Universidad de las Ciencias Informáticas
Facultad 10

Título: Descripción del proceso de construcción del sistema operativo base de la distribución cubana de GNU/Linux Nova.

Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas

Autores:
Jorge Luis Machín Castillo
Anay Corso Bouza

Tutores:
Ing. Miguel Albalat Aguila
Ing. Anielkis Herrera González

Consultante:
Ing. Abel Alfonso Fírvida Donéstevez

Ciudad de La Habana
Mayo del 2010

Declaración de autoría

Declaramos ser autores de la presente tesis y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmamos la presente a los ____ días del mes de _____ del año _____

Jorge Luis Machín Castillo
_____
Firma del Autor

Anay Corso Bouza
_____
Firma del Autor

Miguel Albalat Águila
_____
Firma del Tutor

Anielkis Herrera González
_____
Firma del Tutor

Frase


"...lo más importante en la vida no es el triunfo,
sino la lucha. Lo esencial no es haber vencido,
sino haber luchado bien..."
Pierre de Coubertin

Agradecimientos

De Anay

A mis padres por estar siempre cuando los necesito, brindarme su amor y confiar en mí. Porque gracias a ellos hoy estoy aquí. Por ser los mejores padres del mundo. Los amo.

A mi hermana que es especial para mí, porque la quiero muchísimo y quiero ser un ejemplo para ella.

A mi abuela Edelma por quererme tanto y dedicar parte de su vida a cuidarme. Te quiero mucho.

A mi abuela Migdalia por contagiarme con su alegría.

A mis abuelos que hoy no están pero siempre esperaron lo mejor de mí.

A mi tía Yeny por preocuparse por mí y brindarme su ayuda.

A mi tía Belkis por sus bromas.

A mis tíos Ibrahim y Rafa por estar atentos por mi carrera.

A mis primas Geidy y Glenda por todo el tiempo que hemos compartido juntas. Son como mis hermanas.

A mi novio Serguey, por apoyarme siempre, ayudarme, confiar en mí, brindarme su amor y comprensión. Gracias por tu compañía, ser tan lindo conmigo y hacerme feliz.

A la familia de mi novio por quererme y dejarme formar parte de ellos.

A mi compañero de tesis, por compartir la realización de este trabajo y cinco años de amistad. Por confiar en mí y tenerme presente. Gracias por todo.

A mis compañeros de grupo todos estos años (en especial al incansable Francisco), porque juntos compartimos momentos inolvidables.

A Katia y Yilian por el tiempo que hemos pasado juntas, nuestras diversiones y conversaciones.

A Laura por sus travesuras, realmente es incansable. Por escucharme cuando lo he necesitado.

A Amalia porque aunque hoy no se encuentra en la escuela, se sigue preocupando por mí.

A Abel y Miguel por brindarnos todo su apoyo en la realización de esta tesis y hacer que el trabajo resultara agradable.

A los profes Violena, Angel, Yeni, Laya, porque nos ayudaron en la realización de la tesis.

A mis amigas Eimi y Geysha por dejarme contar con ellas en todo momento y estar pendientes del desarrollo de la tesis.

A las muchachas de mi apartamento: La China, Roxy, Yanet, Caridad, Lianet, Laura, Yilian porque juntas nos hemos divertido muchísimo.

A Yenisel y Maikel por brindarme su ayuda cuando lo he necesitado.
A Yilena, Marisol, Yisel porque también convivieron conmigo.
A los miembros del proyecto Nova por acogerme como si formara parte de él.
A Olga, Arlen, Ernesto, Yenni por brindarme su ayuda.
A todas aquellas personas que de una forma u otra me apoyaron en la realización de este trabajo.
A la UCI por permitirme conocer personas maravillosas que nunca voy a olvidar.
A la Revolución por la oportunidad.
A todas la personas que he conocido estos cinco años.
A los que han formado y forman parte de mi vida y siempre se han preocupado por mí.
A todos los que de una forma u otra aportaron un granito de arena para que este sueño se hiciera realidad.
Muchas gracias a todos, los quiero.

De Jorge Luis

Agradecer a todas las personas que han contribuido a mi formación como ingeniero y ser social. Especialmente a:
Mis padres por ser mi todo, acompañarme en todo momento, por creer siempre en mí y en mis ideas, pero sobre todo por estar.
Mi hermana por ser una de las razones de mi vida, por lo mucho que la quiero.
Mi novia Libis, por su cariño, dedicación y entrega durante todos estos años. Por soportarme como soy y brindarme su apoyo en todo momento.
Mi abuelo y Gladys por su apoyo y ayuda durante todos estos años.
Mi familia por su presencia y el apoyo que me brindan.
Jorgito, Yuniel y Ediel por tantos años de amistad, por los consejos, la ayuda y el apoyo.
Mi compañera de tesis, por compartir la realización de este trabajo y cinco años de amistad.
Mis amigos de Nova: Abel, Miguel, Anielkis, Félix, Raydel, Mijail, Dariem, Ángel, Daniel, Yunier, por contribuir con mi formación y otros que aunque se encuentran en otros lugares siempre estuvieron dispuestos a brindarme su ayuda.
Mi grupo por los gratos momentos que compartimos.
Las profesoras Violena, Laya, Yeni por su asesoramiento y ayuda en el desarrollo de esta investigación.
Abel y Miguel por todo lo que he aprendido de ellos, por tanta ayuda durante mi estancia en el proyecto

Nova, por hacer mi tesis suya.

Todas las personas que compartieron conmigo algún momento durante mi estancia en la Universidad. La Revolución por la oportunidad y la formación.

A todos muchas gracias.

Dedicatoria

De Anay

A mis padres por ser una guía en mi camino, confiar todo el tiempo en mí, apoyarme todos estos años y darme ánimo para seguir adelante.

De Jorge Luis

A mis padres por la confianza, el apoyo y la dedicación que me han brindado durante toda mi vida.
A mi abuela que aunque no esté presente hoy, siempre me acompaña.

Resumen

Con el presente trabajo de diploma titulado “Descripción del proceso de construcción del sistema operativo base de la distribución cubana de GNU/Linux Nova”, se pretende contribuir a la creación de una base común que soporte las líneas de desarrollo del proyecto Nova, a través de la descripción del proceso de construcción del sistema operativo base.

En el documento, se recogen los resultados del estudio del estado del arte, de los procedimientos que siguen algunas distribuciones para construir su sistema operativo base, las notaciones y herramientas que permiten modelar el proceso y las convenciones con las que se describe dicho proceso.

El resultado esperado será una guía capaz de permitir a los miembros del proyecto Nova construir el sistema operativo base de la distribución.

Palabras claves: Nova, sistema base, sistema operativo base.

ÍNDICE

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

Introducción

En el mundo, el uso y la creación de distribuciones de sistemas basados en tecnologías de software libre y código abierto (FOSS)[1], de tipo GNU/Linux, han ido evolucionando y ampliando sus horizontes debido a que les permiten a los usuarios ciertas libertades: estudiar, ejecutar, copiar, distribuir, cambiar y mejorar el software, teniendo como elemento significativo el acceso al código fuente (1).

Cuba por razones de seguridad, política y soberanía tecnológica, es un ejemplo de los países latinoamericanos que han apostado por los sistemas FOSS. En el año 2004 el Consejo de Ministros determinó la necesidad de migrar hacia sistemas de código abierto, para ello se creó el Grupo Ejecutivo Nacional encabezado por el Ministro de Informática y las Comunicaciones. Componen este grupo : el Grupo Legal, encargado de respaldar el proceso; el Grupo de Capacitación, cuya tarea reside en preparar a las personas para el cambio; y el Grupo Técnico, que ha elaborado una guía de migración, referencia para ordenar el proceso y señalar las aplicaciones sustitutas de aquellas en código propietario (2).

Como parte del proceso de migración, en la Universidad de las Ciencias Informáticas en el proyecto Nova, se trabaja en el desarrollo de un sistema operativo basado en GNU/Linux, denominado Nova. Este proyecto surge con la idea de garantizar la compatibilidad del software que se realizaba en la universidad con los sistemas FOSS. Como meta inicial se pretendía apoyar el proceso docente y productivo. Luego de darse a conocer la primera versión estable del sistema operativo, con motivo de la Feria y Convención Internacional “Informática Habana 2009”, varias instituciones se acercaron al proyecto con el deseo de comenzar a utilizar Nova, incrementando el alcance del mismo.

Para aumentar la capacidad ante las peticiones de las instituciones se realizó un análisis, que demostró la necesidad de reestructurar el modelo de producción que existía hasta el momento y dividir el desarrollo hacia las computadoras de escritorio, los servidores y las computadoras sin disco duro como líneas separadas. Para lograrlo se crearon tres líneas fundamentales de desarrollo en el proyecto: Sistema para Escritorios, Servidores y Clientes Ligeros. Para el correcto funcionamiento de este modelo es necesaria la construcción de un sistema operativo que sirva de base común. En la actualidad el proyecto Nova no cuenta con un proceso definido para guiar la construcción de un sistema operativo que sirva de base a todas las líneas de desarrollo.

[1] ***FOSS:*** *Terminología utilizada para nombrar el software libre y de código abierto, por free/libre and open source software, en inglés.*

Por lo antes expuesto se identificó como **problema científico:**

¿Cómo contribuir a la creación de una base común para soportar las líneas de desarrollo del proyecto Nova?

Se define como **objeto de estudio** el proceso de desarrollo en los sistemas operativos GNU/Linux, seleccionándose como **campo de acción** la distribución cubana de GNU/Linux Nova.

El **objetivo general** de esta investigación es definir el proceso de construcción de un sistema operativo que sirva de base al proyecto Nova.

Como **objetivos específicos** se plantearon los siguientes:

1. Analizar el proceso de construcción del sistema operativo base de Nova y de otros sistemas FOSS.
2. Modelar el proceso de construcción del sistema operativo base de Nova.
3. Proponer la descripción del proceso.
4. Construir un sistema operativo base para probar la propuesta de solución.

Para dar cumplimiento a los objetivos específicos se formularon las siguientes **tareas de investigación:**

1. Conversaciones informales con los desarrolladores del sistema operativo base de Nova y análisis de información y conocimiento acumulado.

2. Revisión bibliográfica acerca de la construcción del sistema operativo base de los sistemas basados en las tecnologías de software libre y código abierto.

3. Estudio de las herramientas existentes para modelar procesos.

4. Redacción de la propuesta de solución.

5. Desarrollo del sistema operativo base, con pruebas al sistema.

Para el desarrollo de esta investigación se plantea como **idea a defender**:

La definición del proceso de construcción de un sistema operativo que sirva de base al proyecto Nova, contribuirá al desarrollo de la distribución cubana de GNU/Linux Nova.

Para solucionar las tareas antes mencionadas se utilizaron los siguientes métodos:

Método Empírico:

Observación: Esta técnica permitió tener un registro visual del comportamiento de algunas herramientas que auxilian la construcción de los sistemas base. La vía de empleo de esta técnica es la **observación participante no controlada** porque el observador forma parte del problema a resolver y no existe una guía para realizar la observación.

Técnica de recogida de información:

Conversaciones Informales: Mediante esta técnica se intercambió información de forma informal con miembros del proyecto, que es necesaria tener en cuenta para construir el sistema operativo que sirva de base a las líneas de desarrollo.

Métodos teóricos

Analítico - sintético: Mediante este método se descompuso el problema en pequeñas partes y se realizó un mejor análisis del funcionamiento de los sistemas base, para así entender mejor sus funcionalidades y características.

Modelación: Permitió modelar el proceso a seguir para realizar la construcción del sistema operativo base de Nova, creando una abstracción, con el objetivo de explicar la realidad.

El presente documento estará conformado por: Introducción, Capítulos 1, 2, 3, Conclusiones, Recomendaciones, Bibliografía, Anexos y Glosario de Términos.

En el **Capítulo 1** se realiza un estudio del proceso que siguen algunas distribuciones GNU/Linux, para la construcción de sus sistemas operativos base. Se identifican los lenguajes, notaciones y herramientas que son empleados para modelar procesos. Además se establecen las convenciones que se utilizarán para describir el proceso.

En el **Capítulo 2** se modela el proceso de construcción del sistema operativo base de Nova. Se realiza la descripción de dicho proceso y se construye el sistema para validar la propuesta.

En el **Capítulo 3** se muestran pruebas realizadas al sistema construido como resultado de la investigación realizada.

Capítulo1: Fundamentación teórica

1.1 ¿Qué es un sistema operativo base?

Según los libros Linux From Scratch[2] y Slitaz Scratchbook[3], un sistema operativo base (SOB en lo adelante) GNU/Linux, es la combinación mínima necesaria de aplicaciones que permite la instalación de paquetes en un sistema GNU/Linux, que pueden o no soportar el arranque de la computadora.

1.2 Proceso de construcción del SOB.

1.2.1 Nova

El proyecto Nova no cuenta con un proceso propio para la construcción de su sistema operativo base, puesto que desde su creación ha sido realizado a partir de lo establecido por Gentoo Linux (la base de Nova hasta el momento).

1.2.1 Gentoo Linux[4]

Por su parte la distribución Gentoo Linux crea su SOB desde el código fuente (3) y ofrece a la comunidad un producto primitivo llamado "*stage1*" o primera fase, que provee una cadena de herramientas de compilación (*toolchain*). Pero esta fase no puede ser considerada su SOB debido a que debe ser recompilada (*bootstraped*) ella misma, para generar la segunda fase (*stage2*) sobre la cual se compila el SOB dando a lugar a la base de Gentoo o *stage3.* Es imprescindible destacar que los pasos de una fase a otra están automatizados mediante *scripts* provistos en el *stage1*. Las aplicaciones incluidas en cada fase pueden verse en el **Anexo 1**.

Esta fórmula ofrece la ventaja de que al compilarse el software puede optimizarse su rendimiento. A su vez evidencia desventajas como: la inclusión de herramientas de compilación innecesarias al usuario final en el SOB, trayendo consigo el aumento del espacio ocupado en disco y por consecuencia en los LiveCD, debido a la cantidad de librerías de desarrollo que se encuentran contenidas en el sistema operativo (SO).

1.2.2 Linux From Scratch

Linux From Scratch (LFS) es el nombre de un libro escrito por Gerard Beekmans junto a otros autores. El

[2] *http://www.linuxfromscratch.org*
[3] *http://*www.slitaz.org
[4] *http://www.gentoo.org*

libro es una guía para desarrollar un sistema GNU/Linux desde cero, a partir del código fuente (4).

Al igual que en el proceso que rige la construcción de Gentoo Linux, en LFS es necesario crear una cadena de compilación, recompilarla y luego compilar el SOB. La diferencia entre los dos procesos radica en tres puntos fundamentales:

LFS no ofrece herramientas para automatizar los procesos, puesto que uno de sus objetivos es que el usuario/aprendiz conozca cómo funciona GNU/Linux, desde sus conceptos más básicos.

LFS incluye la compilación del núcleo en su SOB.

LFS describe la realización de *scripts* de configuración y arranque necesarios para que el SO funcione.

1.2.3 Slitaz

SliTaz es una mini distribución y LiveCD del sistema operativo GNU/Linux diseñada para correr en un *hardware* con 128 Mb, y así ser la más pequeña de todas las mini distribuciones GNU/Linux, teniendo 25 Mb de CD y 80 Mb en el disco duro una vez instalada (5).

Para lograr minimizar su tamaño Slitaz desarrolla de una forma peculiar su SO desde la base. Al igual que en los proyectos antes mencionados, se parte de una cadena de compilación. En esta distribución esta cadena es provista por cualquier GNU/Linux. La diferencia radica en que una vez compiladas las aplicaciones se procede a realizar su instalación de forma manual, es decir, en la construcción de Slitaz se toma de cada aplicación compilada solamente lo necesario para que el SO funcione. Esto provee las siguientes ventajas: se optimiza el espacio ocupado en disco y se puede concebir un SO con sólo lo que se necesita, facilitando la escalabilidad del producto.

1.2.4 Red Hat[5]

La concepción del SOB de Red Hat y sus clones Fedora y CentOS no ha sido publicada. El motivo por el cual se incluye esta distribución en la investigación, es porque posee un paquete de software denominado “basesystem” que contiene las instrucciones de instalación y configuración de un SOB y puede ser instalado desde el repositorio. Esto es un punto fundamental si se quiere usar el SOB para crear otras distribuciones.

[5] *http://www.redhat.com*

Las distribuciones estudiadas utilizan como estándares Posix y FHS, por tanto para describir el proceso y construir el SOB se emplearán los mismos. Posix es una colección de documentos que definen claramente una interfaz estándar entre el sistema operativo y sus aplicaciones a nivel de código fuente. En otras palabras, define los servicios que debe proveer un sistema, definiendo de forma exacta los prototipos de las funciones de biblioteca y llamadas al sistema, los tipos de las variables utilizadas, las cabeceras, los códigos de retorno de las funciones y su comportamiento concurrente. También especifica otros aspectos, por ejemplo: la estructura general del sistema de ficheros, consideraciones sobre el set de caracteres usado, sobre las expresiones regulares, las variables del entorno, la interacción con el terminal o las secuencias de escape (6). El estándar FHS será empleado para detallar los nombres, ubicaciones, contenidos y permisos de los archivos y directorios, es decir para especificar una distribución común de los directorios y archivos.

Luego del análisis realizado se puede concluir que para adaptar el producto de esta investigación a los requerimientos de las nuevas metas del SO Nova se necesita cambiar el modelo que existe hasta el momento. El nuevo modelo consistirá en: La obtención de una cadena de compilación (*stage3* de Gentoo), con la que se construirán las aplicaciones, para la instalación de los binarios se utilizará la vía de Slitaz de instalación manual, que permitirá reducir el tamaño y dependencias del SOB y luego se compactará el resultado en un paquete de software que puede ser instalado desde los repositorios de la distribución.

1.3 Modelación de procesos.

El modelado de procesos es utilizado para capturar, documentar o rediseñar procesos de negocio (7).Cuando un proceso es modelado, pueden apreciarse con facilidad las interrelaciones existentes entre distintas actividades, analizarse cada actividad, así como identificarse los subprocesos comprendidos. De igual forma, los problemas pueden ponerse de manifiesto claramente dando la oportunidad de acciones de mejora.
En la actualidad existen un gran número de lenguajes, notaciones y herramientas de software que han sido diseñadas para la modelación de los procesos (8).

1.3.1 IDEF0.

IDEF0 constituye una técnica de modelación gráfica concebida para representar de manera estructurada y jerárquica las actividades que conforman un sistema o empresa, y los objetos o datos que soportan la

interacción de esas actividades. Es considerada una técnica sencilla pero poderosa, ampliamente usada en la industria durante la etapa de análisis en la reingeniería de procesos (9). Los elementos gráficos que se identifican en un proceso en IDEF0 pueden apreciarse en la figura 1.

Figura 1. Representación gráfica de un proceso en IDEF0

- Entrada: muestra los materiales o informaciones que se transformarán en la actividad para obtener la salida.

- Actividad: representa una función, proceso o transformación. Siempre debe ser un verbo o una frase verbal.

- Control: indica las regulaciones que determinan si una actividad se realiza o no. Por ejemplo: normas, guías, reglas, políticas.

- Sujeto: constituye los recursos que ejecutan una actividad. Ejemplo: personas, maquinarias.

- Salida: indica los objetos o informaciones producidos por la ocurrencia de la actividad. (10)

Ventajas

- Permite representar el proceso cronológicamente debido a que se descompone en niveles jerárquicos.

- Permite incorporar en el flujo los datos que entran y salen de las actividades, así como las reglas del negocio y los actores.

- Permite descomponer una actividad como un proceso.

- Permite identificar las actividades que aportan o no valor.

- Es adecuado en el diseño de sistemas complejos y dinámicos.

Desventajas.

- No permite definir responsabilidades fácilmente.

- Requiere una amplia formación y experiencia, tanto de la persona que lo elabora como del que lo interpreta.

- Limitado en la simbología porque el único símbolo utilizado es una caja rectangular que representa una actividad o función.

1.3.2 Notación BPMN

La notación para el modelado de procesos del negocio (BPMN) fue inicialmente desarrollada por la organización *Business Process Management Initiative* (BPMI) (11), que luego se unió con *Object Management Group* (OMG).[6]Define un tipo de diagrama de procesos de negocio (en inglés *Business Process Diagram* o BPD) que se utiliza para modelar procesos de negocio, así como los sub-procesos y tareas que lo componen. Un BPD está formado por los siguientes elementos gráficos:

-Objetos de flujo: Eventos, Actividades, Compuertas (figura 2).

Los eventos afectan el flujo del proceso y suelen tener una causa o resultado. Existen tres tipos de eventos: inicio, intermedio y fin. Las actividades pueden ser tareas o subprocesos y representan el trabajo que es ejecutado dentro de un proceso. El subproceso es una actividad que incluye a su vez un conjunto de actividades. Las compuertas son utilizadas para tomar decisiones y crear nuevos caminos.
Los objetos de flujo se conectan entre ellos en un diagrama para crear el esqueleto básico de la estructura de un proceso de negocio mediante conexiones.

Figura 2.Objetos de flujo

-Objetos de conexión: Flujo de Secuencia, Flujo de Mensaje, Asociación (figura 3).

[6] ***OMG:*** *Organización internacional más importante encargada de crear estándares relacionados con la Ingeniería del Software.*

El flujo de secuencia se usa para mostrar el orden en que las actividades se ejecutarán. El flujo de mensaje se emplea para mostrar los mensajes entre dos participantes y la asociación para mostrar las entradas y salidas de las actividades.

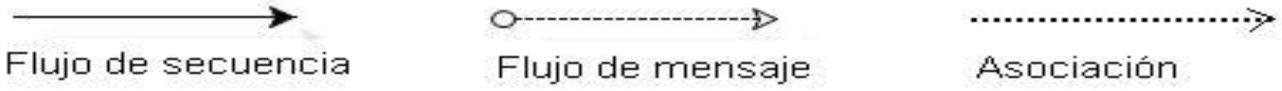

Figura 3.Objetos de conexión

-*Swimlanes* (Canales): Pool, Pista (Lane) (figura 4).

El pool representa un participante y se usa cuando un diagrama implica dos entidades del negocio o participantes separados. Las pistas son sub-particiones dentro de un pool, son usadas para organizar y categorizar actividades.

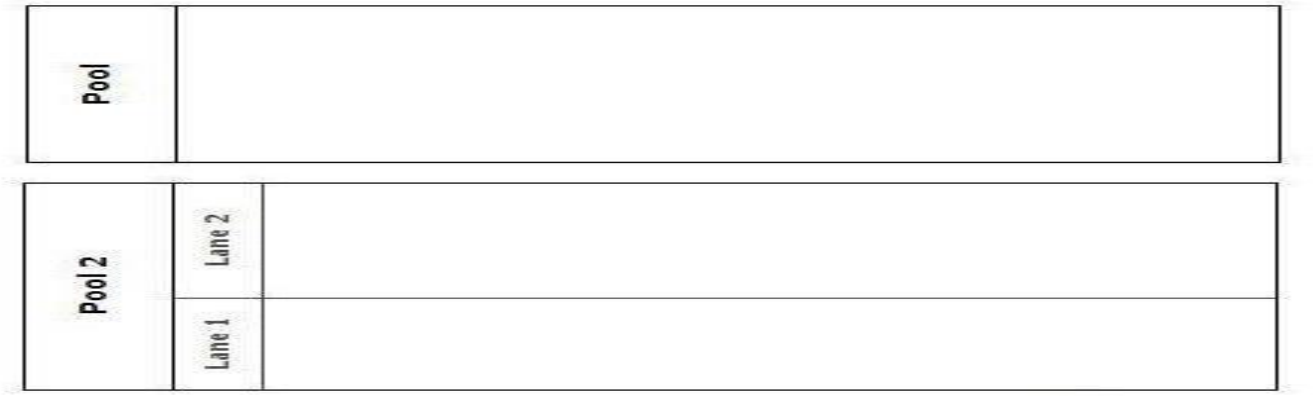

Figura 4.Swilanes

-Artefactos: Objetos de Datos, Grupo, Anotación (figura 5).

Los objetos de datos son un mecanismo para mostrar como los datos son requeridos o producidos por las actividades, están conectados a las actividades a través de asociaciones. El agrupamiento puede ser usado para propósitos de documentación o análisis, y no afecta la secuencia del flujo. Las anotaciones son mecanismos para que un modelador pueda dar información adicional.

Los modeladores pueden crear sus propios artefactos, en caso de desearlo para agregar más detalle al proceso (12).

Figura 5.Artefactos

Ventajas

- Considera un único diagrama para la representación de los procesos (BPD).

- Permite modelar los procesos de una manera unificada y estandarizada permitiendo un entendimiento a todas las personas de una organización.

- Está diseñada para modelar procesos manuales, automáticos, físicos o virtuales.

- Puede ser asignado con naturalidad a lenguajes de ejecución**.**

- Crea un enlace entre la etapa de diseño e implementación.

- Es capaz de representar procesos complejos.

- Está apoyada por organizaciones e instituciones de gran peso en las que participan las empresas más importantes del sector de la informática.

- Es independiente de cualquier metodología de modelado de procesos.

Desventajas

No incluye:

- Estructuras de organización y recursos.

- Reglas del negocio.

- Un mecanismo para el almacenamiento.

1.3.3 Diagrama de actividad de UML.

Los diagramas de actividad tienen entre sus objetivos modelar procesos del negocio.
Entre los elementos de un diagrama de actividad se encuentran (ver figura 6):

-Nodo inicial: Indica el comienzo del flujo de actividades. Se coloca dentro de la calle del actor que inicia el proceso.

-Actividad: Representa la acción que será realizada por el sistema.

-Flujo de control (Transición): representa la secuencia de cada elemento dentro del diagrama.

-Nodo de decisión o ramificación: ocurre cuando existe la posibilidad de que ocurra más de una transición (resultado) al terminar determinada actividad.

-Swimlanes o calles: representa al actor que realiza la actividad.

-Nodo final: Indica el final del flujo de actividades. Pueden existir varios de ellos.

- Flujo de objetos: Muestra el cambio de estado de un objeto al realizarse determinada actividad. (13)

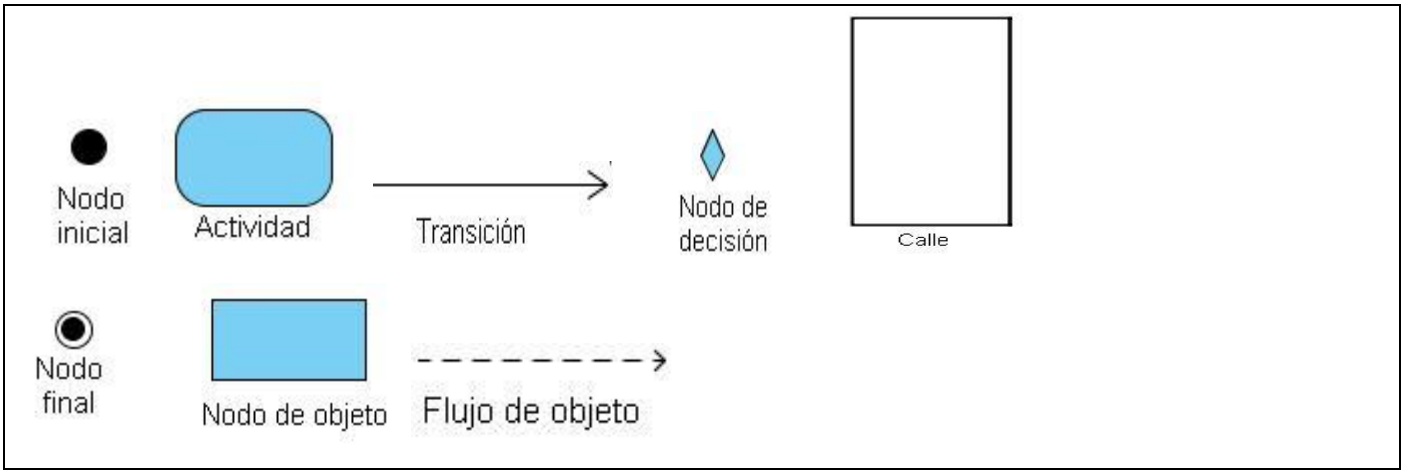

Figura 6.Elementos de un diagrama de actividades

Según comparaciones realizadas en (14) los diagramas de actividad de UML y la notación BPMN, teniendo en cuenta los patrones de flujo de trabajo que permiten medir la expresividad de los lenguajes y notaciones del modelado de procesos del negocio, se llegó a la conclusión de que BPMN es mejor por las siguientes razones:

- Es más rico gráficamente y además sus diagramas son más fáciles de comprender por personas que no son expertas.

- Da soporte a más patrones, es más expresivo.

- Tiene el apoyo de la *WfMC*, una de las organizaciones más importantes en el campo del flujo de trabajo y que es miembro de la OMG.

Después de analizar algunos de los lenguajes y notaciones utilizados para modelar procesos de negocio se determinó utilizar BPMN para modelar el proceso de construcción del SOB de la distribución cubana de GNU/Linux Nova. Se empleará esta notación porque intenta rescatar las mejores ideas de las notaciones anteriormente existentes y consolidarlas en una sola, como es el caso de los diagramas de actividades de UML, IDEF y ADF Diagram. Además es un estándar internacional de modelado de procesos aceptado por la comunidad, rico gráficamente y fácil de comprender.

1.3.4 Herramientas para modelar procesos empleando BPMN.

Existen muchas herramientas que permiten modelar un proceso usando BPMN, entre ellas se encuentran: Intalio, Bizagi y *Bussines Process Visual Architect* (BP-VA).

Intalio

Intalio es un software de código abierto y libre de costo. Está basado en un conjunto de frameworks y arquitecturas muy conocidas en la industria del software y con una madurez aceptable (15). Algunas de sus características son:

- Incluye un Diseñador visual BPMN.
- Incluye un motor de workflow.
- Permite la generación de aplicaciones sin código, es decir funciona como una herramienta CASE.
- Su interfaz es amistosa (web 2.0).
- Incluye BAM (Monitoreo de Actividades).
- Proporciona un esquema de adopción sencillo.
-Cuenta con facilidades para agregar nuevas características.
- Soporta varios lenguajes.

Bizagi

Es una herramienta gratuita, muy poderosa compatible solo con Windows, con el estándar BPMN 1.1 completo incluido en ella (16). Entre sus características se encuentran:

-Permite modelar con facilidad.

- Se pueden escribir propiedades en las figuras.

- Es capaz de generar un documento Word con el diagrama general, los subdiagramas, las figuras, un listado ordenado de las actividades y flujos con las propiedades, lo que permite obtener el documento de procesos automáticamente ahorrando tiempo.

Bussines Process Visual Architect

Esta herramienta es uno de los productos que ofrece Visual Paradigm.[7] Algunas de sus características son:

- Proporciona formas rápidas e intuitivas para visualizar, animar y analizar los procesos de negocio.

- Es fácil de usar.

- Alta velocidad a la hora de cargar y salvar los proyectos. (17)

Luego de estudiar las características de cada herramienta se decidió utilizar BP-VA por ser un completo modelador de procesos de negocio que se encuentra al alcance de los autores.

1.4 Descripción del proceso de construcción.

Para la descripción del proceso se utilizarán las pautas y convenciones establecidas en el libro LFS, estas son las siguientes:

- Los textos que aparecen en un rectángulo gris y en letra negrita pueden teclearse exactamente como aparecen, a menos que se indique lo contrario en el texto subyacente. También se utiliza en las secciones explicativas para identificar el comando al que se hace referencia.
- Los textos escritos en letra cursiva se utilizan para señalar aspectos importantes.
- El siguiente formato se emplea para crear ficheros de configuración

```
cat > $LFS/etc/group << "EOF"
root: x:0:
bin: x:1:
.....
```

[7] **Visual Paradigm:** es un potente conjunto de herramientas CASE UML .Soporta el modelado de negocios, la captura de requisitos, diseño de sistemas, la generación de bases de datos, generación de código y de los productos de gestión de proyectos.

EOF

En donde se encuentra **$LFS/etc/group** se solicita el fichero que se desea crear, a partir de lo que se teclee en las líneas siguientes, hasta encontrar la secuencia de fin del fichero (EOF).

- Los hipervínculos son direcciones de descarga.
- Los textos que se encuentran entre <> se utilizan para encapsular texto que no debe ser escrito tal como aparece.
- Las notas aclaratorias se representan con un rectángulo en amarillo.

Capítulo 2: Modelación y descripción del proceso.

En el primer capítulo se determinaron los elementos a tener en cuenta para crear el nuevo modelo del SOB de Nova. Se establecieron las herramientas que se utilizarán para modelar el proceso de construcción del SOB de Nova y las convenciones que serán empleadas para describir el mismo. Este capítulo se centra en la descripción del proceso de construcción del SOB de Nova.

2.1 Modelo del proceso.

Antes de realizar la descripción de determinado proceso es necesario modelarlo. Modelar el proceso permite a los involucrados tener una idea clara de lo que se pretende realizar, comprender mejor lo que se desea desarrollar y obtener el resultado deseado. En el capítulo anterior se hizo referencia a la herramienta utilizada para realizar el diagrama del proceso.

Para modelar el proceso de construcción del SOB de Nova (figura 7) se tuvieron en cuenta los roles desempeñados por los participantes, las actividades realizadas por cada uno, las decisiones a tomar y los artefactos generados que son producidos o requeridos por determinada actividad.

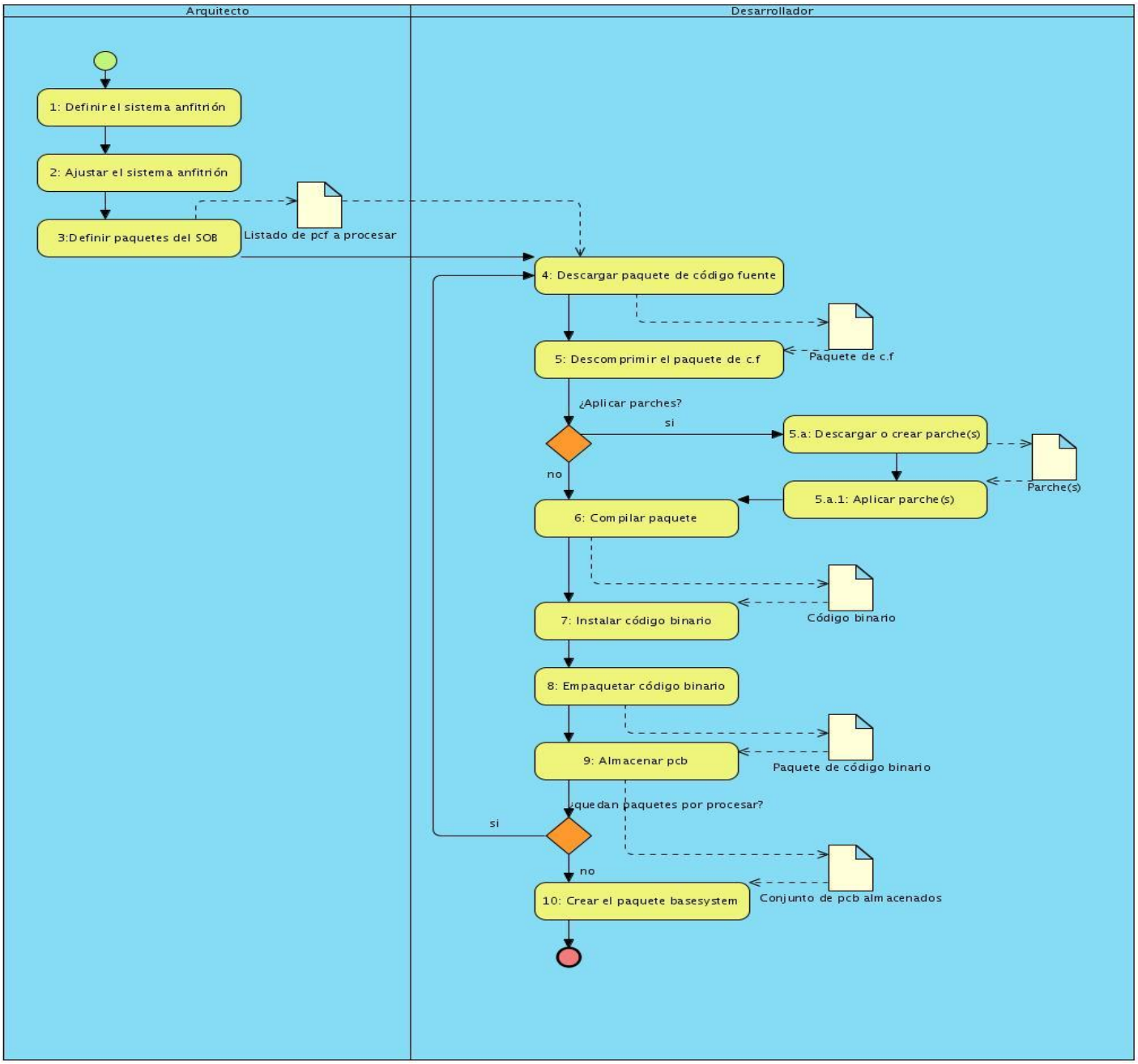

Figura 7 Diagrama del proceso de construcción del SOB de Nova

Para un mayor entendimiento del diagrama del proceso se utilizan fichas que lo describen de manera general (ver AnexoII).

2.2 Descripción del proceso de construcción del SOB de Nova.

Para construir el SOB de Nova el proceso se divide en dos fases.

2.2.1 Primera fase.

Definir el sistema anfitrión.

Lo primero es determinar el sistema anfitrión sobre el cual se va a desarrollar. En esta investigación el sistema elegido es un stage3 de Gentoo, que es utilizado como toolchain. Este stage puede ser descargado desde un mirror de Gentoo.

http://distro.ibiblio.org/pub/linux/distributions/gentoo/releases/x86/current-stage3/stage3-i486-20100216.tar.bz2.

- **Requisitos del sistema anfitrión.**

Luego de haber determinado el sistema anfitrión que se va a utilizar, hay que comprobar que el mismo tenga instalado los paquetes, que según el libro LFS son necesarios para construir un sistema operativo GNU/Linux. Para verificar la existencia de estos paquetes en el sistema anfitrión, se realizó una búsqueda en el directorio "/var/db/pkg/" donde se encuentra la base de datos del software instalado en el SO, corroborándose su capacidad para desarrollar el SOB. Las versiones que se relacionan con cada paquete son las que existen en el sistema seleccionado para la construcción. El listado de estas aplicaciones es el que se presenta a continuación:

+ app-shells/bash-4.0_p35

+ sys-devel/binutils-2.18-r3

+ sys-devel/bison-2.3

+ app-arch/bzip2-1.0.5-r1

+ sys-apps/coreutils-7.5-r1

+ sys-apps/diffutils-2.8.7-r2

+ sys-apps/findutils-4.4.0

+ sys-apps/gawk-3.1.6

+ sys-devel/gcc-4.3.4

+ sys-libs/glibc-2.10.1-r1

+ sys-apps/grep-2.5.4-r1

+ app-arch/gzip-1.4

+ sys-kernel/linux-headers-2.6.27-r2

+ sys-devel/make-3.81

+ sys-devel/patch-2.5.9

+ sys-apps/sed-4.2

+ App-arch/tar-1.20

Ajustar el sistema anfitrión.

Después de haber elegido el sistema anfitrión que se va a utilizar y comprobar que cumple con los requisitos necesarios, el paso siguiente es ajustarlo, para ello:

Se define si el sistema se va a alojar en un directorio o en una partición. Siendo en un directorio, se comprueba que exista y si no cumple este requerimiento, se crea. En caso de ser en una partición se crea y se garantiza que su contenido pueda ser accedido por el sistema. En esta investigación se determinó realizar el proceso de la siguiente forma:

- Se crea la partición
- Se crea el directorio donde se va montar la partición:

```
mkdir /media/sda1
```

- Se monta la partición:

```
mount /dev/sda1 /media/sda1
```

- Se copia el comprimido para la partición:

```
cp $HOME/Descargas/stage3-i486- 20100216.tar.bz2 /media/sda1
```

- Se descomprime manteniendo los permisos:

```
tar -xvjpf stage3-i486-20100216.tar.bz2
```

- Se copian los archivos resolv.conf y make.conf para el sistema y se ajusta.

```
cp /etc/{resolv.conf,make.conf} /media/sda1/etc
```

Nota

Estos archivos se encuentran en el directorio "/etc/" de cualquier sistema Gentoo o sus variaciones. El archivo resolv.conf contiene las direcciones de los servidores de dns del dominio y el make.conf contiene las configuraciones globales del sistema.

- Se montan los dispositivos:

```
mount –v --bind /dev /media/sda1/dev
```

- Se montan los procesos :

```
mount -t proc none /media/sda1/proc
```

- Se realiza el cambio de raíz:

```
chroot /media/sda1
```

- Se crean y actualizan las variables del entorno de trabajo:

```
env-update
```

- Se inicializan las variables que se crearon:

```
source /etc/profile
```

-Se crea un directorio donde se va a realizar la construcción de los paquetes.

```
mkdir $HOME/directorio_construccion
```

- Se crea una variable global NFS(*Nova From Scratch*) y se le asigna el directorio donde se va a realizar la construcción del sistema:

```
export NFS=/home/<zz>/<yy>
```

Reemplazar <zz> y <yy > por los directorios elegidos, en este caso usuario y directorio_construccion respectivamente.

2.2.2 Segunda fase.

Definir los paquetes del SOB.

La investigación realizada brindó como resultado que para construir el SOB de Nova, es necesario definir un listado de aplicaciones mínimas que ofrezcan las funcionalidades básicas de un SO. También es preciso incorporar a este listado un grupo de herramientas de compresión, que permitan al gestor de paquete utilizar el software disponible en los repositorios de la distribución. Si se desea que el sistema creado arranque, se procede a incorporar un grupo de paquetes de software que permitan el cumplimiento de esta funcionalidad. El listado de aplicaciones se pueden observar en la tabla 1.

Tabla 1 Listado de aplicaciones

Listado de aplicaciones		
Básicas	**Compresores**	**Adicionales**
Glibc-2.10.1-r1	Zlib-1.2.3-r1	Linux-2.6.32-r7
Coreutils-7.5-r1	Bzip2-1.0.5-r1	Module-Init-Tools-3.5
Bash-4.0-p37	Gzip-1.4	Aptitude-0.4.11.11
File-5.03	Tar-1.20	
Inetutils-1.7	Lzma-4.65	
Kbd-1.15	Cpio-2.10-r1	
Shadow-4.1.2.2		
Sysvinit-2.87-r3		
Udev-149		
Grep-2.5.4-r1		

Sed-4.2 Gawk-3.1.6		

Las aplicaciones básicas brindan: las librerías fundamentales del sistema y herramientas para ajustar la configuración del mismo, un intérprete de comandos por consola, programas para el trabajo con la red y procesamiento de ficheros, protección de contraseñas, reconocimiento de dispositivos y herramientas para: utilizar el teclado, controlar el arranque, ejecución y cierre del sistema operativo.

El listado de compresores utilizados brinda soporte a las distintas extensiones de compresión existentes; ejemplo de ello es: tar.bz2, tar.gz, tar.lzma, zip.

Construcción de los paquetes.

Luego de haber definido los paquetes del SOB es necesario crear un directorio donde se almacenen los paquetes y parches descargados. También se necesita un directorio de trabajo en el que se desempaqueten y compilen los códigos fuentes. Puede usarse /home/usuario/sources para almacenar los paquetes y parches y como directorio de trabajo directorio_construccion. Para crear el directorio se ejecuta el siguiente comando como usuario root antes de comenzar a descargar los paquetes:

```
mkdir -v $HOME/sources
```

Se crea una variable para tener un fácil acceso al directorio de los fuentes.

```
export SOURCE=$HOME/sources
```

Después hay que permitir que el usuario tenga permisos de escritura y que el propietario sea el único que pueda borrar el fichero. El siguiente comando activará estos permisos:

```
chmod -v a+wt $NFS
```

Más tarde se procede a realizar el trabajo con los paquetes.

- **Glibc-2.10.1-r1**

Descripción: contiene la librería C principal. Esta librería proporciona todas las rutinas básicas para la ubicación de memoria, búsqueda de directorios, abrir y cerrar ficheros, leerlos y escribirlos, manejo de

cadenas, coincidencia de patrones y aritmética (18). Además es portable y soporta gran cantidad de software.

Sitio de descarga:

http://gentoo.uci.cu/gentoo/distfiles/glibc-2.10.1.tar.bz2

http://gentoo.uci.cu/gentoo/distfiles/glibc-libidn-2.10.1.tar.bz2

Preparación del entorno de construcción del paquete:

Se crea un directorio, en este caso con el mismo nombre del paquete para identificarlo.

```
mkdir $NFS/glibc
```

Luego se copia el comprimido del código fuente y de su addon.

```
cp $SOURCE/{glibc-2.10.1.tar.bz2, glibc-libidn-2.10.1.tar.bz2} $NFS/glibc/
```

Se accede al directorio donde se van a desempaquetar los archivos.

```
cd /$NFS/glibc
```

Se descomprime el código fuente del paquete y de libidn.

```
tar -xvjpf glibc-2.10.1.tar.bz2
tar -xvjpf glibc-libidn-2.10.1.tar.bz2
```

Se crea un directorio dentro del código fuente del paquete con el nombre del addon y se mueve su contenido para ese directorio.

```
mkdir –v glibc-2.10.1/libidn
mv –v glibc-libidn-2.10.1/* glibc-2.10.1/libidn
```

La construcción de este paquete es necesario realizarla en un directorio diferente de donde se encuentran el código fuente.

```
mkdir glibc-build
```

Se crea el directorio en donde se va a instalar el paquete.

```
mkdir /system/glibc
```

Configuración del paquete:

Luego se aplican los siguientes scripts de bash para eliminar incoherencias del código fuente.

```
DL=$(readelf -l /bin/sh | sed -n 's@.*interpret.*/tools\(.*\)]$@\1@p') sed -i "s|libs -o|libs -L/usr/lib -
Wl,-dynamic-linker=$DL -o|" \scripts/test-installation.pl unset DL
```

```
sed -i s/utf8/UTF-8/ libio/tst-fgetwc.c
sed -i '/tst-fgetws-ENV/ a\
tst-fgetwc-ENV = LOCPATH=$(common-objpfx)localedata' libio/Makefile
```

```
sed -i \
-e 's/FUTEX_WAIT\( | FUTEX_CLOCK_REALTIME, reg\)/FUTEX_WAIT_BITSET\1/' \
nptl/sysdeps/unix/sysv/linux/i386/i486/lowlevellock.S
```

Luego se accede al directorio de construcción y se adicionan las banderas necesarias para la construcción del paquete:

```
cd ../glibc-build

case `uname -m` in
  i?86) echo "CFLAGS += -march=i486 -mtune=native -O3 -pipe" > configparms ;;
esac
```

-Opciones de configuración**:**

```
CFLAGS="O2 -U_FORTIFY_SOURCE -fno-stack-protector" ../glibc-2.10.1/configure \
--prefix=/system/glibc/usr --libdir=/system/glibc/usr/lib \
 --mandir=/system/glibc/usr/share/man --infodir=/system/glibc/usr/share/info \
--libexecdir=/system/glibc/usr/lib/misc/glibc --disable-profile --enable-add-ons \
--enable-kernel=2.6.0 --without-cvs --without-selinux
```

Significado de las opciones:

--prefix=/system/glibc/usr

Le indica que la instalación se realizará teniendo como raíz el directorio especificado.

--libdir=/system/glibc/usr/lib

Le indica el directorio donde se instalarán las librerías.

--mandir=/system/glibc/usr/share/man

Le indica el directorio donde se instalarán las páginas de los manuales.

--libexecdir=/system/glibc/usr/lib/misc/glibc

Le indica el directorio donde se instalarán las librerías de ejecución.

--disable-profile

Construye las librerías sin información de perfiles.

--enable-add-ons

Le indica que utilice el añadido NPTL como su librería de hilos y los addons que se encuentren en el directorio de código fuente, incluyendo en este caso a libidn.

--enable-kernel=2.6.0

Le indica que construya las librerías con soporte para núcleos Linux 2.6.x.

--without-cvs

Le indica que construya las librerías sin soporte para cvs.

--without-selinux

Le indica que construya las librerías sin soporte para selinux.

Construcción:

Se compila el paquete.

```
make
```

Se ejecuta el banco de pruebas.

```
make check
```

Se instala el paquete.

```
make install
```

Empaquetado y Almacenamiento del código binario:

Se accede al directorio

```
cd /system
```

Se empaqueta el código binario generado, utilizando como herramienta de compresión tar. Teniendo como opciones: c para indicar que es una compresión, v para mostrar la salida de la compresión, j para indicar que el resultado final tiene como extensión tar.bz2, p para mantener los permisos existentes dentro del directorio que se va a empaquetar y f para indicar que se tiene como entrada un directorio.

```
tar -cvjpf glibc-bin-2.10.1-r1.tar.bz2 glibc/*
```

Se crea un directorio donde se van a almacenar los binarios una vez empaquetados.

```
mkdir $HOME/binaries
```

Se copia el paquete de código binario para el directorio creado.

```
cp /system/glibc-bin-2.10.1-r1.tar.bz2 $HOME/binaries
```

- **Coreutils-7.5-r1**

Descripción: contiene utilidades para mostrar y establecer las características básicas del sistema (18). Es una combinación de tres paquetes: utilidades de ficheros (fileutils), utilidades de intérpretes de comandos (shellutils) y utilidades de proceso de textos (textutils) (19).

Sitio de descarga: *http://gentoo.uci.cu/gentoo/distfiles/coreutils-7.5.tar.gz*

Preparación del entorno de construcción del paquete:

Se crea un directorio.

```
mkdir $NFS/coreutils
```

Luego se copia el comprimido del código fuente y los parches.

```
cp $SOURCE /coreutils-7.5.tar.gz $NFS/coreutils/
cp $SOURCE/coreutils-7.5-patches-2.tar.lzma $NFS/coreutils/
```

Se accede al directorio donde se van a desempaquetar los archivos.

```
cd $NFS/coreutils
```

Se descomprimen el código fuente y los parches.

```
tar -xvzpf coreutils-7.5.tar.gz

lzma -cd coreutils-7.5-patches-2.tar.lzma | tar xvp
```

Se accede al directorio.

```
cd coreutils-7.5-r1/
```

Se crea el directorio donde se va a instalar el paquete.

```
mkdir /system/coreutils
```

Nota

La sección **preparación del entorno de construcción**, se realiza de manera similar en los demás paquetes, a diferencia de module-init-tools. Por esta razón esta sección será incluida solamente en module-init-tools.

Configuración del paquete:

-Opciones de configuración.

```
../coreutils-7.5/configure --prefix=/system/coreutils/ \
--bindir=/system/coreutils/usr/bin/ --sysconfdir=/system/coreutils/etc/ \
--libdir=/system/coreutils/usr/lib/ --includedir=/system/coreutils/usr/include/ \
--datarootdir=/system/coreutils/usr/share/ \
--infodir=/system/coreutils/usr/share/info/ \
--localedir=/system/coreutils/usr/share/locale/ \
--mandir=/system/coreutils/usr/share/man/ \
--docdir=/system/coreutils/usr/share/doc/coreutils/
```

Significado de las opciones:

--bindir=/system/coreutils/usr/bin/

Es el directorio donde se guardarán los principales ejecutables del sistema.

--sysconfdir=/system/coreutils/etc/

Indica la dirección donde se almacenarán los archivos de configuración.

--includedir=/system/coreutils/usr/include/

Indica la dirección donde se encontrarán los archivos de cabecera de C.

--datarootdir=/system/coreutils/usr/share/

Indica la raíz del directorio de datos

--localedir=/system/coreutils/usr/share/locale/

Se almacenarán las locales del sistema.

--docdir=/system/coreutils/usr/share/doc/coreutils/

Se almacenará la documentación del paquete.

Construcción:

Se compila el paquete

```
make
```

Se instala

```
make install
```

Se mueven los programas a los lugares especificados por el FHS[8].

```
mv -v /system/coreutils/usr/bin/{cat,chgrp,chmod,chown,cp,date,dd,df,echo} /system/coreutils/bin
mv -v /system/coreutils/usr/bin/{false,ln,ls,mkdir,mknod,mv,pwd,rm} /system/coreutils/bin
mv -v /system/coreutils/usr/bin/{rmdir,stty,sync,true,uname} /system/coreutils/bin
mv -v /system/coreutils/usr/bin/chroot /system/coreutils/usr/sbin
```

[8] FHS: Estándar de la Jerarquía del Sistema de Ficheros en el que se basa el árbol de directorios.

```
mv -v /system/coreutils/usr/bin/{head,sleep,nice} /system/coreutils/bin
```

Empaquetado y Almacenamiento del código binario:

```
cd /system
tar -cvjpf coreutils-bin-7.5-r1.tar.bz2 coreutils/*
cp /system/coreutils-bin-7.5-r1.tar.bz2 $HOME/binaries
```

- **Bash-4.0-p37**

Descripción: contiene el intérprete de lenguaje de comandos definido en la mayoría de los sistemas GNU/Linux. El nombre es un acrónimo de la "Bourne-Again SHell", un juego de palabras sobre Stephen Bourne, el autor (20).

Sitio de descarga: *http://gentoo.uci.cu/gentoo/distfiles/bash-4.0.tar.gz*

Parches que necesita:

La versión 4.0 de Bash elegida tiene 37 correcciones que necesitan aplicarse al paquete.

Dirección de descarga:

http://gentoo.uci.cu/gentoo/distfiles/bash40-001
http://gentoo.uci.cu/gentoo/distfiles/bash40-002
http://gentoo.uci.cu/gentoo/distfiles/bash40-003
http://gentoo.uci.cu/gentoo/distfiles/bash40-004
http://gentoo.uci.cu/gentoo/distfiles/bash40-005
http://gentoo.uci.cu/gentoo/distfiles/bash40-006
http://gentoo.uci.cu/gentoo/distfiles/bash40-007
http://gentoo.uci.cu/gentoo/distfiles/bash40-008
http://gentoo.uci.cu/gentoo/distfiles/bash40-009
http://gentoo.uci.cu/gentoo/distfiles/bash40-010
http://gentoo.uci.cu/gentoo/distfiles/bash40-011
http://gentoo.uci.cu/gentoo/distfiles/bash40-012
http://gentoo.uci.cu/gentoo/distfiles/bash40-013
http://gentoo.uci.cu/gentoo/distfiles/bash40-014

http://gentoo.uci.cu/gentoo/distfiles/bash40-015
http://gentoo.uci.cu/gentoo/distfiles/bash40-016
http://gentoo.uci.cu/gentoo/distfiles/bash40-017
http://gentoo.uci.cu/gentoo/distfiles/bash40-018
http://gentoo.uci.cu/gentoo/distfiles/bash40-019
http://gentoo.uci.cu/gentoo/distfiles/bash40-020
http://gentoo.uci.cu/gentoo/distfiles/bash40-021
http://gentoo.uci.cu/gentoo/distfiles/bash40-022
http://gentoo.uci.cu/gentoo/distfiles/bash40-023
http://gentoo.uci.cu/gentoo/distfiles/bash40-024
http://gentoo.uci.cu/gentoo/distfiles/bash40-025
http://gentoo.uci.cu/gentoo/distfiles/bash40-026
http://gentoo.uci.cu/gentoo/distfiles/bash40-027
http://gentoo.uci.cu/gentoo/distfiles/bash40-028
http://gentoo.uci.cu/gentoo/distfiles/bash40-029
http://gentoo.uci.cu/gentoo/distfiles/bash40-030
http://gentoo.uci.cu/gentoo/distfiles/bash40-031
http://gentoo.uci.cu/gentoo/distfiles/bash40-032
http://gentoo.uci.cu/gentoo/distfiles/bash40-033
http://gentoo.uci.cu/gentoo/distfiles/bash40-034
http://gentoo.uci.cu/gentoo/distfiles/bash40-035
http://gentoo.uci.cu/gentoo/distfiles/bash40-036
http://gentoo.uci.cu/gentoo/distfiles/bash40-037

Configuración del paquete:

-Aplicar parches

```
patch -Np0 -i ../patch/bash40-001
patch -Np0 -i ../patch/bash40-002
patch -Np0 -i ../patch/bash40-003
patch -Np0 -i ../patch/bash40-004
```

```
patch -Np0 -i ../patch/bash40-005
patch -Np0 -i ../patch/bash40-006
patch -Np0 -i ../patch/bash40-007
patch -Np0 -i ../patch/bash40-008
patch -Np0 -i ../patch/bash40-009
patch -Np0 -i ../patch/bash40-010
patch -Np0 -i ../patch/bash40-011
patch -Np0 -i ../patch/bash40-012
patch -Np0 -i ../patch/bash40-013
patch -Np0 -i ../patch/bash40-014
patch -Np0 -i ../patch/bash40-015
patch -Np0 -i ../patch/bash40-016
patch -Np0 -i ../patch/bash40-017
patch -Np0 -i ../patch/bash40-018
patch -Np0 -i ../patch/bash40-019
patch -Np0 -i ../patch/bash40-020
patch -Np0 -i ../patch/bash40-021
patch -Np0 -i ../patch/bash40-022
patch -Np0 -i ../patch/bash40-023
patch -Np0 -i ../patch/bash40-024
patch -Np0 -i ../patch/bash40-025
patch -Np0 -i ../patch/bash40-026
patch -Np0 -i ../patch/bash40-028
patch -Np0 -i ../patch/bash40-029
patch -Np0 -i ../patch/bash40-030
patch -Np0 -i ../patch/bash40-031
patch -Np0 -i ../patch/bash40-032
patch -Np0 -i ../patch/bash40-033
patch -Np0 -i ../patch/bash40-034
patch -Np0 -i ../patch/bash40-035
```

```
patch -Np0 -i ../patch/bash40-036
patch -Np0 -i ../patch/bash40-037
```

-Opciones de configuración.

```
../bash-4.0/configure --prefix=/system/bash/usr/ --bindir=/system/bash/usr/bin/ \
--sysconfdir=/system/bash/etc/ --libdir=/system/bash/usr/lib/\
 --includedir=/system/bash/usr/include/ --datarootdir=/system/bash/usr/share/ \
--infodir=/system/bash/usr/share/info/ --localedir=/system/bash/usr/share/locale/ \
--mandir=/system/bash/usr/share/man/ --docdir=/system/bash/usr/share/doc/bash/ \
--without-bash-malloc
```

Significado de las opciones:

--without-bash-malloc

Desactiva el uso de la función de ubicación de memoria (malloc) de Bash, que se sabe que provoca violaciones de segmento. Al desactivar esta opción Bash utilizará la función malloc de Glibc, que es más estable (18).

Construcción:

Se compila el paquete

```
make
```

Se instala

```
make install
```

Empaquetado y Almacenamiento del código binario:

```
cd /system
tar -cvjpf bash-bin-4.0.tar.bz2 bash/*
cp /system/bash-bin-4.0.tar.bz2 $HOME/binaries/
```

- **File-5.03**

Descripción: contiene una utilidad para determinar el tipo de los ficheros (18).

Sitio de descarga: *http://gentoo.uci.cu/gentoo/distfiles/file-5.03.tar.gz*

Configuración del paquete:

-Opciones de configuración

```
../file-5.03/configure --prefix=/system/file/usr/ --bindir=/system/file/usr/bin/ \
--sysconfdir=/system/file/etc/ --libdir=/system/file/usr/lib/ \
 --includedir=/system/file/usr/include/ --datarootdir=/system/file/usr/share/ \
 --infodir=/system/file/usr/share/info/ --localedir=/system/file/usr/share/locale/ \
--mandir=/system/file/usr/share/man/ --docdir=/system/file/usr/share/doc/file/
```

Construcción:

Se compila el paquete

```
make
```

Se instala

```
make install
```

Empaquetado y Almacenamiento del código binario:

```
cd /system
tar -cvjpf file-bin-5.03.tar.bz2 file/*
cp /system/file-bin-5.03.tar.bz2 $HOME/binaries
```

- **Inetutils-1.7**

Descripción: contiene programas para trabajo básico en red (18).

Sitio de descarga: *http://ftp.gnu.org/gnu/inetutils/inetutils-1.7.tar.gz*

Configuración del paquete:

-Opciones de configuración.

```
../inetutils-1.7/configure --prefix=/system/inetutils/usr/  \
--bindir=/system/inetutils/usr/bin/ --sysconfdir=/system/inetutils/etc/ \
--libdir=/system/inetutils/usr/lib/ --includedir=/system/inetutils/usr/include/ \
--datarootdir=/system/inetutils/usr/share/ --infodir=/system/inetutils/usr/share/info/ \
--localedir=/system/inetutils/usr/share/locale/ \
--mandir=/system/inetutils/usr/share/man/ \
--docdir=/system/inetutils/usr/share/doc/inetutils/ \
--localstatedir=/system/inetutils/var/ --disable-whois --disable-servers
```

Significado de las opciones:

--disable-whois

Desactiva la construcción del cliente whois de Inetutils, que está demasiado anticuado (18).

--disable-servers

Desactiva la construcción de los diferentes servidores incluidos como parte del paquete Inetutils (18).

Construcción:

Se compila el paquete

```
make
```

Se instala

```
make install
```

Empaquetado y Almacenamiento del código binario:

```
cd /system
tar -cvjpf inetutils-bin-1.7.tar.bz2 inetutils/*
cp /system/inetutils-bin-1.7.tar.bz2 $HOME/binaries
```

- **Kbd-1.15**

Descripción: contiene ficheros de mapas de teclado y utilidades para el teclado (18).

Sitio de descarga: *http://gentoo.uci.cu/gentoo/distfiles/kbd-1.15.tar.gz*

Configuración del paquete:

-Opciones de configuración

```
../kbd-1.15/configure --prefix=/system/kbd/usr/ --bindir=/system/kbd/usr/bin/ \
--sysconfdir=/system/kbd/etc/ --libdir=/system/kbd/usr/lib/ \
--includedir=/system/kbd/usr/include/ --datarootdir=/system/kbd/usr/share/ \
--datadir=/system/kbd/lib/kbd --infodir=/system/kbd/usr/share/info/ \
--localedir=/system/kbd/usr/share/locale/ --mandir=/system/kbd/usr/share/man/ \
--docdir=/system/kbd/usr/share/doc/kbd/ --localstatedir=/system/kbd/var/
```

Construcción:

Se compila el paquete

```
make
```

Se instala

```
make install
```

Empaquetado y Almacenamiento del código binario:

```
cd /system
tar -cvjpf kbd-bin-1.15.tar.bz2 kbd/*
cp /system/kbd-bin-1.15.tar.bz2 $HOME/binaries
```

- **Shadow-4.1.2.2**

Descripción: contiene programas para manejar contraseñas de forma segura (18).

Sitio de descarga: *http://gentoo.uci.cu/gentoo/distfiles/shadow-4.1.2.2.tar.bz2*

Configuración del paquete:

-Opciones de configuración

```
../shadow-4.1.2.2/configure --prefix=/system/shadow/usr/ \
--bindir=/system/shadow/usr/bin/ --sysconfdir=/system/shadow/etc/ \
--libdir=/system/shadow/usr/lib/ --includedir=/system/shadow/usr/include/ \
--datarootdir=/system/shadow/usr/share/ --datadir=/system/shadow/lib/shadow \
--infodir=/system/shadow/usr/share/info/ \
--localedir=/system/shadow/usr/share/locale/ \
--mandir=/system/shadow/usr/share/man/ \
--docdir=/system/shadow/usr/share/doc/shadow/ \
--localstatedir=/system/shadow/var/ --enable-shared --without-selinux
```

Significado de las opciones de configuración:

--enable-shared

Permite construir librerías compartidas en el sistema.

Construcción:

Se compila el paquete

```
make
```

Se instala

```
make install
```

Empaquetado y Almacenamiento del código binario:

```
cd /system
tar -cvjpf shadow-bin-4.1.2.2.tar.bz2 shadow/*
cp /system/shadow-bin-4.1.2.2.tar.bz2 $HOME/binaries
```

- **Sysvinit-2.87-r3**

Descripción: contiene programas para controlar el arranque, ejecución y cierre del sistema (18). El programa más importante del paquete es /sbin/init. Es el primer proceso que se inicia al arrancar el sistema y se continúa ejecutando como proceso número 1 hasta que se detiene. Todos los demás procesos descienden de él (21).

Sitio de descarga: *http://gentoo.uci.cu/gentoo/distfiles/sysvinit_dsf.orig.tar.gz*

Parches que necesita: La versión 2.87 de este paquete luego de su tercera revisión, incluye 7 correcciones que son necesarias aplicarle, para el modelo de sistema operativo que se está desarrollando.

0001-shutdown-sync-usage-and-man-page

0002-shutdown-let-non-root-users-see-usage

0003-halt-add-a-k-kexec-flag

0004-shutdown-use-single-user-by-default

0005-init-fix-race-with-utmp-and-children

0006-clean-up-build-system-a-bit

0007-init-update-version

Dirección de descarga (parches): *http://gentoo.uci.cu/gentoo/distfiles/sysvinit-2.87-patches-2.tar.bz2.*

Configuración del paquete:

-Aplicar parches

```
patch -Np1 -i ../patch/0001-shutdown-sync-usage-and-man-page.patch
patch -Np1 -i ../patch/0002-shutdown-let-non-root-users-see-usage.patch
patch -Np1 -i ../patch/0003-halt-add-a-k-kexec-flag.patch
patch -Np1 -i ../patch/0004-shutdown-use-single-user-by-default.patch
patch -Np1 -i ../patch/0005-init-fix-race-with-utmp-and-children.patch
patch -Np1 -i ../patch/0006-clean-up-build-system-a-bit.patch
patch -Np1 -i ../patch/0007-init-update-version.patch
```

Construcción:

Para construir este paquete es necesario crear la estructura de directorios en la que se van a instalar los binarios de forma manual.

```
mkdir /system/sysvinit/{bin,sbin,usr}
mkdir /system/sysvinit/usr/{bin,include,share}
mkdir /system/sysvinit/usr/share/man
mkdir /system/sysvinit/usr/share/man/{man1,man5,man8}
make -C src
make -C src install ROOT=/system/sysvinit/
```

Empaquetado y Almacenamiento del código binario:

```
cd /system
tar -cvjpf sysvinit-bin-2.87-r3.tar.bz2 sysvinit/*
cp /system/sysvinit-bin-2.87-r3.tar.bz2 $HOME/binaries
```

- **Udev-149**

Descripción: contiene programas para la creación dinámica de nodos de dispositivos (18).

Sitio de descarga: *http://gentoo.uci.cu/gentoo/distfiles/udev-149.tar.bz2*

Configuración del paquete:

Se crean algunos dispositivos y directorios que este paquete no puede manejar, debido a que son necesarios al principio del proceso de arranque del SO:

```
install -dv /system/udev/lib/{firmware,udev/devices/{pts,shm}}
mknod -m0666 /system/udev/lib/udev/devices/null c 1 3
mknod -m0600 /system/udev/lib/udev/devices/kmsg c 1 11
ln -sv /proc/self/fd /system/udev/lib/udev/devices/fd
ln -sv /proc/self/fd/0 /system/udev/lib/udev/devices/stdin
ln -sv /proc/self/fd/1 /system/udev/lib/udev/devices/stdout
ln -sv /proc/self/fd/2 /system/udev/lib/udev/devices/stderr
```

```
ln -sv /proc/kcore /system/udev/lib/udev/devices/core
```

-Opciones de configuración.

```
../udev-145/configure --prefix=/system/udev/usr --sysconfdir=/system/udev/etc \
--sbindir=/system/udev/sbin --with-rootlibdir=/system/udev/lib \
--libexecdir=/system/udev/lib/udev --docdir=/system/udev/usr/share/doc/udev-145 \
--disable-extras
```

Significado de las opciones de configuración:

--with-rootlibdir=/system/udev/lib

Indica donde se instalará la librería libudev (22).

--disable-extras

Impide la construcción de librerías adicionales que no son necesarias para un sistema base. (22)

Construcción:

Se compila el paquete

```
make
```

Se instala

```
make install
```

Se instalan reglas para que udev pueda (22):

Reconocer mayor cantidad de dispositivos.

```
install -m644 -v rules/packages/64-*.rules /system/udev/lib/udev/rules.d/
```

Crear vínculos simbólicos para dispositivos que serán montados manualmente.

```
install -m644 -v rules/packages/40-pilot-links.rules /system/udev/lib/udev/rules.d/
```

Para manejar dispositivos de tipo ISDN

```
install -m644 -v rules/packages/40-isdn.rules /system/udev/lib/udev/rules.d/
```

Empaquetado y Almacenamiento del código binario:

```
cd /system
tar -cvjpf udev-bin-149.tar.bz2 udev/*
cp /system/udev-bin-149.tar.bz2 $HOME/binaries
```

- **Grep-2.5.4-r1**

Descripción: contiene programas para buscar dentro de ficheros (18).

Sitio de descarga: *http://gentoo.uci.cu/gentoo/distfiles/grep-2.5.4.tar.bz2*

Configuración del paquete:

-Opciones de configuración:

```
../grep-2.5.4/configure --prefix=/system/grep/usr/ --bindir=/system/grep/usr/bin/ \
--sysconfdir=/system/grep/etc/ --libdir=/system/grep/usr/lib/ \
--includedir=/system/grep/usr/include/ --datarootdir=/system/grep/usr/share/ \
--datadir=/system/grep/lib/grep --infodir=/system/grep/usr/share/info/ \
--localedir=/system/grep/usr/share/locale/ --mandir=/system/grep/usr/share/man/ \
--docdir=/system/grep/usr/share/doc/grep/ --localstatedir=/system/grep/var/ \
--disable-perl-regexp
```

Significado de las opciones de configuración:

--localstatedir=/system/grep/var/

Cambia la localización de la base de datos de locate para que se encuentre en /system/grep/var/ (18).

--disable-perl-regexp

Esto asegura que **grep** no se enlaza con alguna librería PCRE que puede estar presente en el sistema anfitrión (18).

Construcción:

Se compila el paquete

```
make
```

Se ejecuta el banco de pruebas

```
make check
```

Se instala el paquete

```
make install
```

Empaquetado y Almacenamiento del código binario:

```
cd /system
tar -cvjpf grep-bin-2.5.4.tar.bz2 grep/*
cp /system/grep-bin-2.5.4.tar.bz2 $HOME/binaries
```

- **Sed-4.2**

Descripción: El paquete Sed permite manipular flujos de datos, como por ejemplo cortar líneas, buscar y reemplazar texto (con soporte de expresiones regulares) (23).
Sitio de descarga: *http://gentoo.uci.cu/gentoo/distfiles/sed-4.2.tar.bz2.*

Configuración del paquete:

-Opciones de configuración:

```
../sed-4.2/configure --prefix=/system/sed/usr/ --bindir=/system/sed/usr/bin/ \
--sysconfdir=/system/sed/etc/ --libdir=/system/sed/usr/lib/ \
--includedir=/system/sed/usr/include/ --datarootdir=/system/sed/usr/share/ \
--infodir=/system/sed/usr/share/info/ --localedir=/system/sed/usr/share/locale/ \
 --mandir=/system/sed/usr/share/man/ --docdir=/system/sed/usr/share/doc/sed/ \
  --localstatedir=/system/sed/var/
```

Construcción:

Se compila el paquete

```
make
```

Se instala el paquete

```
make install
```

Empaquetado y Almacenamiento del código binario:

```
cd /system

tar -cvjpf sed-bin-4.2.tar.bz2 sed/*

cp /system/sed-bin-4.2.tar.bz2 $HOME/binaries
```

- **Gawk-3.1.6**

Descripción: contiene programas para manipular ficheros de texto (18).

Sitio de descarga: *http://gentoo.uci.cu/gentoo/distfiles/gwak-3.1.6.tar.bz2.*

Configuración del paquete:

-Opciones de configuración:

```
../gawk-3.1.6/configure --prefix=/system/gawk/usr/ --bindir=/system/gawk/usr/bin/ \
--sysconfdir=/system/gawk/etc/ --libdir=/system/gawk/usr/lib/ \
--libexecdir=/system/gawk/usr/libexec/ --includedir=/system/gawk/usr/include/ \
--datarootdir=/system/gawk/usr/share/ --infodir=/system/gawk/usr/share/info/ \
--localedir=/system/gawk/usr/share/locale/ --mandir=/system/gawk/usr/share/man/ \
--docdir=/system/gawk/usr/share/doc/gawk/ \
--localstatedir=/system/gawk/var/
```

Construcción:

Se compila el paquete

```
make
```

Se ejecuta el banco de pruebas

```
make check
```

Se instala el paquete

```
make install
```

Empaquetado y Almacenamiento del código binario:

```
cd /system
tar -cvjpf gawk-bin-3.1.6.tar.bz2 gawk/*
cp /system/gawk-bin-3.1.6.tar.bz2 $HOME/binaries
```

Listado de compresores

- **Zlib-1.2.3-r1**

Descripción: contiene rutinas de compresión y descompresión usadas por algunos programas (18).

Sitio de descarga: *http://gentoo.uci.cu/gentoo/distfiles/zlib-1.2.3.tar.bz2.*

Configuración del paquete:

-Opciones de configuración:

```
../zlib-1.2.3/configure --prefix=/system/zlib/usr/ --libdir=/system/zlib/usr/lib/ --shared
```

Significado de las opciones de configuración:

--shared

Para construir las librerías compartidas.

Construcción:

Se compila el paquete

```
make
```

Se ejecuta el banco de pruebas

```
make check
```

Se instalan las librerías compartidas del paquete.

```
make install
```

Se construye la librería estática

```
make clean
./configure --prefix=/system/zlib/usr/
make
```

Se ejecuta nuevamente el banco de pruebas.

```
make check
```

Se instala la librería estática.

```
make install
```

Se corrigen los permisos de la librería estática.

```
chmod -v 644 /usr/lib/libz.a
```

Empaquetado y Almacenamiento del código binario:

```
cd /system
tar -cvjpf zlib-bin-1.2.3.tar.bz2 zlib/*
cp /system/zlib-bin-1.2.3.tar.bz2 $HOME/binaries
```

- **Bzip2-1.0.5-r1**

Descripción: contiene programas para comprimir y descomprimir ficheros (18).

Sitio de descarga: *http://gentoo.uci.cu/gentoo/distfiles/bzip2-1.0.5.tar.gz*

Construcción:

Se compila el paquete

```
make
```

Se instala el paquete para el directorio indicado en PREFIX.

```
make PREFIX=/system/bzip2/ install
```

Empaquetado y Almacenamiento del código binario:

```
cd /system
tar -cvjpf bzip2-bin-1.0.5.tar.bz2 bzip2/*
cp /system/bzip2-bin-1.0.5.tar.bz2 $HOME/binaries
```

- **Gzip-1.4**

Descripción: contiene programas para comprimir y descomprimir ficheros (18).

Sitio de descarga: *http://gentoo.uci.cu/gentoo/distfiles/gzip-1.4.tar.gz*

Configuración del paquete:

-Opciones de configuración

```
../gzip-1.4/configure --prefix=/system/gzip/usr/ --bindir=/system/gzip/bin/ \
--sysconfdir=/system/gzip/etc/ --libdir=/system/gzip/usr/lib/ \
--includedir=/system/gzip/usr/include/ --datarootdir=/system/gzip/usr/share/ \
--infodir=/system/gzip/usr/share/info/ --localedir=/system/gzip/usr/share/locale/ \
--mandir=/system/gzip/usr/share/man/ --docdir=/system/gzip/usr/share/doc/gzip/ \
--localstatedir=/system/gzip/var/
```

Construcción:

Se compila el paquete

```
make
```

Se ejecuta el banco de pruebas

```
make check
```

Se instala el paquete

```
make install
```

Se mueven los programas a los lugares especificados por el FHS.

```
mv -v /system/gzip/bin/{gzexe,uncompress,zcmp,zdiff,zegrep} /system/gzip/usr/bin
mv -v /system/gzip/bin/{zfgrep,zforce,zgrep,zless,zmore,znew} /system/gzip/usr/bin
```

Empaquetado y Almacenamiento del código binario:

```
cd /system

tar -cvjpf gzip-bin-1.4.tar.bz2 gzip/*

cp /system/gzip-bin-1.4.tar.bz2 $HOME/binaries
```

- **Tar-1.20**

Descripción: contiene un programa de archivado (18).

Sitio de descarga: *http://gentoo.uci.cu/gentoo/distfiles/tar-1.20.tar.bz2.*

Configuración del paquete:

-Opciones de configuración

```
../tar-1.20/configure --prefix=/system/tar/usr/ --bindir=/system/tar/bin/ \
--sysconfdir=/system/tar/etc/ --libdir=/system/tar/usr/lib/ \
--includedir=/system/tar/usr/include/ --datarootdir=/system/tar/usr/share/ \
--infodir=/system/tar/usr/share/info/ --localedir=/system/tar/usr/share/locale/ \
 --mandir=/system/tar/usr/share/man/ --docdir=/system/tar/usr/share/doc/tar/ \
```

```
--localstatedir=/system/tar/var/ --libexecdir=/system/tar/usr/sbin/
```

Construcción:

Se compila el paquete

```
make
```

Se ejecuta el banco de pruebas

```
make check
```

Se instala el paquete

```
make install
```

Empaquetado y Almacenamiento del código binario:

```
cd /system

tar -cvjpf tar-bin-1.20.tar.bz2 tar/*

cp /system/tar-bin-1.20.tar.bz2 $HOME/binaries
```

- **Lzma-4.65**

Descripción: es el método de compresión por defecto y general del formato 7z. Es muy apropiado para aplicaciones de sistemas embebidos (24)**.**

Sitio de descarga: *http://gentoo.uci.cu/gentoo/distfiles/lzma465.tar.bz2*

Construcción:

Se accede al directorio donde se encuentra el archivo de construcción

```
cd /CPP/7zip/Compress/LZMA_Alone/
```

Se compila y se crean los directorios para instalar el binario

```
make -f makefile.gcc
mkdir /system/lzma/usr
```

```
mkdir /system/lzma/usr/bin
cp /CPP/7zip/Compress/LZMA_Alone/lzma /system/lzma/usr/bin/lzma_alone
```

Empaquetado y Almacenamiento del código binario:

```
cd /system
tar -cvjpf lzma-bin-4.65.tar.bz2 lzma/*
cp /system/lzma-bin-4.65.tar.bz2 $HOME/binaries
```

- **Cpio-2.10-r1**

Descripción: Es un formato de compresión con un algoritmo similar a gzip. Es utilizado en algunos sistemas como formato estándar de su initrd.

Sitio de descarga: *http://gentoo.uci.cu/gentoo/distfiles/cpio-2.10.tar.bz2*

Configuración del paquete:

-Opciones de configuración

```
../cpio-2.10/configure --prefix=/system/cpio/usr/ --bindir=/system/cpio/bin/ \
--sysconfdir=/system/cpio/etc/ --libdir=/system/cpio/usr/lib/ \
--includedir=/system/cpio/usr/include/ --datarootdir=/system/cpio/usr/share/ \
 --infodir=/system/cpio/usr/share/info/ --localedir=/system/cpio/usr/share/locale/ \
--mandir=/system/cpio/usr/share/man/ --docdir=/system/cpio/usr/share/doc/cpio/ \
--localstatedir=/system/cpio/var/ --libexecdir=/system/cpio/usr/sbin/
```

Construcción:

Se compila el paquete

```
make
```

Se ejecuta el banco de pruebas

```
make check
```

Se instala el paquete

```
make install
```

Empaquetado y Almacenamiento del código binario:

```
cd /system
tar -cvjpf cpio-bin-2.10.tar.bz2 cpio/*
cp /system/cpio-bin-2.10.tar.bz2 $HOME/binaries
```

Listado de aplicaciones adicionales.

- **Linux-2.6.32-r7**

Descripción: es el núcleo Linux y utilizado por la mayoría de los sistemas GNU.

Sitio de descarga: *http://gentoo.uci.cu/gentoo/distfiles/linux-2.6.32.tar.bz2*

Configuración del paquete:

-Opciones de configuración:

```
make menuconfig
```

Nota

El comando anterior muestra en la consola un menú que permite seleccionar las opciones con las que se desea construir el núcleo del sistema.

Construcción:

Se compila el paquete

```
make
```

Se crean lo módulos

```
make modules
```

Empaquetado y Almacenamiento del código binario:

```
make tarbz2-pkg
mv -v $NFS /linux/linux-2.6.32.tar.bz2 $HOME/binaries
```

- **Module-Init-Tools-3.5**

Descripción: contiene un conjunto de programas para cargar, insertar y eliminar los módulos del núcleo Linux. (25)

Sitio de descarga: http://gentoo.uci.cu/gentoo/distfiles/module-init-tools-3.5.tar.bz2

Preparación del entorno de construcción del paquete:

Se crea un directorio.

```
mkdir /$NFS/module-init-tools
```

Luego se copia el comprimido del código fuente.

```
cp $SOURCE /module-init-tools-3.5.tar.bz2 /$NFS/module-init-tools
```

Se accede al directorio donde están los archivos descomprimidos.

```
cd /$NFS/ module-init-tools
```

Se descomprime el código fuente.

```
tar -xvjpf module-init-tools-3.5.tar.bz2
```

Se accede al directorio

```
cd module-init-tools-3.5/
```

Se crea el directorio en donde se va a instalar el paquete.

```
mkdir /system/module-init-tools
```

En este paquete es necesario ejecutar las pruebas antes de configurarlo. Las pruebas se realizan de la siguiente forma:

```
./configure
make check
./tests/runtests
make clean
```

Configuración del paquete:

-Opciones de configuración:

```
../module-init-tools/configure --prefix=/system/module-init-tools/usr/ \
--bindir=/system/module-init-tools/usr/bin/ \
--sysconfdir=/system/module-init-tools/etc/ \
 --libdir=/system/module-init-tools/usr/lib/ \
--includedir=/system/module-init-tools/usr/include/ \
 --datarootdir=/system/module-init-tools/usr/share/ \
--datadir=/system/module-init-tools/lib/module-init-tools/ \
--infodir=/system/module-init-tools/usr/share/module-init-tools/ \
--localedir=/system/module-init-tools/usr/share/locale/ \
--mandir=/system/module-init-tools/usr/share/man/ \
--docdir=/system/module-init-tools/usr/share/doc/kbd/ \
--localstatedir=/system/module-init-tools/var/ --enable-zlib
```

Significado de las opciones de configuración:

--enable-zlib

Habilita la librería de comprensión de datos zlib.

Construcción:

Se compila el paquete

```
make
```

Se instala

```
make install
```

Empaquetado y Almacenamiento del código binario:

```
cd /system
tar -cvjpf module-init-tools-bin-3.5.tar.bz2 module-init-tools/*
cp /system/module-init-tools-bin-3.5.tar.bz2 $HOME/binaries
```

Construcción del SOB.

Una vez concluida la construcción de los paquetes se procede a construir el SOB. El primer paso es crear la estructura de directorios del SO utilizando el estándar FHS. Luego se descomprimen e instalan manualmente los paquetes de código binario.

Creación del sistema de directorios:

```
mkdir final_system
cd final_system/
mkdir -pv /{bin,boot,etc/opt,home,lib,mnt,opt}
mkdir -pv /{media/{floppy,cdrom},sbin,srv,var}
install -dv -m 0750 /root
install -dv -m 1777 /tmp /var/tmp
mkdir -pv /usr/{,local/}{bin,include,lib,sbin,src}
mkdir -pv /usr/{,local/}share/{doc,info,locale,man}
mkdir -v /usr/{,local/}share/{misc,terminfo,zoneinfo}
mkdir -pv /usr/{,local/}share/man/man{1..8}
for dir in /usr /usr/local; do
  ln -sv share/{man,doc,info} $dir ; done
mkdir -v /var/{lock,log,mail,run,spool}
mkdir -pv /var/{opt,cache,lib/{misc,locate},local}
```

- Instalación de paquetes y configuración.

La instalación de los paquetes se realizará comenzando por los paquetes básicos del sistema y sus

dependencias.

Primero se declara una variable global que contiene la dirección del sistema que se está construyendo.

```
export FS=$HOME/sistema_final
```

Nota

En algunos casos se brinda la explicación y no los comandos que se deben introducir porque es extensa la cantidad de directorios y archivos contenidos en las direcciones que se especifican.

Después se comienzan a instalar los paquetes.

- **Glibc-2.10.1-r1**

```
cd $HOME/binaries/
tar -xvjpf glibc-bin-2.10.1.tar.bz2
mv –v glibc/usr/bin/{catchsegv,gencat,getconf,getent,iconv,ldd,lddlibc4,locale,\
localedef,pcprofiledump,rpcgen,sprof,tzselect,xtrace} $FS/usr/bin
mv –v glibc/usr/include/* $FS/usr/include/
mv –v glibc/usr/lib/* $FS/usr/lib/
mv -v glibc/usr/sbin/{iconvconfig,nscd,rpcinfo,zdump,zic} $FS/usr/sbin/
mv -v glibc/usr/sbin/{ldconfig,sln} $FS/sbin/
mv -v glibc/usr/etc/* $FS/etc/
```

Se mueve el contenido del directorio **glibc/usr/share/** para el directorio **$FS/usr/share/**, sin modificar los archivos existentes**.**

Luego se borra el directorio de binarios.

```
rm -rf glibc/
```

- **Coreutils-7.5-r1**

```
cd $HOME/binaries/
```

```
tar -xvjpf coreutils-bin-7.5-r1.tar.bz2
mv -v coreutils/bin/* $FS/bin
mv -v coreutils/usr/sbin/chroot $FS/usr/sbin
mv -v coreutils/usr/bin/* $FS/usr/bin
mv -v coreutils/usr/lib/* $FS/usr/lib
mv -v coreutils/usr/share/info/* $FS/usr/share/info
mv -v coreutils/usr/share/locale/* $FS/usr/share/locale
mv -v coreutils/usr/share/man/man1/* $FS/usr/share/man/man1
```

Se borra el directorio de binarios

```
rm -rf coreutils/
```

- **Bash-4.0-p37**

```
cd $HOME/binaries/
tar -xvjpf bash-bin-4.0-p37.tar.bz2
mv -v bash/usr/bin/bash $FS/bin
mv -v bash/usr/bin/bashbug $FS/usr/bin
mv -v bash/usr/share/doc/bash/{bash.html,bashref.html} $FS/usr/share/doc/bash/
mv -v bash/usr/share/info/* $FS/usr/share/info
mv -v bash/usr/share/man/man1/* $FS/usr/share/man/man1/
mv -v bash/usr/share/locale/af/LC_MESSAGES/* $FS/usr/share/locale/af/
mv -v bash/usr/share/locale/bg/LC_MESSAGES/* $FS/usr/share/locale/bg/
mv -v bash/usr/share/locale/ca/LC_MESSAGES/* $FS/usr/share/locale/ca/
mv -v bash/usr/share/locale/cs/LC_MESSAGES/* $FS/usr/share/locale/cs/
mv -v bash/usr/share/locale/de/LC_MESSAGES/* $FS/usr/share/locale/de/
mv -v bash/usr/share/locale/en@boldquot/LC_MESSAGES/* $FS/usr/share/locale/en@enboldquot/
mv -v bash/usr/share/locale/en@quot/LC_MESSAGES/* $FS/usr/share/locale/en@quot/
mv -v bash/usr/share/locale/eo/LC_MESSAGES/* $FS/usr/share/locale/eo/
mv -v bash/usr/share/locale/es/LC_MESSAGES/* $FS/usr/share/locale/es/
mv -v bash/usr/share/locale/et/LC_MESSAGES/* $FS/usr/share/locale/et/
```

```
mv -v bash/usr/share/locale/fr/LC_MESSAGES/* $FS/usr/share/locale/fr/
mv -v bash/usr/share/locale/hu/LC_MESSAGES/* $FS/usr/share/locale/hu/
mv -v bash/usr/share/locale/id/LC_MESSAGES/* $FS/usr/share/locale/id/
mv -v bash/usr/share/locale/ja/LC_MESSAGES/* $FS/usr/share/locale/ja/
mv -v bash/usr/share/locale/lt/LC_MESSAGES/* $FS/usr/share/locale/lt/
mv -v bash/usr/share/locale/nl/LC_MESSAGES/* $FS/usr/share/locale/nl/
mv -v bash/usr/share/locale/pl/LC_MESSAGES/* $FS/usr/share/locale/pl/
mv -v bash/usr/share/locale/pt_BR/LC_MESSAGES/* $FS/usr/share/locale/pt_BR/
mv -v bash/usr/share/locale/ro/LC_MESSAGES/* $FS/usr/share/locale/ro/
mv -v bash/usr/share/locale/ru/LC_MESSAGES/* $FS/usr/share/locale/ru/
mv -v bash/usr/share/locale/sk/LC_MESSAGES/* $FS/usr/share/locale/sk/
mv -v bash/usr/share/locale/sv/LC_MESSAGES/* $FS/usr/share/locale/sv/
mv -v bash/usr/share/locale/tr/LC_MESSAGES/* $FS/usr/share/locale/tr/
mv -v bash/usr/share/locale/vi/LC_MESSAGES/* $FS/usr/share/locale/vi/
mv -v bash/usr/share/locale/zh_TW/LC_MESSAGES/* $FS/usr/share/locale/zh_TW
```

Se borra el directorio de binarios

```
rm -rf bash/
```

- **File-5.03**

```
cd $HOME/binaries/
tar -xvjpf file-bin-5.03.tar.bz2
mv -v file/bin/file $FS/usr/bin/
mv -v file/include/magic.h $FS/usr/include/magic.h
mv -v file/lib/{libmagic.a,libmagic.la,libmagic.so,libmagic.so.1,libmagic.so.1.0.0} $FS/usr/lib
mv -v file/share/man/man1/file.1 $FS/usr/man/man1/
mv -v file/share/man/man3/libmagick.3 $FS/usr/man/man3/
mv -v file/share/man/man4/magic.4 $FS/usr/man/man4/
mv -v file/share/misc/magic.mgc $FS/usr/misc/
```

Se borra el directorio de binarios

```
rm -rf file/
```

- **Inetutils-1.7**

```
cd $HOME/binaries/
tar -xvjpf inetutils-bin-1.7.tar.bz2
mv -v inteutils/bin/{hostname,ping,ping6} $FS/bin
mv -v inteutils/bin/{ftp,logger,rcp,rexec,rlogin,rsh,telnet,tftp,traceroute} $FS/usr/bin
mv -v inteutils/bin/{ifconfig} $FS/sbin
mv -v inteutils/share/info/{dir,inetutils.info} $FS/usr/share/info/
mv -v inteutils/share/man/man1/* $FS/usr/man/man1/
```

Se borra el directorio de binarios

```
rm -rf inetutils/
```

- **Kbd-1.15**

```
cd $HOME/binaries/
tar -xvjpf kbd-bin-1.15.tar.bz2
mv -v kbd/bin/{loadkeys,setfont} $FS/bin/
mv -v  kbd/bin/* $FS/usr/bin/
mv -v kbd/share/man/man1/* $FS/usr/share/man/man1/
cp kbd/share/man/man5/keymaps.5 $FS/usr/share/man/man5/
cp kbd/share/man/man8/* $FS/usr/share/man/man8/
```

Se borra el directorio de binarios

```
rm -rf kbd/
```

- **Shadow-4.1.2.2**

```
cd $HOME/binaries/
tar -xvjpf shadow-bin-4.1.2.2.tar.bz2
mv -v shadow/etc/* $FS/etc/
mv -v shadow/usr/bin/* $FS/usr/bin/
```

```
mv -v shadow/usr/sbin/* $FS/usr/sbin/
```

Luego se mueve el contenido de **shadow/usr/share/** para el directorio similar existente en **$FS/usr/share/** sin borrar los archivos existentes en el último.

```
mv -v shadow/usr/share/locale/cs/LC_MESSAGES/* $FS/usr/share/locale/cs/LC_MESSAGES/
mv -v shadow/usr/share/locale/da/LC_MESSAGES/* $FS/usr/share/locale/da/LC_MESSAGES/
mv -v shadow/usr/share/locale/de/LC_MESSAGES/* $FS/usr/share/locale/de/LC_MESSAGES/
mkdir -v /$FS/usr/share/locale/dz/
mv -v shadow/usr/share/locale/dz/LC_MESSAGES/* $FS/usr/share/locale/dz/LC_MESSAGES/
mv -v shadow/usr/share/locale/el/LC_MESSAGES/* $FS/usr/share/locale/el/LC_MESSAGES/
mv -v shadow/usr/share/locale/es/LC_MESSAGES/* $FS/usr/share/locale/es/LC_MESSAGES/
mv -v shadow/usr/share/locale/eu/LC_MESSAGES/* $FS/usr/share/locale/eu/LC_MESSAGES/
mv -v shadow/usr/share/locale/fi/LC_MESSAGES/* $FS/usr/share/locale/fi/LC_MESSAGES/
mv -v shadow/usr/share/locale/fr/LC_MESSAGES/* $FS/usr/share/locale/fr/LC_MESSAGES/
mv -v shadow/usr/share/locale/gl/LC_MESSAGES/* $FS/usr/share/locale/gl/LC_MESSAGES/
mkdir /$FS/usr/share/locale/he
mv -v shadow/usr/share/locale/he/LC_MESSAGES/* $FS/usr/share/locale/he/LC_MESSAGES/
mv -v shadow/usr/share/locale/hu/LC_MESSAGES/* $FS/usr/share/locale/hu/LC_MESSAGES/
mv -v shadow/usr/share/locale/id/LC_MESSAGES/* $FS/usr/share/locale/id/LC_MESSAGES/
mv -v shadow/usr/share/locale/it/LC_MESSAGES/* $FS/usr/share/locale/it/LC_MESSAGES/
mv -v shadow/usr/share/locale/ja/LC_MESSAGES/* $FS/usr/share/locale/ja/LC_MESSAGES/
mkdir -v /$FS/usr/share/locale/km
mv -v shadow/usr/share/locale/km/LC_MESSAGES/* $FS/usr/share/locale/km/LC_MESSAGES/
mv -v shadow/usr/share/locale/ko/LC_MESSAGES/* $FS/usr/share/locale/ko/LC_MESSAGES/
```

```
mv -v shadow/usr/share/locale/nb/LC_MESSAGES/* $FS/usr/share/locale/nb/LC_MESSAGES/
mkdir -v /$FS/usr/share/locale/ne
mv -v shadow/usr/share/locale/ne/LC_MESSAGES/* $FS/usr/share/locale/ne/LC_MESSAGES/
mv -v shadow/usr/share/locale/nl/LC_MESSAGES/* $FS/usr/share/locale/nl/LC_MESSAGES/
mkdir -v /$FS/usr/share/locale/nn
mv -v shadow/usr/share/locale/nn/LC_MESSAGES/* $FS/usr/share/locale/nn/LC_MESSAGES/
mv -v shadow/usr/share/locale/pl/LC_MESSAGES/* $FS/usr/share/locale/pl/LC_MESSAGES/
mv -v shadow/usr/share/locale/pt/LC_MESSAGES/* $FS/usr/share/locale/pt/LC_MESSAGES/
mv -v shadow/usr/share/locale/pt_BR/LC_MESSAGES/*
$FS/usr/share/locale/pt_BR/LC_MESSAGES/
mv -v shadow/usr/share/locale/ro/LC_MESSAGES/* $FS/usr/share/locale/ro/LC_MESSAGES/
mv -v shadow/usr/share/locale/ru/LC_MESSAGES/* $FS/usr/share/locale/ru/LC_MESSAGES/
mkdir -v $FS/usr/share/man/es
mv -v shadow/usr/share/man/es/* $FS/usr/share/man/es
mv -v shadow/usr/share/man/es/* $FS/usr/share/man/es
```

Se elimina el directorio de los binarios.

```
rm -rf shadow/
```

- **Sysvinit-2.87-r3**

```
cd $HOME/binaries/
tar -xvjpf sysvinit-bin-2.87-r3.tar.bz2
mv -v sysvinit/bin/{mountpoint,sidof} $FS/bin
mv -v sysvinit/sbin/{bootlogd,halt,init,killall5,poweroff,reboot,runlevel,shutdown,
sulogin,telinit} $FS/sbin
mv -v sysvinit/usr/bin/{last,lastb,mesg,utmpdump,wall} $FS/usr/bin
```

```
mv -v sysvinit/usr/include/initreq.h $FS/usr/include
mv -v sysvinit/usr/share/man/man1/* $FS/usr/share/man/man1/
mv -v sysvinit/usr/share/man/man5/* $FS/usr/share/man/man5/
mv -v sysvinit/usr/share/man/man8/* $FS/usr/share/man/man8/
```

Se elimina el directorio de los binarios.

```
rm -rf sysvinit/
```

- **Udev-149**

```
cd $HOME/binaries/
tar -xvjpf udev-bin-149.tar.bz2
mv -v udev/sbin/{udevadm,udevd} $FS/sbin/
```

Se mueve el contenido del directorio **udev/etc** para **$FS/etc/**

Se mueve el contenido del directorio **udev/lib** para **$FS/lib/**

Se copia el contenido del directorio **udev/usr/** para **$FS/usr/**

Se elimina el directorio de los binarios.

```
rm -rf udev/
```

- **Grep-2.5.4-r1**

```
cd $HOME/binaries/
tar -xvjpf grep-bin-2.5.4-r1.tar.bz2
mv -v grep/usr/bin/* $FS/bin
mv -v grep/usr/share/man/man1/* $FS/usr/share/man/man1/
mv -v grep/usr/share/info/* $FS/usr/share/info/
```

Se mueven los archivos que se encuentran dentro del directorio **grep/usr/share/locale/** al directorio **$FS/usr/share/locale/** sin sobreescribir los archivos que se encuentran en este último.

Se elimina el directorio de los binarios.

```
rm -rf grep/
```

- **Sed-4.2**

```
cd $HOME/binaries/
tar -xvjpf sed-bin-4.2.tar.bz2
mv -v sed/usr/bin/* $FS/bin
mv -v sed/usr/share/man/man1/* $FS/usr/share/man/man1/
mv -v sed/usr/share/info/* $FS/usr/share/info/
```

Se mueven los archivos que se encuentran dentro del directorio **sed/usr/share/locale/** al directorio **$FS/usr/share/locale/** sin sobreescribir los archivos que se encuentran en este último.

Se elimina el directorio de los binarios.

```
rm -rf sed/
```

- **Gawk-3.1.6**

```
cd $HOME/binaries/
tar -xvjpf gawk-bin-3.1.6.tar.bz2
mv -v gawk/usr/bin/* $FS/bin/
mkdir -v $FS/usr/libexec/
mv -v gawk/usr/libexec/* $FS/usr/libexec/
mkdir -v $FS/usr/share/awk/
mv -v gawk/usr/share/awk/* $FS/usr/share/awk/
mv -v gawk/usr/share/info/* $FS/usr/share/info/
mv -v gawk/usr/share/* $FS/usr/share/
mv-v gawk/usr/share/man/man1/* $FS/usr/share/man/ man1
```

Se mueven los archivos que se encuentran dentro del directorio **gawk/usr/share/locale/** al directorio **$FS/usr/share/locale/** sin sobreescribir los archivos que se encuentran en este último.

Se elimina el directorio de los binarios.

```
rm -rf gawk/
```

- **Zlib-1.2.3-r1**

```
cd $HOME/binaries/
tar -xvjpf zlib-bin-1.2.3-r1.tar.bz2
mv -v zlib/usr/lib/zlib.a  $FS/usr/lib/
mv -v zlib/usr/lib/* $FS/lib/
mv -v zlib/usr/share/man/man3/zlib.3 $FS/usr/share/man/man3/
```

Se elimina el directorio de los binarios.

```
rm -rf zlib/
```

- **Bzip2-1.0.5-r1**

```
cd $HOME/binaries/
tar -xvjpf bzip2-bin-1.0.5-r1.tar.bz2
mv -v bzip2/bin/{bunzip,bzcat,bzip2} $FS/bin/
mv -v bzip2/bin/{bzcmp,bzdiff,bzegrep,bzfgrep,bzgrep,bzip2recover,bzless,bzmore} $FS/usr/bin/
mv -v bzip2/include/ $FS/usr/include/
mv -v bzip2/lib/libbz2.a $FS/usr/lib/
mv -v bzip2/man/man/man1/* $FS/usr/share/man/man1/
```

Se elimina el directorio de los binarios.

```
rm -rf bzip2/
```

- **Gzip-1.4**

```
cd $HOME/binaries/
tar -xvjpf gzip-bin-1.4.tar.bz2
mv -v gzip/bin/{gunzip,gzip,zcat} $FS/bin/
mv -v gzip/usr/bin/{uncompress} $FS/bin/
mv -v gzip/usr/bin/* $FS/usr/bin/
mv -v gzip/usr/share/info/* $FS/usr/share/info/
mv -v gzip/usr/share/man/man1/* $FS/usr/share/man/man1/
```

Se elimina el directorio de los binarios.

```
rm -rf gzip/
```

- **Tar-1.20**

```
cd $HOME/binaries/
tar -xvjpf tar-bin-1.20.tar.bz2
mv -v tar/bin/tar $FS/bin/
mv -v tar/usr/sbin/rmt FS/usr/sbin/
mv -v tar/usr/share/info/* FS/usr/share/info/
```

Se mueven los archivos que se encuentran dentro del directorio **tar/usr/share/locale/** al directorio **$FS/usr/share/locale/** sin sobreescribir los archivos que se encuentran en este último.

Se elimina el directorio de los binarios.

```
rm -rf tar/
```

- **Lzma-4.65**

```
cd $HOME/binaries/
tar -xvjpf lzma-bin-4.65.tar.bz2
mv -v lzma/usr/bin/lzma_alone $FS/usr/bin/
```

Se elimina el directorio de los binarios.

```
rm -rf lzma/
```

- **Cpio-2.10-r1**

```
cd $HOME/binaries/
tar -xvjpf cpio-bin-2.10-r1.tar.bz2
mv -v cpio/bin/cpio $FS/bin/
mv -v cpio/usr/share/info/* $FS/usr/share/info/
mv -v cpio/usr/share/man/man1/* $FS/usr/share/man/man1/
```

Se mueven los archivos que se encuentran dentro del directorio **cpio/usr/share/locale/** al directorio

$FS/usr/share/locale/ sin sobreescribir los archivos que se encuentran en este último.

Se elimina el directorio de los binarios.

```
rm -rf cpio/
```

- **Module-init-tools-3.5**

```
cd $HOME/binaries/
tar -xvjpf module-init-tools-bin-3.5.tar.bz2
mv -v module-init-tools/usr/bin/lsmod $FS/bin
mv -v module-init-tools/usr/sbin/* $FS/sbin
mv -v module-init-tools/usr/share/man/man5/* $FS/usr/share/man/man5
mv -v module-init-tools/usr/share/man/man8/* $FS/usr/share/man/man8
```

Se elimina el directorio de los binarios.

```
rm -rf module-init-tools/
```

- **Linux-2.6.32**

```
cd $HOME/binaries/
tar -xvjpf linux-2.6.32.tar.bz2
mv -v linux-2.6.32/boot/* $FS/boot/
```

Se mueven los módulos existentes en el directorio **linux-2.6.32/lib/** para el directorio **$FS/lib** del sistema final.

Se elimina el directorio de los binarios.

```
rm -rf linux-2.6.32/
```

- **Aptitude-0.4.11.11**

Se incluyó este gestor de paquete en modo texto en el SOB, para la posterior instalación de paquetes binarios en el sistema.

```
cd $HOME/binaries/aptitude
```

Se mueve el contenido de etc/ para $FS/etc.

Se mueve el contenido de usr/ para $FS/usr.

Se mueve el contenido de var/ para $FS/var.

Luego de haber realizado la construcción e instalación de los paquetes que componen el SOB, se debe entrar al sistema operativo creado. Para ello es preciso abandonar el entorno de construcción (dejar el entorno chroot en el que se había trabajado hasta el momento). Después se siguen los siguientes pasos para proceder a configurar el sistema:

- Se montan los dispositivos:

```
mount –v --bind /dev $FS/dev
```

- Se montan los procesos :

```
mount -t proc none $FS/proc
```

- Se realiza el cambio de raíz:

```
chroot $FS
```

Configuración del sistema base:

Después de realizar la instalación de los paquetes de software en la estructura de directorios elaborada, es preciso crear ciertos archivos y aplicar configuraciones, para lograr que el sistema arranque y funcione correctamente. Las configuraciones serán aplicadas a los paquetes Glibc, Shadow, Sysvinit y Bash; a la consola de Linux y a otros ficheros del sistema.

-Configuración de Glibc:

Se necesita crear el fichero /etc/nsswitch.conf, porque aunque Glibc facilita los valores por defecto, cuando este fichero no se encuentra o está corrupto, los valores por defecto no funcionan bien en un entorno de red. Por esto se crea el fichero /etc/nsswitch.conf ejecutando:

```
cat > /etc/nsswitch.conf << "EOF"
# Begin /etc/nsswitch.conf
passwd: files
group: files
shadow: files
hosts: files dns
networks: files
protocols: files
services: files
ethers: files
rpc: files
# End /etc/nsswitch.conf
EOF
```

También es necesario indicar la zona horaria. Para lo cual se crea el fichero /etc/localtime ejecutando:

```
cp -v --remove-destination /usr/share/zoneinfo/<xxx> /etc/localtime
```

Sustituyendo <xxx> con el nombre de la zona horaria seleccionada.

Se configura además el cargador dinámico, el cual busca por defecto en /lib y /usr/lib las librerías que necesitan los programas cuando se ejecutan. No obstante, si hay librerías que no se encuentran en estos directorios, se deben adicionar al fichero de configuración. Para esto se crea el fichero /etc/ld.so.conf ejecutando lo siguiente:

```
cat > /etc/ld.so.conf << "EOF"
# Begin /etc/ld.so.conf
/usr/local/lib
/opt/lib
# End /etc/ld.so.conf
EOF
```

-Configuración de Shadow

Se habilitan las contraseñas ocultas:

```
pwconv
```

Se habilitan las contraseñas de grupo ocultas:

```
grpconv
```

Se adicionan los usuarios:

```
useradd
```

Se establece la contraseña del usuario root

```
passwd root
```

-Configuración de Sysvinit

Se crea el fichero /etc/inittab

```
cat > /etc/inittab << "EOF"
# Begin /etc/inittab
id:3:initdefault:
si::sysinit:/etc/rc.d/init.d/rc sysinit
l0:0:wait:/etc/rc.d/init.d/rc 0
l1:S1:wait:/etc/rc.d/init.d/rc 1
l2:2:wait:/etc/rc.d/init.d/rc 2
l3:3:wait:/etc/rc.d/init.d/rc 3
l4:4:wait:/etc/rc.d/init.d/rc 4
l5:5:wait:/etc/rc.d/init.d/rc 5
l6:6:wait:/etc/rc.d/init.d/rc 6
ca:12345:ctrlaltdel:/sbin/shutdown -t1 -a -r now
su:S016:once:/sbin/sulogin
1:2345:respawn:/sbin/agetty tty1 9600
2:2345:respawn:/sbin/agetty tty2 9600
3:2345:respawn:/sbin/agetty tty3 9600
```

```
4:2345:respawn:/sbin/agetty tty4 9600
5:2345:respawn:/sbin/agetty tty5 9600
6:2345:respawn:/sbin/agetty tty6 9600
# End /etc/inittab
EOF
```

El guión setclock lee la hora del reloj interno del ordenador, conocido también como reloj BIOS. Si el reloj hardware está establecido a la hora UTC, este guión la convierte a la hora local mediante el fichero / etc/ localtime (que le indica al programa **hwclock** en qué zona horaria se encuentra el usuario). No hay manera de detectar automáticamente si el reloj utiliza UTC o no, así que esto se debe configurar manualmente.

Si no puedes recordar si el reloj hardware está en UTC o no, averígualo ejecutando el comando **hwclock--localtime --show**. Si dicha hora coincide con la de tu reloj, entonces el reloj hardware está a la hora local. Esto mostrará la hora actual según el reloj hardware. Cambia abajo el valor de la variable UTC a 0 (cero) si el reloj hardware no utiliza la hora UTC (18). Se crea el fichero /etc/sysconfig/clock para utilizar la hora que tiene el reloj interno de la computadora, ejecutando:

```
cat > /etc/sysconfig/clock << "EOF"
# Begin /etc/sysconfig/clock
UTC=1
# End /etc/sysconfig/clock
EOF
```

Configurar la consola Linux.

Para configurar la consola de linux se crea el fichero /etc/sysconfig/console de la siguiente forma:

```
cat > /etc/sysconfig/console << "EOF"
# Inicio de /etc/sysconfig/console
UNICODE="1"
KEYMAP="en_us-utf8"
FONT="Verdana-16"
```

```
# Fin de /etc/sysconfig/console
EOF
```

Nota

El fichero /etc/sysconfig/console controla solamente la consola de texto Linux.

-Configuración Bash

Se crea el fichero /etc/profile una vez que se definen los ajustes del idioma.

```
cat > /etc/profile << "EOF"
# Begin /etc/profile
export LANG=<11>_<CC>
# End /etc/profile
EOF
```

Sustituye <*11*> con el código de dos letras del idioma deseado (por ejemplo, “en”) y <*CC*> con el código de dos letras de tu país (por ejemplo, “CU”).

-Configuración de localnet.

El guión localnet se encarga de establecer el nombre de la máquina. Se configura en el fichero /etc/ sysconfig/network. Se crea el fichero /etc/sysconfig/network y se introduce el nombre de la computadora en que se está trabajando, ejecutando el siguiente comando:

```
echo "HOSTNAME=<nombre > /etc/sysconfig/network
```

Se sustituye <nombre > por el nombre de la computadora en que se está trabajando**.**

Luego se configura el fichero /etc/hosts para identificar los nombres de dominio a través de una dirección IP. Se crea ejecutando el siguiente comando:

```
cat > /etc/hosts << "EOF"
# Begin /etc/hosts (network card version)
127.0.0.1 localhost
<192.168.1.1> <HOSTNAME.example.org>
```

```
EOF
```

Cambiar los valores <192. 168. 1. 1> y <HOSTNAME. example. org> por los de la computadora donde se está trabajando.

En el caso de desear configurar una tarjeta de red se procede a crear los ficheros de configuración de la interfaz de red. El siguiente comando crea un fichero ipv4 de ejemplo para el dispositivo eth0:

```
cd /etc/sysconfig/network-devices
mkdir -v ifconfig.eth0
cat > ifconfig.eth0/ipv4 << "EOF"
ONBOOT=yes
SERVICE=ipv4-static
IP=192.168.1.1
GATEWAY=192.168.1.2
PREFIX=24
BROADCAST=192.168.1.255
EOF
```

Nota

Los valores de estas variables se deben cambiar en todos los ficheros por los valores apropiados. Si la variable ONBOOT tiene el valor “yes”, el guión network activará la NIC (Interfaz de Tarjeta de Red) correspondiente durante el arranque del sistema. Si contiene cualquier otro valor, el guión network ignorará la NIC correspondiente y no la activará.

La entrada SERVICE define el método usado para obtener la dirección IP.

La variable GATEWAY contiene la dirección IP de la puerta de enlace por efecto, si hay alguna. Si no, se comenta la variable.

La variable PREFIX debe contener el número de bits usados en la subred. Cada octeto de una dirección IP tiene 8 bits.

Si la máscara de subred es 255.255.255.0, estonces está usando los primeros tres octetos (24 bits) para especificar el número de red.

Si el sistema va a estar conectado a Internet, necesitará algún tipo de resolución de nombres DNS para

resolver los nombres de dominio de Internet a direcciones IP y viceversa. Esto se consigue colocando la dirección IP del servidor DNS, facilitado por el ISP o administrador de red, en / etc/ resolv. conf. Este fichero se crea ejecutando lo siguiente:

```
cat > /etc/resolv.conf << "EOF"
# Begin /etc/resolv.conf
domain <nombre de dominio >
nameserver <dirección IP del servidor primario>
nameserver <dirección IP del servidor secundario>
# End /etc/resolv.conf
```

El fichero /etc/fstab lo utilizan ciertos programas para determinar dónde deben montarse los sistemas de ficheros, en qué orden y cuales deben comprobarse (por fallos de integridad) antes de montarse. Se crea una tabla de ficheros de la siguiente forma:

```
cat > /etc/fstab << "EOF"
# Begin /etc/fstab
# file system     mount-point     type          options          dump   fsck
/dev/<sda1>       /               < reiserfs>   defaults           1    1
/dev/<sda2>       swap            swap          pri=1              0    0
proc              /proc           proc          defaults           0    0
sysfs             /sys            sysfs         defaults           0    0
devpts            /dev/pts        devpts        gid=4,mode=620     0    0
shm               /dev/shm        tmpfs         defaults           0    0
# End /etc/fstab
EOF
```

Reemplazar <*sda1*>, <*sda2*> y <*reiserfs*> con los valores apropiados para tu sistema.

Para obtener un sistema funcional, aún es necesario aplicar ciertas configuraciones e incluir algunos ficheros. Es necesario garantizar que el sistema inicie correctamente. Para cumplir este requerimiento se

necesita un núcleo y su correspondiente initrd. Teniendo en cuenta que el primer requisito fue abordado anteriormente, se explicará cómo se ajustará el initrd al núcleo del sistema.

-Se copia el initrd de un sistema que funcione para un directorio en que se tengan permisos de escritura.

```
cp -v /boot/initrd $FS/tmp/
```

Se accede al directorio

```
cd $FS/tmp/
```

Se crea un directorio y se accede a él.

```
mkdir -v cache
cd cache/
```

Se descomprime el initrd

```
cpio -i < ../initrd
```

Se sustituye el contenido de /lib/modules del initrd por el de /lib/modules del sistema creado.

```
rm -rf lib/modules/
mkdir lib/modules
cp -v /lib/modules/* lib/modules/
```

Luego se comprime el initrd, se copia para el directorio /boot del sistema, se elimina el directorio cache y se hace un cambio de directorio hacia la raíz del sistema.

```
find . -print |cpio -o -H newc |gzip --best >../initrd
cp -v ../initrd $FS/boot/
rm -rf ../cache
cd ../../
```

Luego de concluido este paso se comprime el sistema, obteniéndose de esta forma el basesystem.

```
tar -xvjpf sistema20100426.tar.bz2 .
```

De esta forma queda listo el sistema para ser instalado.

El proceso descrito será de gran utilidad para los miembros del proyecto Nova y para las personas que se

encuentren interesadas en el tema.

Capítulo3. Validación de la propuesta.

En todo sistema o aplicación creada es preciso ejecutar pruebas de software para medir su calidad y validar su funcionalidad. Por tanto, es necesario realizar pruebas al SOB construido con el fin de detectar posibles errores que impidan que el sistema funcione correctamente y soporte las líneas de desarrollo del proyecto Nova.

3.1 Pruebas realizadas.

Las pruebas de software realizadas al SOB construido son: pruebas de unidad y del sistema.

3.1.1 Pruebas de unidad

Las pruebas de unidad evalúan el interior de los sistemas, es decir, el código, su implementación y los posibles errores. Algunos de los paquetes del SOB creado proporcionan un banco de pruebas que permite probar el código. Ejecutar el banco de pruebas para un paquete recién construido es, generalmente, una buena idea, ya que puede proporcionar la seguridad de que todo se ha compilado correctamente.

Algunos bancos de pruebas son más importantes que otros. Por ejemplo, los bancos de pruebas de los paquetes de las herramientas principales – Glibc, Module-init-tools − son de mayor importancia debido a su papel central en el correcto funcionamiento del sistema (18).
Se realizaron los bancos de prueba a los paquetes que los poseen, ejecutando para cada uno:

```
make check
```

obteniendo un resultado satisfactorio. Sin embargo, esto no garantiza que el paquete está totalmente libre de errores. Por esto es necesario realizar pruebas del sistema.

3.1.2 Pruebas del sistema

Están dirigidas a verificar el programa final, después que todos los componentes de software y hardware han sido integrados. Tienen como objetivo ejercitar profundamente el sistema comprobando la integración de la información globalmente, verificando el funcionamiento correcto de las interfaces entre los distintos subsistemas que lo componen y con el resto de sistemas de información con los que se comunica (26).

Para realizar las pruebas del sistema se diseñaron casos de pruebas que permiten verificar que el producto cumple con los requerimientos y se comporta como se desea. Un caso de prueba es un conjunto

de entradas de pruebas, condiciones de ejecución y resultados esperados desarrollados para cumplir un objetivo en particular o una función esperada, es la entidad más simple que siempre es ejecutada como una unidad, desde el comienzo hasta el final. Los casos de pruebas aparecen a continuación.

3.1.2.1 Casos de pruebas.

Caso de prueba 1

CPR-01. Arranque del sistema

Nombre de la prueba	**Arranque del sistema**
Identificador	CPR-01
Descripción	Se realiza con el objetivo de comprobar que el sistema arranca.
Dependencias : núcleo del sistema e initrd	

Paso:	1
Usuario:	Selecciona el sistema a arrancar en el grub.
Sistema:	Inicia el núcleo del sistema y llama a su initrd.
Resultado esperado:	Se descomprima el initrd en la ram y mande a iniciar el sistema.

Caso de prueba 2

CPR-02 Autenticarse

Nombre de la prueba	**Autenticarse**
Identificador	CPR-02
Descripción	Se realiza con el objetivo de comprobar que las personas pueden entrar al sistema.
Dependencias: Inicio correcto del sistema y del paquete shadow.	

Paso:	1
Usuario:	Introduce su usuario y contraseña
Sistema:	Validar que el usuario y la contraseña son correctos.
Resultado esperado:	El usuario accede al sistema

Caso de prueba 3

CPR-03. Comprimir

Nombre de la prueba	**Comprimir**
Identificador	CPR-03
Descripción	Se realiza con el objetivo de comprobar que el sistema soporta distintos modelos de compresión.
Dependencias: Tar, Zlib,Bzip2,Gzip,Tar, Lzma, Cpio	

Paso:	1
Usuario:	Determina el modelo de compresión que desea utilizar y pregunta al sistema si soporta ese modelo.
Sistema:	Devuelve si soporta el modelo seleccionado.
Resultado esperado:	Soporta el modelo
Paso:	2
Usuario:	Selecciona el archivo(s) o directorio(s) que desea comprimir e introduce las opciones de compresión.
Sistema:	Comprime
Resultado esperado:	Compresión satisfactoria.

Caso de prueba

CPR-04 Descomprimir

Nombre de la prueba	**Descomprimir**
Identificador	CPR-04
Descripción	Se realiza con el objetivo de comprobar que el sistema soporta distintos modelos de descompresión.
Dependencias: Tar, Zlib,Bzip2,Gzip,Tar, Lzma, Cpio	

Paso:	1
Usuario:	Pregunta al sistema si soporta el modelo con el cual se encuentra comprimido el archivo.
Sistema:	Devuelve si soporta el modelo seleccionado.
Resultado esperado:	Soporta el modelo
Paso:	2
Usuario:	Selecciona el archivo(s) o directorio(s) que desea descomprimir e introduce las

	opciones de descompresión.
Sistema:	Descomprime
Resultado esperado:	Descompresión satisfactoria.

Caso de prueba 5

CPR-05. Instalar paquete.

Nombre de la prueba	**Instalar paquete**
Identificador	CPR-05
Descripción	Se realiza con el objetivo de comprobar la configuración de la red, su estado, el gestor de paquetes y que los permisos han sido otorgados correctamente a los directorios.
Dependencias: Gestor de paquetes y herramientas del paquete inetutils.	

Paso:	1
Usuario:	Realiza una búsqueda en los repositorios disponibles y selecciona el paquete que desea instalar.
Sistema:	Realiza la búsqueda del paquete a través de un patrón.
Resultado esperado:	Devolución del estado del paquete
Paso:	2
Usuario:	Si el paquete no se encuentra instalado en el sistema se procede a instalarlo
Sistema:	Descarga, descomprime e instala el paquete.
Resultado esperado:	La instalación se realiza satisfactoriamente.

Caso de prueba 6

CPR-06. Desinstalar paquete.

Nombre de la prueba	**Desinstalar paquete**
Identificador	CPR-06
Descripción	Se realiza con el objetivo de comprobar el funcionamiento del gestor de paquete.
Dependencias: Gestor de paquete y que el proceso de instalación se haya realizado en alguna ocasión.	

Paso:	1
Usuario:	Selecciona el paquete que desea desinstalar.
Sistema:	Busca el paquete
Resultado esperado:	Devuelve el estado.
Paso:	2
Usuario:	Si el paquete se encuentra en el sistema se procede a desinstalarlo.
Sistema:	Desinstala el paquete
Resultado esperado:	La desinstalación se realiza correctamente.

Las pruebas realizadas al SOB creado permitieron comprobar que el sistema cumple con las aplicaciones necesarias para que el sistema funcione correctamente y que en caso de desearlo pueden ser añadidas nuevas funcionalidades.

Conclusiones generales

Para el cumplimiento de los objetivos trazados se realizó un estudio en varias distribuciones GNU/Linux, con el propósito de analizar el proceso que siguen para efectuar la construcción de sus SOB y de esta forma poder identificar los elementos necesarios para efectuar la construcción del SOB de Nova. En la mayoría de las distribuciones estudiadas se identificaron pocos elementos del proceso constructivo de su sistema base. Por lo que fue necesario realizar un proceso de ingeniería inversa a algunos de estos sistemas, para obtener otros elementos que permitieran crear un proceso simple.

Con la realización de este trabajo se cumplieron los objetivos propuestos:

- Se definió el proceso de construcción del SOB de Nova; que puede ser utilizado como base común por las líneas de desarrollo del proyecto Nova y como guía para desarrollos similares o para la diversificación de este estudio.
- Se realizó la construcción del SOB de Nova el cual se encuentra listo para realizar personalizaciones.
- Se ejecutaron un conjunto de pruebas que demostraron la calidad del proceso.

Recomendaciones

Los autores de este trabajo recomiendan utilizar este proceso:

- Con fines académicos y productivos.
- Como una guía para realizar construcciones de sistemas operativos.
- Continuar la investigación sobre aspectos que no fueron tratados profundamente en el trabajo.
- Los miembros del proyecto Nova pueden realizar cambios para enfocarlo a cada producto que liberan las líneas de desarrollo
- Ajustar el proceso para arquitecturas de 64 bits.

Referencias bibliográficas

1. **Bella A.; Sanchez J.; Santos.R y Segovia M.A. 2003.** *Libro Blanco del Software Libre en España.* 2003.

2**. Sosa, A. Lianet. 2009.** Sin recetas acabadas, pero con lucidez: Migración al Software Libre. *El Habanero Digital.* 18 de 11 de 2009, pág. 1.

3. **Robbins, Daniel.** Gentoo Linux. [En línea] 2010. [Citado el: 10 de 01 de 2010.] http://www.gentoo.org.

4. **Carrero, David**. fai.informazione.it. [En línea] 26 de 03 de 2007 [Citado el: 10 de 02 de 2010.]

http://fai.informazione.it/p/BBC5ABC7-E6BF-4117-BB4D-025A368A5357/Linux-From-Scratch-es-una-forma-de-instalar-un-sistema-GNULinux-desarrollando-todos-los-componentes.

5. Linuxlandia. [En línea] [Citado el: 25 de 02 de 2010.] http://linuxlandia.es.tl/SliTaz-la-m%E1s-peque%F1a-de-todas-las-minidistribuciones-GNU-Linux.htm

6. **García, Luis Martín.** Estándares abiertos. [En línea] 29 de 07 de 2006. [Citado el: 05 de 05 de 2010.] http://www.estandaresabiertos.com/index.php?option=com_content&task=view&id=58&Itemid=1.

7. **Mora, Beatriz y otros.2009** *Experiencia en transformación de modelos de procesos de negocios desde BPMN a XPDL.*España: s.n.2009

8. **Rodríguez, Andrés. 2008.** *Lenguajes, notaciones y herramientas para el modelado y análisis de procesos.*http://www.gestiopolis.com/administracion-estrategia/lenguajes-notaciones-y-herramientas-en-analisis-de-procesos.htm

9. **SANCHIS, Raquel; POLER, Raúl y ORTIZ, Ángel**. Técnicas para el Modelado de Procesos de Negocio en Cadenas de Suministro. *Inf. tecnol.* [En línea]. 2009, vol.20, n.2 [Citado el: 20 de 04 de 2010], pp.29-40.Disponible en:http://www.scielo.cl/scielo.php?pid=S0718-07642009000200005&script=sci_arttext

10. **Alonso, R. Yisel y otros. 2008.** GestioPolis. [En línea] 10 de 03 de 2008. [Citado el: 09 de 03 de 2010.]http://www.gestiopolis.com/administracion-estrategia/rup-tecnologia-aplicada-al-modelo-de-negocios.htm

11. **Aguilar, H.Violena. 2009.** *Protofase a la ingeniería de requisitos para facilitar la comprensión del negocio a informatizar en el desarrollo de software de gestión.* Ciudad Habana : s.n., 2009.

12. **Barriento, Manuel Sánchez. 2008.** aprendergratis. [En línea] 02 de 11 de 2008. [Citado el: 15 de 02 de 2010.] http://www.aprendergratis.com/introduccion-a-bpmn.html.
Experiencia en transformación de modelos de procesos de negocios desde BPMN a XPDL.

13. **2007.** Scribd. [En línea] 2007. [Citado el: 12 de 03 de 2010.] http://www.scribd.com/doc/2568098/UML-Diagramas-de-actividad.

14**. Peréz J.Juan; Durán A., Ruiz T.Antonio. 2007.** *¿Por qué OMG ha elegido BPMN para modelar de Procesos de Negocios si ya existe UML?* Sevilla : s.n., 2007.

15. **Cejas, Julio.** *Intalio (BPM + BPMN + BPEL + Open Source).* 2007.

16.**2007**. Free Download Manager. [En línea] 2007. [Citado el: 20 de 02 de 2010.] http://www.freedownloadmanager.org/es/downloads/Proceso_Comercial_ARQUITECTO_Visual_%28YO%29_35331_p/.

17. **Valenzuela, Andrés Bustamante. 2008.** Dr. Oscar Barros V. - MBE Universidad de Chile. [En línea] 27 de 10 de 2008. [Citado el: 10 de 02 de 2010.] http://blog.obarros.cl/archives/40.

18. **Beekams, Gerard. 2007.** *Linux From Scratch versión 6.3.* 2007.

19.GNU Operating System. [En línea] 2010. [Citado el: 02 de 03 de 2010.]

http://www.gnu.org/software/coreutils/.

20. GNU Operating System. [En línea] 20 de 11 de 2006 [Citado el: 22 de 03 de 2010.]

http://www.gnu.org/software/bash/.

21. debian. [En línea] 2010 [Citado el: 10 de 03 de 2010.]

http://packages.debian.org/es/squeeze/sysvinit

22. **Beekams, Gerard. 2008.** *Linux From Scratch versión 6.5.* 2008

23. debian. [En línea] 2010 [Citado el: 14 de 03 de 2010.] http://packages.debian.org/es/squeeze/sed

24.COMPRESION.es. [En línea] 26 de 03 de 07. [Citado el: 08 de 04 de 2010.] http://www.compresion.es/formato-7z/.

25.debian. [En línea] 2010 [Citado el: 20 de 03 de 2010.] http://packages.debian.org/es/squeeze/module-

init-tools.

26. **López, Yisel Tornés. 2009.** *Diseño y aplicación de pruebas al producto Primicia, Plataforma de Televisión Informativa.* Habana : s.n., 2009.

Bibliografía consultada

Agramonte R. Alisney; García R.Olivier. *Modelo genérico para diseñar procesos de negocio a partir de las herramientas BPMS.*

Aguilar, H.Violena. 2009. *Protofase a la ingeniería de requisitos para facilitar la comprensión del negocio a informatizar en el desarrollo de software de gestión.* Ciudad Habana : s.n., 2009.

Alonso, Yisel y otros. 2008. GestioPolis. [En línea] 10 de 03 de 2008. [Citado el: 09 de 03 de 2010.] http://www.gestiopolis.com/administracion-estrategia/rup-tecnologia-aplicada-al-modelo-de-negocios.htm.

Barriento, Manuel Sánchez. 2008. aprendergratis. [En línea] 02 de 11 de 2008. [Citado el: 15 de 02 de 2010.] http://www.aprendergratis.com/introduccion-a-bpmn.html.
Experiencia en transformación de modelos de procesos de negocios desde BPMN a XPDL.

Barros, Oscar. 2008. Modelamiento de procesos de negocio con BPMN. 02 de 11 de 2008.

Beekmans, Gerard. 2007. *Linux From Scratch, Versión 6.3.* 2007.

Beekmans, Gerard. 2008. *Linux From Scratch, Versión 6.5.* 2008.

B. Manuel, Sanchéz. 2008. aprendergratis. [En línea] 02 de 11 de 2008. [Citado el: 11 de 03 de 2010.] http://www.aprendergratis.com/introduccion-a-bpmn.html.

Carrero,David.fai.informazione.it. [En línea] 26 de 03 de 2007 [Citado el: 10 de 02 de 2010.] http://fai.informazione.it/p/BBC5ABC7-E6BF-4117-BB4D-025A368A5357/Linux-From-Scratch-es-una-forma-de-instalar-un-sistema-GNULinux-desarrollando-todos-los-componentes.

Cejas, Julio. *Intalio (BPM + BPMN + BPEL + Open Source).* 2007.

debian. [En línea] 2010 [Citado el: 20 de 03 de 2010.] http://packages.debian.org/es/squeeze/module-init-tools.

debian. [En línea] 2010 [Citado el: 14 de 03 de 2010.] http://packages.debian.org/es/squeeze/sed

debian. [En línea] 2010 [Citado el: 10 de 03 de 2010.]

http://packages.debian.org/es/squeeze/sysvinit

Free Download Manager. [En línea] 2007. [Citado el: 20 de 02 de 2010.]

http://www.freedownloadmanager.org/es/downloads/Proceso_Comercial_ARQUITECTO_Visual_%28YO%29_35331_p/.

García, Luis Martín. Estándares abiertos. [En línea] 29 de 07 de 2006. [Citado el: 05 de 05 de 2010.] http://www.estandaresabiertos.com/index.php?option=com_content&task=view&id=58&Itemid=1.

Giraldo,Luis ;Zapata, Yuliana. *HERRAMIENTAS DE DESARROLLO DE INGENIERIA DE SW PARA LINUX.* 2005.

GNU Operating System. [En línea] 20 de 11 de 2006 [Citado el: 22 de 03 de 2010.]

http://www.gnu.org/software/bash/.

GNU Operating System. [En línea] 2010. [Citado el: 02 de 03 de 2010.]

http://www.gnu.org/software/coreutils/.

Jiménez, P. Juan.2009. España: s.n., 29 de 12 de 2009.

Jupamea, L.Dulce. 2008. *Aplicación de la gestión por procesos en una fábrica de muebles.* 2008. **2008.** *Slitaz Scratchbook.* 2008.

López, Yisel Tornés. 2009. *Diseño y aplicación de pruebas al producto Primicia, Plataforma de Televisión Informativa.* Habana : s.n., 2009.

Mora, Beatriz y otros.2009 *Experiencia en transformación de modelos de procesos de negocios desde BPMN a XPDL.*España: s.n.2009

Peréz, J.Juan. 2007 *Notaciones y lenguajes de procesos. Una visión global.* España : s.n.2007

Peréz J.Juan, Durán A.,Ruiz T.Antonio. 2007. *¿Por qué OMG ha elegido BPMN para modelar de Procesos de Negocios si ya existe UML?* Sevilla : s.n., 2007.

Robbins, Daniel. Gentoo Linux. [En línea] 2010. [Citado el: 10 de 01 de 2010.] http://www.gentoo.org.

Rodríguez A., Fernández E.,Piattiani M. 2005. *Hacia la definición de procesos de negocios seguros basados en una arquitectura dirigida por modelos.* Castilla : s.n., 2005.

Rodriguez, Andrés 2008. GestioPolis. [En línea] 16 de 06 de 2008. [Citado el: 22 de 02 de 2010.] http://www.gestiopolis.com/administracion-estrategia/lenguajes-notaciones-y-herramientas-en-analisis-de-procesos.htm.

SANCHIS, Raquel; POLER, Raúl y ORTIZ, Ángel. Técnicas para el Modelado de Procesos de Negocio

en Cadenas de Suministro. *Inf. tecnol.* [En línea]. 2009, vol.20, n.2 [Citado el: 20 de 04 de 2010], pp.29-40.Disponible en:http://www.scielo.cl/scielo.php?pid=S0718-07642009000200005&script=sci_arttext

Scribd. [En línea] 2007. [Citado el: 12 de 03 de 2010.] http://www.scribd.com/doc/2568098/UML-Diagramas-de-actividad

Sosa, Lianet Arias. 2009. Sin recetas acabadas, pero con lucidez: Migración al Software Libre. *El Habanero Digital.* 18 de 11 de 2009, pág. 1.

Valenzuela, Andrés Bustamante. 2008. Dr. Oscar Barros V. - MBE Universidad de Chile. [En línea] 27 de 10 de 2008. [Citado el: 10 de 02 de 2010.] http://blog.obarros.cl/archives/40.

Anexos

Anexol

Tabla 2 Aplicaciones

Aplicaciones		
Stage1	**Stage2**	**Stage3**
app-admin/perl-cleaner	sys-apps/baselayout	app-arch/bzip2
app-admin/python-updater	app-arch/bzip2	app-arch/cpio
dev-lang/perl	app-arch/cpio	app-arch/gzip
dev-lang/python	app-arch/tar	app-arch/tar
app-shells/bash	app-shells/bash	app-arch/xz-utils
app-editors/nano	dev-lang/perl	app-i18n/man-pages-es
net-misc/rsync	dev-lang/python	app-shells/bash
net-misc/wget	net-misc/iputils	dev-perl/Locale-gettext
sys-kernel/linux-headers	net-misc/rsync	dev-util/pkgconfig
app-arch/bzip2	net-misc/wget	net-misc/iputils
app-arch/tar	sys-apps/coreutils	net-misc/openssh
app-arch/gzip	sys-apps/debianutils	net-misc/rsync
app-misc/pax-utils	sys-apps/diffutils	net-misc/wget
dev-libs/libxml2	sys-apps/file	sys-apps/baselayout
dev-libs/expat	sys-apps/findutils	sys-apps/busybox
dev-libs/popt	sys-apps/gawk	sys-apps/coreutils
virtual/editor	sys-apps/grep	sys-apps/diffutils
virtual/init	sys-apps/kbd	sys-apps/file
virtual/libiconv	sys-apps/net-tools	sys-apps/findutils
sys-devel/libtool	sys-apps/portage-2.0.51.22	sys-apps/gawk
sys-devel/automake	sys-process/procps	sys-apps/grep
sys-devel/autoconf	sys-process/psmisc	sys-apps/help2man
sys-devel/libperl	sys-apps/sed	sys-apps/kbd
sys-devel/gcc	sys-apps/shadow	sys-apps/man
sys-devel/gettext	sys-apps/texinfo	sys-apps/man-pages
sys-devel/binutils	sys-apps/which	sys-apps/module-init-tools
sys-devel/bison	sys-devel/autoconf	sys-apps/net-tools
sys-devel/binutils	sys-devel/automake	sys-apps/portage
sys-devel/make	sys-devel/binutils	sys-apps/sed
sys-devel/flex	sys-devel/bison	sys-apps/shadow
sys-devel/patch	sys-devel/flex	sys-apps/texinfo
sys-devel/gcc-config	sys-devel/gcc	sys-apps/util-linux
sys-devel/autoconf-wrapper	sys-devel/gnuconfig	sys-apps/which
sys-devel/automake-wrapper	sys-devel/libtool	sys-devel/autoconf

sys-devel/gnuconfig sys-devel/m4 sys-libs/glibc sys-libs/timezone-data sys-libs/zlib sys-libs/ncurses sys-apps/portage sys-apps/diffutils sys-apps/man-pages sys-apps/baselayout sys-apps/man sys-apps/less sys-apps/coreutils sys-apps/groff sys-apps/file sys-apps/sed sys-apps/gawk sys-apps/net-tools sys-apps/grep sys-apps/sandbox sys-apps/findutils sys-apps/sysvinit sys-apps/texinfo Lugar de donde se obtuvo: *http://distro.ibiblio.org/pub/linux/distributions/gentoo/releases/x86/2008.0/stages*	sys-devel/m4 sys-devel/make sys-devel/patch sys-fs/e2fsprogs sys-libs/ncurses sys-libs/readline sys-libs/zlib virtual/dev-manager virtual/editor virtual/gzip virtual/libc virtual/man virtual/modutils virtual/os-headers virtual/pager virtual/portage virtual/ssh Lugar de donde se obtuvo: Sistema de Gentoo en la siguiente dirección: usr/portage/profiles/base/package	sys-devel/automake sys-devel/binutils sys-devel/bison sys-devel/flex sys-devel/gcc sys-devel/gnuconfig sys-devel/automake-wrapper sys-devel/libtool sys-devel/m4 sys-devel/make sys-devel/patch sys-fs/e2fsprogs sys-fs/udev sys-kernel/linux-headers sys-libs/glibc sys-libs/gpm sys-libs/ncurses sys-libs/readline sys-libs/zlib sys-process/psmisc sys-process/procps virtual/editor virtual/pager Estas aplicaciones se obtuvieron luego de hacer un chroot al stage3 y aplicar el comando emerge--pretend system.

AnexoII

Fichas del proceso

Tabla 3 Ficha del proceso de construcción del SOB de Nova

Ficha de procesos		
Id: 1	**Nombre del proceso:** Construcción del SOB de Nova	**Responsables:** Arquitecto y desarrollador.

Misión: Crear una base común para las líneas de desarrollo del proyecto Nova				
Síntesis del proceso				
Flujo Normal del Proceso				
Disparador/ Proveedor/ Cliente	**Entrada**	**Actividad**		**Salida**
		Id	**Nombre**	
Arquitecto		1	Definir el sistema anfitrión	
Arquitecto		2	Ajustar el sistema anfitrión	
Arquitecto		3	Definir paquetes del SOB	Listado de pcf[9] a procesar
Desarrollador	Listado de pcf a procesar	4	Descargar paquete de código fuente	Paquete de cf [10]
Desarrollador	Paquete de cf	5	Descomprimir el paquete de cf	
Desarrollador		6	Compilar paquete	Código binario
Desarrollador	Código binario	7	Instalar código binario	
Desarrollador		8	Empaquetar código binario	Paquete de código binario
Desarrollador	Paquete de código binario	9	Almacenar pcb[11].	Conjunto de pcb almacenados
Desarrollador	Conjunto de pcb almacenados	10	Crear el paquete basesystem	

[9] ***Pcf:*** *Paquete de código fuente.*
[10] ***Cf:*** *Código fuente.*
[11] ***Pcb:*** *Paquete de código binario.*

Tabla 4 Flujo alterno aplicar parches

Flujos alternos del proceso : Construcción del sistema operativo base de Nova				
Actividad a partir de la cual se genera el flujo: 5 **Alternativa a seguir**: Aplicar parches				
Disparador/ Proveedor/ Cliente	**Entrada**	**Actividad**		**Salida**
		Id	**Nombre**	
Desarrollador		5.a	Descargar o crear parche(s)	Parche(s)
Desarrollador	Parche(s)	5.a.1	Aplicar parche(s)	
Alternativa a seguir: No aplicar parches				
Se compila el paquete.				

Tabla 5 Flujo alterno quedan paquetes por procesar

Flujos alternos del proceso : Construcción del sistema operativo base de Nova				
Actividad a partir de la cual se genera el flujo: 9 **Alternativa a seguir:** Quedan paquetes por procesar.				
Disparador/ Proveedor/ Cliente	**Entrada**	**Actividad**		**Salida**
		Id	**Nombre**	
Desarrollador		4	Descargar paquete de código fuente.	
Alternativa a seguir: No quedan paquetes por procesar.				

Se crea el paquete basesystem y se termina el proceso.

Tabla 6 Descripción de las entradas y salidas

Descripción de las Entradas/Salidas					
Nombre	**Estado**	**Entrada**	**Salida**	**Id de la actividad con la que se relaciona**	**Responsable**
Listado de pcf a procesar	Creado		X	3	Arquitecto
Listado de pcf a procesar	Consultado	X		4	Desarrollador
Paquete de cf	Descargado		X	4	Desarrollador
Paquete de cf	Consultado	X		5	Desarrollador
Parche(s)	Descargado o creado		X	5.a	Desarrollador
Parche(s)	Consultado	X		5.a.1	Desarrollador
Código binario	Compilado		X	6	Desarrollador
Código binario	Consultado	X		7	Desarrollador
Paquete de código binario	Creado		X	8	Desarrollador

Paquete de código binario	Consultado	X		9	Desarrollador
Conjunto de pcb almacenados	Creado		X	9	Desarrollador
Conjunto de pcb almacenados	Consultado	X		10	Desarrollador

Glosario de términos

Banco de pruebas: Es una plataforma para la experimentación de proyectos. Brindan una forma de comprobación rigurosa, transparente y repetible de teorías científicas, elementos computacionales, y otras nuevas tecnologías.

Código fuente: Puede definirse como:

- Un conjunto de líneas que conforman un bloque de texto, escrito según las reglas sintácticas de algún lenguaje de programación destinado a ser legible por humanos.
- Un programa en su forma original, tal y como fue escrito por el programador, no es ejecutable directamente por el computador, debe convertirse en lenguaje de máquina mediante compiladores, ensambladores o intérpretes.

Normalmente está destinado a ser traducido a otro código, llamado código objeto, ya sea lenguaje máquina nativo para ser ejecutado por una computadora o código byte para ser ejecutado por un intérprete.

Dependencias del paquete: Es cuando un paquete de software depende del contenido de otro ya sea para compilarse, instalarse o ejecutarse.

Dispositivos: Cualquier elemento que se puede unir al sistema mediante algún medio de conexión.

Empaquetar: Operación que permite que grupo de ficheros (o uno sólo) se incluyan dentro de otro fichero, ocupando así menos espacio. El empaquetado es similar a la compresión de ficheros. La diferencia entre empaquetado y compresión es la herramienta con que se realiza la operación. Por ejemplo, la herramienta *tar se utiliza para empaquetar*, mientras que la herramienta *zip o gzip -WinZip- se utiliza para comprimir*.

Enlazar: Es una instrucción que conecta una parte de un programa de software, un elemento o una lista de ellos, a otro programa o lista.

Gestor de paquete: es una colección de herramientas que sirven para automatizar el proceso de instalación, actualización, configuración y eliminación de paquetes de software. El término se usa comúnmente para referirse a los gestores de paquetes en sistemas UNIX. Algunos de los sistemas de gestión de paquetes más avanzados tienen la capacidad de desinstalar los paquetes recursivamente o en

cascada, de forma que se eliminan todos los paquetes que dependen del paquete a desinstalar y todos los paquetes de los que el paquete a desinstalar depende, respectivamente.

Initrd: Es un pequeño sistema de archivos que el núcleo puede cargar en la memoria Ram. Proporciona un entorno Linux mínimo que habilita la ejecución de programas antes de que se monte el sistema de la raíz.

Instalación: Término informático que se refiere a la actualización del sistema operativo con nuevos paquetes que pueden ser de software, documentación, código fuente, bibliotecas.

Kernel: Ver definición de Núcleo.

Librería: Es una colección de subrutinas o clases utilizados para desarrollar software.

Librerías compartidas: Son librerías que ofrecen algún tipo de intercambio, permitiendo ser utilizadas por varios programas al mismo tiempo.

Librerías estáticas: Es un conjunto de rutinas que se copian en una aplicación de destino por el compilador, enlazador, o una carpeta, produciendo archivos de objetos y un archivo ejecutable independiente.

Módulos del núcleo: Son unidades de programa discretas e identificables con respecto a la compilación, la combinación con otras unidades y la carga.

Montar: Es una línea de comandos Unix que instruye al sistema operativo de que un sistema de archivos está listo para usarse, y lo asocia con un punto en particular en el sistema operativo.

Núcleo: En informática, el **núcleo,** también conocido en español con el anglicismo *kernel*, de raíces germánicas como *kern*, es la parte fundamental de un sistema operativo. Es el software responsable de facilitar a los distintos programas acceso seguro al hardware de la computadora o en forma más básica, es el encargado de gestionar recursos, a través de servicios de llamada al sistema. Es la parte del sistema operativo responsable de mantener la fluidez de datos entre los discos duros, la memoria, las impresoras, la pantalla de video y todas las partes que se hallan unidas al mismo.

Paquete de software: Un paquete de software es una serie de programas que se distribuyen conjuntamente. Algunas de las razones para ello suelen ser que el funcionamiento de cada uno complementa o requiere a los demás, que sus objetivos están relacionados o como estrategia de mercadotecnia.

Muchos sistemas operativos modernos emplean sistemas de control de paquetes que permiten que el administrador del sistema instale o desinstale estos, sin que en ningún momento queden programas instalados que no funcionen por falta de otros incluidos en su paquete. El sistema de control de paquetes usualmente también se ocupa de mantener las dependencias entre paquetes: Si algún paquete depende de otro, el sistema se encarga de instalarlo.

Shell: Es una aplicación por la cual un usuario puede comunicarse con el sistema operativo en modo texto.

Sistema base: Prototipo funcional del sistema operativo, sin interfaz gráfica y solamente con los programas mínimos necesarios para garantizar la instalación de paquetes.

Sistema embebido: Un sistema embebido o empotrado es un sistema de computación diseñado para realizar una o algunas funciones dedicadas, frecuentemente en un sistema de computación en tiempo real.

Sistema operativo: Un sistema operativo es un software de sistema, es decir, un conjunto de programas de computadora destinado a permitir una administración eficaz de sus recursos. Comienza a trabajar cuando se enciende el computador, y gestiona el hardware de la máquina desde los niveles básicos, permitiendo también la interacción con el usuario.

Variables de entorno: Las variables de entorno son un conjunto de valores dinámicos que normalmente afectan el comportamiento de los procesos en una computadora.