incorporar actualizaciones futuras mediante el uso de repositorios de software.

Por otro lado, la idea de crear una distribución propia basada en una plataforma GNU/Linux pone a disposición de los usuarios la posibilidad de estudiar aspectos del mismo software que se les proporciona. Esta característica es propia del software libre, lo que permite que los propios usuarios analicen el código y practiquen los conocimientos adquiridos. A su vez, esta apuesta supone una oportunidad para muchos alumnos de dar el paso definitivo en la migración a sistemas GNU/Linux, beneficiándose de las ventajas de su uso con fines académicos frente a otras opciones propietarias más complejas.

Como resultado, la *comunidad open source,* y a la vez el soporte del software libre aumentará, de forma que se mejore y fomente el uso de herramientas con licencia pública, además de servir de vehículo en el aprendizaje de los estudios de informática.

1.4 Alcance

La selección del contenido que incorpora UOCbuntu se ha realizado en base a la propia experiencia como alumno de la Universitat Oberta de Catalunya. Se ha realizado una recopilación del software utilizado en todas las asignaturas cursadas para cumplir con los requisitos de cualquier estudiante. Sin embargo, esta distribución también puede ser utilizada por estudiantes de otras ingenierías relacionadas como pueden ser: ingeniería técnica en informática de gestión, grado en ingeniería informática, e incluso ingenierías industriales y de telecomunicaciones. En última instancia, también puede ser utilizada por estudiantes de módulos de grado medio y superior, e incluso por estudiantes de certificaciones técnicas especializadas.

1.5 Justificación

La creación o adaptación de una distribución basada en sistemas GNU/Linux para cubrir las necesidades de los estudiantes de informática de la UOC no es algo nuevo. A lo largo de estos años han sido varias las propuestas como proyecto final de carrera de otros compañeros, sin embargo, la UOC no termina de lanzarse en la creación definitiva de una versión oficial para el alumnado.

Como consecuencia, a día de hoy cuando un alumno de informática termina sus estudios puede comprobar la acumulación de cds recibidos al inicio de cada semestre (en mi caso este número asciende a quince), entre distribuciones, tanto de uso específico como general, y aplicaciones.

Por tanto, mediante este proyecto se pretende demostrar la viabilidad de utilizar una distribución propia, aprovechando la evolución de la última versión de escritorio de Ubuntu y la madurez de los sistemas basados en GNU/Linux. Por otro lado, también se pretende demostrar la autosuficiencia en la utilización del software libre en estudios superiores.

2 Planificación

2.1 Estimaciones iniciales

Dado que el TFC se imparte como asignatura semestral nos vemos en la obligación de limitar el desarrollo a la duración del semestre y a las fechas impuestas para las diferentes entregas. Se ha optado por una división temporal equitativa y acorde en proporción con las pruebas de evaluación continua (PACs). Según la complejidad se ha realizado una estimación de tiempo incial en la que se tuvieron en cuenta las horas necesarias para cada una de las fases por las que ha de transcurrir el desarrollo del proyecto.

Por tanto, la estimación realizada en un primer momento para el plan de trabajo inicial es relativa a los aspectos tecnológicos y operativos, descartando obviamente los aspectos económicos.

En cuanto a la implementación existen diferentes métodos para crear una distribución basada en GNU/Linux:

- Linux From Scratch
- A partir de otras distribuciones existentes
- A partir de otras distribuciones liveCD existentes
- Mediante herramientas automáticas: Remastersys, UCK, Reconstructor, etc.

La utilización del método influye directamente en la estimación de tiempo realizado para el TFC. El coste temporal puede sufrir un aumento exponencial según el método elegido. Por otro lado, esta elección también influye en el nivel de personalización y adaptación a las necesidades requeridas.

La elección basada en la modificación de una distribución liveCD de escritorio existente nos proporciona una estimación inicial adecuada al tiempo disponible durante el semestre. Además, supone una base a partir de la que empezar el trabajo evitando la fase previa relativa a la compilación de las fuentes.

2.2 Viabilidad técnica

Tal y como se menciona antes el desarrollo del TFC no sólo supone la creación de una distribución live adaptada, sino también el análisis de herramientas libres útiles en los estudios de informática. Por tanto, el mismo desarrollo del TFC se realiza íntegramente mediante software de código libre.

El desarrollo del TFC necesita la utilización de herramientas en un entorno controlado y con las característica necesarias para ello. La solución se basa en la utilización de máquinas virtuales donde ejecutar las diferentes versiones de la distribución. Dichas máquinas se ejecutarán sobre una máquina guest basada en una plataforma GNU/Linux. En la máquina virtual se realizarán operaciones de montaje, cambios de raíz en el sistema de archivos, modificación y gestión del sistema, para luego, crear el archivo .iso que contendrá todos los archivos necesarios para su ejecución en modo live en cualquier equipo.

El hardware utilizado debe dispone de potencia suficiente para la ejecución de máquinas virtuales y tareas de compresión y empaquetado en un tiempo razonable. Para ello se cuenta con un equipo portátil con las siguientes características:

Figura 1: Características del equipo utilizado

Modelo	Asus A53SD
Procesador	Intel® Core™ i7-2670QM velocidad: 2.2 Ghz núcleos: 4 (8 hilos de ejecución) caché: 6 MB
Memoria	6GB DDR3 velocidad: 1333 Mhz
Tarjeta Gráfica	nVidia 610M
Disco Duro	500GB sATA

2.3 Viabilidad operativa

Las capacidades necesarias para hacer frente al desarrollo de este TFC son las relacionadas a la administración de sistemas unix/linux: actualizaciones, des/instalación de software, soporte (repositorios), conocimientos de la estructura del sistema de archivos y utilización de la consola (*shell*) a nivel intermedio. También son necesarios conocimientos en la utilización de software adicional *open source*: máquinas virtuales y herramientas de diseño gráfico para la parte visual de la distribución. Por otro lado, los aspectos de sistemas operativos vistos en los estudios de ETIS, también han sido de gran ayuda en el desarrollo de este proyecto.

2.4 Programa de trabajo

A partir de las estimaciones iniciales y el análisis de viabilidad del TFC llegamos a la planificación del proyecto. El siguiente paso trató el desglose de las tareas del proyecto siguiendo una división por fases acorde con las entregas parciales y el porcentaje de trabajo completado sugerido en el plan de estudios.

A continuación se presenta el plan de trabajo utilizado en forma de tabla. Se pueden comprobar las fechas concretas de inicio y término de las tareas realizadas durante el semestre. Este plan de trabajo se ha respetado en todo momento y ha sido el guión principal en el desarrollo de este proyecto:

Figura 2: Plan de trabajo

	Nombre	Inicio	Terminado
1	**⊟Fase Previa**	21/02/12 17:00	5/03/12 17:00
2	Elección TFC	21/02/12 17:00	5/03/12 17:00
3	**⊟Fase Inicial**	5/03/12 17:00	1/04/12 17:00
4	Lectura manuales	5/03/12 17:00	1/04/12 17:00
5	Análisis distribuciones live actuales	5/03/12 17:00	15/03/12 17:00
6	Elección distribución live base	15/03/12 17:00	19/03/12 17:00
7	**⊟Preparación entorno trabajo**	19/03/12 17:00	1/04/12 17:00
8	Instalación software entorno trabajo	19/03/12 17:00	23/03/12 17:00
9	Preparación equipo guest	19/03/12 17:00	28/03/12 17:00
10	Testing inicial	28/03/12 17:00	1/04/12 17:00
11	**⊟Fase de diseño y desarrollo**	1/04/12 17:00	15/05/12 17:00
12	Análisis software ETIS UOC	1/04/12 17:00	10/04/12 17:00
13	Búsqueda de software GPL equivalente	10/04/12 17:00	19/04/12 17:00
14	**⊟Actualización distribución live base**	2/04/12 17:00	11/05/12 17:00
15	Eliminación software prescindible	2/04/12 17:00	15/04/12 17:00
16	Actualización kernel a última versión con soporte	15/04/12 17:00	20/04/12 17:00
17	Instalación software ETIS UOC	20/04/12 17:00	11/05/12 17:00
18	Actualización repositorios distribución	20/04/12 17:00	11/05/12 17:00
19	**⊟Modificación diseño distribución**	11/05/12 17:00	15/05/12 17:00
20	Diseño logotipo distribución	11/05/12 17:00	12/05/12 17:00
21	Wallpaper	12/05/12 17:00	15/05/12 17:00
22	Iconos	12/05/12 17:00	15/05/12 17:00
23	Splash	12/05/12 17:00	15/05/12 17:00
24	**⊟Fase Final**	15/05/12 17:00	21/05/12 17:00
25	Pruebas finales	15/05/12 17:00	20/05/12 17:00
26	Análisis de resultados	20/05/12 17:00	21/05/12 17:00
27	Memoria	21/05/12 17:00	8/06/12 17:00
28	Presentación Virtual	21/05/12 17:00	15/06/12 17:00

2.4.1 Fase previa

En un primer momento podemos considerar la gestación del proyecto a desarrollar como la fase previa, donde se discutieron con el consultor las posibilidades planteadas al inicio. Finalmente, se decidió por la creación de una distribución específica basada en Ubuntu.

2.4.2 Fase inicial

Una vez elegido el TFC a desarrollar se pasó a la fase inicial. A partir de aquí se empezó a investigar sobre los distintos métodos de creación de distribuciones GNU/Linux y las versiones existentes en modo live. En esta fase la colaboración de la **comunidad open source** ha sido clave, gracias a todas las aportaciones en forma de manuales y *how to´s* que existen en internet. A esto hay que añadir el soporte existente en foros para ayudar de forma desinteresada en aspectos técnicos de las distintas distribuciones.

Una vez elegida la distribución que se utilizaría como base para el desarrollo de nuestro proyecto se pasó a la preparación del entorno de trabajo. Este consiste en la instalación de la máquina virtual con la distribución base y una primera fase de pruebas sobre el mismo entorno.

2.4.3 Fase de diseño y desarrollo

Dentro de esta etapa destacamos las tareas que componen el eje principal del trabajo: el análisis y recopilación del software utilizado durante la carrera, la búsqueda de alternativas de licencia libre y su incorporación a la distribución final. Además, también se han realizado cambios en la distribución base, como es la eliminación de software considerado poco útil para los estudiantes, actualización del núcleo para mayor compatibilidad de hardware y la actualización de los repositorios de software.

Otro tipo de modificaciones son aquellas relativas al aspecto visual. El objetivo ha sido dotar a la distribución de personalidad propia mediante la creación de iconos, wallpapers y modificación de scripts del sistema para mantener un equilibrio estético general. También se han organizado los menús, submenús y accesos a las aplicaciones para proporcionar al estudiante un acceso cómodo y sencillo desde el escritorio.

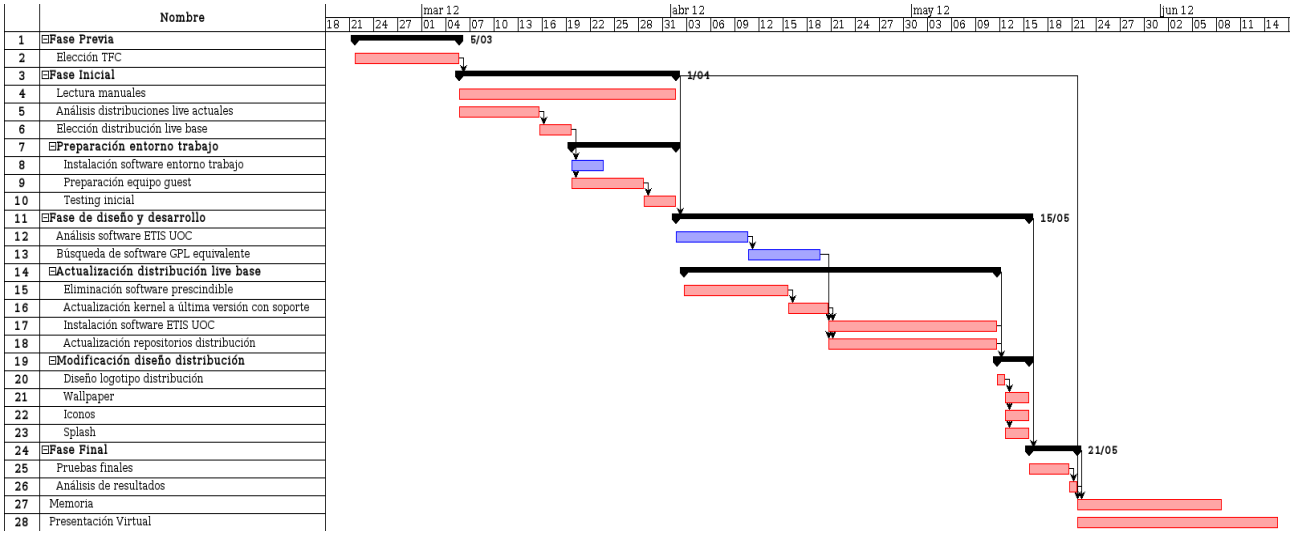
2.4.4 Fase final

En esta última etapa del TFC se realizan todas las tareas relativas a las pruebas finales y al análisis de resultados. Es necesario que todo el software incorporado sea completamente funcional para que se cumplan los objetivos propuestos. También se han realizado pruebas en otros equipos para comprobar la portabilidad del sistema y el funcionamiento de las aplicaciones en diferentes configuraciones de hardware.

2.5 Diagrama de Gantt

A partir del programa de trabajo anterior podemos construir el diagrama de Gantt correspondiente utilizando **OpenProj**, una herramienta *open source* de administración de proyectos. Este diagrama refleja la programación del TFC con una visión conjunta de las tareas y la sucesión de éstas en el desarrollo del trabajo. Se destacan como hitos las fechas de finalización de cada una de las fases mencionadas:

Figura 3: Diagrama de Gantt

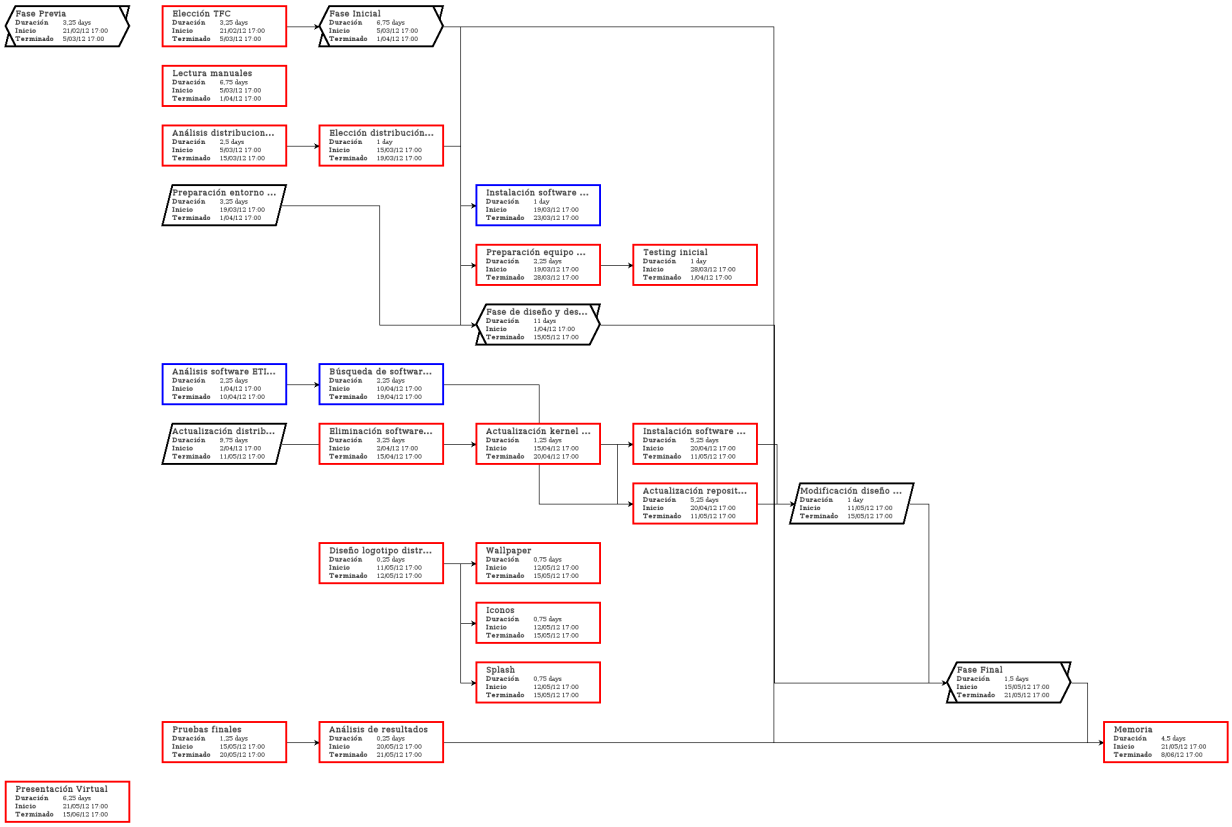


Viendo el anterior diagrama podemos comprobar que los requisitos técnicos nos obligan a una metodología en cascada salvo excepciones. El único elemento horizontal a destacar es el *testing* continuo que se realiza en las diferentes fases así como la recopilación de documentación durante todo el desarrollo del TFC.

2.6 Diagrama de PERT

Una vez creado el diagrama de Gantt indicando las tareas, su duración y la ruta crítica, también podemos generar con la herramienta OpenProj el diagrama PERT, donde se muestra la interdependencia de las diferentes tareas que se llevarán a cabo:

Figura 4: Diagrama de PERT



3 Distribuciones GNU/Linux

Una vez consolidada la combinación entre las herramientas del proyecto GNU y el núcleo Linux dando lugar al primer sistema operativo libre a principio de los noventa empezaron a surgir proyectos derivados. La finalidad de éstos se centraba en la publicación de este sistema en forma de *distribuciones* y cada una de ellas consistía en una variante del sistema incorporando un conjunto específico de herramientas, bibliotecas, versión del núcleo, administrador de ventanas, entorno de escritorio y gestor

de paquetes destinados a un tipo de uso específico.

A día de hoy el número de distribuciones GNU/Linux ha aumentado de forma considerable, dando lugar a una oferta libre de gran variedad.

3.1 Distribuciones de escritorio

Las distribuciones cuyo uso está orientado a ser de carácter general se denominan *distribuciones de escritorio*. Se caracterizan en que el software que incorporan cubre las necesidades básicas de cualquier tipo de usuario: escritorio gráfico, herramientas para administración del sistema, conectividad, ofimática básica, herramientas básicas para la utilización del hardware del equipo y núcleos compilados para ofrecer la mayor compatibilidad posible.

En la década de los noventa existieron contadas distribuciones de escritorio y éstas requerían conocimientos avanzados de sistemas unix/linux para su utilización. A partir de finales de los noventa, dada la evolución y auge de la comunidad, el número de distribuciones se multiplicó y ahora muchas de éstas no requieren de los conocimientos de antaño para su uso.

De todas las distribuciones existentes en la actualidad nombramos algunas de las más importantes que marcan tendencia:

Figura 5: Principales distribuciones de escritorio

fedora	Distribución patrocinada por Red Hat y soportada por la comunidad. Fácil de instalar y buena calidad. Sigue un ciclo de desarrollo rápido y continúa destinada al gran público
debian	Distribución con muy buena calidad. Algunas de las ventajas son: numero gigantesco de paquetes, programa de fácil instalación de software (APT), distribución completamente *open source*, centrada en proporcionar estabilidad antes que últimos avances.
ubuntu	Distribución basada en Debian y centrada en el usuario final y facilidad de uso, compatible con los paquetes Debian, instalador de gran simpleza, nuevas versiones cada 6 u 8 meses, muy popular y con mucho soporte en la comunidad. El entorno de escritorio por defecto es GNOME.
	Distribución basada en Ubuntu con fuerte énfasis en la usabilidad. Incorpora un pequeño paquete de software propio y el escritorio Cinnamon como principales distintivos frente a Ubuntu.
openSUSE	Adquirida por Novell, es la versión libre de la distribución SuSE. Sencillez de administración, instalación y uso gracias a la herramienta YaST. Utiliza el sistema de paquetes de Red Hat RPM. Amplia comunidad de soporte.

Mandriva	Anteriormente conocida como Mandrake, es una distribución derivada de Red Hat. Su proceso de instalación es considerado un modelo a seguir y su utilización resulta de las más sencillas. Actualmente Mandriva se encuentra en un proceso de reorientación por parte de sus desarrolladores.
gentoo linux	Considerada una distribución fuente, el proceso de instalación es largo y puede resultar complicado ya que se encarga de compilar las fuentes a partir de las indicaciones recibidas durante la instalación por parte del usuario. El resultado es aumentar las prestaciones de los programas.
slackware linux	Esta distribución es una de las más antiguas. Su instalador se reduce a la mínima expresión y casi todas las configuraciones deben establecerse a mano y sin asistente. No cuenta con un gestor de paquetes.
archlinux	Distribución simple y ligera, enfocada a un diseño elegante y minimalista. No posee herramientas de configuración automáticas. Los paquetes son gestionados por la aplicación PACMAN. Es un sistema en constante evolución. Su gestor de paquetes permite a sus usuarios mantener el sistema actualizado.

3.2 Distribuciones basadas en Ubuntu

Muchas de las distribuciones existentes en la actualidad surgen a partir de otras anteriores, heredando algunas de sus características y mejorando otras en pos de cumplir una finalidad concreta. Como resultado, actualmente podemos encontrar una gran cantidad de opciones que pueden cubrir muchas de las necesidades de un perfil concreto de usuario.

Ubuntu se ha convertido en poco tiempo en un germen para la aparición de nuevas distribuciones debido en parte a las características ya comentadas. Algunas de ellas tienen una orientación específica y otras siguen la finalidad de distribución de escritorio. A continuación se enumeran las variantes de Ubuntu reconocidas por la Fundación Ubuntu, las cuales poseen lanzamientos simultáneos con la distribución base:

- Kubuntu — Utiliza el escritorio KDE en lugar de Gnome.
- Edubuntu — Distribución orientada al uso en escuelas.
- Xubuntu — Utiliza el escritorio ligero Xfce.
- Lubuntu — Utiliza el escritorio ligero LXDE.
- Ubuntu Studio — Distribución orientada a la edición multimedia.
- Mythbuntu — Incorpora el software MythTV y el escritorio Xfce.
- Ubuntu JeOS — Distribución orientada a máquinas virtuales.
- Ubuntu Netbook Remix — Distribución orientada a netbooks.

Además de las denominadas oficiales por la Fundación Ubuntu existen otras distribuciones. En este gráfico podemos ver la mayoría de distribuciones originadas a partir de ella, tanto oficiales como no oficiales. Como podemos apreciar, desde la aparición en 2004 de Ubuntu han surgido de forma constante nuevas distribuciones basadas en ésta. Algunas de ellas han pasado a tener un desarrollo descontinuado pero la gran mayoría aún sigue en activo a comienzos de 2012.

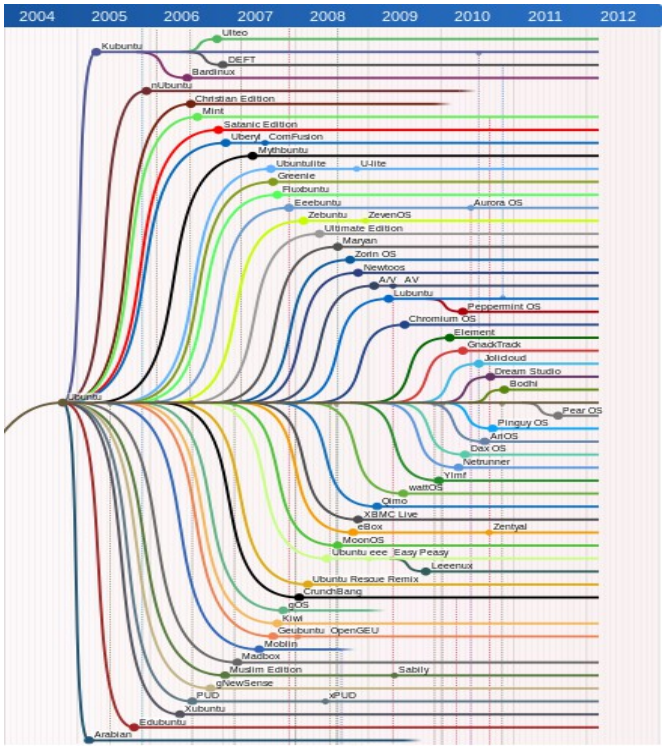


Figura 6: Distribuciones basadas en Ubuntu
(Fuente: wikipedia.org)

3.3 Distribuciones live

Las distribuciones *live* se diferencian del resto de distribuciones en que el sistema se carga por completo en memoria a partir de un medio extraíble. Tras la carga del núcleo y la carga en memoria del sistema de archivos inicial, se descomprime el sistema de archivos raíz y se monta por completo en memoria. De esta forma cualquier cambio realizado no afecta al equipo, a no ser que así se indique de forma expresa por parte del usuario.

La utilización de las distribuciones live no se popularizó hasta principios de la década del 2000 con la aparición de Knoppix, una de las primeras distribuciones en ejecutarse completamente desde un medio extraíble soportando una interfaz gráfica con escritorio. Knoppix esta orientada a la gestión de redes, para lo cual incorpora una amplia gama de herramientas de utilidad para administradores. De esta forma aparecía el concepto de distribución live, lo cual abría el camino para la creación de distribuciones de este tipo tanto para versiones de uso específico como versiones de escritorio.

De las distribuciones de escritorio mencionadas antes, todas disponen de su respectiva versión live. Esto proporciona a los usuarios la posibilidad de probar el sistema antes de instalarlo para comprobar así: la compatibilidad con el hardware, el

software incluido o el funcionamiento en general. Por otro lado, la utilización de un sistema en versión live permite la realización de tareas específicas aprovechando la ejecución completa en memoria. Según las distintas tareas surgen diferentes distribuciones live con objetivos específicos: educación, rescate, clustering, seguridad, entretenimiento, juego, medicina, diagnóstico, firewall, forense, servidor. La lista de distribuciones para fines específicos es aún mayor que la lista anterior y también se va incrementando cada año.

3.4 ¿Por qué Ubuntu?

La elección de Ubuntu como distribución base para el proyecto se debe en parte a las ventajas comentadas anteriormente y también a las mencionadas a continuación:

- Amplia variedad de manuales, how to´s y foros oficiales de soporte: askubuntu, ubuntuforums, ubuntu-es, forosubuntu y muchos más según la región seleccionada.
- Publicación de nuevas versiones con las últimas actualizaciones disponibles en un periodo de 6 u 8 meses.
- Soporte de Canonical para cada versión.
- Soporte extendido para las versiones LTS (Long Term Support).
- Madurez con que cuenta tras casi una década en desarrollo.
- Facilidad de uso para usuarios noveles.
- Amplios repositorios con software actualizado.

3.5 Proyectos relacionados

Además de las distribuciones basadas en Ubuntu analizadas, existen a día de hoy proyectos consolidados cuya finalidad es el ámbito universitario. Estas distribuciones son las más cercanas al presente proyecto:

Figura 7: Proyectos relacionados

Proyecto LULA Linux de Universidades LatinoAmericanas	Distribución académica basada en Ubuntu diseñada para cubrir las necesidades específicas de la docencia práctica en el ámbito de las universidades latinoamericanas .
CAE Linux open-source powered engineering	Distribución basada en Ubuntu centrada en software de ingeniería y ciencias. Está orientada tanto a estudiantes como a profesionales.

4 Software específico de UOCbuntu

4.1 Análisis de requisitos

El análisis de requisitos constituye la etapa de recogida de la información necesaria para cumplir las necesidades de un grupo específico de usuarios. En este trabajo estas necesidades vienen definidas por el software requerido por estudiantes de ingenierías informáticas en cada una de las asignaturas.

En el presente proyecto esta etapa ha supuesto un recorrido por todas las asignaturas de ETIS cursadas en la Universitat Oberta de Catalunya. Como alumno, esta tarea ha sido de menor dificultad con la que normalmente se encuentran otros proyectos ya que desempeñé el papel de director y cliente de forma simultánea.

El resultado de este análisis dio lugar a una lista de software en la que se encontraban herramientas libres y propietarias. Para estas últimas es posible utilizar opciones libres pero en algunos casos no existen alternativas. Esto ha sido así para herramientas como:

- SiMR: Simulador de la Máquina Rudimentaria
- Modellus: Software de simulación de modelos matemáticos

Y para el resto de software propietario utilizado, aunque existen alternativas, en algunas ocasiones no cuentan con el soporte necesario dentro del aula.

4.2 Software incluido

Tras el análisis de requisitos anterior se realizó una búsqueda de software relacionado y de calidad suficiente para cumplir con los objetivos. Principalmente esta búsqueda se realizó en:

- Repositorios de Ubuntu, donde existe una gran cantidad de software disponible.
- Centro de fuentes de software de licencia libre basado en web, como por ejemplo SourceForge.net.
- Otras distribuciones de uso específico como Knoppix o Backtrack.
- Foros de consulta y el buscador Google en última instancia.

Como resultado se ha obtenido una amplia lista con una gran cantidad de opciones para software de un mismo tipo por lo que se tomó la decisión de incorporar las mejores alternativas y dejar al usuario la elección de la herramienta a utilizar. Todo este software se muestra en la siguiente tabla:

Figura 8: Software específico de UOCbuntu

Tecnología de Computadores	
- TKGate	Software de diseño y simulación de circuitos digitales
- Logisim	Herramienta para simulación de circuitos digitales
- xSPIM	Emulador de procesadores MIPS R2000/R3000
- GNUSim	Simulador, ensamblador y depurador para el microprocesador intel8085
- MCU 8051 IDE	Entorno de desarrollo integrado para microcontroladores de la familia intel 8051
- NASM	Ensamblador para la plataforma x86
Sistemas Operativos	
- CSCope	Análisis del código fuente de linux
- VIM	Versión mejorada del editor de texto vi
- GVIM	Versión gráfica del editor de texto vim
- VirtualBox	Software de virtualización de arquitecturas x86/amd64
- QEmu	Emulador y virtualizador genérico de CPU
- Qemulator	Administrador gráfico de máquinas virtuales qemu
Ciencias	
- Maxima	Sistema de álgebra computacional especializado en operaciones simbólicas
- wxMaxima	Interfaz gráfica para el software Maxima
- FreeMat	Entorno de cálculo numérico para tareas computacionales y programación
-GNU Octave	Sistema de álgebra computacional orientado al análisis numérico con un lenguaje propio
- QtOctave	Entorno gráfico para GNU Octave
- Scilab	Software matemático con un lenguaje de programación de alto nivel orientado al uso de matrices y vectores
- GNUPlot	Utilidad en línea de comandos para la construcción de diagramas
- GTKWave	Analizador de señales digitales y analógicas a partir de archivos estándar
- RKWard	Interfaz gráfica para el lenguaje de programación estadístico R
- GNU PSPP	Software para análisis estadístico de datos
Programación	
- Netbeans	Entorno de desarrollo integrado multilenguaje
- Eclipse	Entorno de desarrollo integrado multilenguaje
- Monoevelop	IDE para C# y lenguajes .NET
- KompoZer	Editor web
- Bluefish	Editor HTML GTK+ para diseño web
- Cssed	Creación y edición de hojas de estilo
- IDLE	IDE para python

- Code::Blocks	IDE configurable y extensible
- Geany	IDE rápido y ligero basado en GTK
Bases de Datos	
- PostgreSQL	Sistema de gestión de base de datos relacional orientada a objetos
- PgAdminIII	Herramienta gráfica para gestión, desarrollo y administración de PostgreSQL
- SquirreL	Cliente SQL gráfico para administración de bases de datos
- SQLite3	Sistema de gestión de bases de datos relacionales
- Sqliteman	Herramienta para la administración y desarrollo de bases de datos Sqlite3
- MySQL	Sistema de gestión de bases de datos relacionales multiusuario y multihilo
- PhpMyAdmin	Herramienta web de gestión de bases de datos MySQL
- PhpPgAdmin	Herramienta web de gestión de bases de datos PostgreSQL
Redes	
- Apache2	Servidor web
- Dynamips	Emulador de routers Cisco
- GIP	Calculadora de direcciones IP
- GNS3	Simulador gráfico de red
- Lighttpd	Servidor web ligero
- Quagga	Software de gestión de protocolos de enrutamiento
- Wireshark	Analizador de protocolos de red
- Chromium	Navegador web de la compañía Google
Seguridad	
- dSniff	Conjunto de herramientas para análisis y auditorias de seguridad de la red
- GADMIN-OpenVPN	Herramienta gráfica para OpenVPN
- Hping3	Herramienta de envío de paquetes IP/TCP arbitrarios
- Iptunnel	Herramienta para la construcción y administración de túneles IP
- Kleopatra	Herramienta gráfica para la administración de certificados X.509 y claves pgp
- Nmap	Herramienta de seguridad y escaneo de puertos
- OpenSSH	Cliente/Servidor para conexiones shell seguras
- OpenSSL	Herramienta para la utilización de las librerías criptográficas ssl
- OpenVAS	Sistema de identificación de vulnerabilidades
- OpenVPN	Aplicación para creación de redes privadas virtuales
- Snort	Sistema de detección de intrusos en red
- Zenmap	Interfaz gráfica para la herramienta nmap
Gráficos	
- BOUML	Herramienta Case de diagramas UML
- Dia	Creación de diferentes tipos de diagramas
- Gimp	Programa de edición de imágenes digitales
- Inkscape	Editor de gráficos vectoriales
- Umbrello	Modelador de diagramas UML
- XCircuit	Editor de diagramas de circuitos y otros tipos de diagramas

Oficina	
- openProj	Software de administración de proyectos
- PDF Editor	Editor de archivos PDF
- Planner	Gestión de proyectos
- Lyx	Edición de textos científicos utilizando Latex
Sistema	
- Administrador de opciones de Compiz	Administración del escritorio Compiz Fusion
- GParted	Edición de particiones de Gnome
- OpenJDK7	Entorno de ejecución java
- Rar	Gestión de archivos comprimidos .rar
- Synaptic	Herramienta gráfica para la gestión de paquetes
- UNetbootin	Herramienta para la creación de unidades live USB
- Wine	Permite la ejecución de programas diseñados para MSDOS y versiones de Microsoft Windows
- Gnome-Classic	Escritorio clásico de Gnome
- Flashplugin-installer	Última versión flash disponible
- Manpages-es	Versión en castellano de las páginas del manual de linux
Vídeo	
- Avidemux	Herramienta para la edición de vídeo
- RecordMyDesktop	Grabación y captura del escritorio activo
- WinFF	Interfaz gráfica para el conversor FFmpeg

4.3 Software descartado

Al comienzo del proyecto se pensó en incluir el mayor número de herramientas posibles de aquellas utilizadas durante los estudios, ya fuesen de licencia propietaria como software diseñado para otros sistemas de pago. Por esta razón se incluye el emulador de aplicaciones de windows wine. Sin embargo, las funcionalidades de estas herramientas emuladas con wine no son comparables con la opción nativa, así que se tomó la decisión de descartarlas. Por otro lado conseguimos como resultado una distribución completa basada en opciones libres, aunque también se incluye wine en caso de que algún alumno desee ejecutar otro tipo de software.

Aparte de SiMR y Modellus, ya mencionados antes, se han descartado otras herramientas por estas razones. En el siguiente cuadro podemos ver sus características:

Figura 9: Software descartado

Herramienta	Categoría	Licencia	Plataforma
Digital Works	Diagramas circuitos	Propietaria	Windows
Dev-C++	IDE	GNU	Windows
Magic Draw	Herramienta Case	Propietaria	Multiplataforma
Wiris	Cálculo simbólico	Propietaria	Windows, online
Packet Tracer	Simulación redes	Propietaria	Windows, Linux
Nessus	Diagnóstico seguridad	Propietaria	Multiplataforma
Minitab	Estadística	Propietaria	Windows
Microsoft Project	Proyectos	Propietaria	Windows
SiMR	Computador pedagógico	Propietaria	Windows
Modellus	Modelos matemáticos	Propietaria	Windows

5 Entorno de trabajo

5.1 Entorno de trabajo seguro

El proyecto se ha desarrollado sobre un entorno de trabajo seguro. Por ello el desarrollo se realiza en todo momento sobre máquinas virtuales ya que las operaciones relacionadas con los cambio de sistemas de archivos raíz pueden desestabilizar el sistema principal en caso de producirse algún error durante su ejecución.

La distribución host corresponde a la máquina física sobre la que ejecutaremos las diferentes máquinas virtuales para las pruebas de configuración y *testing*. Se utiliza la versión 12.1 de openSUSE ya que es la distribución con la que habitualmente trabajo. La facilidad de administración de openSUSE mediante la herramienta YaST permite una instalación sencilla del software de virtualización.

5.2 Software de virtualización

Las dos opciones principales en la elección de software de virtualización *open source* son: **QEMU** y la versión libre de VirtualBox, **VirtualBox OSE.**

QEMU es un emulador pero también sirve como virtualizador utilizando el módulo KVM para el núcleo Linux. Cuando se utiliza para tareas de virtualización consigue resultados cercanos a los nativos mediante la ejecución de código directamente en la CPU de la máquina host. QEMU puede virtualizar máquinas guest de tipo x86, servidores, embedded PowerPC y S390. No incluye un entorno gráfico para el usuario pero existen para ello proyectos como Qemulator, QEMU Manager o QEMU Launcher.

VirtualBox OSE puede virtualizar sistemas basados en arquitecturas x86 y AMD64/Intel64 en entornos empresariales y domésticos. VirtualBox es desarrollado de forma activa con nuevas versiones que aparecen de forma frecuente y una lista de características que crece constantemente. Incorpora un entorno gráfico desarrollado por la misma compañía, Oracle.

El desarrollo de UOCbuntu se realiza principalmente con VirtualBox OSE, aunque QEMU también se utiliza en otros aspectos del proyecto y relacionadas con la virtualización de software de networking. Esta elección se hace en base a la experiencia con este software tras su uso en la asignatura *Seguridad en Redes de Computadores* durante el semestre pasado.

5.3 Configuración de VirtualBox OSE

El software de virtualización permite compartir los recursos de la máquina anfitrión para la ejecución de las máquinas virtuales que utilizaremos. Esto supone un mayor consumo de recursos, por lo que es crucial una asignación coherente para evitar desestabilizar el sistema y mantener una fluidez en la ejecución del desarrollo. Por tanto, la configuración elegida para VirtualBox y las máquinas virtuales que crearemos está condicionada por las características del equipo utilizado.

Para preparar el entorno seguimos los siguientes pasos:

a) Creamos una máquina virtual nueva, le damos un nombre, seleccionamos Ubuntu como sistema y le asignamos una cantidad de memoria equivalente a la mitad de la ofrecida por el equipo. Elegimos disco dinámico de virtual box (VDI) reservado dinámicamente y de tamaño 10GB para no tener problemas de espacio.

b) Ahora pasaremos a la configuración avanzada navegando por las pestañas de la configuración.

   i. Sistema: Procesador = 2 CPUs
      Características extendidas = Habilitar PAE/NX
   ii. Pantalla: Memoria de vídeo = 128 MB
      Funcionalidades extendidas = Habilitar aceleración 3D
   iii. Almacenamiento: Se agrega un segundo disco para mayor flexibilidad

iv. USB: Agregar filtro desde dispositivo
v. Carpetas compartidas: Agregamos la carpeta del host a compartir

c) Descargamos la iso de Ubuntu 12.04 LTS desde www.ubuntu.com/download/desktop. En almacenamiento seleccionamos la unidad de CD-DVD e indicamos la ruta de dicha imagen.

d) Por último le damos a Iniciar y seleccionamos instalar Ubuntu. Seguimos los pasos y una vez terminada la instalación reiniciamos. Una vez iniciado el escritorio instalamos las *guest additions* para poder utilizar las funcionalidades avanzadas. En el caso de openSUSE fue necesario instalar los módulos *devel* del núcleo para poder utilizar esta característica.

En este punto ya disponemos de un entorno virtualizado y completamente seguro para el desarrollo de la distribución UOCbuntu.

6 Implementación

6.1 Preparación

Tal y como se ha comentado UOCbuntu toma como base la distribución liveCD Ubuntu 12.04 LTS. Para realizar las modificaciones a la imagen iso descargada son necesarios unos pasos previos una vez nos encontramos dentro de la máquina virtual. A continuación se explican en detalle:

En primer lugar instalamos las herramientas necesarias para poder trabajar con el sistema de archivos comprimido que incorporan las distribuciones liveCD. Este tipo de sistemas, denominado **squashfs**, contiene todos los inodos, directorios y datos. Para ello instalamos la herramienta squashfs-tools desde la consola utilizando la herramienta **apt**:

$ sudo apt-get install squashfs-tools

Una vez instalada cargamos los módulos correspondientes a esta herramienta en el núcleo:

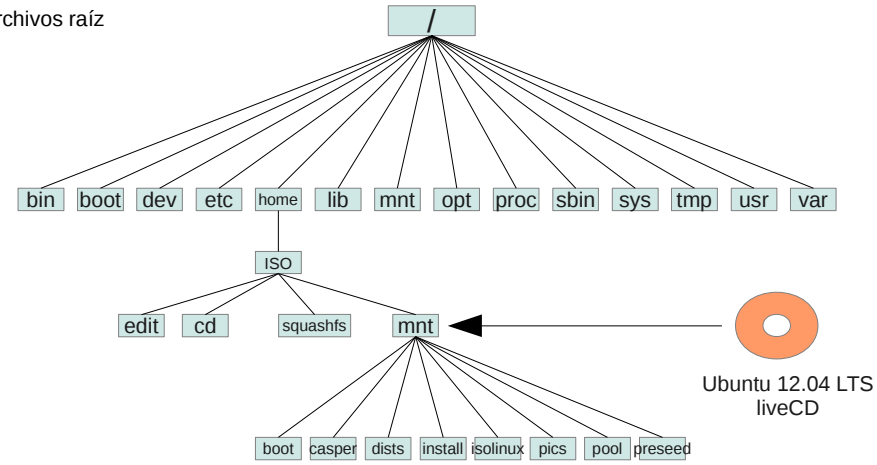
$ sudo modprobe squashfs

Ahora ya podemos extraer el sistema de archivos incluido en el liveCD. Antes, montamos la imagen iso para trabajar con los archivos que contiene. Este proceso

consiste en elegir un directorio en nuestro sistema de archivos donde colgarán los directorios y ficheros contenidos en la imagen iso. En este caso se trabaja en el directorio *home*, donde se crea otro denominado *TFC* y del que colgarán aquellos que crearemos para trabajar con cada sistema de archivos. Para la extracción de la iso se crea otro directorio denominado *mnt* y se ejecuta la siguiente orden:

$ sudo mount -o loop ../ubuntu-12.04-desktop-i386.iso mnt

Figura 10: Montaje en el sistema de archivos raíz



En la anterior orden el flag *-o loop* indica que el montaje se ha de realizar dentro del mismo sistema de archivos raíz.

Una vez hemos extraído los archivos de la iso tenemos que realizar la copia exacta en otro directorio ya que en este momento el sistema es de sólo lectura y nuestro objetivo es realizar modificaciones. Para esta tarea utilizamos la herramienta **rsync**:

$ mkdircd
$ sudo rsync --exclude=/casper/filesystem.squashfs -a mnt/ cd

Como resultado creamos un directorio denominado *cd*, en el cual copiamos el contenido exacto de los archivos montados en el directorio *mnt* (opción *-a mnt/*) a excepción del sistema de archivos comprimido squashfs (opción *--exclude*), que extraeremos en un directorio aparte:

$ mkdir squashfs
$ sudo mount -t squashfs -o loop mnt/casper/filesystem.squashfs squashfs

En este caso se indica a la herramienta mount que se trata de un sistema de archivos squashfs mediante la opción *-t squashfs* y se vuelve a utilizar la opción *loop* para trabajar desde el mismo sistemas de archivos raíz. Sin embargo, el contenido de squashfs es de sólo lectura, por lo que copiamos su contenido a otro directorio para poder realizar modificaciones:

$ mkdir edit
$ sudo cp -a squashfs/ edit/*

La opción *-a* indica que se copien todos los archivos del directorio *squashfs.*

6.2 La *jaula* chroot

"chroot en los sistemas operativos derivados de Unix, es una operación que invoca un proceso, cambiando para este y sus hijos el directorio raíz del sistema. "chroot" puede referirse a la llamada de sistema chroot(2) o al programa ejecutable chroot(8). Comúnmente, el entorno virtual creado por chroot a partir de la nueva raíz del sistema se conoce como "jaula chroot"."

(Fuente: wikipedia.org)

Utilizaremos la herramienta **chroot** (change root) para cambiar el directorio raíz al nuevo árbol de directorios que hemos extraído de squashfs y copiado en el directorio *edit/*. De esta forma podremos trabajar en un entorno seguro e invocar procesos pertenecientes al sistema de archivos de la distribución que queremos modificar. Decimos que es un entorno seguro porque ninguno de los procesos ni sus hijos podrán acceder a ficheros que se encuentren por encima del nuevo directorio raíz.

Antes de invocar a chroot necesitamos ejecutar algunas órdenes previas para tener completa funcionalidad dentro de la jaula. En primer lugar copiamos archivos de configuración necesarios para utilizar la red. Éstos son los datos relativos a los servidores DNS y los datos de configuración del mismo host local:

$ sudo cp /etc/resolv.conf edit/etc/
$ sudo cp /etc/hosts edit/etc/

También es necesario montar en el nuevo sistema de archivos el directorio */dev* del árbol primario. Éste directorio contiene los descriptores del hardware y por tanto, son indispensables. Utilizaremos la herramienta **mount** con la opción *–bind*, la cual nos permite montar parte de la jerarquía del árbol primario en el directorio */edit*:

$ sudo mount --bind /dev/ edit/dev

En este punto ya podemos realizar la llamada a chroot para realizar el cambio de raíz al directorio *edit/*:

$ sudo chroot edit

Una vez dentro de la jaula es necesario montar ciertos directorios utilizados por el núcleo para el funcionamiento del sistema:

 # mount -t proc none /proc

El directorio */proc* contiene una jerarquía de archivos especiales que representan el estado actual del kernel.

 # mount -t sysfs none /sys

El directorio */sys* contiene las configuraciones de los periféricos. Podemos encontrar un archivo por cada periférico.

 # mount -t devpts none /dev/pts

Dentro de */dev/pts* se crearán los ficheros relativos a las pseudoterminales creadas. Éstos son utilizados por el núcleo.

Configuramos las variables locales:

 # export HOME=/root
 # export LC_ALL=C

Además, para la ejecución de otros procesos en la jaula necesitaremos crear un UUID (Universally Unique IDentfier) para el Desktop-Bus (D-Bus) que consiste en un sistema de intercomunicación entre procesos (IPC):

 # dbus-uuidgen > /var/lib/dbus/machine-id

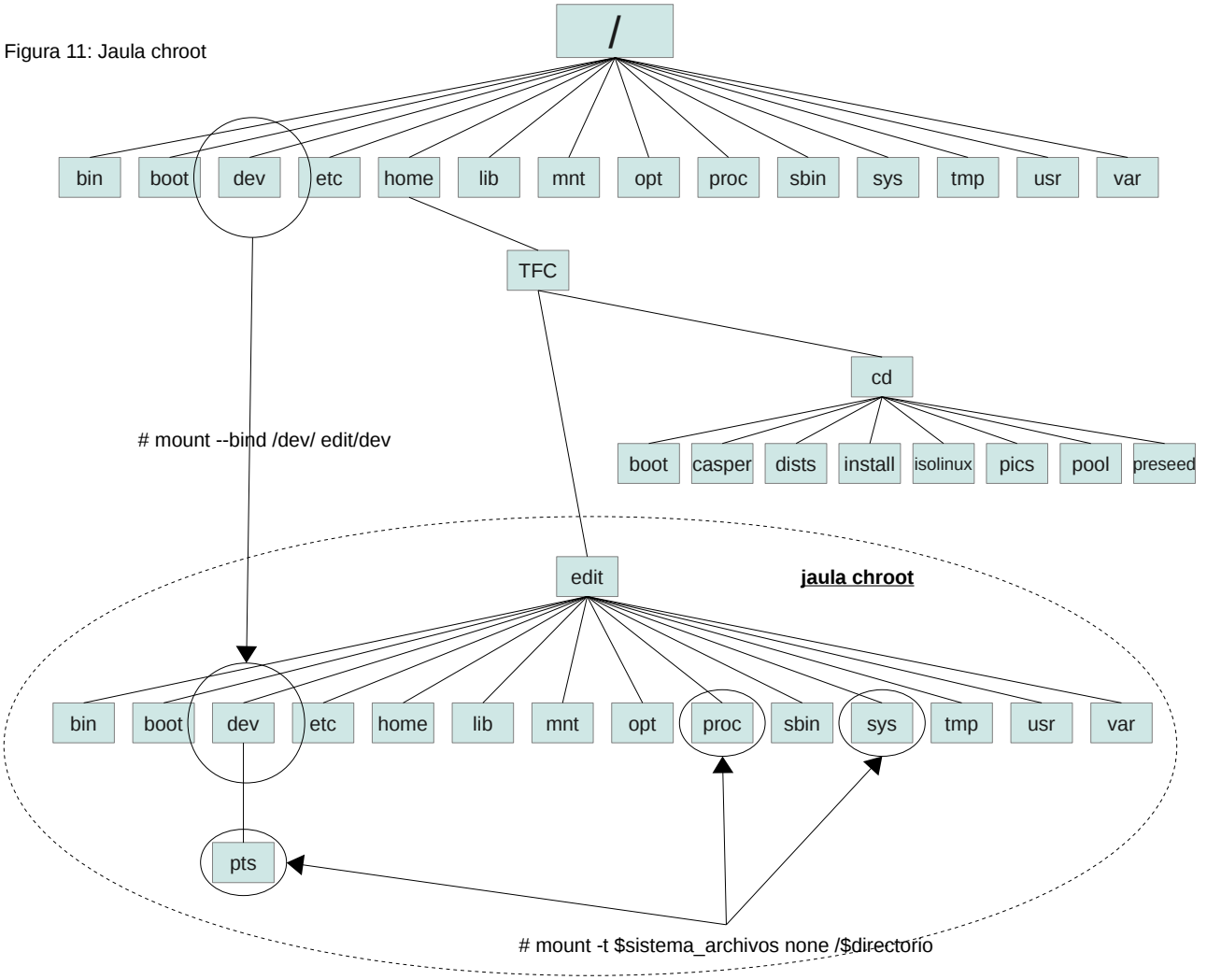
De esta forma se consigue que los procesos dentro de la jaula no compartan la zona de memoria utilizada por los procesos existentes externos a ella. Por último hemos de ejecutar:

 # dpkg-divert --local --rename --add /sbin/initctl
 # ln -s /bin/true /sbin/initctl

dpkg-divert se ejecuta para que a la hora de ejecutar la herramienta dpkg dentro del entorno chroot instale los ficheros relativos al nuevo software en una ubicación redirigida al nuevo sistema de archivos raíz. El enlace simbólico creado soluciona problemas con ciertos servicios que utilizan upstart en lugar de los clásico scripts init.

En la siguiente figura podemos ver el árbol de directorios resultante al terminar de montar el entorno chroot y la relación con los archivos y directorios del sistemas de archivos raíz:

Figura 11: Jaula chroot



Una vez dentro del nuevo entorno, antes de realizar ningún cambio es necesaria la creación de un usuario con permisos para ello. Por defecto, las distribuciones live crean un usuario denominado *live sessión user* en el momento del inicio del sistema para que el usuario pueda arrancar el entorno de escritorio y pueda realizar cambios.

Los ficheros asociados con los usuarios creados al inicio son copiados desde la ubicación */etc/skel*, que conforma el esqueleto de la configuración que se copiará sobre la carpeta */home*. Éstos consisten en configuraciones relativas a programas y archivos que el usuario podrá utilizar.

En nuestro caso, para que todos los cambios realizados dentro de la jaula sean copiados al usuario en vivo que inicie el sistema primero tendremos que crear un usuario con permisos. Ejecutamos:

 # adduser uoc

Elegimos un password y aceptamos las preguntas. Para concederle

permisos de superusuario abrimos el archivo *sudoers*:

 # visudo -f /etc/sudoers

Y añadimos bajo la siguiente línea los permisos al usuario uoc:

 # User privilege specification
uoc ALL=(ALL) ALL

Para que los cambios relativos a los usuarios sean aplicados a cualquier usuario en vivo creado en el arranque antes de salir de la jaula hemos de copiar los archivos del directorio /home en /etc/skel:

 # cp -r ~/. /etc/skel/.

6.3 Eliminación de software

Para la administración de software en Ubuntu existe la biblioteca de funciones **APT** (Advanced Packaging Tool) heredada del proyecto Debian. APT es un sistema de gestión de paquetes que permite la instalación y desinstalación de software en sistemas GNU/Linux mediante las herramientas **apt-get** y **apt-cache**. En sus inicios APT sólo soportaba paquetes .deb mediante la utilización de la herramienta de bajo nivel **dpkg** pero actualmente también puede gestionar paquetes .rpm utilizando **apt-rpm**.

Las herramientas APT funcionan en línea de comandos pero existen diferentes *front-ends* para simplificar su utilización:

- Centro de software de Ubuntu. Este es el único front-end incluido en las últimas versiones de Ubuntu. Es una aplicación desarrollada íntegramente por el proyecto Ubuntu y está orientada a usuarios noveles de GNU/Linux.
- Synaptic. Ofrece una interfaz gráfica basada en GTK+ y hasta ahora se incorporaba en las distribuciones Ubuntu. Fue suprimido en las últimas versiones. Está orientada a un usuario medio y proporciona más posibilidades de configuración.
- Adept. Interfaz gráfica basada en librerías Qt e incluido en la distribución Kubuntu.
- Aptitude. Una de las primeras interfaces gráficas que existieron para APT. Destaca por su capacidad para resolver las dependencias y eliminar paquetes inservibles del sistema. Se basa en librerías ncurses.

Desde la jaula chroot se ha intentado utilizar en la medida de lo posible herramientas en modo consola para evitar errores derivados de la ejecución de interfaces gráficas en un entorno “cerrado” y no nativo cien por cien como es el caso de chroot.

APT

Para trabajar con APT en primer lugar actualizamos la lista de software de los repositorios a su última versión mediante:

 # apt-get update

Para conocer los paquetes instalados en el sistema y visualizarlos de forma más cómoda:

 # apt-cache pkgnames | less

Para desinstalar paquetes con APT utilizamos la siguiente orden:

 # apt-get remove –purge <paquete>

La opción *–purge* indica a APT que desinstale aquellos paquetes derivados y no utilizados por otra aplicación.

Los paquetes eliminados de la distribución base se pueden ver en la siguiente tabla:

Figura 12: Paquetes desinstalados

Paquete	**Motivo**
Ubiquity	Suprimida la posibilidad de instalación
Unity	
unity-2d	Escritorio Unity
unity-2d-commons	

Synaptic

Para utilizar Synpatic desde la jaula chroot en primer lugar necesitamos concederle acceso al servidor X para poder ejecutar aplicaciones con interfaz gráfica. Para ello desde fuera de la jaula ejecutamos como superusuario la orden:

$ sudo xhost +

A continuación abrimos Synaptic desde la jaula:

 # synaptic

A partir de aquí simplemente utilizando las opciones de synaptic podemos visualizar los paquetes instalados, sus dependencias, paquetes disponibles en los repositorios y más opciones para la gestión de paquetes. Para proceder con los cambios

se pulsa sobre *Apply* y una vez terminado cerramos la ventana o pulsamos ctrl+c desde la consola en curso dentro de la jaula.

6.4 Actualización del núcleo

El núcleo de las distribuciones basadas en Ubuntu cuenta con la peculiaridad de que son núcleos específicos compilados por Canonical con una configuración y parches testados para proporcionar la estabilidad con el resto de software.

Actualmente, la última versión estable disponible del núcleo Linux es la 3.3.6, mientras que la incorporada en Ubuntu 12.04 LTS es la 3.2.0. Esto se debe a que previamente, el núcleo ha de ser adaptado para que la convivencia dentro del sistema no sea propicia a errores. Sin embargo, dentro del mismo sistema tenemos el archivo de configuración necesario para compilar cualquier núcleo, aunque sin los parches adecuados. Este tipo de kernel es el que ofrecen los desarrolladores de Canonical en paquetes .deb a través del proyecto *kernel mainline.*

Las pruebas realizadas con la instalación de estos núcleos no dieron los resultados esperados (kernel 3.3.5 mainline), así que finalmente se optó por la estabilidad del sistema a pesar de dar un paso atrás en la versión del kernel de la distribución. La elección final es la versión 3.2.0-24 disponible en los repositorios oficiales de Ubuntu, que cuenta con pequeñas, pero estables mejoras respecto de la versión 3.2.0-23 incorporada de serie.

La versión que instalamos es aquella que incluye la característica PAE (*Physical Address Extension*) para arquitecturas i386 con una cantidad de memoria RAM superior a los 4GB.

Para instalar un nuevo kernel en Ubuntu podemos utilizar APT indicando los paquetes a instalar con la herramienta apt-get.

 # apt-get install linux-headers-3.2.0-24 linux-headers-3.2.0-24-generic-pae linux-image-3.2.0-24-generic-pae

Una vez terminada la instalación podemos eliminar la versión anterior:

 # apt-get remove --purge linux-headers-3.2.0-23 linux-headers- 3.2.0-23-generic-pae linux-image-3.2.0-23-generic-pae

6.5 Instalación del software específico de UOCbuntu

Casi todo el software elegido para UOCbuntu está disponible en los repositorios incorporados por defecto en Ubuntu. Por tanto para su instalación desde el entorno chroot procedemos como antes. Podemos utilizar tanto el centro de software como Synaptic o APT.

Utilizando apt-get ejecutamos:

 # apt-get install <paquete>

Los paquetes necesarios podemos verlos desde fuera de la jaula chroot mediante Synaptic o el centro de software, o bien buscando coincidencias desde la consola al utilizar apt-cache:

 # apt-cache search <paquete>

Para el software no disponible en los repositorios, como ha sido el caso de Squirrel, logisim, openproj, GNS3-0.8.2 y qemu-1.1.0-rc2, podemos obtenerlo a partir de la misma página del proyecto o a través de SourceForge.net, una web que almacena una gran cantidad de software de licencia pública.

El software que obtenemos puede encontrarse en diferentes formatos:

Paquetes .deb

En caso de tratarse de un instalador empaquetado para distribuciones Debian podemos utilizar directamente la herramienta **dpkg** desde la terminal. Para ello nos situamos en el directorio donde se encuentra el fichero y ejecutamos:

 # dpkg –install <paquete>.deb

Al tratarse de un paquete de instalación, la misma herramienta dpkg automatiza todo el proceso y sólo necesita la confirmación del usuario.

Paquetes .rpm

En caso de obtener un paquete rpm (*Red hat Packet Manager*) podemos instalar la herramienta **apt-rpm** e instalarlo directamente o convertir el paquete a .deb mediante la aplicación **alien** e instalarlo. Los pasos son los siguientes:

 # apt-get install alien
 # alien -d -i <paquete>.rpm

La opción *-d* realiza la conversión a .deb y mediante *-i* instala el paquete.

Paquetes .jar

En caso de un instalador en lenguaje java nos encontraremos con un formato .jar. Para poder tratar estos archivos necesitamos tener instalado jdk y jre. UOCbuntu incorpora de la distribución base la versión 6 de **openjdk** aunque se ha instalado la versión 7, más actualizada. Para instalar el paquete ejecutamos desde el directorio donde se encuentra:

 # java -jar <paquete_instalador>.jar

En caso de que el software a instalar no incorpore un instalador y tan sólo existan las fuentes compiladas en java se ha optado por alojarlas en el directorio */uocbuntu* creado expresamente para este software. Para su ejecución se indica la misma orden:

 # java -jar <lanzador>.jar

Archivos fuente

En entornos GNU/Linux muchas veces se trabajan con las mismas fuentes dada su orientación libre. Así que para instalar muchos de los programas que existen para esta plataforma necesitaremos compilar el mismo código fuente. Sin embargo, el proyecto GNU desarrolló su propia versión de la herramienta **make** utilizada en unix. Dicha aplicación automatiza el proceso de creación de reglas, compilación e instalación de fuentes.

Una vez descargadas y descomprimidas las fuentes en el directorio ejecutamos desde la misma localización:

 # ./configure
 # make
 # make install

El comando *./configure* proporciona las dependencias y crea el fichero *Makefile*, este fichero contiene el conjunto de reglas, rutas y opciones necesarias para la compilación mediante el comando *make*. En último lugar procedemos con la instalación mediante **make install**.

Así, según el formato del paquete instalamos cada herramienta de aquellas seleccionadas para la distribución UOCbuntu. Como resultado, el sistema de archivos crecerá desde los 700MB de la distribución base hasta aproximadamente 2,4GB. A cambio, UOCbuntu dispondrá de todo el software necesario para los estudiantes de ingeniería informática.

6.6 Actualizaciones

En el apartado anterior se ha explicado como instalar software a partir de los repositorios incluidos por defecto en Ubuntu y almacenados en el fichero */etc/apt/sources.list*. En caso de no estar disponible hemos comentado la posibilidad de descargar los paquetes desde la web del proyecto correspondiente. Sin embargo, en ocasiones, los mismos desarrolladores ofrecen repositorios que podemos agregar para descargar e instalar los paquetes e incluso instalar actualizaciones cuando se encuentren disponibles.

Una vez conocemos el nombre del repositorio podemos agregarlo a la distribución mediante la herramienta APT:

 # add-apt-repository ppa:<repositorio>

La denominación PPA corresponde a las siglas *Personal Package Archive*, relativa al software para el que queremos obtener soporte. También se pueden añadir repositorios mediante una url utilizando la misma orden. Otra opción consiste en añadir los repositorios editando el mismo fichero */etc/sources.list*.

Dado que la mayoría del software es accesible mediante los repositorios estándar de Ubuntu 12.04, apenas ha sido necesaria la inclusión de nuevos:

- ppa:tiheum/equinox: paquete de iconos Faenza.
- ppa:libreoffice/ppa: para actualizaciones de idiomas de LibreOffice.

El software instalado en el sistema puede ser actualizado mediante los repositorios utilizando también la herramienta APT. Primero actualizamos la base de datos de éstos:

 # apt-get update

Y a continuación realizamos la actualización de aquellos paquetes para los que haya nuevas versiones disponibles mediante:

 # apt-get upgrade

En caso de existir una nueva versión de la distribución base podríamos realizar la actualización mediante:

 # apt-get dist-upgrade

El último paso en este aspecto es la actualización de initrd (*Initial Ram Disk*), necesario para la carga inicial del sistema de archivos raíz. Ha de ser actualizado en caso de instalación de algún módulo nuevo en el núcleo ya que influye en el inicio del sistema. Para ello simplemente ejecutamos:

 # update-initramfs -u

El resultado es la actualización del archivo initrd.img-3.2.0-24-generic-pae localizado en el directorio */boot.*

6.7 Salida de la jaula chroot

Antes de salir del entorno chroot son necesarios unos pasos para almacenar los cambios realizados y también para asegurar la salida del entorno de forma segura sin desestabilizar el sistema de archivos origen.

Primero copiamos el contenido de la carpeta del usuario creado (uoc) a */etc/skel* por los motivos comentados anteriormente. Eliminamos el usuario y le quitamos los permisos concedidos:

 # cp -r ~/. /etc/skel/.
 # userdel -rf uoc
 # visudo -f /etc/sudoers

Eliminamos los ficheros de los paquetes descargados con apt-get:

 # apt-get clean

Se eliminan también los archivos temporales, el historial del bash (terminal) y los archivos copiados previamente a la ejecución de chroot. También se elimina el UUID y el enlace creado al inicio.

 # rm -rf /tmp/ ~/.bash_history*
 # rm /etc/hosts
 # rm /etc/resolv.conf
 # rm /var/lib/dbus/machine-id
 # rm /sbin/initctl
 # dpkg-divert --rename --remove /sbin/initctl

Por último se procede al desmontaje de aquellas partes de la jerarquía del árbol de archivos:

 # umount /proc || umount -lf /proc
 # umount /sys
 # umount /dev/pts

Y salimos mediante la orden *exit*.

6.8 Actualización de vmlinuz e initrd.lz

Las versiones *live* de las distribuciones GNU/Linux utilizan **Syslinux** para el arranque desde medios extraíbles. Esta herramienta carga el núcleo Linux en memoria pero antes de descomprimir el sistema de archivos raíz *squashfs*, el núcleo necesita de ciertos servicios y un sistema de archivos temporal reducido con diferentes herramientas. Éste sistema, mencionado antes, se denomina *initrd* y se carga completamente en memoria. Una vez se realiza el cambio al sistema de archivos raíz se libera de la memoria y ya no se utiliza.

Ambos componentes son necesarios para el arranque del sistema y se alojan en el directorio *casper* del sistema de archivos del medio extraíble que es la ubicación utilizada por Syslinux para su ejecución. El núcleo se encuentra bajo el nombre **vmlinuz** e initrd se encuentra comprimido mediante lzma con el nombre **initrd.lz**.

En el sistema de archivos raíz, ambos ficheros se encuentran en el directorio */boot* y son utilizados en el arranque una vez el sistema se instala en disco.

Dados los cambios de versión del núcleo, ahora tenemos una versión de vmlinuz e initrd diferentes de las originales, que son las que se encuentran en el directorio *casper/*. Así que pasamos a actualizar ambas versiones simplemente reemplazando los ficheros viejos por los nuevos. Ejecutamos desde una terminal:

$ sudo cp edit/boot/vmlinuz-3.2.0-24-generic-pae cd/casper/vmlinuz
$ sudo cp edit/boot/initrd.img-3.2.0-24-generic-pae cd/casper/initrd.lz

Ahora Syslinux encontrará en la misma ubicación los ficheros actualizados bajo el mismo nombre.

7 Open networking en UOCbuntu

En el área de redes se ha incorporado a la distribución el simulador gráfico GNS3 en la última versión disponible. Se trata de un simulador de red de gran potencia que permite, gracias al motor Dynamips, ejecutar sistemas operativos de las principales marcas de networking. Para su funcionamiento necesita las imágenes de los sistemas Cisco IOS o JunOS. Por otro lado, GNS3 también incorpora la posibilidad de ejecutar imágenes con software de virtualización como Qemu y VirtualBox, capacidad que utilizaremos para poder ejecutar imágenes de sistemas GNU/Linux en GNS3.

7.1 Vyatta

Vyatta es un proyecto comercial para proporcionar soluciones open source en sistemas operativos orientados al networking. El sistema Vyatta Core está basado en la distribución Debian e incorpora software específico para tareas de networking, realizando funciones de router, firewall, VPN, prevención de instrusiones, DHCP, NAT, etc. Para ello incorpora software específico como: Quagga, SSL, OpenVPN y otras herramientas relacionadas. De forma similar a los sistemas propietarios incluye un modo estándar de administración por consola.

Algunas de las características más destacables de Vyatta son:

- ✔ Implementación de BGP en redes escalables.
- ✔ Firewall basado en inspección de estados.
- ✔ Conectividad VPN.
- ✔ Ejecución en entornos virtualizados.

7.2 Open vSwitch

Open Virtual Switch es un switch virtual multicapa bajo licencia Apache 2.0. Está diseñado para permitir la automatización de tareas en redes amplias mediante la implementación de software específico. Soporta la gestión estándar de interfaces y protocolos utilizados en la conmutación de circuitos. En la web del proyecto podemos encontrar versiones de Open vSwitch ejecutándose sobre diferentes versiones del núcleo Linux, de las que cabe destacar aquellas versiones de tamaño reducido ya que aumentan las posibilidades de este software.

Las características principales son:

- ✔ Visibilidad en comunicaciones entre máquinas virtuales mediante NetFlow, sFlow(R), SPAN, RSPAN y tunelado GRE.
- ✔ LACP (IEEE 802.1AX-2008).
- ✔ Modelo estándar 802.1Q VLAN.
- ✔ 802.1ag CCM.
- ✔ STP (IEEE 802.1D-1998).
- ✔ Granularidad en el ratio de QoS.
- ✔ Soporte para HFSC qdisc.
- ✔ Políticas para el tráfico entre máquinas virtuales.

- ✔ Balanceo de carga basado en MAC, active backup, y hashing de nivel 4.
- ✔ Soporte para el protocolo OpenFlow.
- ✔ Soporte IPv6.
- ✔ Soporte para múltiples protocolos de tunelado (Ethernet sobre GRE,
- ✔ CAPWAP, IPsec, GRE sobre Ipsec).

7.3 Linux TinyCore y Linux MicroCore

La distribución GNU/Linux **TinyCore** destaca por incorporar una versión muy reducida del núcleo Linux y una selección mínima de herramientas. Su tamaño de alrededor de 10MB la convierten en una de las distribuciones más reducidas. Incorpora una interfaz gráfica de usuario basada en el escritorio FLK. Dado su tamaño, TinyCore y los programas que incluye pueden ejecutarse completamente desde la memoria. Otra de las virtudes derivadas de dicho tamaño es la rapidez con la que puede llegar a ejecutarse, y además, incorpora todo el software necesario para proporcionar conectividad al equipo.

Otra versión pero aún más reducida de Linux TinyCore es la denominada **MicroCore,** que incluye todas las características de TinyCore a excepción del escritorio. Esta versión no dispone de sistema X windows y se ejecuta en modo línea de comandos.

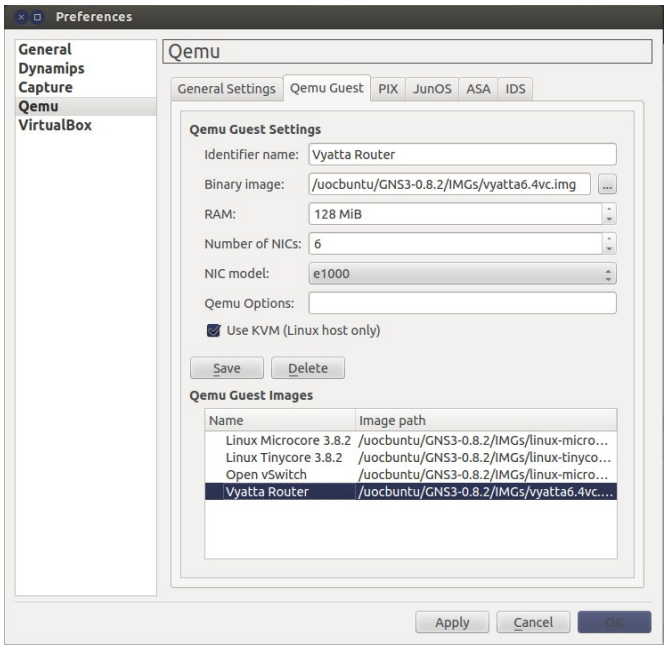
7.4 Configuración de GNS3

Conocidas las diferentes opciones *open source* en el ámbito del networking pasamos a incorporarlas a la configuración de GNS3 en UOCbuntu. Para ello se utilizan imágenes Qemu listas para su utilización y disponibles en la misma página web del proyecto GNS3. Se copian al sistema de ficheros de UOCbuntu y ejecutamos GNS3 desde el entorno chroot para ejecutar la siguiente configuración:

En primer lugar vamos al apartado de configuración siguiendo la ruta:

Edit > Preferences > Qemu > Qemu Guest

Y para cada imagen Qemu descargada creamos una máquina virtual nueva con las características apreciadas en la captura, de las que podemos destacar:



- 128 MB de memoria RAM
- Modelo NIC e1000
- 6 interfaces ethernet
- Utilización de Linux KVM (Kernel based Virtual Machine)

Figura 13: Configuración GNS3

A continuación creamos nuevos símbolos accesibles para cada imagen de Qemu creada y los asociamos al tipo de nodo que corresponde a la función de cada uno. Mediante el gestor de símbolos de GNS3 (Edit > Symbol Manager) realizamos los cambios.

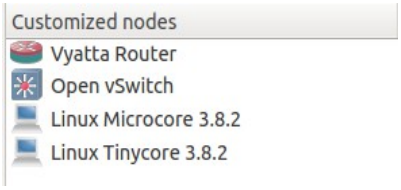


Figura 14: Nodos creados en GNS3

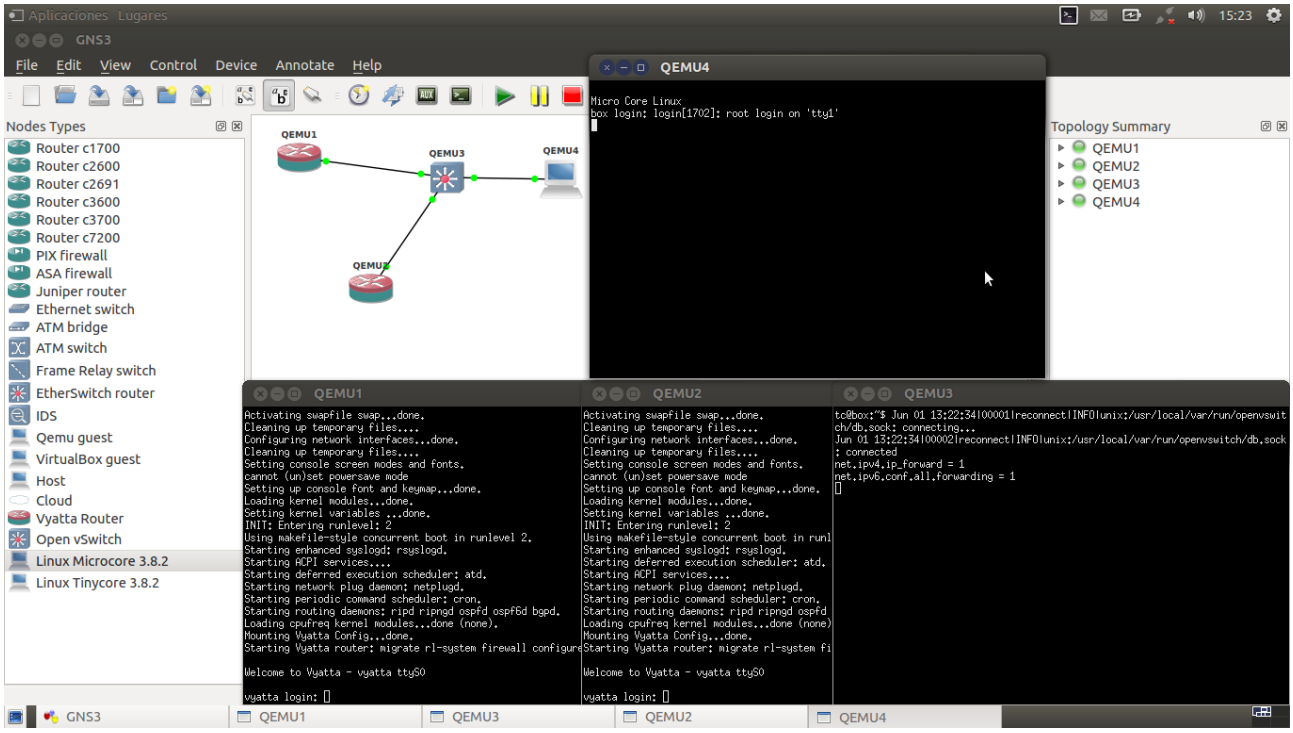
7.5 Ejemplo de utilización de GNS3

La utilización de GNS3 con los nuevos nodos no difiere del uso tradicional con Cisco IOS o JunOS. En el panel lateral correspondiente a los nodos veremos los creados para UOCbuntu, así que el usuario tan sólo ha de arrastrarlos hacia el área de trabajo. El programa le preguntará qué imagen Qemu creada quiere utilizar y éste ha de seleccionar aquella que corresponda.

Dado que la cantidad de memoria asignada para cada imagen es de 128 MB, el usuario puede crear una topología con varias imágenes a la vez sin una utilización excesiva de recursos. Por tanto, una vez creada la red añadiendo enlaces entre los nodos puede iniciar las imágenes pulsando en *Run/Iniciar* . Tras el arranque de cada imagen puede abrir las terminales asociadas pulsando en *Console*.

En la siguiente captura podemos ver una topología básica ejecutándose en UOCbuntu:

Figura 15: GNS3 en UOCbuntu



8 Modificaciones en el aspecto visual

8.1 Escritorio de UOCbuntu

Uno de los cambios más significativos de UOCbuntu en el aspecto visual es la sustitución del escritorio **Unity** de Ubuntu por **Gnome Classic**. De esta forma se pueden editar con mayor facilidad los menús y submenús desplegables, además de proporcionar una experiencia de escritorio más clásica y simple para usuarios noveles.

La instalación de Gnome Classic se realiza desde el entorno chroot. Instalamos el paquete *gnome-session-fallback*, configuramos el nuevo escritorio como sesión por defecto y por último desinstalamos Unity:

user@laptop:/# apt-get install gnome-session-fallback
user@laptop:/# /usr/lib/lightdm/lightdm-set-defaults -s gnome-session
user@laptop:/# apt-get remove –purge unity unity-2d unity-2d-commons

Con esta configuración se inicia por defecto la versión con efectos de Gnome a no ser que exista alguna incompatibilidad, en cuyo caso se iniciaría la sesión sin efectos.

8.2 Tema de iconos

Figura 16: Iconos Faenza

Para diferenciar UOCbuntu de la distribución base se ha optado por la instalación del tema de iconos **Faenza**, disponible en gnome-look.org bajo licencia GNU/GPL. De esta forma se consigue un aspecto visual renovado del paquete de iconos incluidos por defecto en los temas de Ubuntu. Como podemos ver Faenza destaca por su colorido, vanguardismo, calidad y cantidad de iconos disponibles.

La instalación de Faenza se ha realizado a partir del repositorio específico añadido a UOCbuntu.

user@laptop:/# add-apt-repository ppa:tiheum/equinox
user@laptop:/# apt-get install faenza-icon-theme

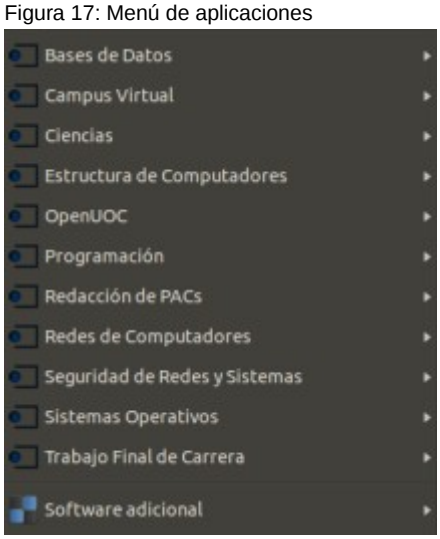
Para habilitar el uso de Faenza en el tema por defecto de Ubuntu se ha utilizado la herramienta **dconf**, una herramienta que permite la configuración de aspectos avanzado del sistema, disponible en los repositorios de Ubuntu.

8.3 Menú de aplicaciones

La herramienta de edición **Alacarte**, incluida en Ubuntu supone una de las piezas centrales en el desarrollo del presente trabajo. Mediante dicha aplicación se han conseguido editar de forma fácil y rápida los menús de aplicaciones para UOCbuntu. Así, se han creado lanzadores para las diferentes aplicaciones, tanto para aquellas que se ejecutan en modo gráfico como en línea de comandos.

La distribución de los accesos a las aplicaciones de UOCbuntu se ha organizado de forma que éstas se agrupen en base al área de conocimiento que corresponde dentro de los estudios de ingeniería informática. También se han creado menús adicionales donde el estudiante puede encontrar más herramientas útiles para su aprendizaje. Éstos se denominan:

Figura 17: Menú de aplicaciones



- *Campus Virtual*: proporciona accesos a la página de inicio del campus virtual de la UOC a través de los navegadores Firefox y Chromium. También incluye software relacionado como correo, mensajería instantánea y redes sociales.
- *OpenUOC*: incorpora un par de enlaces a los materiales en abierto de la Universitat Oberta de Catalunya a través del proyecto OpenCourseWare.
- *Redacción de PACs*: engloba el software de ofimática y aquel que puede resultar útil en esta tarea.
- El resto de software no relacionado directamente con estas necesidades concreta se agrupa en otro menú aparte denominado *Software adicional*.

Con esto se ha suprimido la organización clásica de los escritorios Gnome en pos de una nueva y personalizada estructura orientada a los estudiantes. El resultado, junto con el icono diseñado a partir del logotipo de la UOC puede apreciarse en la imagen superior.

Por otra parte, existe software instalado en la distribución que por defecto hace que se ejecuten determinados servicios en el arranque del sistema. Para dichos servicios se ha incorporado un acceso para que sea el mismo usuario quien elija cuándo iniciarlos y detenerlos. Se han creado accesos con Alacarte para se ejecute la orden desde consola mediante:

 # sh -c “service <servicio> start|stop; sleep 4

Y para que no arranquen al inicio del sistema se ha utilizado la herramienta **update-rc.d** desde el entorno chroot, que permite la modificación de aquellos servicios que inician con el sistema según el nivel de ejecución al que pertenecen. La sintaxis utilizada es:

 # update-rc.d -f <servicio> remove

En cuanto a la parada de estos servicios al cierre del equipo ya se encuentra configurado por defecto con los scripts correspondientes en los niveles de ejecución 0 (*halt*) y 6 (*reboot*).

8.4 Artwork

La distribución UOCbuntu también queda influenciada en el aspecto visual por el logotipo y el color propio de la Universitat Oberta de Catalunya. Se han utilizado las herramientas *open source* **Gimp** e **Inkscape** para el diseño, creación y edición de iconos, logotipos e imágenes. Por otro lado se ha tratado de mantener un equilibrio en las principales tonalidades utilizadas, basadas principalmente en un gris sobre el que destaca el azul oscuro propio de la UOC.

A continuación se hace una breve relación de los archivos modificados:

- Splash (*plymouth*). Se modifican los archivos:

  /lib/plymouth/themes/ubuntu-logo/plymouth.script
  /lib/plymouth/themes/ubuntu-logo/ubuntu_logo.png
  /lib/plymouth/themes/ubuntu-logo/ubuntu_logo16.png
  */lib/plymouth/themes/ubuntu-logo/progress_dot_**

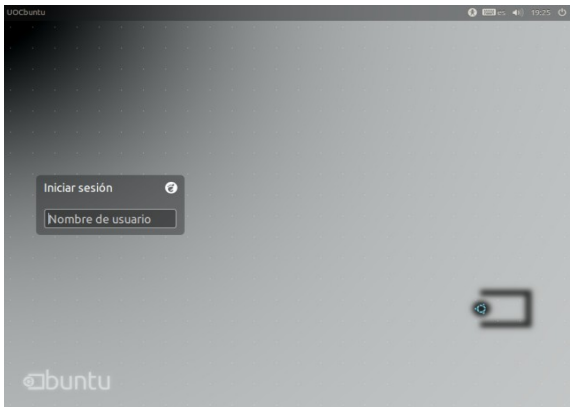
Figura 18: Splash UOCbuntu

El splash de Ubuntu para baja resolución, denominado *ubuntu-text* se ha eliminado mediante APT.

- Login (*unity-greeter*). Pantalla de login, aunque no se muestra al incio de la sesión live.

  /usr/share/unity-greeter/logo.png

Figura 19: Login UOCbuntu



- Icono de la distribución. Se muestra el icono con el logotipo de la UOC.

  /usr/share/themes/Ambiance/gtk-3.0/apps/gnome-panel.css

Figura 20: Logotipo inicio UOCbuntu

- Color UOC. Se sustituye el naranja de Ubuntu por el azul UOC (*#2d3454*)

  /usr/share/themes/Ambiance/gtk-3.0/gtk.css
  /usr/share/themes/Ambiance/gtk-3.0/settings.ini
  /usr/share/themes/Ambiance/gtk-3.0/gtkrc

- Botones de las ventanas. También sustituidos por el azul UOC.

Figura 21: Botones ventanas UOCbuntu

- Wallpapers. Diseñados con Gimp a partir del logotipo creado. Se incluyen en */usr/share/backgrounds/* y se seleccionan por defecto desde el entorno chroot.

Figura 22: Escritorio de UOCbuntu

8.5 UOCbuntu *flavour*

Como ya se ha comentado anteriormente, el fichero initrd.lz contiene un sistema de archivos temporal que el sistema carga en memoria al inicio para permitir al núcleo montar el sistema de archivos raíz. Este sistema de archivos temporal se encuentra en el directorio denominado *casper* y comprimido mediante el algoritmo lzma dentro de la imagen iso.

El trabajo realizado sobre initrd ha consistido en la extracción de los archivos, su modificación para darle un nuevo "sabor" diferente a la distribución, y su posterior compresión. Esto se realiza ejecutando desde la ruta *cd/casper/* la siguiente orden:

$ sudo lzma -dc -S .lz ../initrd.lz | cpio -imvd --no-absolute-filenames

Y modificamos los archivos:

- */etc/casper.conf*. Para editar el nombre de usuario y de host en modo live.
- */etc/default/console-setup*. Para las opciones predeterminadas de la consola.
- */etc/defautl/keyboard*. Para la localización del teclado.
- */lib/plymouth/themes/ubuntu-logo/**. Para incluir el splash creado antes.

Una vez realizados todos los cambios volvemos a comprimir el sistema de archivos mediante:

$ sudo find . | cpio --quiet --dereference -o -H newc | lzma -7 > ../initrd.lz

9 Isolinux

El sistema de arranque utilizado en las versiones live de Ubuntu es Isolinux. Este sistema es utilizado para permitir el arranque desde sistemas de archivos ISO9660 y forma parte del proyecto **Syslinux** de Peter H. Anvin.

Para mantener la línea con el resto de distribuciones basadas en Ubuntu se mantiene la configuración principal del sistema de arranque y se modifican los siguientes aspectos de los archivos existentes en el directorio *isolinux*:

- Etiqueta de modo en vivo al inicio (*label live*) en el archivo *txt.cgf* para modificar el nombre de la distribución. También se añade como fijada la resolución de pantalla a 1024x768x16 con la opción *vga=0x317.*

- Eliminado el tiempo de espera de la consola (*timeout*) en el archivo *gfxboot* que muestra la pantalla con las opciones. De esta forma se obliga al usuario a seleccionar la opción de inicio.
- Los idiomas posibles se restringen al castellano ya que los archivos de idiomas tendrían que ser modificados uno por uno. Para ello se crea un archivo denominado *langlist* con el único valor *"es"*.
- Las opciones para los teclados no se modifican.
- En el aspecto visual se modifican los archivos *.pcx* y *.png* con el color de fondo y el logotipo de UOCbuntu utilizando Gimp.

El resultado final de la pantalla de inicio es el siguiente:

Figura 23: Isolinux UOCbuntu

El resto de opciones visibles: ayuda, modos, accesibilidad y otras opciones son completamente funcionales, tal y como sucede con la distribución base. Cabe destacar la utilidad de la opción F6, donde el usuario puede editar las opciones de arranque en caso de problemas de compatibilidad con el hardware de su equipo.

Una vez el sistema inicia con la configuración de pantalla adecuada se muestra al usuario el splash durante la carga del sistema.

10 Creación de la imagen final

Una vez realizados todos los cambios sobre la imagen base de Ubuntu hemos de volver a crear la imagen con los nuevos cambios. Son necesarios los siguientes pasos:

En primer lugar damos permisos de escritura al archivo *filesystem.manifest:*

$ sudo chmod +w cd/casper/filesystem.manifest

Este archivo es el que ha de incluir todos los paquetes existentes dentro del sistema de archivos *squashfs*. Para que incorpore los paquetes añadidos ejecutamos:

$ sudo chroot edit dpkg-query -W --showformat='${Package} ${Version}\n' > cd/casper/filesystem.manifest

Se utiliza la herramienta **dpkg-query** para esta tarea desde la jaula chroot. Las opciones añadidas sirven para indicar a dpkg-query el formato a la hora de escribir los resultados en el archivo *filesystem.manifest*. Actualizamos también el archivo *filesystem.manifest-desktop:*

$ sudo cp cd/casper/filesystem.manifest cd/casper/filesystem.manifest-desktop

Filtramos el contenido del archivo con la utilización del editor de cadena **sed**:

$ sudo sed -i '/casper/d' cd/casper/filesystem.manifest-desktop

Eliminamos el antiguo sistema de archivos squashfs en caso de existir uno previo:

$ sudo rm cd/casper/filesystem.squashfs

Y por último creamos el nuevo *squashfs* a partir del sistema de archivos editado a través de chroot. Mediante la herramienta **mksquashfs** se indica el origen y el destino del nuevo sistema a crear:

$ sudo mksquashfs edit cd/casper/filesystem.squashfs

Una vez creado el sistema de archivos squashfs podemos editar el archivo *README.diskdefines* con el nombre de UOCbuntu y a continuación se calcula el nuevo MD5 para proporcionar integridad al archivo iso.

$ sudo rm md5sum.txt
$ sudo find -type f -print0 | sudo xargs -0 md5sum | grep -v isolinux/boot.cat | sudo tee md5sum.txt

En este punto ya está todo listo para la creación de la imagen final. En este caso utilizamos la herramienta **mkisofs** para crear una imagen ISO9660. Ejecutamos:

$ sudo mkisofs -D -r -V "" -cache-inodes -J -l -b isolinux/isolinux.bin -c isolinux/boot.cat -no-emul-boot -boot-load-size 4 -boot-info-table -o ../ubuntu-.iso .

11 Optimización de UOCbuntu

La cantidad de software específico incluido en UOCbuntu hace que la imagen .iso final obtenida sea del tamaño de unos 3 Gigabytes. Esto nos obliga a la utilización de soportes de mayor tamaño para su ejecución en modo live, dependiendo también de las necesidades que queramos cumplir. También hay que recordar que queda descartada la posibilidad de instalación en disco de UOCbuntu, característica eliminada con la desinstalación de Ubiquity.

11.1 Versión live-DVD

La primera opción es la creación de una versión live en soporte DVD. De forma muy sencilla se puede utilizar el grabador de discos ópticos **Brasero**. Se trata de una herramienta *open source* incluida tanto en Ubuntu como en otras distribuciones GNU/Linux. Para ello simplemente seleccionamos la ruta de la imagen .iso creada, el dispositivo óptico y la velocidad de grabación.

Como resultado se obtiene una versión live-DVD de UOCbuntu completamente funcional que permite a los usuarios beneficiarse de las ventajas de disponer de un sistema en modo live con las herramientas específicas.

11.2 Versión live-USB persistente

La segunda opción intenta mejorar aquellas características ofrecidas por la versión live-DVD. En este caso UOCbuntu se instala en una memoria USB de tamaño mayor a 8 Gigabytes para proporcionar persistencia en los cambios realizados por el usuario durante la sesión live. Aunque mediante la versión live-DVD el usuario puede almacenar sus cambios en cualquier disco duro del equipo, mediante esta opción los cambios realizados en el sistema quedan almacenados en la misma

memoria USB. Esta característica permite al usuario portar además del sistema, los cambios realizados, y poder utilizarlos en otros equipos gracias a la enorme compatibilidad que ofrece Ubuntu 12.04.

Para crear esta versión live persistente en una memoria USB existen diferentes métodos:

1. Método manual. Se utiliza la herramienta **fdisk** de administración de particiones de Ubuntu para crear dos particiones FAT, una para el sistema y otra para los cambios. Luego se extraen los archivos contenidos en la imagen .iso y se renombra el directorio y archivos con denominación isolinux a syslinux para permitir la compatibilidad con sistemas de archivos FAT.

2. Método automático. Ubuntu dispone de la herramienta **Creador de discos de arranque** que en modo gráfico permite realizar las mismas tareas anteriores de forma automatizada. La aplicación permite desde la interfaz gráfica: borrar la unidad a utilizar, indicar la ruta de la imagen .iso, seleccionar el dispositivo destino y el tamaño del dispositivo a utilizar para datos persistentes. Con estos datos la aplicación crea la versión live-USB de la imagen .iso seleccionada.

Además de la persistencia en los datos, la utilización de UOCbuntu en un soporte USB extraíble proporciona mayor velocidad de ejecución frente a la capacidad ofrecida en la ejecución desde medios ópticos.

12 Evaluación final

12.1 Evaluación de UOCbuntu

Las pruebas correspondientes a la evaluación final del proyecto se han realizado sobre la siguiente relación equipos, de los que se especifican las características de cada uno:

Figura 24: Características de los equipos de pruebas

	PC A	PC B	PC C
Modelo	Asus A53SD	Compaq 610	Compaq Presario V4000
Procesador	Intel® Core™ i7-2670QM velocidad: 2.2 Ghz núcleos: 4(8 hilos de ejecución) caché: 6 MB	Intel® Core™ 2 Duo T5870 velocidad: 2 Ghz núcleos: 2 caché: 2 MB	Intel® Pentium Centrino velocidad: 1.86 Ghz núcleos: 1 caché: 2 MB
Memoria	6GB DDR3 velocidad: 1333 Mhz	4GB DDR2 velocidad: 800 Mhz	1GB DDR2 velocidad: 667 Mhz
Tarjeta Gráfica	nVidia 610M	Mobile™ Intel® Graphics Media Accelerator X3100	Mobile Intel 915 GM/GMS

Estas pruebas han tratado la ejecución en modo live-DVD y live-USB persistente de UOCbuntu. El *testing* realizado en estos equipos ha seguido los siguientes puntos:

- ✔ Compatibilidad de hardware.
- ✔ Ejecución de aplicaciones en modo gráfico.
- ✔ Ejecución de aplicaciones en modo consola.
- ✔ Inicio y parada de servicios incorporados.
- ✔ Ejecución de una topología básica en GNS3 utilizando los nodos creados.
- ✔ Funcionamiento general del sistema en modo live.

12.1.1 Evaluación de errores

En términos generales, la ejecución de UOCbuntu sobre estos equipos no presenta graves errores, ya que el sistema inicia bien y permite una gran compatibilidad con el hardware gracias a las características mencionadas sobre Ubuntu 12.04. Sin embargo existen algunos **problemas** y **errores** que se resumen a continuación:

Figura 25: Problemas y errores detectados en UOCbuntu

Aplicación/Servicio	Problema/Error	Posible causa
- Inicio del sistema en versión live-DVD	Velocidad muy lenta	Velocidad limitada de la unidad lectora
- Splash	A veces no se muestra	Resolución pantalla. Compatibilidad hardware
- Parada del sistema (halt)	A veces requiere de apagado manual	Compatibilidad con apagado ACPI
- Nautilus	Inicio lento	Desconocida
- GNS3	Las terminales xterm son cerradas de inmediato en las pruebas sobre PCs B y C	Desconocida
	Wireshark no captura	No es posible en modo live con una configuración por defecto
	Switch ethernet de GNS3 no es compatible con los nodos creados	Desconocida

Aparte de estos errores también podemos añadir el problema que supone la utilización del usuario por defecto en versión live con permisos de *root* sin una contraseña configurada. Esto facilita algunas tareas al usuario novel pero puede desestabilizar el sistema si se realiza un mal uso de la orden **sudo**.

En cuanto al aspecto visual también podemos destacar la incompatibilidad con las librerías gtk de algunas aplicaciones, lo que hace que se ejecuten en modo gráfico básico.

12.1.2 Limitaciones

La utilización de UOCbuntu implica una serie de limitaciones, de entre las que destacamos:

- ✗ Velocidad de ejecución desde la unidad DVD lectora. Puede suponer un hándicap en caso de utilizar la versión live-DVD.
- ✗ Idioma restringido al castellano. Puede ser motivo de rechazo por estudiantes de provincias como Catalunya, País Vasco o Galicia, así como de países extranjeros.
- ✗ Resolución de pantalla fijada a 1024x768x16 en el inicio del sistema, lo cual puede presentar problemas de compatibilidad con equipos no soportados.
- ✗ Plugins para los diferentes lenguajes de programación en los IDEs. Queda como tarea del usuario la instalación de aquellos plugins a utilizar.
- ✗ Complejidad añadida en la utilización de software como Vyatta y Open vSwitch. Puede influir en la curva de aprendizaje respecto de la utilización de software propietario.
- ✗ Tamaño de la imagen de Vyatta incluida en UOCbuntu (700MB).
- ✗ Ausencia de software relativo a las asignaturas de idiomas de la UOC por quedar bajo licencia copyright.
- ✗ Ausencia de alternativas libres para software como: SiMR y Modellus.

12.2 Evaluación general del proyecto

Las diferentes fases del desarrollo de este trabajo final de carrera han supuesto la adquisición de conocimientos relativos al movimiento *open source* y al funcionamiento del sistema operativo GNU/Linux.

En la primera parte del proyecto se han conocido y probado las distribuciones GNU/Linux más importantes y se han podido comprobar algunas de las características ofrecidas. La fase relativa a la búsqueda de alternativas libres me ha

llevado a realizar un recorrido por las asignaturas cursadas, con lo que se constata que la Universitat Oberta de Catalunya hace una firme apuesta por el software libre pero no llega a abarcar todo el software utilizado. Como resultado de la búsqueda de estas alternativas se ha podido comprobar la cantidad de software existente en los repositorios de Ubuntu y en webs como SourceForge.net. Gracias al breve análisis de este software para su incorporación final he conocido la cantidad de software de calidad disponible, así como la variedad existente.

Para llevar a cabo la implementación han resultado básicos los manuales y how-to´s disponibles en foros de soporte para Ubuntu y otras distribuciones. Esta ayuda ha simplificado el desarrollo, aunque ha sido necesario leer diferentes versiones de distintos autores para una mejor comprensión y llevar a cabo la implementación de UOCbuntu con éxito.

En la fase de implementación se han producido problemas que han retrasado el desarrollo:

- Problemas en el seguimiento de algunos manuales incompletos.
- Inconsistencia del sistema base tras una mala ejecución en las órdenes para la preparación del entorno chroot. Después de esto no iniciaba.
- Problemas con la instalación de núcleos obtenidos de kernel/mainline. Se consiguió su instalación mediante *unionfs*, pero el funcionamiento de la distribución reportaba problemas derivados del kernel.
- Penalización de tiempo a la hora de realizar pruebas sobre cambios realizados, dado el tiempo invertido en tareas de compresión y pruebas.

Por otro lado, en esta fase también se han adquirido conocimientos sobre la organización del sistema de archivos, herramienta *mount*, administración de paquetes y servicios del sistema.

Por último, la organización del menú principal ha resultado de gran sencillez gracias a la herramienta *Alacarte* lo cual ha supuesto un ahorro de tiempo y la posibilidad de incorporar nuevas funcionalidades. Las modificaciones realizadas en el aspecto gráfico ha significado el aprendizaje a nivel básico de herramientas como Gimp e Inkscape, así como la edición de archivos relativos a configuraciones del escritorio Gnome.

12.3 Objetivos conseguidos

Los resultados del presente proyecto han sido bastante satisfactorios en cuanto a los objetivos planteados al comienzo. La distribución UOCbuntu es completamente funcional y las herramientas incorporadas cumplen con los requisitos planteados. Como resultado, los alumnos de las ingenierías informáticas tienen a su disposición una plataforma completamente *open source* como apoyo en sus estudios. El software incluido es amplio y autosuficiente, con la posibilidad *open networking* añadida en el ámbito de la simulación de redes.

12.4 Trabajo futuro

Aparte de la solución de los errores comprobados en la evaluación final, UOCbuntu puede ser mejorado para ofrecer nuevas y renovadas características a los estudiantes de ingeniería de la Universitat Oberta de Catalunya. Algunas de ellas se resumen a continuación:

- Mayor integración con los servicios ofrecidos por la universidad, como pueden ser el campus virtual, los materiales de las asignaturas y acceso a la base de datos.
- Escritorio activo que permita la ejecución de contenido web. De esta forma, el trabajo del estudiante giraría siempre en torno al campus virtual.
- Instalación de diferentes escritorios personalizados con el tema UOCbuntu para permitir al usuario la elección al inicio del sistema.
- Inclusión de materiales de las asignaturas en caso de existir bajo licencia pública.
- Inclusión de herramientas relativas a asignaturas de idiomas.
- Creación de un repositorio propio para dotar a los estudiantes con las últimas novedades.
- Disponibilidad del resto de idiomas al inicio.

Todas estas mejoras supondrían convertir a UOCbuntu en una auténtica distribución para la enseñanza superior en ingenierías, convirtiéndose en pionera como distribución de carácter académico. A cambio sería necesario realizar una inversión en soporte técnico independiente del ofrecido por Canonical para proporcionar a los usuarios las últimas novedades y soluciones a los problemas planteados.

15 Conclusiones

Actualmente las distribuciones GNU/Linux gozan de gran madurez y son la primera opción en servidores y equipos orientados a alta computación. Como sistema de escritorio no cuentan como primera elección para la mayoría de usuarios pero igualmente se encuentran en un gran momento dada la evolución del núcleo Linux así como el desarrollo de herramientas *open source*. La consecuencia es el auge de distribuciones de escritorio de gran calidad.

Otra consecuencia directa de la evolución de los sistemas GNU/Linux es la cantidad de distribuciones de uso específico. Podemos encontrar una gran variedad de éstas dada la evolución tecnológica y la aparición de nuevas necesidades, impulsada por la capacidad de ejecución de estos sistemas en modo *live*. A raíz de este último punto han aparecido diferentes opciones para que cualquier usuario cree su propia versión GNU/Linux a partir de otras distribuciones de forma sencilla y automatizada. Sin embargo, en el desarrollo de este proyecto se ha utilizado el entorno *chroot* como pieza fundamental, que permite modificaciones más allá de las permitidas por estas herramientas automáticas a cambio de añadir complejidad al proceso.

Otra pieza fundamental del presente proyecto es el software libre incorporado. Éste dota de sentido a la distribución y su inclusión no podría haber sido posible sin la comunidad *open source* y la cantidad de proyectos en desarrollo a día de hoy. La oferta existente se amplía cada año y la calidad ofrecida es suficiente para cubrir las necesidades de diferentes perfiles de usuarios.

Aunque en ocasiones este software queda totalmente desplazado por las opciones propietarias, tal y como hemos visto en algunas asignaturas cursadas, UOCbuntu es una muestra de que sí es posible una alternativa completamente libre.

Bibliografía

Nemeth, E.; Snyder, G.; R. Hein, T. (2008). *Administración de sistemas Linux* (2ª ed.). Anaya Multimedia. [Ed. orig.: *Linux Administration Handbook*, Person Education]

Rohaut, S. (2010). *Preparación para la certificación LPIC-1*. Eni ediciones.

Artículos:

M. Tim, J. (2006). *Linux Initial RAM disk (initrd) overview.* www.ibm.com/developerworks.

Laird, C. (2003). *Knoppix gives bootable, one-disk Linux.* www.ibm.com/developerworks.

M. Tim, J. (2006). *Inside the Linux boot process.* www.ibm.com/developerworks.

Linux Home Networking. (2012). *The Linux boot process.* http://www.linuxhomenetworking.com/wiki/index.php.

Manuales:

Community help wiki. (2012). *LiveCDCustomization.* help.ubuntu.com/community/LiveCDCustomization.

Wiki. (2011). *SYSLINUX.* www.syslinux.org/wiki/index.php/SYSLINUX.

Community help wiki. (2012). *CustomizeLiveInitrd.* wiki.ubuntu.com/CustomizeLiveInitrd.

Popescu, C. (2011). *How to make/create a live CD/DVD from your harddisk* securityfocus.eu/?p=149.

González, S. (2008). *Creando un CDROM autoejecutable con ISOLINUX.* www.sergio-gonzalez.com/metadistros/isolinux/html/.

Wiki administradores. (2007). *Creación de un entorno chroot.* administradores.educarex.es/wiki/index.php/Creación_de_un_Entorno_chroot.

Larry, M. (2010). *Edit grub´s boot menu options.* thpc.info/how/edit_grub_menu.html.

Community help wiki. (2012). *LiveUsbPendrivePersistent.* wiki.ubuntu.com/LiveUsbPendrivePersistent.

Índice de figuras