

Universidad de las Ciencias Informáticas

Facultad 1



**Herramienta informática para la creación de
personalizaciones de la distribución cubana GNU/Linux**

Nova

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

Autor

Sulema Hernández Garrido

Tutores

M Sc. Juan Manuel Fuentes Rodríguez

M Sc. Alicia Reina Senra Mugica

La Habana, abril del 2020



“ El tema relativo al conocimiento y la tecnología es de especial relieve en nuestra agenda, porque en él abordamos los problemas que deciden, en buena medida, el futuro de nuestros países”

La Habana, 19 de septiembre de 1999.

Fidel Castro Ruz.

Declaración de autoría

Declaro por este medio que yo Sulema Hernández Garrido, con carné de identidad 96091208778 soy la autora principal del trabajo titulado Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova y autorizo a la Universidad de las Ciencias Informáticas a hacer uso de la misma en su beneficio, así como los derechos patrimoniales con carácter exclusivo.

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

Firma de la autora

Sulema Hernández Garrido

Firma del tutor

Msc. Juan Manuel Fuentes Rodríguez

Firma de la tutora

Msc. Alicia Reina Senra Mugica

Datos de contacto

Msc. Juan Manuel Fuentes Rodríguez

Universidad de las Ciencias Informáticas, La Habana, Cuba

Correo: jfuentesr@uci.cu

Msc. Alicia Reina Senra Mugica

Universidad de las Ciencias Informáticas, La Habana, Cuba

Correo: alicia@uci.cu

Agradecimientos:

Este trabajo ha sido una gran carrera donde han influido muchas personas:

- *Mis padres, ellos han sido siempre mi guía brindando amor, dedicación y apoyo incondicional, hoy en día estoy celebrando este triunfo gracias a ellos.*
- *A mi familia, en general a mi hermana y mis sobrinos que han dado siempre amor y dedicación, a mis tíos y a mi abuela Tita que siempre me despedían con orgullo cuando regresaba a la beca. A mi abuela Eva la cual quiero mucho.*
- *Mi novio Ivan, por estar conmigo en todo momento, por la dedicación, paciencia y el cariño, además pasar juntos todos los momentos que vivimos en la universidad. A su mamá Ileana que me acogió como a su hija y nunca faltaron los consejos y dedicación de su parte.*
- *A mis amigos de la universidad; a mi grupo, junto pasamos muy buenos momentos de fiesta, guardias, estudios para las pruebas son momentos únicos los cuales recordaré al pasar de los años.*
- *A mis tutores, Juan Manuel Fuentes y Alicia Senra Mugica que han dedicado horas de su tiempo y esfuerzo para atenderme y ayudarme en todo lo que necesitaba.*
- *Al claustro de profesores de la Facultad 1, gracias a ellos me formé como una profesional en esta casa de altos estudios.*
- *A la Universidad de las Ciencias Informáticas por acogerme como una docente más entre sus aulas, beca, gimnasios y centros culturales, de este lugar me llevo los mejores momento de mi vida, momentos que solo se viven una vez como muchos dicen los cuales te forman ante la sociedad como todo un ingeniero en Ciencias Informáticas.*

Dedicatoria:

A mis padres, todo este esfuerzo ha sido por ellos, querer reflejar el fruto de su educación y sacrificio.

Resumen

La Distribución Cubana GNU/Linux Nova es el sistema operativo seleccionado para ser desplegado en los Organismos de Administración Central del Estado por tener grandes capacidades de procesamiento y administración de sus recursos, manteniendo siempre como pilar importante la seguridad. Este sistema operativo presenta una versión para Escritorios a la que frecuentemente se le realizan personalizaciones que consumen mucho tiempo debido a que se desarrolla de forma manual. El objetivo del presente trabajo de diploma es desarrollar una herramienta informática que permita mejorar la creación de personalizaciones de la distribución cubana GNU/Linux Nova. Para guiar el proceso de construcción de la propuesta de solución se utilizó la metodología de desarrollo de software Variación de AUP para la UCI y en la implementación se emplearon tecnologías y herramientas de software libre. La evaluación de la propuesta de solución se realizó mediante la aplicación de pruebas que garantizan el correcto funcionamiento de la herramienta y demostraron la satisfacción del cliente hacia el sistema desarrollado. Al culminar la presente investigación se obtuvo una herramienta informática que permite mejorar la creación de personalizaciones de la distribución cubana GNU/Linux Nova.

Palabras claves: GNU/Linux Nova, herramienta informática, personalizaciones, sistema operativo.

Índice

INTRODUCCIÓN	11
CAPÍTULO1. FUNDAMENTACIÓN TEÓRICA DE LA HERRAMIENTA INFORMÁTICA PARA LA CREACIÓN DE PERSONALIZACIONES DE LA DISTRIBUCIÓN CUBANA GNU/LINUX NOVA.	15
1.1 Introducción	15
1.2 Conceptos y definiciones asociados al dominio del problema.....	15
1.3 Estructura de un ISO de GNU/Linux	16
1.3.1 Herramienta de compresión de sistemas operativos.....	18
1.3.2 Gestor de arranque de los ISOS de las distribuciones de GNU/Linux	18
1.3.3 Herramientas para el uso de UEFI/GPT en los ISOS de las distribuciones de GNU/Linux.....	19
1.4 Análisis de sistemas homólogos	21
1.4.1 Novo Builder	21
1.4.2 Distroshare Ubuntu Imager	21
1.4.3 Ubuntu Customized Kit.....	22
1.4.4 Sistema de Construcción de Personalizaciones de Nova	22
Resultados del estudio de sistemas homólogos.....	22
1.5 ¿Cómo se hacen los ISOS de Nova?.....	23
1.6 Metodología de desarrollo de software	24
1.7 Lenguajes y herramientas para el modelado de la propuesta de solución	25
1.7.1 Lenguajes de Modelado	25
1.7.2 Herramienta de modelado	26
1.7.3 Herramientas y tecnologías de implementación.....	26
1.7.4 Lenguaje de programación	26
1.7.5 Entorno de Desarrollo Integrado.....	27
1.7.6 Control de versiones.....	28
1.8 Conclusiones del Capítulo.....	28
CAPÍTULO 2: ANÁLISIS Y DISEÑO DE LA HERRAMIENTA INFORMÁTICA PARA LA CREACIÓN DE PERSONALIZACIONES DE LA DISTRIBUCIÓN CUBANA GNU/LINUX NOVA.....	30
Introducción	30
2.1 Modelo del negocio.....	30
2.1.1 Modelo Conceptual	30

2.1.2 Diccionario de datos de personalización	31
2.1.3 Descripción de proceso de negocio	32
2.1.4 Diagrama de procesos del negocio personalización de Nova.	32
2.2 Propuesta de solución	33
2.3 Requisitos	35
2.3.1 Fuentes de obtención de requisitos	35
2.3.2 Técnicas de identificación de requisitos.....	36
2.3.4 Especificación de requisitos de software	36
2.3.5 Descripción de requisitos de software por proceso.....	38
2.4 Análisis y Diseño	39
2.4.1 Modelado del diseño	39
2.4.2 Patrones de Diseño.....	40
2.4.3 Diseño arquitectónico	41
2.5 Conclusiones del Capítulo.....	43
CAPÍTULO 3: IMPLEMENTACIÓN, PRUEBAS Y EVALUACIÓN DE LA HERRAMIENTA INFORMÁTICA PARA LA CREACIÓN DE PERSONALIZACIONES DE LA DISTRIBUCIÓN CUBANA GNU/LINUX NOVA	44
3.1 Introducción	44
3.1.1 Implementación.....	44
3.1.2 Estándares de codificación	44
3.1.3 Diagrama de Componentes	46
3.1.4 Diagrama de Despliegue.....	47
3.2 Pruebas de software.....	47
3.2.1 Pruebas a realizar	48
3.2.2 Métodos de prueba	48
3.2.3 Técnicas de prueba.....	49
3.3 Aplicación de las pruebas de software	49
3.3.1 Pruebas internas.....	49
3.4 Evaluación del objetivo de la investigación	54
3.4.1 Resultados de la técnica de ladov	55
3.5 Conclusiones del Capítulo.....	59
CONCLUSIONES GENERALES.....	60
ANEXOS.....	74

Anexo 1: Entrevista realizada a especialistas del proyecto Nova.....74

Anexo 2: Guía de observación para el proceso de creación de personalizaciones de la distribución cubana GNU/Linux Nova.74

Anexo 3: Diseño de Caso de Prueba: Instalar Paquetes.....75

Anexo 4: Diseño de Caso de Prueba: Ejecutar Hooks.....76

Anexo 5: Caso de Prueba. Ruta Básica 277

Anexo 6: Caso de Prueba. Ruta Básica 378

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

INTRODUCCIÓN

Un sistema operativo es el software principal o conjunto de programas de un sistema informático. Gestiona los recursos de hardware y provee servicios a los programas de aplicación de software. Sus funciones básicas son administrar recursos, coordinar el hardware, organizar archivos y directorios en dispositivos de almacenamiento (Piñeiro, 2015).

Los sistemas operativos que se desarrollan tienen un enfoque específico, donde son adaptados para hardware determinado; o genérico que posibilitan un mayor soporte de hardware que pueden ser utilizados en varios dispositivos, incluyendo no sólo computadoras tradicionales sino también *Tablet* y celulares (Jiménez, 2015).

GNU/Linux es un ejemplo de sistema operativo genérico, se ha caracterizado por su estabilidad de operación, velocidad, seguridad como pilar importante y capacidad para la administración eficiente de los recursos de la computadora como son la memoria, el poder de la unidad central de procesamiento y espacio en disco (EPSILONMAG, 2017).

Todas las características antes mencionadas han propiciado que se hayan reproducido disímiles distribuciones de GNU/Linux. Además, brindan a sus usuarios la oportunidad de crear un sistema operativo personalizable de software libre, sin necesidad de costear sus actualizaciones, permitiendo la libertad de ejecutar, copiar, cambiar y mejorar el software. Llevándolo así a diferenciarse del resto de los sistemas operativos del mundo, propiciando su uso a varias esferas de la sociedad y convirtiéndose en una alternativa a nivel mundial (EPSILONMAG, 2017).

En América Latina existen países como Venezuela, Brasil y Bolivia que han hecho del software libre y sus distribuciones, proyectos bandera en el desarrollo científico y tecnológico. Estos países coinciden ideológicamente en su oposición a la privatización, a favor de la autonomía tecnológica y el ahorro económico (Hipertextual, 2015).

Cuba en aras de garantizar la soberanía tecnológica y la informatización del país lleva a cabo el proceso de la migración a software libre donde la Distribución Cubana GNU/Linux Nova, es el sistema operativo seleccionado para ser desplegado en las estaciones de trabajo de los Organismos de la Administración Central del Estado (OACE) (Fuentes, 2018).

Dicha distribución es desarrollada por el proyecto Nova, perteneciente al Centro de Software Libre (CESOL) de la Universidad de las Ciencias Informáticas. Desde sus inicios ha apostado su desarrollo en la soberanía tecnológica, la seguridad y la sostenibilidad de la nación, permitiendo interactuar con un

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

sistema operativo que propicie la adaptabilidad del cliente. A pesar de ser un proyecto relativamente joven, ha alcanzado importantes resultados en la personalización de sistemas operativos de GNU/Linux, ejemplos de ellos son: NOVA NAS¹, NOVA UNIFICADO², NOVA CLONIX³, NOVA SPEX⁴ (NOVA, 2019).

Mediante una entrevista realizada a especialista del centro CESOL (ver Anexo 1), se evidencia que en la actualidad se cuenta con sistemas y herramientas informáticas que permiten la creación de las distribuciones cubanas GNU/Linux Nova, no así para la creación de personalizaciones, donde el especialista tiene que llevar a cabo un proceso manual de interactuar con la computadora para que se ejecuten opciones y configuraciones para su desarrollo, esto trae como consecuencias pérdida de tiempo y recursos.

Partiendo de la problemática anterior se representa el siguiente **problema de la investigación**: ¿cómo desarrollar una herramienta informática que permita mejorar la creación de personalizaciones de la Distribución Cubana GNU/Linux Nova?, para orientar la investigación se identifica como **objeto de estudio**: la creación de personalizaciones de las Distribuciones GNU/Linux, y para enmarcar el alcance del mismo se cuenta con el **campo de acción**: la creación de personalizaciones de la Distribución Cubana GNU/Linux Nova.

Para realizar la investigación se trazó como **objetivo general**: Desarrollar una herramienta informática que permita mejorar la creación de personalizaciones de la Distribución Cubana GNU/Linux Nova.

Para guiar el proceso investigativo se plantean las siguientes **preguntas científicas**:

1. ¿Cuáles son los principales referentes teórico-metodológicos que sustentan el desarrollo de la herramienta informática para la creación de personalizaciones de la Distribución Cubana GNU/Linux Nova?
2. ¿Cuáles metodologías, tecnologías y herramientas, conforman el ambiente de desarrollo de la herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova?

¹ NOVA NAS- Personalización de Nova Servidor para la administración centralizada de un servidor de almacenamiento conectado a la red

² NOVA UNIFICADO- Personalización de Nova Servidor para la administración centralizada de servicios telemáticos.

³ NOVA CLONIX- Personalización de Nova Servidor para la clonación de imágenes, particiones y discos en la fábrica

⁴ NOVA SPEX- Personalización de Nova Servidor para realizar pruebas de envejecimiento a componentes de hardware en la fábrica

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

3. ¿Cómo implementar los artefactos a partir del análisis y diseño de la herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova?
4. ¿Qué pruebas aplicar en la evaluación de la herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova?

Con el propósito de dar cumplimiento a las preguntas científicas planteadas se definen las siguientes **tareas de investigación:**

1. Elaboración del marco teórico-metodológico para la propuesta de desarrollo de la herramienta informática para la creación de personalizaciones de la Distribución Cubana GNU/Linux Nova.
2. Análisis y diseño de la herramienta informática para la creación de personalizaciones de la Distribución Cubana GNU/Linux Nova
3. Implementación de la herramienta informática para la creación de personalizaciones de la Distribución Cubana GNU/Linux Nova.
4. Evaluación de la herramienta informática para la creación de personalizaciones de la Distribución Cubana GNU/Linux Nova.

Para el desarrollo de la investigación se utilizaron los siguientes **métodos de la investigación científica:**

Métodos teóricos:

- **Analítico-Sintético:** se utiliza para la realización del estudio y el análisis de un grupo de herramientas de personalización de sistemas operativos, posibilitando la comprensión de sus características generales y las relaciones esenciales entre ellas.
- **Deductivo-Inductivo:** permite llevar a cabo un procedimiento que a partir de conocimientos generales se concluyeron casos particulares por un razonamiento lógico.
- **Modelación:** se utiliza para modelar los diagramas de requisitos, análisis y diseño e implementación en la construcción de la herramienta de creación de personalizaciones de Nova.
- **Histórico Lógico:** permite conocer la evolución de las herramientas para la creación de personalizaciones de Nova, así como los elementos que la caracterizan.

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

Métodos Empíricos:

- **Observación:** se utiliza para analizar cómo se realiza actualmente la creación de personalizaciones de la Distribución Cubana GNU/Linux Nova, obteniendo conocimientos que serían usados en la implementación.
- **Entrevista:** se utiliza para adquirir conocimientos importantes sobre las características del proceso de desarrollo de personalizaciones de Nova.

El presente trabajo está compuesto por: resumen, introducción, tres capítulos, conclusiones, recomendaciones, referencias bibliográficas y anexos.

Capítulo1. Fundamentación teórica de la herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova: se presentan los conceptos fundamentales, la definición de la metodología de desarrollo de software que se utiliza y se realiza la descripción de la propuesta de solución. Además, se describen las principales herramientas que se utilizan, lenguajes de programación y tecnologías informáticas con que se implementa la herramienta informática para la creación de personalizaciones de GNU/Linux Nova.

Capítulo2. Análisis y Diseño de la herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova: se realiza el análisis y diseño de la propuesta de solución haciendo uso de la metodología de desarrollo seleccionada. Además, se especifica los requisitos funcionales y no funcionales que requieren la herramienta para su desarrollo y la arquitectura utilizada.

Capítulo3. Implementación, pruebas y evaluación de la herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova: se documenta el proceso de implementación de la herramienta informática para la creación de personalizaciones de Nova a partir de los resultados del Análisis y Diseño, además de elaborar y aplicarle las pruebas al sistema.

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

CAPÍTULO 1. FUNDAMENTACIÓN TEÓRICA DE LA HERRAMIENTA INFORMÁTICA PARA LA CREACIÓN DE PERSONALIZACIONES DE LA DISTRIBUCIÓN CUBANA GNU/LINUX NOVA.

1.1 Introducción

En el presente capítulo se presentan los conceptos fundamentales al tema en cuestión, el estudio sobre las principales herramientas de personalizaciones que existen de la distribución GNU/Linux, así como la metodología de desarrollo de software a utilizar y las distintas herramientas y tecnologías que serán empleadas en el desarrollo de la solución.

1.2 Conceptos y definiciones asociados al dominio del problema

Distribución de GNU/Linux

Según el autor del libro *The Debian administrator's Handbook* de Raphael Hertzog y otros autores donde plantean que: un sistema operativo basado en una distribución de GNU/Linux puede definirse como una distribución de software basada en el núcleo Linux que incluye determinados paquetes de software para satisfacer las necesidades de un grupo específico de usuarios. Las mismas pueden contener o no controladores privativos (Hertzog, et al., 2015).

Manuel Alejandro Sánchez en su tesis de grado plantea, una distribución de GNU/Linux permite la posibilidad de emplearlo o modificarlo para que sea utilizado con un determinado uso u objetivo específico, y esto se debe a que se puede contar con su código fuente. A estas modificaciones que se le realizan a una distribución determinada con el fin de suplir necesidades particulares se le denominan: personalizaciones (Sánchez, 2015).

Por lo anteriormente planteado se toma como concepto en la presente investigación que una distribución de GNU/Linux es el conjunto de pasos y modificaciones realizadas a una distribución para agregar, quitar o modificar características al sistema dependiendo de la necesidad y los usuarios finales.

Sistema Operativo Base

En el libro “**Fundamentos de los Sistemas Operativos**” Gunnar Wolf y otros autores, definen un sistema operativo base como: La combinación mínima necesaria de aplicaciones que permiten la instalación de paquetes en un sistema operativo, es aquel que sirve de base para cualquier otro (WOLF, y otros, 2015).

Según La Real Academia de la Lengua Española un sistema operativo base son conjunto de programas que realizan funciones básicas y permiten el desarrollo de otros programas en un Sistema Operativo.

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

En la presente investigación se asume como concepto de Sistema Operativo Base al conjunto de aplicaciones que permiten realizar las funciones básicas que necesita un Sistema Operativo para funcionar, además el mismo puede ser la base o no de cualquier otro. En esta base es donde se realizan los cambios necesarios que constituyen distintivos de una personalización.

Personalizaciones de GNU/Linux

Según lo expresado por Manuel Enrique Cruz Peiso en su tesis de maestría plantea que al utilizar distribuciones de GNU/Linux permiten la posibilidad de emplearlo o modificarlos para que sea utilizado con un determinado uso u objetivo específico, y esto se debe a que se puede contar con su código fuente, es por eso que se puede definir personalizaciones de GNU/Linux como: el conjunto de pasos y modificaciones realizadas a una distribución para agregar, quitar o modificar características al sistema dependiendo de la necesidad y los usuarios finales. (Cruz, 2018)

Se asume como concepto de personalizaciones de GNU/Linux como la modificación del sistema operativo GNU/Linux a través de acciones realizadas sobre un conjunto de aplicaciones y configuraciones que lo confirmen, con el objetivo de agregar características al sistema y adaptarlo a las necesidades y gustos de sus usuarios finales.

1.3 Estructura de un ISO de GNU/Linux

Una estructura de un ISO de GNU/Linux es como se encuentra conformada la jerarquía del sistema de archivos en GNU/Linux, la misma sigue un estándar conocido como FHS (*Filesystem Hierarchy Standard*) que fue creada por la comunidad que define los directorios y el contenido de los directorios en los sistemas operativos GNU/Linux (Peiso, 2018).

Mediante FHS se consigue disponer de un sistema de archivos completamente estructurado, coherente y estandarizado obteniendo como ventajas (LinuxAdictos, 2017):

- El software que se tiene instalado en nuestro ordenador sabe en todo momento las carpetas y los permisos de las carpetas del ordenador. Por lo tanto en todo momento se sabe dónde encontrar y almacenar la información que necesita el software para su funcionamiento.
- Los usuarios saben en todo momento el contenido que hay en cada una de las carpetas del ordenador.
- Ayuda a la hora de realizar el mantenimiento de un sistema operativo.

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

- Ayuda a otorgar los permisos pertinentes a cada uno de los archivos del sistema operativo que se está ejecutando.

A continuación, se muestra en la Figura 1. Estructura de un ISO de GNU/Linux:

```
----boot
----casper
----dists
----EFI
----install
----isolinux
----md5sum.txt
----pics
----pool
----preseed
----README diskdefine
```

Figura 1. Estructura de un ISO de GNU/Linux

(Fuente: elaboración propia)

- **boot:** almacena las configuraciones del cargador de arranque del sistema para instalar en particiones GPT11. Esto incluye las características gráficas que el mismo debe poseer, así como la dirección exacta del núcleo y del archivo de configuración del liveCD dentro del sistema de fichero.
- **casper:** contiene los archivos que utiliza el sistema para cargar el liveCD. Esto incluye el sistema comprimido (.squashfs), los archivos que necesita el instalador para su funcionamiento (.manifest, .manifest-desktop), el tamaño del sistema comprimido una vez que se descomprime (.size), el sistema de inicio de Linux (initrd, lz) y el núcleo del sistema (vmlinuz).
- **dists:** almacena los índices del repositorio interno, esto es utilizado por el instalador si necesita instalar alguna nueva aplicación.
- **EFI:** contiene las configuraciones necesarias para cargar el sistema con EFI.
- **isolinux:** almacena las configuraciones del cargador de arranque para instalar en particiones MBR. Esto incluye las características gráficas que el mismo debe poseer, así como la dirección exacta del núcleo y del archivo de configuración del liveCD dentro del sistema de fichero.

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

- **pool:** contiene los paquetes de código binario del repositorio interno, esto es utilizado por el instalador si necesita instalar alguna nueva aplicación.
- **. disk:** incluye informaciones del sistema operativo, tales como nombre, versión, arquitectura, etc.
- **md5sum.txt:** es un archivo que almacena las sumas de verificación de cada uno de los archivos de la imagen.
- **preseed:** conjuntos de instrucciones propias del sistema que permite habilitar y deshabilitar funcionalidades específicas.
- **ubuntu:** enlace simbólico a la misma carpeta donde se encuentra la estructura de la imagen.
- **pics:** conjunto de imágenes heredadas de Debian.

1.3.1 Herramienta de compresión de sistemas operativos

Actualmente cualquier distribución de GNU/Linux incluye tanto tar⁵ como los paquetes con las librerías y utilidades para comprimir y descomprimir ficheros (TICARTE, 2014). Numerosas son las herramientas para realizar dichos procesos; en este su epígrafe el autor dedicó una explicación a la herramienta para la compresión **squashfs** porque se adapta a las características de la propuesta de solución.

Squashfs (. sfs) es un sistema de archivos comprimido de solo lectura para Linux. Comprime archivos y directorios soportando tamaños de bloque de hasta 1024 KB⁶ para mayor compresión es también software libre para acceder a sistemas de archivos Squashfs. Está pensado para su uso como sistema de archivos genérico de solo lectura y en dispositivos de bloques/sistemas de memoria limitada, donde se requiere poca sobrecarga (Nikkel, 2016).

1.3.2 Gestor de arranque de los ISOS de las distribuciones de GNU/Linux

Un Gestor de arranque es un pequeño programa que se ejecuta una vez completado el inicio normal de la BIOS y que permite seleccionar el sistema operativo en caso de disponer de arranque múltiple. (Raya Cabrera, et al., 2014).

En las distribuciones de GNU/Linux se utiliza como gestor de arranque **Isolinux** el cual proporciona abundante control sobre el proceso de inicio al permitir que el usuario ingrese diferentes etiquetas y opciones de inicio para controlar qué imagen se inicia y qué características de esa imagen están activadas

⁵ Tar: Comando para unir fichero y directorios en un único fichero.

⁶ KB: kilobyte es una unidad de almacenamiento de información y equivale a 10³ (mil) byte.

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

o desactivadas. Isolinux es uno de varios proyectos en el conjunto de cargadores de arranque Syslinux. Los diferentes cargadores de arranque Syslinux difieren según el tipo de medio que cada uno está hecho para arrancar, que incluyen lo siguiente (Negus, 2014):

- *Isolinux*: se utiliza para arrancar CD-ROM estándar ISO 9660. El cargador Isolinux es el que se usa con más frecuencia para arrancar CD y DVD en vivo (o imágenes hechas para esos medios).
- *Syslinux*: se utiliza para arrancar el sistema de archivos MS-DOS y Windows FAT.

1.3.3 Herramientas para el uso de UEFI/GPT en los ISOS de las distribuciones de GNU/Linux UEFI

Interfaz de Firmware⁷ Extensible Unificada; es una interfaz entre el sistema operativo y el hardware de la plataforma en cuestión. Pretende reemplazar la tradicional Interfaz del Sistema Básico de Entrada y Salida (BIOS). La mayoría de las computadoras modernas son compatibles con el modo UEFI y el modo BIOS el modo de inicio que se debe usar depende de factores como: Coincidencia en el modo de arranque, Soporte de hardware y Problemas del gestor de arranque (Rothman, y otros, 2017).

Ventajas que presenta UEFI frente a BIOS (Richarte, 2018):

- Funciona en computadoras tanto de 32 bits como 64 bits.
- Se pueden conectar discos duros con más de 4 particiones y más de 2,2 TB cada una.
- Arquitectura y drives independientes de la CPU.
- Compatibilidad y emulación del BIOS para los sistemas operativos solo compatibles con esta última.
- Soporte completo para la Tabla de particiones GUID (GPT) [concepto explicado en lo adelante], se pueden crear hasta 128 particiones por disco, con una capacidad total de 8 ZB⁸.
- Uso de unidades cifradas gracias al arranque seguro.
- Entorno amigable y flexible Pre-Sistema Operativo, incluyendo capacidades de red.

⁷ Firmware: es un programa informático que establece la lógica de más bajo nivel que controla los circuitos electrónicos de un dispositivo de cualquier tipo.

⁸ ZB (zettabyte): es una unidad de almacenamiento de información equivale a 10^{21} bytes.

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

UEFI/GPT

La tabla de partición GUID (GPT) es un esquema de partición que forma parte de la especificación de la Interfaz de Firmware Extensible Unificada (UEFI); utiliza Identificadores Únicos Globales (GUID), para abordar las particiones y contiene un Master Boot Record, Registro de Arranque Principal (MBR) de protección para dirigir las computadoras más antiguas en términos de manejo de la partición. Su capacidad para manejar discos de hasta 256 TB⁹ por lo que puede soportar hasta 128 particiones (Rohaut, 2017).

Entre las herramientas para el uso de UEFI/GPT en los ISOS de las distribuciones GNU/Linux se encuentra:

1. **Fdisk (Disco Fijo o Disco de Formato):** es una utilidad de manipulación de disco basada en la línea de comandos más utilizada para sistemas Linux/Unix. Con la ayuda del comando *fdisk*, se pueden realizar operaciones como: ver, crear, cambiar de tamaño, eliminar, copiar y mover particiones en un disco duro utilizando su propia interfaz de menú basada en texto fácil de usar (Bresnahan, et al., 2015).
2. **GNU Parted:** es una herramienta de línea de comandos que permite administrar fácilmente las particiones del disco duro. Puede ayudarlo a agregar, eliminar, reducir y extender particiones de disco junto con los sistemas de archivos ubicados en ellas (Smith, 2011).
3. **GParted:** es un editor de particiones, el cual nos permite crear, eliminar, redimensionar, inspeccionar y copiar particiones como también los archivos que se encuentran en ellas. Permite al usuario la posibilidad de cambiar la organización de las particiones conservando el contenido de la organización. Utiliza la biblioteca **GNU Parted** para detectar y manipular dispositivos y tablas de partición, así como varias herramientas de sistema de archivos que dan mantenimiento a los no soportados por *libparted* (Librería de Particionamiento) (GPARED, 2017).
4. **GNOME Disks:** es una utilidad central del sistema utilizada para la gestión de particiones de disco y la supervisión de S.M.A.R.T. Se utiliza para formatear y crear particiones en unidades, montar y desmontar particiones. Se entrega con el conocido entorno de escritorio GNOME¹⁰. Su última

⁹ TB (Terabyte): equivalente a 10¹² byte.

¹⁰ GNOME: es un entorno de escritorio e infraestructura de desarrollo para sistemas operativos GNU/Linux.

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

versión 3.0.0 contiene nuevas características para agregar, redimensionar particiones, verificar los sistemas de archivos en busca de daños y repararlos (Sobell, 2014).

Según la entrevista realizada a especialistas de CESOL ver **Anexo1** los cuales desarrollan a diario aplicaciones encaminadas a la migración de Software Libre; se determinó que las herramientas para el uso de UEFI/GPT en las distribuciones de GNU/Linux Nova como distribución cubana: GParted y GNOME Disk.

1.4 Análisis de sistemas homólogos

En el análisis de la bibliografía realizada se tienen en cuenta para el estudio de la presente investigación los siguientes sistemas. A continuación, se exponen los resultados obtenidos de su estudio:

1.4.1 Novo Builder

Novo Builder (ver Anexo2) es una herramienta de Debian GNU/ Linux que permite crear sus propias variantes actualizadas de Ubuntu personalizada para incluir los programas, temas y configuraciones personalizadas que el usuario desee. Proporciona una interfaz conveniente para editar y construir un Sistema Operativo en menos de una hora (MAKETECHEASIER, 2010).

Otras Características (Leontis, 2012):

- Permite compartir lo que el usuario desee con mucha facilidad permitiendo que otras personas utilicen tus ideas para crear sus propios proyectos.
- Puede almacenar cualquier número de proyectos, por lo que puede tener una gran colección de preajustes que se puede ejecutar.
- Puede construir un sistema operativo para i386 y amd64¹¹
- Personaliza que repositorios usa para construir y que paquetes incluye.

1.4.2 Distroshare Ubuntu Imager

Distroshare Ubuntu Imager, es un script basado en las instrucciones que se pueden encontrar en la página oficial de Ubuntu, el mismo permite crear un Live instalable que se puede ejecutar desde el terminal para automatizar el proceso de personalización y creación. Como su nombre indica, permite crear una distribución basada en Ubuntu gracias a un fichero de `.conf` comentado, el cual se puede ir

¹¹ amd64: es la versión de 64 bits del conjunto de instrucciones x86.

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

personalizando sus valores hasta dejar la distribución a gusto del usuario y luego ejecutar el `.sh` para generar la ISO¹² (UBUNLOG, 2017).

1.4.3 Ubuntu Customized Kit

Ubuntu Customized Kit (ver Anexo 3), es un script que guía y automatiza la edición de archivos de una imagen básica de Ubuntu, por lo que para crear un *live-cd* se necesita una imagen básica de Ubuntu. Permite además instalar el software y el escritorio que se desee; uno de sus objetivos principales es facilitar la creación de versiones del sistema Operativo en uno más idiomas beneficiando al que o instale ya que la versión oficial de Ubuntu puede no contener todos los paquetes de idiomas disponibles debido a cuestiones de espacio (Sarmah, 2019).

1.4.4 Sistema de Construcción de Personalizaciones de Nova

El Sistema de Construcción de Personalizaciones de Nova es una tesis realizada por el ingeniero Eugenio Durán Aroche en el año 2012 en la Universidad de las Ciencias Informáticas, dicha tesis tuvo como objetivo general crear un sistema de construcción de personalizaciones como un servicio de la plataforma Zentyal¹³ que brinde la posibilidad de seleccionar el perfil y arquitectura disponible, además debe de gestionar repositorios y añadir paquetes locales a la instalación (Aroche, 2012); pero de dicho sistema sólo se tiene un estudio teórico impidiendo que el proceso fuese automatizado por el proyecto NOVA.

Resultados del estudio de sistemas homólogos

En la propuesta de solución se tomó como base teórica el uso de la tesis desarrollada por Ing. Eugenio Durán Aroche, ya que esta brinda la posibilidad de profundizar en base de lo que ya está realizado ayudando así a un mejor desarrollo de la propuesta de solución.

Con respecto a las herramientas restantes se realizó un análisis con el objetivo de conocer si cumplen con las necesidades existentes en la creación de personalizaciones para la distribución de GNU/Linux NOVA.

Criterios de Análisis: se escogen como criterios los mostrados posteriormente porque los mismos forman parte de las necesidades que se requieren para el desarrollo de la propuesta de solución.

- GPT+UEFI: permite realizar particionamiento de discos y unidades mediante esta tecnología.

¹² ISO: archivo donde se guarda una copia o imagen exacta de un sistema de fichero.

¹³ Zentyal: servidor de red unificada de código abierto, gestiona las amenazas de seguridad como servidor de oficina o como servidor de comunicaciones unificadas.

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

- Configuración desde un archivo: que las configuraciones que se hagan para crear las personalizaciones se a través de un archivo de configuración entrado por el usuario al sistema para especificar de qué manera quiere su distribución.
- Construcción automática: que no requiera de la interacción manual por parte del usuario para que se ejecute el sistema, sino que sea automatizado todo el proceso de creación de la personalización de la distribución cubana GNU/Linux Nova.
- Construcción desde cero: permita realizar todas las configuraciones desde cero dígase desde la conformación del sistema operativo base.

A continuación, se muestra la Tabla 1 Resultados del análisis de las Herramientas con distribuciones de GNU/Linux.

Tabla 1: Resultados del análisis de las Herramientas con distribuciones de GNU/Linux
(Fuente: elaboración propia)

Criterios de Análisis	Herramientas Seleccionadas		
	Novo Builder	Distroshare Ubuntu Imager	Ubuntu Customized Kit
GPT+UEFI	No	No	Si
Configuración desde un archivo	No	No	No
Construcción automática	No	Si	No
Construcción desde cero	No	No	No

El análisis realizado demostró la necesidad de desarrollar una nueva solución ya que estas herramientas no resuelven las carencias que presenta el proyecto Nova para la creación de sus personalizaciones.

1.5 ¿Cómo se hacen los ISOS de Nova?

Pasos y procesos que intervienen en la construcción de Nova (Peiso, 2018):

Construcción del repositorio de código fuente: se crea un repositorio que contiene el código fuente de las aplicaciones etiquetadas de “important” en el repositorio oficial de Ubuntu.

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

Estas son las aplicaciones básicas que necesita un sistema para funcionar correctamente.

- **Modificación del código fuente** de algunas aplicaciones, compilación y empaquetado: se modifica el código fuente de varias aplicaciones que permiten darle identidad a la distribución cubana de GNU/Linux Nova. Entre ella se encuentran: *distro-info* y *distro-info-data* (información de la distribución), *devscripts* (scripts para el uso de los desarrolladores), *novakeyring* (llaves públicas de Nova), *debootstrap* (construcción de sistemas base). Luego se compilan y empaquetan todas las aplicaciones albergadas en el repositorio para crear los instalables de las mismas.
- **Construcción del repositorio de código binario:** con los instalables generados se crea el repositorio de código binario que los almacena. En este punto se puede decir que se posee un repositorio de código fuente y binarios de los paquetes básicos que necesita un sistema para funcionar correctamente.
- **Modificación del entorno de escritorio:** se incluye el código fuente del entorno de escritorio y todas sus dependencias en el repositorio y se comienza a modificar teniendo en cuenta la arquitectura de información definida en el departamento de Sistemas Operativos.
- **Modificación de aplicaciones:** se incluye el código fuente de varias aplicaciones que responden a los requisitos detectados en la ingeniería de requisitos, así como sus dependencias, y se comienzan a modificar, siempre que sea necesario, para que se integren correctamente con el entorno de escritorio.
- **Construcción de imágenes de Nova:** a lo largo de cada uno de los procesos anteriores se construyen imágenes para verificar el correcto funcionamiento de cada uno de ellos y para detectar posibles incompatibilidades entre algunas de las aplicaciones requeridas.

1.6 Metodología de desarrollo de software

Una metodología de desarrollo de software es un marco de trabajo usado para estructurar, planificar y controlar el proceso de desarrollo en sistemas de introducción. (OBSBUSINESS, 2019) Además, cuenta con factores como los costes, la planificación, la calidad y las dificultades asociadas que se manejan en el proceso que se suele seguir para diseñar una solución o un programa específico (Piñeiro, 2015)

La metodología seleccionada para el desarrollo de la propuesta de solución es una variación de la metodología “Proceso Unificado Ágil” (AUP) en unión con el modelo CMMMI-DEV fue creada por la Universidad de las Ciencias Informáticas denominada AUP-UCI, (León, 2017) la cual se adapta a las

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

características de cada proyecto de desarrollo de dicha universidad, además es su documento rector y se ajusta al ciclo de vida definido para la actividad productiva. Es una metodología flexible que no requiere de una gran cantidad de desarrolladores. Contiene un flujo de trabajo que cuenta de tres fases: (Rodríguez, 2015)

- **Inicio:** durante el inicio se llevan a cabo todos los planes del proyecto, recursos humanos, cronograma de tareas, plan de riesgo. Se realiza un estudio inicial de la organización cliente que permite obtener información fundamental acerca del alcance del proyecto, realizar estimaciones de tiempo, esfuerzo y costo y decidir si se ejecuta o no el proyecto.
- **Ejecución:** en esta fase se ejecutan las actividades requeridas para desarrollar el software, incluyendo el ajuste de los planes del proyecto considerando los requisitos, la arquitectura y el diseño, además de implementarse el producto.
- **Cierre:** en el cierre se analizan tanto los resultados del proyecto como su ejecución y se realizan las actividades formales de cierre del proyecto.

El presente trabajo de diploma se centra en la fase de Ejecución y transitará por las siguientes disciplinas propuestas en la metodología: Modelado del negocio, Requisitos, Análisis y Diseño, Implementación, Pruebas internas y Pruebas de aceptación. Los productos generados por la investigación y por lo tanto en la propuesta de solución estarán basados en el Escenario No 3 porque se aplica a los proyectos que hayan evaluado el negocio a informatizar y como resultado obtengan un negocio con procesos, independientes de las personas que los manejan y ejecutan, proporcionando objetividad, solidez, y su continuidad. Este escenario es muy conveniente si se desea representar una gran cantidad de niveles de detalles y relaciones entre los procesos identificados (Peñalver, 2017).

1.7 Lenguajes y herramientas para el modelado de la propuesta de solución

En el modelado de la propuesta de solución se utilizaron los siguientes lenguajes y herramientas:

1.7.1 Lenguajes de Modelado

UML 2.0 (Lenguaje de Modelado Unificado) es el lenguaje de modelado gráfico utilizado para visualizar y documentar un sistema. Posibilita un estándar para describir los modelos, incluyendo aspectos conceptuales como procesos de negocio, funciones del sistema, expresiones de lenguajes de programación, esquemas de bases de datos y componentes reutilizables (UML, 2017).

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

BPMN (Modelo y Notación de Procesos de Negocio), es una notación gráfica estandarizada que permite el modelado de procesos de negocio, su principal objetivo como lenguaje de modelado es proporcionar una notación gráfica estándar que sea fácilmente legible y entendible; además tiene la finalidad de servir como lenguaje común para cerrar la brecha de comunicación que frecuentemente se presenta entre el diseño de los procesos de negocio y sus implementación (Chakray, 2017).

1.7.2 Herramienta de modelado

Visual Paradigm 8.0 UML, es una herramienta CASE¹⁴ multiplataforma, se selecciona como herramienta para el modelado porque permite la representación de las etapas por las que trasciende un producto de software. Ayuda a una rápida construcción de aplicaciones de calidad y con el menor costo posible. Permite dibujar todos los tipos de diagramas de clases, generar código desde diagramas y desarrollar información (Cabriales, 2013).

1.7.3 Herramientas y tecnologías de implementación

En el desarrollo de la propuesta de solución se utilizaron las siguientes herramientas y tecnologías.

1.7.4 Lenguaje de programación

Un lenguaje de programación es un lenguaje natural o artificial que le proporciona a una persona, la capacidad de escribir una serie de instrucciones o secuencias de órdenes en forma de algoritmos con el fin de controlar el comportamiento físico y lógico de una computadora (Rockcontent, 2019).

Python 3.5 es un lenguaje potente, flexible y con una sintaxis clara y concisa que no requiere dedicar tiempo a su compilación debido a que es interpretado. Es de código abierto y como tal cualquiera puede contribuir a su desarrollo y divulgación. Además, no es necesario pagar ninguna licencia para distribuir el software desarrollado en este lenguaje. Hasta su intérprete se distribuye de forma gratuita. Puede ser utilizado en diversas plataformas y sistemas operativos, entre los que podemos destacar los más populares, Windows y Linux; pero además puede funcionar en Smartphone (PYTHONES, 2019).

Las principales características que tiene este lenguaje son (Nolasco, 2018):

- Es un lenguaje interpretado, **no compilado**, usa tipado dinámico, fuertemente tipado.
- Es multiplataforma, lo cual es ventajoso para hacer ejecutable su código fuente entre varios Sistemas Operativos.

¹⁴ CASE (Ingeniería de Software Asistida por Computadoras), es una herramienta formada por programas y ayudas para la automatización de tareas en el desarrollo de un software, completamente o en alguna de sus fases.

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

- Es un lenguaje de programación multiparadigma, el cual soporta varios paradigmas de programación como orientación a objetos, estructurada, programación imperativa y, en menor medida, programación funcional.
- En Python, el formato del código es estructural.

Para realizar la implementación de la propuesta de solución se decide utilizar como lenguajes de programación Python porque es un lenguaje de programación reutilizable, y posibilita realizar las configuraciones deseadas y así llegar a la personalización de su sistema operativo, en este caso Nova como distribución cubana de GNU/Linux.

1.7.5 Entorno de Desarrollo Integrado

Un entorno de desarrollo integrado o entorno de desarrollo interactivo (IDE), es una aplicación informática que proporciona servicios integrales para facilitarle al desarrollador o programador el desarrollo de software. Consiste en un editor de código fuente, herramientas de construcción automáticas y un depurador (SOFTWAREPROG, 2018).

El límite entre un IDE y otras partes del entorno de desarrollo de software más amplio no está bien definido. Muchas veces, a los efectos de simplificar la construcción de la interfaz gráfica de usuario (GUI, por sus siglas en inglés) se integran un sistema controlador de versión y varias herramientas. Muchos IDE modernos también cuentan con un navegador de clases, un buscador de objetos y un diagrama de jerarquía de clases, para su uso con el desarrollo de software orientado a objetos (FERGARCIA, 2013).

Visual Studio Code es un editor de código fuente desarrollado por Microsoft para Windows, Linux Y macOS. Incluye soporte para la depuración, control integrado de GIT, resaltado de sintaxis, finalización inteligente de código, fragmentos y refactorización de código. También es personalizable, por lo que los usuarios pueden cambiar el tema del editor, los atajos de teclado y las preferencias. Es gratuito y de código abierto (SPRINGER, 2018).

Se utiliza como IDE para el desarrollo de la propuesta de solución porque ofrece como ventajas:

- Resaltado de sintaxis: capacidad para el tratamiento de textos, para diferenciar elementos de textos mediante diversos colores o estilos tipográficos, dependiendo de las categorías sintácticas de sus términos.
- Depuración: identifica y corrige errores.

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

- *Snippet*: pequeñas partes reusables de código fuente, de máquina o texto.
- Refactorización: ofrece la posibilidad de reestructurar un código fuente, alterando su estructura interna sin cambiar su comportamiento externo.

1.7.6 Control de versiones

Un sistema de control de versiones es la gestión de los diversos cambios que se realizan sobre los elementos de algún producto o una configuración del mismo. Una versión, revisión o edición de un producto, es el estado en que se encuentra el mismo en un momento dado de su desarrollo o modificación.

El utilizado en la propuesta de solución es el Control de Versiones de Software **GIT** el cual cuenta con características como (Paz, 2017):

- Fue diseñada por Linus Torvalds.
- No depende de un repositorio central.
- Es de software libre.
- Se puede llevar a cabo un historial completo de versiones.
- Posibilita que el usuario pueda visualizar las revisiones de código y desplazarse por el de manera ágil.
- Posee un sistema de trabajo con ramas, las cuales están destinadas a provocar proyectos divergentes de un proyecto final, brindando la posibilidad de realizar experimentos o probar nuevas funcionalidades.

1.8 Conclusiones del Capítulo

En el capítulo se definen los conceptos esenciales para la investigación relacionados con el proceso de elaboración de personalizaciones de las distribuciones GNU/Linux, fundamentales en la propuesta de solución.

La comparación de las diferentes herramientas de personalización analizadas permitió demostrar la necesidad de desarrollar una herramienta informática porque las existentes no resuelven las carencias que presenta el proyecto Nova para la creación de personalizaciones de su sistema operativo.

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

Para la propuesta de solución se determina el uso de Variación de AUP para la UCI como metodología de desarrollo, *Visual Paradigm* como herramienta para el modelado, Python como lenguaje de programación, *Visual Studio Code* como entorno de desarrollo y *Git* como control de versiones.

CAPÍTULO 2: ANÁLISIS Y DISEÑO DE LA HERRAMIENTA INFORMÁTICA PARA LA CREACIÓN DE PERSONALIZACIONES DE LA DISTRIBUCIÓN CUBANA GNU/LINUX NOVA.

Introducción

En el presente capítulo, se desarrolla el análisis y diseño partiendo de la propuesta de solución del proceso de creación de personalizaciones y los elementos planteados por la metodología Variación de AUP para la UCI en el escenario 3. Se definen los requisitos del sistema y la arquitectura del mismo.

2.1 Modelo del negocio

El modelado del negocio es la disciplina destinada a comprender los procesos de negocio de una organización. Se comprende cómo funciona el negocio que se desea informatizar para tener garantías de que el software desarrollado va a cumplir su propósito (Jiménez, 2018).

2.1.1 Modelo Conceptual

El modelo conceptual es el encargado de representar los conceptos fundamentales para el desarrollo de una aplicación y las diferentes relaciones que existen entre ellos. Son presentados como clases los aspectos esenciales y las interrelaciones denotan una estructura de funcionamiento, brindando una mejor comprensión del medio actual para la concepción de un futuro sistema (Pantaleo, 2016).

En la figura 2 se muestra el modelo conceptual del dominio del negocio de la propuesta de solución.

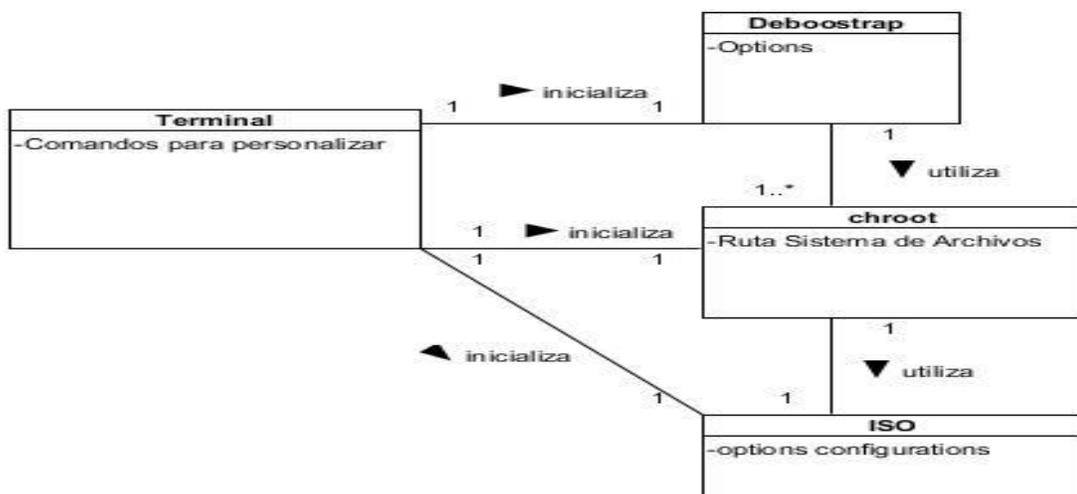


Figura 2 Modelo Conceptual

(Fuente: elaboración propia)

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

Conceptos:

- **Terminal:** es el dispositivo electrónico que le permite al usuario realizar funcionalidades para poder interrumpir cualquiera de los procesos para personalizar basado a sus requerimientos el ISO.
- **Debootstrap:** herramienta que permite crear el Sistema Operativo Base.
- **Chroot:** herramienta que permite Configurar paquetes necesarios y adicionales para crear una distribución.
- **ISO:** proceso donde se realizan las configuraciones necesarias para crear la imagen del Sistema Operativo.

2.1.2 Diccionario de datos de personalización

Tabla 2. Diccionario de personalización

(Fuente: elaboración propia)

Descripción	Una personalización es la versión de la imagen ISO inicial con una configuración diferente y personalizada a partir de las necesidades del usuario, donde se contestan todas o la mayoría de las preguntas que una instalación normalmente hace. Además incluye un grupo de software que se instalarán automáticamente antes de que el sistema arranque por primera vez en servicio.					
Atributos						
Nombre	Descripción	Tipo	¿Puede ser nulo?	¿Único?	Restricciones de clase	
					Válidas	No válidas
Comandos para personalizar	Comandos que se utilizan para poder detener el proceso que se esté ejecutando en el momento y así poder ejecutar cualquier parámetro que necesite el usuario para su personalización.	Alfanumérico	No	No	No aplica	Caracteres extraños
<i>Options</i>	Representa las opciones necesarias para que se pueda crear el Sistema	Alfanumérico	No	No	No aplica	Caracteres extraños

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

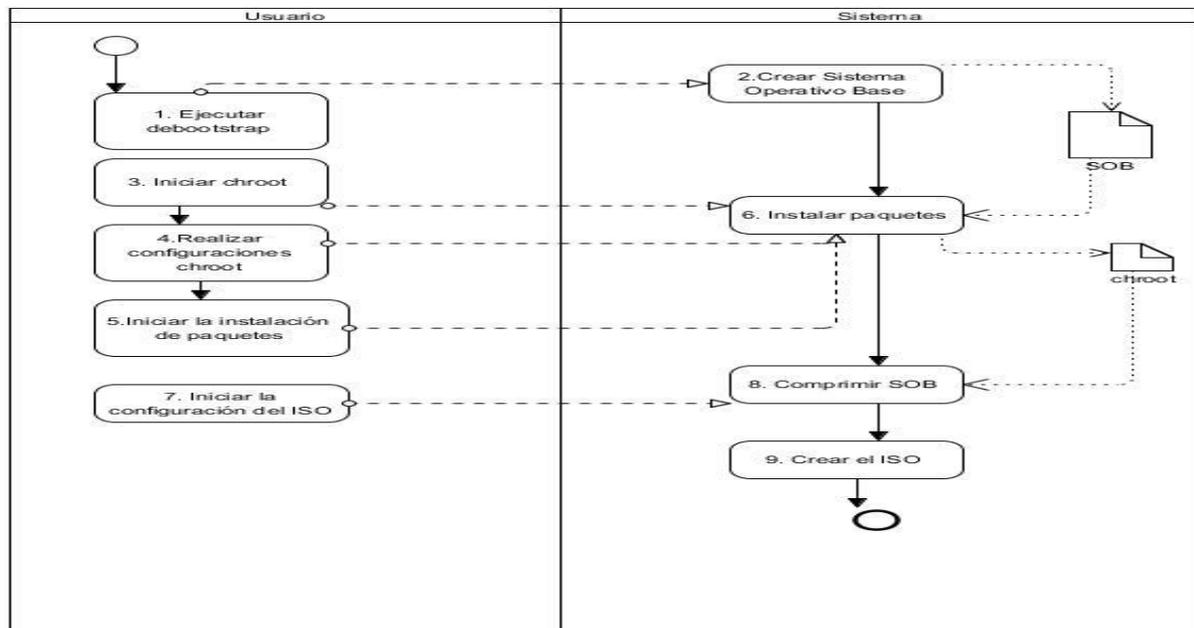
	Operativo Base.					
Ruta de Sistema de Archivos	Contiene la ruta donde se encuentra los archivos necesarios para que funcionen de manera correcta la configuración de paquetes.	Alfanumérico	No	Sí	No aplica	Caracteres extraños
<i>Options de Configuration</i>	Representa las configuraciones que se realizan para poder realizar el ISO.	Alfanumérico	No	No	No aplica	Caracteres extraños

2.1.3 Descripción de proceso de negocio

Un proceso de negocio es un conjunto de tareas relacionadas lógicamente llevadas a cabo para lograr un resultado de negocio definido. Cada uno de los procesos tiene sus entradas, funciones y salidas. Las entradas son requisitos que deben tenerse en cuenta antes de que una función pueda ser aplicada (Pérez, 2015).

2.1.4 Diagrama de procesos del negocio personalización de Nova.

A continuación, se presenta el diagrama de proceso del negocio creador de personalizaciones de Nova y su descripción.



Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

Figura 3: Diagrama de proceso del negocio Creador de personalizaciones de Nova.

(Fuente: elaboración propia)

Tabla 3: Descripción de procesos de Creador de personalizaciones de Nova

(Fuente: elaboración propia)

Objetivo	Definir el flujo de eventos de la creación de personalizaciones de la distribución cubana GNU/Linux Nova.
Evento(s) que lo genera(n)	No aplica
Pre condiciones	ISO de Nova Escritorio, computadora con Linux.
Marco legal	No aplica.
Clientes internos	No aplica.
Clientes externos	No aplica.
Entradas	Archivo de Configuración
Flujo de eventos	
Flujo Básico	
1.	El usuario da la orden al sistema de ejecutar el debootstrap. Introduce las opciones del Debootstrap por consola.
2.	El sistema recibe la orden y crea el Sistema Operativo Base con las opciones que introdujo el usuario.
3.	El usuario inicia el chroot introduciendo la ruta de la ubicación de los ficheros de archivos.
4.	El usuario introduce las configuraciones del chroot. Introduce la lista de paquetes que quiere instalar.
5.	El usuario da la orden de iniciar la instalación. Mediante comandos le pide al sistema que realice la instalación de paquetes.
6.	El sistema instala los paquetes. La instalación se realiza mediante todas las configuraciones entradas por el usuario para realizar el chroot.
7.	El usuario inicia la configuración del ISO. Mediante líneas de comandos introduce al sistema configuraciones del ISO para que el sistema lo ejecute.
8.	El sistema comprime el SOB. Con todas las configuraciones realizadas anteriormente el sistema comprime el SOB para así poder realizar el ISO.
9.	El sistema crea el ISO. Luego de estar comprimido el SOB y las configuraciones realizadas por el usuario para crear el ISO el sistema crea el ISO.
Pos-condiciones	Se creó el ISO personalizado.
Salidas	ISO personalizado.
Asuntos pendientes	No aplica.

2.2 Propuesta de solución

Luego de haber realizado un análisis del negocio y su comportamiento actual se tiene que hoy en día se lleva a cabo un proceso manual de creación de las personalizaciones de las distribuciones GNU/Linux, donde el usuario tiene que estar interactuando con el sistema operativo para realizar todos los procesos de creación e introduciéndoles opciones y configuraciones para que se puedan ejecutar. Es por eso que se plantea como propuesta de solución una herramienta informática que automatice la creación de

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

personalizaciones de la distribución cubana GNU/Linux Nova. A continuación, se realiza una explicación del proceso de la propuesta de solución.

Mediante un archivo de configuración creado por el usuario donde introduce arquitectura (*arch*), distribución (*Distribution*), componentes (*components*), ubicación del repositorio de donde se va a realizar la descarga de todos sus requerimientos (*mirror*), incluir los paquetes que quiere instalar (*include*), las configuraciones adicionales (*hooks*) y realizar las configuraciones de *isolinux*, *casper* y del repositorio. La herramienta informática se va a dividir en varios procesos:

1. La herramienta analiza sintácticamente (*parsing*) el archivo de configuración entrado por el usuario: la herramienta analiza todos los requerimientos que necesita el usuario para su personalización.

(A continuación, se muestra x.1 el flujo básico de la herramienta y en el x.2 el Flujo alterno que no es más que si no ocurre un subprocesso ocurre el otro)

2. Crear el Sistema Operativo Base:
 - 2.1 La herramienta borra cualquier carpeta de sistema de archivo que se haya ejecutado anteriormente, luego verifica si está descomprimido algún debootstrap (clase que permite crear SOB) antes realizado si no es el caso, crea el sistema operativo base, ejecuta las configuraciones adicionales entradas por el usuario y comprime el Sistema Operativo Base una vez configurado y creado.
 - 2.2 La herramienta borra cualquier carpeta de sistema de archivo que se haya ejecutado anteriormente, luego verifica si está descomprimido algún debootstrap antes realizado, si es el caso, carga el anterior, luego ejecuta las configuraciones adicionales y comprime el Sistema Operativo Base una vez configurado y creado.
3. Instalar Paquetes:
 - 3.1 La herramienta verifica si ha sido creado algún chroot (clase que permite instalar paquetes) sino es el caso descomprime el debootstrap, luego monta los Sistemas de Ficheros, instala los paquetes, limpia
4. Configurar directorio ISO: la herramienta con el chroot compreso ejecuta las configuraciones para poder crear el ISO.
5. Hacer el ISO: mediante las configuraciones y el chroot compreso hace el ISO. Devuelve un ISO personalizado.

A continuación, se muestra en la Figura 4. Propuesta de Solución.

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

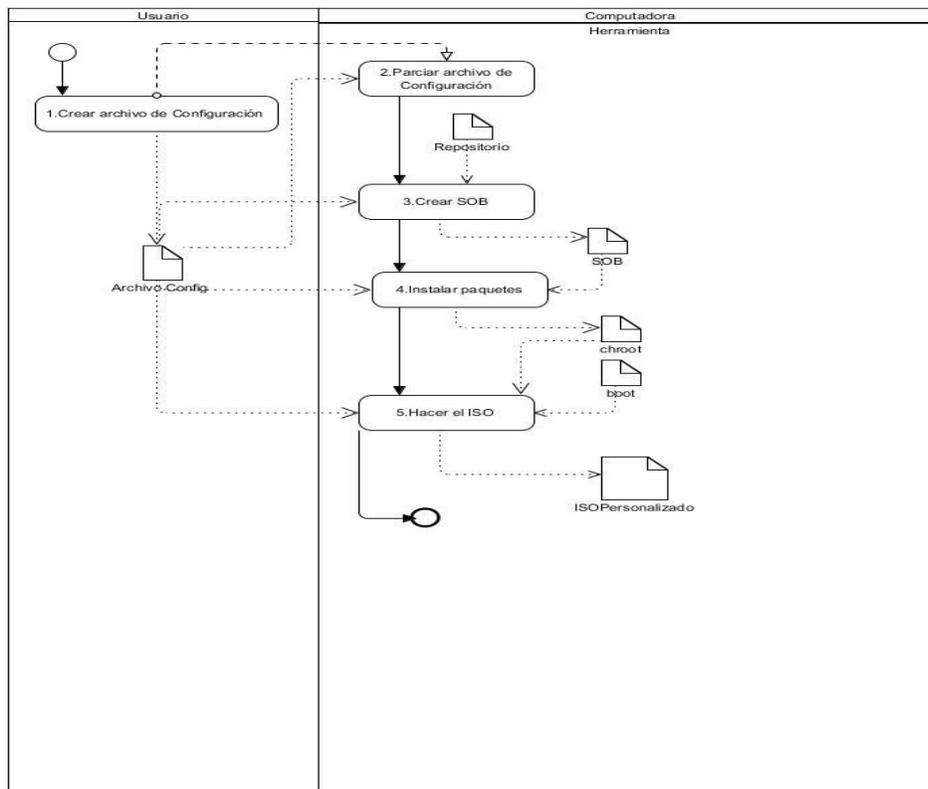


Figura 4. Propuesta de Solución.

(Fuente: elaboración propia)

2.3 Requisitos

Un requisito es simplemente una declaración abstracta de alto nivel de un servicio que debe proporcionar el sistema o una restricción de éste. En el otro extremo, es una definición formal y detallada de una función del sistema (Sommerville, 2005). El objetivo principal en la disciplina Requisitos es desarrollar un modelo del sistema que se va a construir. Esta disciplina comprende la administración y gestión de los requisitos funcionales y no funcionales del producto (Sánchez, 2015).

2.3.1 Fuentes de obtención de requisitos

Las fuentes de obtención de requisitos utilizadas fueron:

- Modelo Conceptual (Ver figura 2)
- Análisis de las herramientas existentes (Ver epígrafes 1.7 y 1.8)
- Especialistas de CESOL.

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

2.3.2 Técnicas de identificación de requisitos

Las técnicas de identificación de requisitos de software permiten identificar las necesidades de negocio de los clientes y los usuarios. Son mecanismos que se utilizan para recolectar la información necesaria en la obtención de los requerimientos de una aplicación, permiten investigar aspectos generales para posteriormente ser especificados con un mayor detalle, requieren ser adecuadamente orientadas para cubrir la información que se requiere capturar (Pressman, 2010).

Entrevista

La entrevista es una de las técnicas más utilizadas en la identificación de requisitos ya que responde a identificar los principales clientes y a mantener conversaciones técnicas para la identificación de los requisitos del sistema que va a ser desarrollado. En general, son escogidas para entrevistar a las principales personas y grupos de personas que usarán o recibirán algún impacto del sistema a desarrollar y su objetivo es determinar cuáles serán los usos que estos actores del negocio harán del sistema, el entorno que utilizarán, cuáles serán las limitaciones, cualidades y uso que harán de la herramienta para la creación de personalizaciones de Nova (Arias, 2014).

Observación

Por medio de esta técnica se obtiene información de primera mano sobre la forma en que se efectuarán las actividades. Este método permite observar la forma en que se llevan a cabo los procesos y, por otro, verificar que realmente se sigan los pasos especificados (Guerra, 2017).

2.3.4 Especificación de requisitos de software

La especificación de Requisitos de Software es una de las tareas más importantes en el ciclo de vida del desarrollo de software, porque en ella se determinan los planos de la nueva herramienta. Se realiza una descripción completa del comportamiento del sistema que se va a desarrollar. Y además de los requisitos funcionales y no funcionales (Vázquez, 2018).

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

Requisitos funcionales

Expresan las capacidades que debe poseer la solución, se identifican a partir de necesidades de negocio o necesidades de los clientes y usuarios. Estos requisitos describen beneficios que recibirán la organización y/o sus clientes con el desarrollo del sistema (Durango, 2015).

En la tabla 4 se realiza la especificación de los requisitos funcionales de la propuesta de solución, su descripción y complejidad, determinada mediante el producto de trabajo Evaluación de Requisitos del expediente de proyecto 4.0 disponible para la actividad productiva de la universidad.

Tabla 4. Especificación de requisitos funcionales

(Fuente: elaboración propia)

Número	Requisito	Descripción	Complejidad
RF1	Crear Sistema Operativo Base	El sistema debe permitir crear el Sistema Operativo Base mediante la herramienta Debootstrap.	Alto
RF2	Instalar paquetes	El sistema debe permitir instalar paquetes mediante la herramienta Chroot.	Alto
RF2.1	Instalar paquetes auxiliares	El sistema debe permitir instalar paquetes auxiliares para que los principales puedan realizar y ser borrados luego de la instalación.	Medio
RF2.2	Instalar el Kernel	El sistema debe permitir la instalación del kernel para que pueda ser ejecutado el Sistema Operativo Base Linux.	Alto
RF3	Comprimir el Sistema Operativo	El sistema debe permitir comprimir el Sistema Operativo para poderse ejecutar luego en un booteable.	Alto
RF5	Hacer el ISO	El sistema debe permitir hacer el ISO mediante la herramienta squasfs.	Alto
RF6	Ejecutar Hooks	El sistema debe permitir al usuario crear configuraciones adicionales.	Medio

Requisitos No Funcionales del sistema

Son restricciones de los servicios o funciones ofrecidas por la solución informática (tiempo, proceso o estándares del dominio de negocio), además de cualidades que se le imponen al sistema en cuanto al diseño y la implementación. Normalmente afectan al sistema en su totalidad más que una característica o servicio en particular (Durango, 2015).

Usabilidad

RNF 1: El sistema no debe presentar una interfaz gráfica, sino que utiliza una interfaz en línea de comandos.

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

Software

RNF 2: El sistema debe ser ejecutado en el sistema operativo GNU/Linux Nova.

RNF 3: El sistema debe ser programado en lenguaje de programación Python como restricción de diseño.

Seguridad

RNF 4: El sistema para poder ser utilizado debe ser ejecutado como root.

2.3.5 Descripción de requisitos de software por proceso

A continuación, se muestra la descripción del requisito funcional 1 Crear Sistema Operativo Base.

Descripción de RF 1 por proceso. Crear Sistema Operativo Base

Tabla 5.RF1. Crear Sistema Operativo Base

(Fuente: elaboración propia)

Precondiciones	El usuario rellena un archivo de configuración según sus necesidades.
Flujo de eventos	
Flujo básico crear Sistema Operativo Base	
1.	Remover debootstrap anterior.
2.	Obtener del archivo de configuración las opciones deseadas por el usuario y crear Sistema Operativo Base.
3.	Ejecutar hooks.
4.	Comprimir debootstrap.
Pos-condiciones	
1.	Se creó el Sistema Operativo Base.
Flujos alternos	
1.a El usuario creó un debootstrap anteriormente	
1.	Remover debootstrap anterior.
2.	Descomprimir la versión previa del debootstrap.
3.	Ejecutar hooks.
4.	Comprimir debootstrap.
Pos-condiciones	
1.	Se creó el Sistema Operativo Base.
Validaciones- Los parámetros del archivo de configuración deben seguir las siguientes validaciones para que el sistema pueda crear el Sistema Operativo Base.	
1.	La Architecture puede contener números (0-255) y letras (a-z y A-Z) con no más 255 caracteres.
2.	La Distribution solo puede contener cuatro números entre 0 y 255.

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

3.	El campo Components debe contener letras (a-z y A-Z).
4.	El campo Mirror debe contener un conjunto de caracteres que están compuestos por protocolo dos puntos y seguidos <i>backslash</i> (barra inversa invertida) subdominio separado por un punto le sigue el dominio separado por un punto y finalmente la extensión.

2.4 Análisis y Diseño

En esta disciplina, si se considera necesario, los requisitos pueden ser refinados y estructurados para conseguir una comprensión más precisa de estos y una descripción que sea fácil de mantener y ayude a la estructuración del sistema (incluyendo su arquitectura). Además en esta disciplina se modela el sistema y su forma para que soporte todos los requisitos, incluyendo los requisitos no funcionales. Los modelos desarrollados son más formales y específicos (Arias, 2014).

2.4.1 Modelado del diseño

A continuación, se presenta el modelado del diseño de la propuesta de solución.

Diagrama de clases del diseño

Un diagrama de clases es una presentación gráfica de la vista estática, que muestra una colección de elementos declarativos (estáticos) del modelo, como clases, tipos, contenidos y relaciones. Contiene ciertos elementos materializados de comportamiento, como operaciones, pero cuya dinámica se puede representar en otros diagramas, como diagramas de estados o diagramas de colaboración (Aroche, 2014).

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

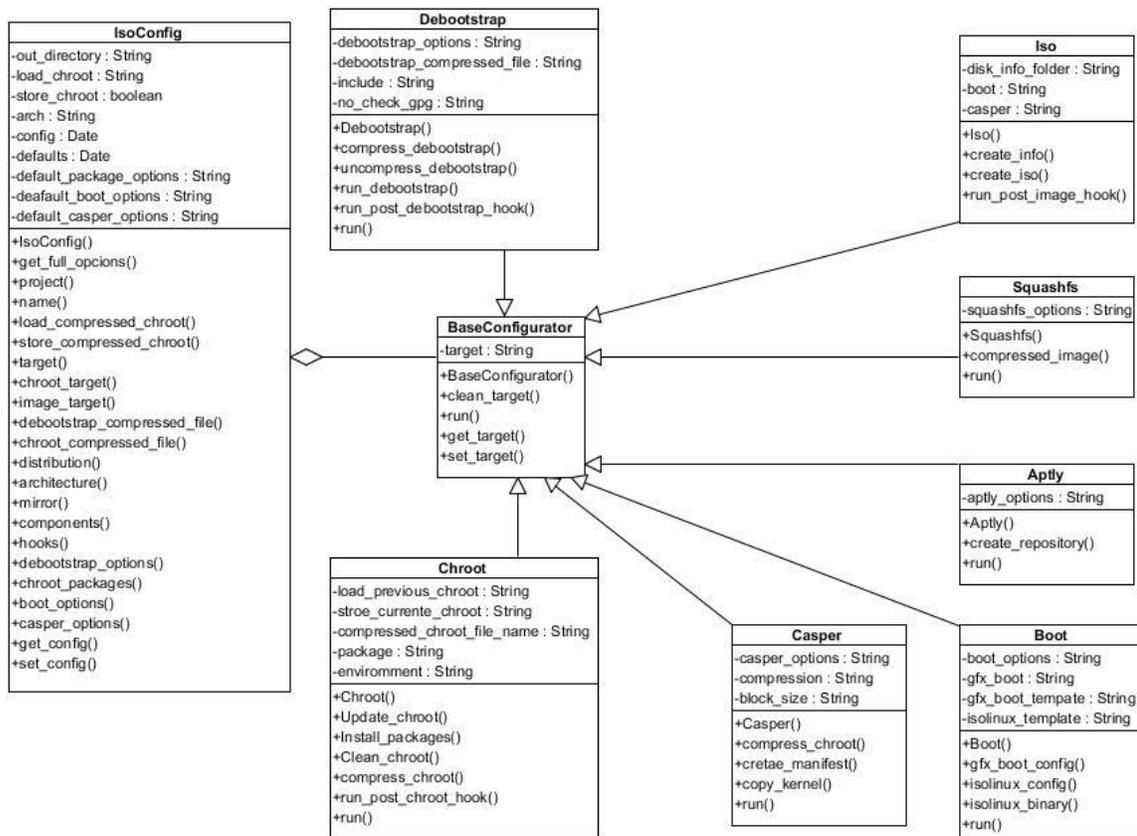


Figura 5: Diagrama de Clases del Diseño

(Fuente: Elaboración Propia)

El diagrama de clases del diseño permitió definir la estructura estática de las clases en el sistema, su contenido y las relaciones que se establecen entre ellas.

2.4.2 Patrones de Diseño

Los patrones GRASP describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en formas de patrones. GRASP es un acrónimo que significa *General Responsibility Assignment Software Patterns*. El nombre se eligió para indicar la importancia de captar estos principios, si se quiere diseñar eficazmente el software orientado a objetos.

A continuación, se explican los patrones utilizados:

- **Experto en información:** El GRASP de experto en información es el principio básico de asignación de responsabilidades. Indica, por ejemplo, que la responsabilidad de la creación de un objeto o la implementación de un método, debe recaer sobre la clase que conoce toda la

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

información necesaria para crearlo. De este modo se obtiene un diseño con mayor cohesión y así la información se mantiene encapsulada.

Ejemplo de ello es la clase *IsoConfig*, la cual permite capturar y analizar sintácticamente la información del archivo de configuración que introduce el usuario para realizar las configuraciones pertinentes.

- **Bajo acoplamiento:** Es la idea de tener las clases lo menos ligadas entre sí que se pueda. De tal forma que, en caso de producirse una modificación en alguna de ellas, se tenga la mínima repercusión posible en el resto de clases, potenciando la reutilización, y disminuyendo la dependencia entre las clases. Un ejemplo de lo antes expuesto lo constituye la clase *Debootstrap* la cual se evidencia que en caso de ocurrir un cambio en alguna de sus funcionalidades no va a repercutir en el resto del funcionamiento de los demás procesos.

Los patrones **GOF**, acrónimo de *Gang o Four*, Pandilla de los Cuatro, en su traducción al español. Los patrones de diseño, conocidos como GoF se clasifican en tres grandes categorías basadas en su propósito: creadores, estructurales y de comportamiento. A continuación, se describen los que fueron utilizados en el desarrollo de la propuesta de solución.

- **Patrones de Creación:** El objetivo de estos patrones es de abstraer el proceso de instanciar y ocultar los detalles de cómo son creados e inicializados los objetos. Ejemplo de este patrón se evidencia en la clase *IsoConfig* la cual permite realizar toda la configuración mediante los métodos *get_config* y *set_config*.

2.4.3 Diseño arquitectónico

En el diseño de la propuesta de solución se utiliza como patrón arquitectónico Tuberías y Filtros donde los componentes o procesos se denominan filtros, ya que actúan como traductores que toman la entrada, la transforman según el algoritmo o funcionalidad que se desarrolle en el proceso, y generan la salida hacia un conducto de comunicaciones. Los conductos de entrada y salida se denominan tubos (Molinero Parra, 2018).

Estilo Arquitectónico

Para la implementación de la herramienta se selecciona el tipo Tuberías y filtros ya que este enfoque pone énfasis en la transformación gradual de los datos por componentes sucesivos. El flujo de datos es conducido por datos y todo el sistema se descompone en componentes de origen de datos, filtros, tuberías, y los sumideros de datos. Las conexiones entre los módulos son de flujo de que puede ser flujo

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

de bytes, caracteres o cualquier otro tipo de este tipo. La característica principal de esta arquitectura es su ejecución concurrente (Ramos, 2017).

Los llamados filtros no realizan forzosamente tareas de filtrado, como ser eliminación de campos o registros, sino que ejecutan formas variables de transformación, una de las cuales puede ser el filtrado.

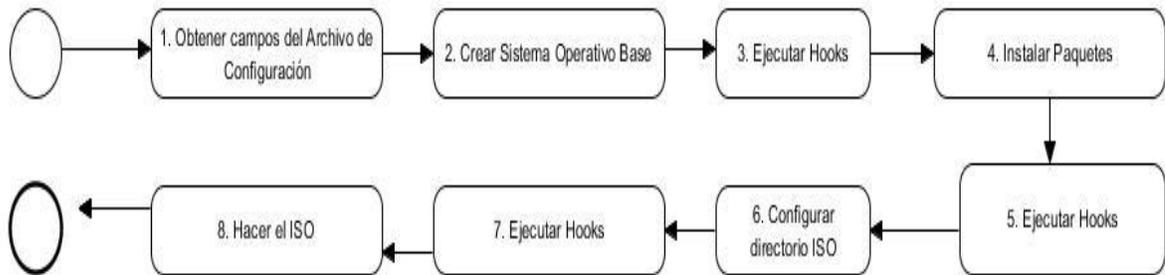
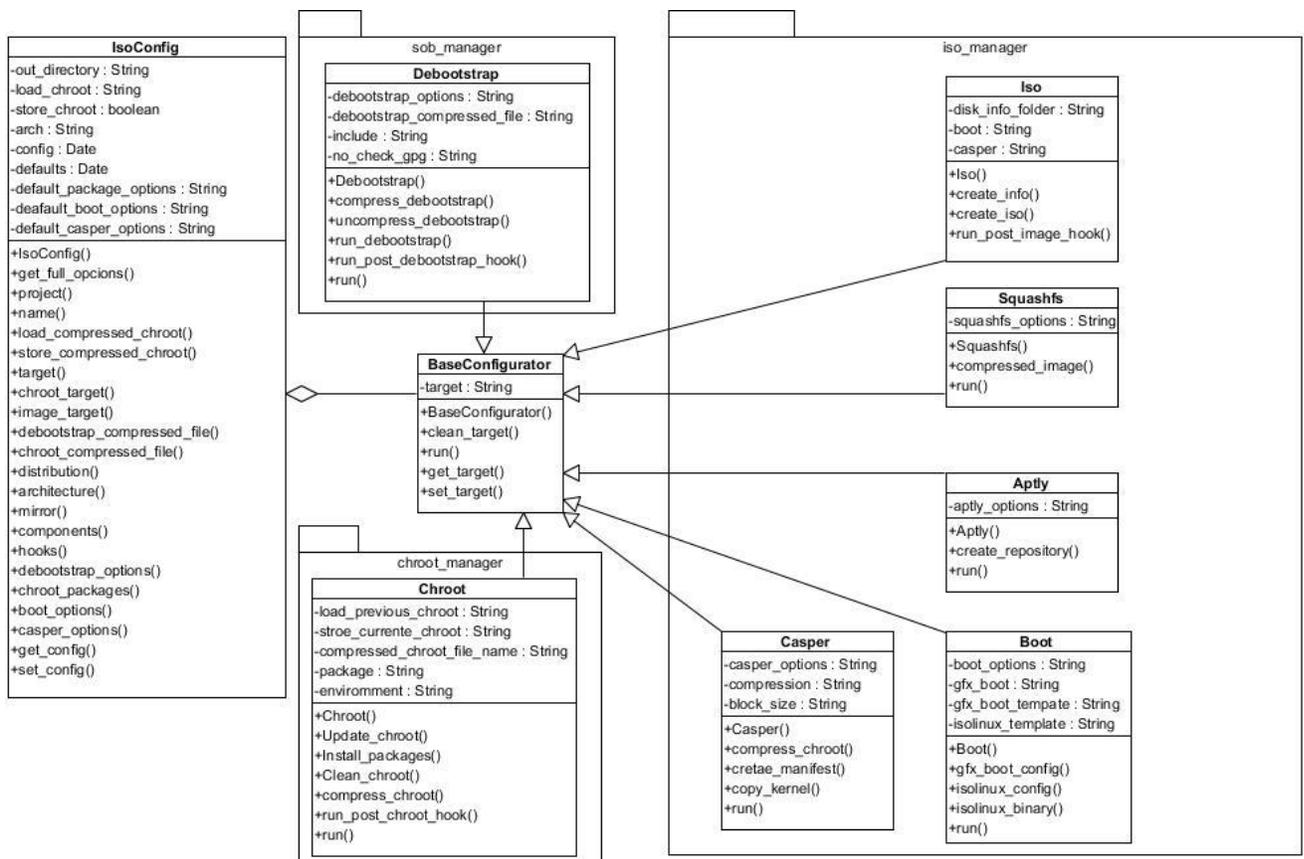


Figura 6: Tuberías y Filtros

(Fuente: Elaboración propia)



Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

Figura 7: Estructura del Sistema

(Fuente: Elaboración Propia)

2.5 Conclusiones del Capítulo

El análisis y diseño de la propuesta de solución permitió realizar una descripción detallada de las características de la Herramienta Informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova, lo que permitió un mejor entendimiento para la fase de implementación al tener los principales artefactos para su desarrollo. Se definieron 6 requisitos funcionales y 4 no funcionales los cuales proporcionan una guía de desarrollo de las funcionalidades de la herramienta.

Se seleccionó como arquitectura Flujo de Datos ya que esta responde a las necesidades de la herramienta Informática.

La definición del estilo arquitectónico y la realización de los diagramas de clases permitieron un mejor entendimiento del funcionamiento del sistema.

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

CAPÍTULO 3: IMPLEMENTACIÓN, PRUEBAS Y EVALUACIÓN DE LA HERRAMIENTA INFORMÁTICA PARA LA CREACIÓN DE PERSONALIZACIONES DE LA DISTRIBUCIÓN CUBANA GNU/LINUX NOVA

3.1 Introducción

El presente capítulo se enfoca en la construcción del sistema a partir de los resultados de la disciplina de Análisis y Diseño. Se elaboran los diagramas de componentes y de despliegue y se definen los estándares de codificación a utilizar en la implementación de la solución. Además, se realizan pruebas de software con el objetivo de descubrir y corregir errores y se evalúa la propuesta de solución.

3.2 Implementación

La etapa de implementación del software es el proceso de convertir una especificación del sistema en un sistema ejecutable (Sommerville, 2011). Esta fase comprende la materialización, en forma de código, de todos los artefactos, descripciones y arquitectura propuestos en la etapa de análisis y diseño; con el objetivo de conformar el producto final requerido por el cliente (Larman, 2004).

Una vez desarrollado el software, el mismo debe ser sometido a una serie de pruebas que muestren que el sistema se ajusta a su especificación y que cumple con las expectativas del cliente. Según (Sommerville, 2011), esta etapa es conocida como la validación e implica una serie de procesos de comprobación, como las inspecciones y revisiones.

3.3 Estándares de codificación

Los estilos de programación, también llamado estándares de código, es un término que describe convenciones para escribir código fuente en un cierto lenguaje de programación. El estilo de programación es dependiente del lenguaje de programación que se haya elegido. Ellos garantizan el mejoramiento de la comunicación en los equipos de desarrollo de software, reducen los errores de programación y mejoran la calidad del software. Todo esto repercute en la competitividad de las empresas de software y en la productividad de sus trabajadores porque se mejora su facilidad de mantenimiento teniendo un gran impacto en la reducción de los costos de mantenimiento (PEP8, 2015).

Para la codificación de la propuesta de solución se utiliza el estándar de codificación para Python utilizado en el Universidad de las Ciencias Informáticas. En este documento se listan distintas convenciones utilizadas en el código Python comprendido en la librería estándar de la distribución principal de Python (PEP8, 2015).

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

Indentación (Sangría):

Se utilizaron 4 espacios por cada nivel de indentación.

Tabuladores o espacios:

No se mezclaron tabuladores y espacios en la codificación. Las formas más populares de indentar en Python es utilizando sólo espacios o sólo tabuladores, el código indentado con una mezcla de tabuladores y espacios se reformateó y se usaron espacios exclusivamente.

Tamaño máximo de línea:

Todas las líneas están limitadas a un máximo de 79 caracteres.

Convenciones de nombres:

- No se utilizaron los caracteres 'l' (letra ele minúscula), 'O' (letra o mayúscula) o 'I' (letra i mayúscula) como nombres de variables de un solo caracter debido a que, en algunas fuentes, estos caracteres son indistinguibles de los numerales uno y cero.
- Se utilizó CapWords (palabras que comienzan con mayúsculas) para los nombres de las clases.
- Los nombres de funciones están en letras minúsculas, con palabras separadas mediante guiones bajos según sea necesario para mejorar la legibilidad.

```
class Debootstrap:
    """
    debootstrap --arch=ARCH --include=alpha,beta --components=alpha,beta \
    SUITE TARGET [MIRROR [SCRIPT]]
    """
    def __init__(self):
        self.debootstrap_options = get_config().debootstrap_options
        self.target = path.join(get_config().target, 'chroot')
        self.include = ""
        if 'include' in self.debootstrap_options:
            self.include = self.debootstrap_options['include']
```

CapWords

Espacios a cada lado de los operadores

Minúsculas separadas mediante guiones bajo.

Figura 8: Aplicación de los Estándares de Codificación

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

(Fuente: elaboración propia)

3.4 Diagrama de Componentes

El diagrama de componentes representa cómo va a estar dividido los componentes y sus dependencias en la propuesta de solución. Un componente de software es considerado unidades autónomas encapsuladas dentro de un sistema que proporciona una o varias interfaces. Los diagramas de componentes son generalmente dirigidos al personal de aplicación de un sistema, además representa una comprensión temprana del sistema global que se está construyendo (Bejerano, 2019).

A continuación, se muestra en la Figura 9 el Diagrama de Componentes de la propuesta de solución.

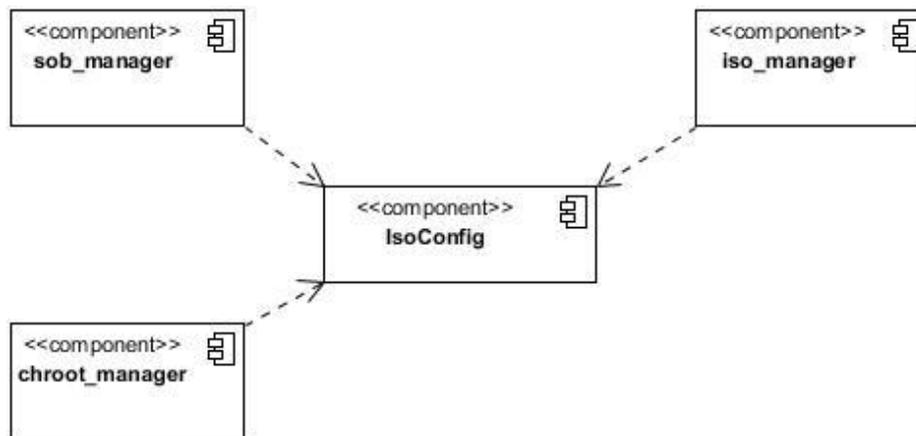


Figura 9: Diagrama de Componentes de la Propuesta de Solución

(Fuente: elaboración propia)

Descripción de los Componentes:

- **sob_manager:** contiene el Sistema Operativo Base.
- **Chroot_manager:** contiene la instalación de paquetes.
- **Iso_manager:** contiene el ISO configurado y comprimido.
- **IsoConfig:** permite analizar sintácticamente el archivo de configuración.

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

3.5 Diagrama de Despliegue

El diagrama de despliegue consiste en una representación estructural de la arquitectura del sistema desde el punto de vista de la distribución de los artefactos del software en los destinos de despliegue; definiendo a los artefactos como representaciones de elementos concretos en el mundo físico que son el resultado de un proceso de desarrollo (UCC, 2017)

A continuación, se muestra en la Figura 10 el Diagrama de Despliegue de la propuesta de solución.

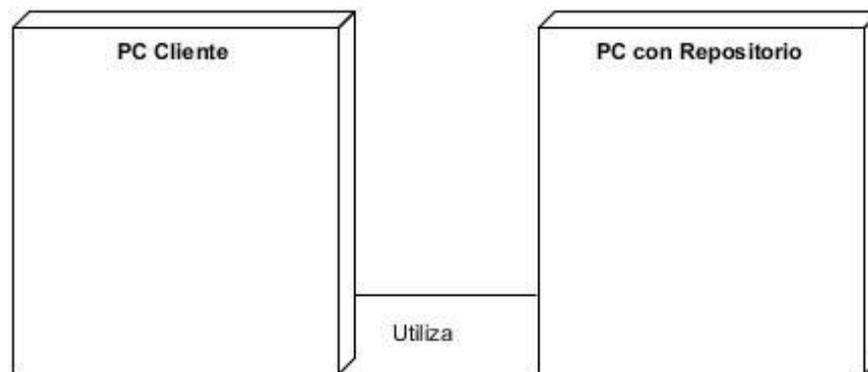


Figura 10: Diagrama de Despliegue

(Fuente: elaboración propia)

Descripción de los nodos:

- **PC Cliente:** contiene la Herramienta informática para la creación de personalizaciones, el Archivo de Configuración creado por el usuario y la carpeta que contiene los Hooks que permiten realizar las configuraciones adicionales.
- **PC con Repositorio:** es el repositorio de Nova y proporciona el ISO y los paquetes necesarios para la instalación.

3.6 Pruebas de software

Luego del desarrollo de la propuesta de solución, se hace necesario verificar y validar el sistema implementado a través de una estrategia de pruebas que permitan comprobar el cumplimiento de las especificaciones del diseño y de la codificación e identificar los posibles errores cometidos.

Las pruebas de software son un conjunto de herramientas, técnicas y métodos que evalúan el desempeño y la calidad de un software, así como evaluar los resultados. Las técnicas son variadas y se realizan

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

desde el personal de prueba hasta herramientas automatizadas que facilitan el trabajo y el rango de tiempo en que se realizan estas (Hidalgo, 2017).

3.6.1 Pruebas a realizar

Pruebas Funcionales

Las pruebas funcionales se concentran en las acciones visibles para el usuario y en la salida del sistema que este puede reconocer. La validación se alcanza cuando el software funciona de tal manera que satisface las experiencias razonables del cliente (Pressman, 2010).

Pruebas unitarias

Las pruebas unitarias se concentran en el esfuerzo de verificación de la unidad más pequeña del software, el componente o módulo de software. Tomando como guía de partida la descripción del diseño a nivel de componentes, se prueban importantes caminos de control para descubrir errores dentro de los límites del código. Centran su actividad en verificar la funcionalidad y la estructura (lógica interna) de cada elemento individualmente, una vez que ha sido codificado (Pressman, 2010).

3.6.2 Métodos de prueba

Caja blanca permite realizar verificaciones y validaciones directamente con el código fuente, para ello las **pruebas unitarias** son realizadas por el programador cada vez agrega una funcionalidad o método para asegurar la calidad del código. Esta técnica se basa en el diseño de casos de prueba que usa la estructura de control del diseño procedimental para derivarlos.

Permite al ingeniero de software obtener casos de prueba que:

1. Garanticen que se ejerciten por lo menos una vez todos los caminos independientes de cada módulo, programa o método.
2. Ejerciten todas las decisiones lógicas en las vertientes verdaderas y falsas.
3. Ejecuten todos los bucles en sus límites operacionales.
4. Ejerciten las estructuras internas de datos para asegurar su validez.

Se considera a la prueba de **caja blanca** como uno de los tipos de pruebas más importantes que se le aplican al software, logrando como resultado que disminuya en un gran porcentaje el número de errores existentes en los sistemas y por ende una mayor calidad y confiabilidad (Pressman, 2010) .

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

3.6.3 Técnicas de prueba

Partición de equivalencia (posibles valores divididos en clases, valores de entrada y valores de salida). Se agrupan todos los valores para los cuales se espera que el programa tenga un comportamiento común (rango de valores) y esa es una clase de equivalencia, existen clases de equivalencia válidas y clases de equivalencia inválidas (FING, 2018)

Camino Básico permite obtener una medida de la complejidad lógica de un diseño procedimental y que use esta medida como guía para definir un conjunto básico de rutas de ejecución. Los casos de pruebas derivados para ejercitar el conjunto básico deben garantizar que se ejecuta cada instrucción por lo menos una vez durante la prueba (Pressman, 2010).

3.7 Aplicación de las pruebas de software

3.7.1 Pruebas internas

Pruebas Unitarias

Las pruebas unitarias se desarrollaron utilizando la técnica del camino básico del método de prueba caja blanca. En esta técnica se utilizó la métrica del software complejidad ciclomática que proporciona una medida cuantitativa de la complejidad lógica de un programa o procedimiento. Cuando se utiliza en el contexto de prueba el valor calculado mediante la complejidad ciclomática define el número de caminos independientes en el conjunto básico de un programa o procedimiento y proporciona un límite superior para el número de pruebas que se deben realizar para asegurar que se ejecuta cada sentencia al menos una vez (Pressman, 2010). En la figura 11 se presenta el procedimiento de la funcionalidad Run dentro del RF. Crear Sistema Operativo Base.

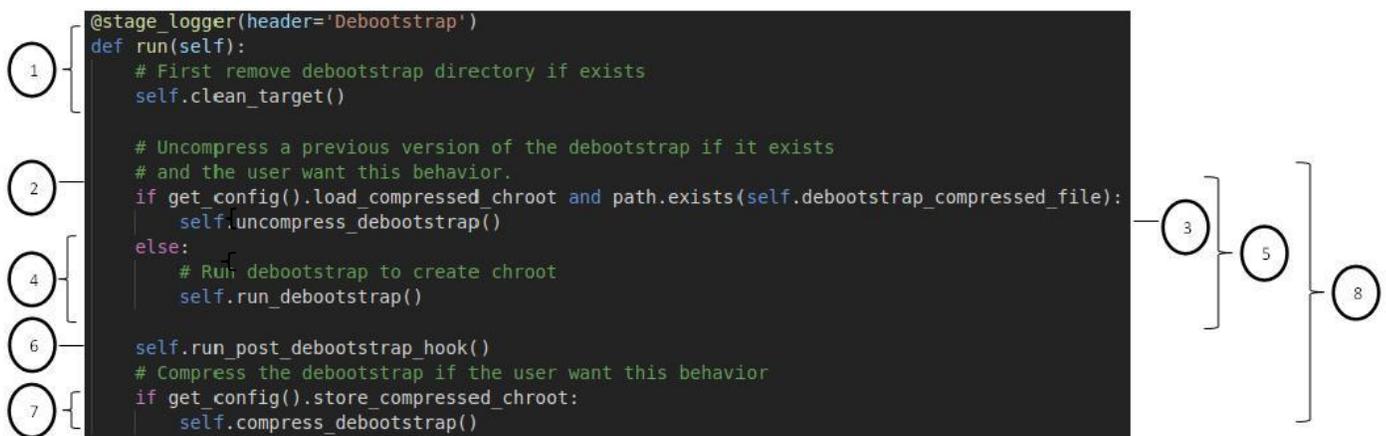


Figura 11: Procedimiento Run del RF Crear Sistema Operativo Base

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

(Fuente: elaboración propia)

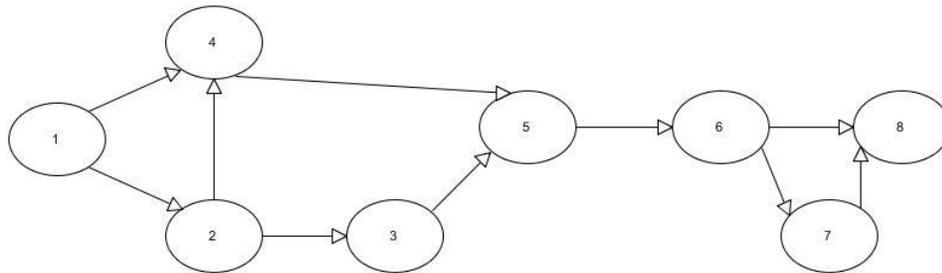


Figura 12: Grafo de flujo

(Fuente: elaboración propia)

La complejidad ciclomática es una medición de software que proporciona una evaluación cuantitativa de la complejidad lógica de un programa. Cuando se usa en el contexto del método de prueba de la ruta básica o camino básico, el valor calculado por la complejidad ciclomática define el número de rutas independientes del conjunto básico de un programa. Brinda una cota superior para el número de pruebas que debe realizar, a fin de asegurar que todos los enunciados se ejecutaron al menos una vez. La complejidad ciclomática tiene fundamentos en la teoría de gráficos y proporciona una medición de software extremadamente útil. La complejidad se calcula de tres formas:

1. El número de regiones del gráfico de flujo corresponde a la complejidad ciclomática.
2. La complejidad ciclomática $V(G)$ para un gráfico de flujo G se define como:

$V(G) = E - N + 2$ donde E es el número de aristas del gráfico de flujo y N el número de nodos del gráfico de flujo.

$V(G) = P + 1$ donde P es el número de nodos predicado¹⁵ contenidos en el gráfico de flujo G .
(Pressman, 2010)

En el gráfico de flujo anterior, la complejidad ciclomática es calculada usando cada uno de los algoritmos indicados:

1. El gráfico de flujo tiene tres regiones.
2. $V(G) = 10$ aristas - 8 nodos + 2 = 4.

$$V(G) = 2 \text{ nodos predicado} + 1 = 3.$$

15 Un nodo predicado es el que representa una condicional if o case, es decir, que de él salen varios caminos.

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

La complejidad ciclomática del gráfico de flujo asociado al método Run () tiene valor 3. V (G) proporciona la cota superior sobre el número de rutas linealmente independientes a través de la estructura de control del programa. Por tanto, las rutas o caminos posibles son las siguientes:

Tabla 6: Rutas básicas del grafo de flujo

(Fuente: elaboración propia)

Número de ruta	Rutas Básicas
1	1-2-3-5-6-8
2	1-2-4-5-6-7-8
3	1-3-5-6-7-8

Una vez determinadas las rutas o caminos básicos de flujo, se pasa a ejecutar los casos de pruebas para cada camino resultante. Los datos deben elegirse de modo que las condiciones en los nodos predicados se establezcan de manera adecuada conforme se prueba cada ruta. Cada caso de prueba se ejecuta y compara con los resultados esperados. Una vez completados todos los casos de prueba, el examinador puede estar seguro de que todos los enunciados del programa se ejecutaron al menos una vez.

Tabla 7: Caso de prueba Ruta Básica 1

(Fuente: elaboración propia)

Ruta Básica 1: 1-2-3-5-6-8	
Descripción	Proceso para correr todos los métodos del Debootstrap
Condición de ejecución	El usuario haya creado un Debootstrap anteriormente y no desea comprimir al final del proceso el Debootstrap.
Resultado esperado	La herramienta descomprime el Debootstrap, y ejecuta los Hooks.
Resultado	Satisfactorio.

Después de aplicado el método del camino básico, en el cual se realizaron 3 iteraciones de acuerdo a la cantidad de rutas básicas obtenidas, se concluye que, de los 3 casos de pruebas realizados (los restantes se encuentran en los Anexos), todos tuvieron resultados satisfactorios. Las no conformidades detectadas fueron resueltas, por tanto, la aplicación de la prueba fue satisfactoria en su totalidad.

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

Pruebas Funcionales

Las pruebas funcionales se aplican a un software determinado, con el objetivo de verificar que las funcionalidades implementadas se desempeñan de acuerdo a las especificaciones de los requisitos definidos con anterioridad. Al conocer las funciones específicas que se le asignaron a una aplicación para su realización, pueden llevarse a cabo casos de pruebas que demuestren que cada función es completamente operativa mientras que al mismo tiempo se buscan errores en cada función (Pressman, 2010). Este tipo de pruebas es basado en requisitos y se vincula a los datos de entrada.

A continuación se muestra el Caso de Prueba: Crear Sistema Operativo Base que contienen clases de equivalencia válida y no válida, para cada campo de entrada del sistema.

Escenario	Descripción	Arquitectura	Distribución	Componentes	Mirror	Respuesta del Sistema	Flujo Central
EC 1.1 Crear Sistema Operativo Base con datos introducidos por el usuario.	El sistema permite crear Sistema Operativo Base en caso de que no exista ningún error.	Válido	Válido	Válido	Válido	Crea Sistema Operativo Base con datos entrados por el usuario.	1. Leer el archivo de configuración entrado por el usuario. 2. Crear Sistema Operativo Base.
		amd64	2015	extendido	http://novarepo.cu/nova		
			Válido 2017				
		Válido	Válido	Válido			
		i386	2019	principal			
EC 1.2 Crear Sistema Operativo Base insertando valores inválidos.	El sistema permite verificar si existen valores inválidos.	Inválido	Inválido	Inválido	Inválido	No crea Sistema Operativo Base, muestra errores donde se introdujo campos inválidos.	1. Leer el archivo de configuración entrado por el usuario. 2. Verificar si existen campos vacíos. 3. Enviar error en línea de comandos en caso de que hayan valores inválidos para que el usuario arregle el error en el campo de configuración.
		i485	2021	estático	http://nova.cu		
EC 1.3 Crear Sistema Operativo dejando campos vacíos.	El sistema permite verificar si existen campos vacíos.	Válido	Válido	Válido	Válido	Crea Sistema Operativo Base con datos entrados por el usuario y datos por	1. Leer el archivo de configuración entrado por el usuario. 2. Verificar si existen campos vacíos. 3. Crear Sistema Operativo Base
		vacío	vacío	vacío	vacío		
		Válido	Válido	Válido	Válido		
		amd64	2019	extendido	http://novarepo.cu/nova		

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

				Válido principal		defecto entrados por el sistema en caso de que no hayan sido rellenados por el usuario.	con los datos entrados por el usuario y tomados por el sistema como por defecto.
EC 1.4 El Sistema Operativo Base ya se encuentra creado.	El sistema le permite al usuario cargar un debootstrap que ya ha sido creado anteriormente	Válido	Válido	Válido	Válido	El Sistema le permite al usuario en caso de que así lo desee cargar un debootstrap anteriormente creado.	1. El usuario le introduce al sistema la ruta donde se encuentra el Debootstrap anteriormente creado. 2. El sistema carga de la ruta introducida por el usuario el Debootstrap.
		amd64	2015	extendido	http://n ovarepo .cu/nov a		
			Válido				
			2017				
		Válido	Válido	Válido			
i386	2019	principal					

Tabla 8 Diseño de Caso de Prueba: Crear Sistema Operativo Base (Fuente: elaboración propia)

Resultados de las pruebas

Como resultado final de las pruebas funcionales, se obtuvo, en una primera iteración, un total de (10) no conformidades de las cuales fueron resueltas 6 y quedaron pendiente 4. En la segunda iteración se detectaron 3 no conformidades que fueron resueltas y también las 3 pendientes de la primera iteración y en la tercera iteración no se identificaron nuevas no conformidades, obteniendo, de esta manera, resultados satisfactorios. La siguiente gráfica, muestra los resultados antes descritos:

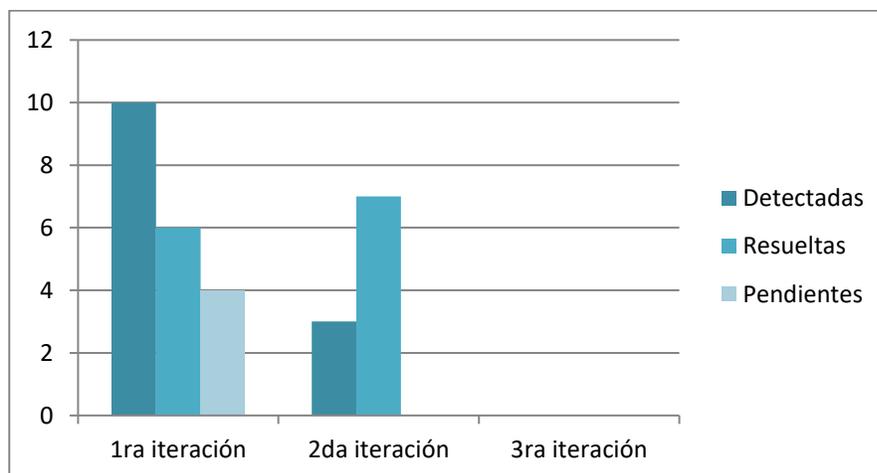


Figura 13. Resultados de las pruebas funcionales

(Fuente: elaboración propia)

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

3.7.2 Pruebas de Aceptación

Se aplica este tipo de prueba con el objetivo de determinar, por parte del cliente, la aceptación o rechazo del sistema desarrollado. Este tipo de prueba es realizada por el usuario final en lugar de los ingenieros de software. Una prueba de aceptación puede variar desde una prueba de conducción informal hasta una serie de pruebas planificadas y ejecutadas sistemáticamente. La prueba de aceptación puede realizarse durante un periodo de semanas o meses, y mediante ella descubrir errores acumulados que con el tiempo puedan degradar el sistema. (León, 2018)

Entre los tipos de pruebas de aceptación que existen en este caso en particular se realizan las pruebas **alfa** que son las que se desarrollan en la propia compañía desarrolladora del software con el equipo de pruebas del software. A continuación, se muestra el Caso de prueba correspondiente RF_2 Instalar paquetes.

- **Código:** identificador de la prueba realizada sugerente a la HU a la que hace referencia.
- **Nombre:** nombre de la prueba a realizar.
- **Nombre del probador:** nombre de la persona que realiza la prueba.
- **Descripción:** se describe la funcionalidad que se desea probar.
- **Condiciones de Ejecución:** muestra las condiciones que deben cumplirse para poder llevar a cabo el caso de prueba, estas condiciones deben ser satisfechas antes de la ejecución del caso de prueba para que se puedan obtener los resultados esperados.
- **Entradas/Pasos de Ejecución:** descripción de cada uno de los pasos seguidos durante el desarrollo de la prueba, se tiene en cuenta cada una de las entradas que hace el usuario con el objetivo de ver si se obtiene el resultado esperado.
- **Resultado esperado:** breve descripción del resultado que se espera obtener con la prueba realizada.
- **Evaluación de la prueba:** acorde al resultado de la prueba realizada se emite una evaluación sobre la misma. Esta evaluación tiene uno de los dos valores que a continuación se describen:
 - Satisfactorio
 - Insatisfactorio

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

Caso de prueba aceptación		
Código de caso de prueba: RF_2	Nombre del requisito: Instalar paquetes.	
Nombre de la persona que realiza la prueba: Juan Manuel Fuentes		
Descripción de la prueba: Prueba la funcionalidad instalar paquetes		
Entrada/Pasos de ejecución: Luego de crearse el Sistema Operativo Base, la herramienta ejecuta el chroot que es el encargado de instalar los paquetes. <ol style="list-style-type: none"> 1. Descomprimir el Debootstrap. 2. Montar el Sistema de Archivos. 3. Actualizar chroot. 4. Instalar paquetes con datos introducidos por el usuario. 		
Escenarios:	Resultados Esperado	Evaluación de la prueba:
EC 2.1 Instalar paquetes introducidos por el usuario.	Instala Paquetes con datos introducidos por el usuario.	Satisfactoria.
EC 2.2 Instalar paquetes insertando valores inválidos.	Instala paquetes válidos, en caso de que no exista o contiene algún error dicho paquete el sistema no los ejecuta.	Satisfactoria.
EC 2.3 Instalar paquetes dejando campos vacíos.	Crea Sistema Operativo Base con datos entrados por el usuario y datos por defecto entrados por el sistema en caso de que no hayan sido rellenados por el usuario.	Satisfactoria.

Tabla 9: Caso de prueba. Instalar paquetes

(Fuente: elaboración propia)

Resultados de las pruebas

Las pruebas de aceptación arrojaron como resultado que no existía ninguna no conformidad por parte del cliente. Los resultados en los casos de pruebas fueron satisfactorios por lo que se acepta el producto.

3.8 Evaluación del objetivo de la investigación

Para la validación del resultado, se utilizó la **Técnica de ladov**: Aplicada a una muestra de especialistas del proyecto NOVA para medir mediante el Índice de Satisfacción Grupal (ISG) el nivel de aceptación de la herramienta informática desarrollada, por parte de los usuarios finales.

3.8.1 Resultados de la técnica de ladov

Cuando se realiza una propuesta, es recomendable retroalimentarse con la opinión de los usuarios potenciales. Esta información es útil para conocer las debilidades de la propuesta y profundizar en sus

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

fortalezas. En ese sentido, **la técnica de ladov** es un instrumento que ayuda a conocer el grado de satisfacción de los potenciales usuarios (Silega, 2017).

Mediante esta técnica se puede determinar, de forma indirecta, Índice de Satisfacción Grupal (ISG) de los individuos involucrados en el proceso que está siendo objeto de análisis. La técnica de ladov se basa en un cuestionario conformado por cinco preguntas, de las cuales tres son cerradas y dos abiertas. Las preguntas cerradas guardan una relación entre sí, que previamente no es de conocimiento por parte del sujeto al que se le aplica la técnica (Ramírez, 2018).

Estas tres preguntas se relacionan a través del "Cuadro Lógico de ladov" el cual permite ubicar a cada encuestado, según el cuadro lógico en una escala de satisfacción, para luego calcular el ISG. Las respuestas a cada una de estas preguntas permiten determinar la posición de cada sujeto en la escala de satisfacción que toma valores desde 1 hasta 6, distribuido de la siguiente manera: 1-Clara satisfacción, 2-Más satisfecho que insatisfecho, 3-No definida, 4-Más insatisfecho que satisfecho, 5-Clara insatisfacción y 6-Contradictoria (Ramírez, 2018). El cuadro lógico utilizado en la investigación se muestra en la siguiente tabla.

¿Qué tan satisfecho se siente con los resultados de la herramienta informática para la creación de personalizaciones de Nova?	¿Estima provechosa la herramienta informática desarrollada para el trabajo en el proyecto Nova?								
	Sí			No sé			No		
	¿Utilizaría la herramienta informática para la creación de personalizaciones de Nova?								
	Sí	No sé	No	Sí	No sé	No	Sí	No sé	No
Me satisface	1	2	6	2	2	6	6	6	6
Me resulta más satisfactorio que insatisfactorio	2	2	3	2	3	3	6	3	6
Me son indiferentes	3	3	3	3	3	3	3	3	3
Me resulta más insatisfactorio que satisfactorio	6	3	6	3	4	4	3	3	4
No me satisfacen en lo absoluto	6	6	6	6	4	4	6	6	5
No sé decir	2	3	6	3	3	3	6	6	4

Tabla 10 Cuadro lógico de ladov utilizado

(Fuente: Elaboración propia)

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

Para medir el grado de satisfacción de los usuarios respecto al sistema desarrollado, se tomó como muestra a siete especialistas vinculados del proyecto Nova. La selección se realizó teniendo en cuenta la experiencia como analista y desarrolladores en el proceso de creación de la distribución cubana GNU/Linux Nova. Los resultados obtenidos para la satisfacción de forma individual se exponen a continuación.

A partir de la cantidad de respuestas por categoría es posible calcular el Índice de Satisfacción Grupal (ISG) siguiendo la siguiente fórmula:

$$ISG = \frac{A(+1) + B(+0.5) + C(0) + D(-0.5) + E(-1)}{N}$$

Las variables representan las cantidades de participantes agrupados por las escalas del índice de satisfacción individual. La cantidad de participantes que expresaron tener una clara satisfacción son representados por A, la cantidad que se sienten más satisfechos que insatisfechos se expresan mediante B, no definido y contradicción se evidencia mediante C, los que se sienten más insatisfechos que satisfechos mediante D, E es la cantidad de participantes que expresan una clara insatisfacción. El valor de N representa el total de participantes.

El valor del ISG permite identificar las siguientes categorías grupales:

- Máxima insatisfacción: -1
- Más insatisfecho que satisfecho: -0.5
- No definido y contradictorio: 0
- Más satisfecho que insatisfecho: 0.5
- Máximo de satisfacción: +1

Esta técnica permite determinar el índice de satisfacción grupal (ISG), que representa los niveles de satisfacción en una escala numérica que abarca el intervalo desde -1 hasta 1. Los valores que se encuentran comprendidos entre -1 y -0,5 indican insatisfacción; los comprendidos entre -0,49 y +0,49 evidencian contradicción y los que están entre 0,5 y 1 indican que existe satisfacción.

Resultados obtenidos

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

1. Los resultados obtenidos de la aplicación de la encuesta se presentan a continuación:

Categorías grupales de satisfacción	Escala	Participantes en la escala
Clara satisfacción	A	4
Más satisfecho que insatisfecho	B	3
No definido	C	0
Más insatisfecho que satisfecho	D	0
Clara insatisfacción	E	0
Contradictorio	C	0

Tabla 11 Escala del índice de satisfacción individual

Fuente: *Elaboración propia*

2. Cálculo del Índice de Satisfacción Grupal

$$\text{ISG} = \frac{A(+1) + B(+0.5)}{N}$$

$$\text{ISG} = \frac{4(+1) + 3(+0.5)}{7} = 0.79$$

3. Interpretación del resultado del ISG

El proceso de validación mediante la técnica de ladov de la consulta a los usuarios donde se ha implementado la herramienta informática para la creación de personalizaciones de Nova propuesta, confirmó su factibilidad de uso, expresado cuantitativamente en el alto Índice de Satisfacción Grupal (ISG= 0.79) y cualitativamente en los criterios emitidos donde evidencian su satisfacción por la contribución del sistema, lo que refleja aceptación de la propuesta y un reconocimiento a su utilidad. Las respuestas a las preguntas abiertas brindadas por los encuestados reafirman los beneficios que traerá la utilización del sistema propuesto. Por lo anteriormente planteado se puede afirmar que se cumplió el objetivo de la investigación.

El formulario presentado a los participantes incluye dos preguntas abiertas, mostradas a continuación:

1. ¿Qué importancia le concede a la herramienta informática para la creación de personalizaciones de Nova?
2. ¿Qué aspectos a su juicio potencian o limitan el uso de la herramienta informática para la creación de personalizaciones de Nova?

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

Sobre la primera pregunta, los participantes manifestaron que resulta de gran importancia porque permite crear de manera automatizada la creación de personalizaciones de la distribución cubana GNU/Linux Nova.

Sobre la segunda pregunta los participantes expresaron que la creciente necesidad de mejorar el proceso de creación de personalizar la distribución cubana GNU/Linux Nova.

3.9 Conclusiones del Capítulo

En el capítulo concluido se realizó la elaboración del diagrama de componentes, facilitó la comprensión de la estructura general del sistema a través de sus componentes.

Se establecieron los estándares de codificación a tener en cuenta para la implementación de la Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova, permitiendo una mejor legibilidad y comprensión del código, facilitando su mantenimiento.

La implementación del sistema permitió la obtención de una aplicación funcional y completamente operativa.

La ejecución de la estrategia de pruebas especificada, permitió detectar y corregir deficiencias presentes en la solución y ofrecer una aplicación con mayor calidad, seguridad y usabilidad.

Se evidenció la satisfacción de la propuesta de solución a partir de la técnica de ladov, la cual propició la evaluación satisfactoria de la herramienta para la creación de personalizaciones de Nova.

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

CONCLUSIONES GENERALES

La investigación realizada cumple con los objetivos planteados mediante el desarrollo de la Herramienta Informática para la creación de personalizaciones de la Distribución Cubana GNU/Linux Nova y se arriba a las siguientes conclusiones:

- El análisis de los referentes teóricos y de los sistemas informáticos estudiados evidenció la necesidad de desarrollar una herramienta informática para mejorar el proceso de creación de personalizaciones de la Distribución Cubana GNU/Linux Nova.
- Se obtuvo una herramienta informática que permite mejorar la construcción de personalizaciones de la Distribución Cubana GNU/Linux Nova.
- La evaluación de la investigación mediante las pruebas realizadas garantizan el correcto funcionamiento de la aplicación y demostraron la satisfacción del cliente hacia la herramienta desarrollada.

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

Recomendaciones

- Utilizar UEFI-GPT como tecnología de particionado de disco.
- Utilizar *Docker* para reducir los tiempos de procesamientos del Debootstrap y Chroot.

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

REFERENCIAS

- Martinez. 2015.** PostgreSQL . [En línea] 2 de octubre de 2015. [Citado el: 25 de octubre de 2018.] <http://www.postgresql.org>.
- Abran, Alain and Moore, James W. 2004.** *Swelok. Guide to the Software Engineering Body of Knowledge*. Estados Unidos de América : IEEE Computer Society Professional Practices Committee, 2004. 204.
- Abreu, Ramon. 2016.** 2016.
- Acunetix Vulnerability Scanner. 2016.** Audit Your Web Security with Acunetix Vulnerability Scanner . [En línea] 2016. [Citado el: 22 de 03 de 2019.] <https://www.acunetix.com/vulnerability-scanner/>.
- AFT. 2012.** Administracion. [En línea] 11 de junio de 2012. [Citado el: 2016 de 10 de 10.] <http://www.administracionmoderna.com/2012/06/activo-fijo-tangible.html>.
- Alegre Ramos, María del Pilar. 2019.** Sistemas Operativos Monopuesto. [en línea]. [En línea] 11 de mayo de 2019. [Citado el: 2 de octubre de 2019.] https://books.google.com/cu/books?id=qt-ZDwAAQBAJ&printsec=frontcover&dq=que+es+un+sistema+operativo&hl=es&sa=X&ved=0ahUKEwjH35S_h-rIAhXExFkKHYgdBEAQ6AEINTAC#v=onepage&q=que%20es%20un%20sistema%20operativo&f=false. ISBN 978-84-283-4139-4.
- Almeira, Adriana. 2007.** *Arquitectura de Software: Estilos y Patrones*. Argentina : Universidad Nacional De La Patagonia San Juan Bosco, 2007.
- Álvarez, Miguel Angel. 2017.** DesarrolloWeb. [En línea] 14 de septiembre de 2017. [Citado el: 07 de noviembre de 2018.] <https://DesarrolloWeb.com>.
- Amparo López Gaona. 2012.** El modelo Entidad-Relación. [En línea] Universidad Nacional Autónoma de México. Facultad de Ciencias, 2012. [Citado el: 14 de 04 de 2017.] <http://hp.fciencias.unam.mx/~alg/bd/>.
- Arias, Angel. 2014.** *Aprende sobre la Ingeniería del Software*. [https://books.google.com/cu/books?id=0E3mCgAAQBAJ&pg=PT85&lpg=PT85&dq=entrevista+como+t%C3%A9cnica+de+identificaci%C3%B3n+de+requisitos+del+software&source=bl&ots=BiZALVmCtd&sig=ACfU3U3PyqFqdi9UokZnVWtkf-GO6yrk2A&hl=es&sa=X&ved=2ahUKEwIjz922uqznAhVyuVkkHZ] s.l. : IT Campus Academy, 2014. 978-1497417656.
- Aroche, Eugenio Durán. 2012.** *Sistema de Construcción de Personalizaciones de Nova*. Universidad de las Ciencias Informáticas. La Habana : s.n., 2012. Trabajo de diploma para optar por el título de Ingeniero en Ciencias Informáticas.
- . 2014. Trabajo de diploma para optar por ingeniero en Ciencias Informáticas. *Sistema de Construcción de Personalizaciones de Nova*. La Habana : Universidad de las Ciencias Informáticas, 2014.
- Arrete, Juan Pablo. 2016.** Una introducción a MongoDB. [En línea] 09 de marzo de 2016. [Citado el: 14 de enero de 2018.] <https://www.blogs-informaticos.com/MongoDB>.

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

Bartak, Pedro. 2013. [En línea] 9 de 12 de 2013. [Citado el: 5 de 4 de 2017.]

<http://www.grandespymes.com.ar/2013/12/29/definicion-y-componentes-del-sistema-de-control-de-gestion-en-las-empresas/>.

Bejerano, Yunier Pñerez. 2019. *Sistema de Gestión de Licencias del Personal Aeronáutico del Instituto de la Aeronáutica Civil de Cuba versión 2.0.* Universidad de las Ciencias Informáticas. La Habana : s.n., 2019. Trabajo de diploma para optar por el título de Ingeniero en Ciencias Informáticas.

Berberat, Esteve. 2012. *Visual Paradigm.* Universidad de Madrid. Madrid : s.n., 2012. Tesis de Maestría.

Boltton, Vladimir. 2017. UML_Secuencia. [En línea] 15 de 02 de 2017. [Citado el: 13 de 02 de 2019.]

<https://uml.org/uml-secuencia>.

Bresnahan, Christine y Blum, Richard. 2015. LPIC-1: Linux Professional Institute Certification Study Guide. [En línea] 2015. [Citado el: 22 de septiembre de 2019.]

<https://books.google.com.cu/books?id=jf3zBgAAQBAJ&pg=PA140&dq=fdisk&hl=es&sa=X&ved=0ahUKEwiS3cuM5JmAhUSvVkkKHU6qAHMQ6AEIUTAE#v=onepage&q=fdisk&f=false>. ISBN 978-1-119-02118-6.

Brito, Eugenio González. 2015. Pruebas de rendimiento de software. [En línea] 18 de 07 de 2015. [Citado el: 21 de 03 de 2019.]

https://cgrw01.cgr.go.cr/rup/RUP.es/LargeProjects/core.base_rup/guidances/concepts/performance_testing_37A31809.html.

Cabriales, Andy Michel Moreno. 2013. *Manejador de Copias para NOVA 4.0.* Universidad de las Ciencias Informáticas. La Habana : s.n., 2013. Título para optar por el título como Ingeniero en Ciencias Informáticas.

Canales, Pedro Ramos. 2017. Visual Paradigm. *Cómo se realizan los diagramas en Visual Paradigm.* [En línea] 22 de octubre de 2017. [Citado el: 20 de enero de 2019.] <https://www.adictosaltrabajo.com/tutoriales/vparadigm/>.

CELERY. 2017. Celery: Distributed Task Queue. [En línea] 20 de octubre de 2017. [Citado el: 4 de mayo de 2019.]

https://celery/distributed_task_queue. ISBN: 974-48-448-1.

Chakray. 2017. ¿Qué es el BPMN y para que sirve? [En línea] 20 de septiembre de 2017. [Citado el: 10 de noviembre de 2019.] <https://www.chakray.com/es/que-es-el-bpmn-y-para-que-sirve/>.

Condori, Jose Luis. 2016. *Python - DjangoFramework de desarrollo web para perfeccionistas* Basado en el Modelo MTV. 2016.

Creed, Manuel. 2017. Marco de Desarrollo de la Junta de Andalucía. [En línea] 2017.

Cruz, Manuel Enrique Peiso. 2018. *Procedimiento de construcción de imágenes de la personalización de GNU/Linux Nova Escritorio.* Universidad de las Ciencias Informáticas. La Habana : s.n., 2018. Trabajo presentado en opción al título de Máster en Informática Aplicada.

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

- CSS, Learn to style HTML using. 2017.** Mozilla Developer Network. [En línea] 23 de 11 de 2017. [Citado el: 25 de 05 de 2018.] <https://developer.mozilla.org/en-US/docs/Learn/CSS..>
- Durango, Alicia. 2015.** *Diseño de software: 2 Edición.* [https://books.google.com/cu/books?id=1ShpCwAAQBAJ&pg=PA15&dq=que+es+un+requisito+funcional&hl=es&sa=X&ved=0ahUKEwj958Grz6znAhXSt1kKHZXBdzEQ6AEIRDAE#v=onepage&q=que%20es%20un%20requisito%20funcional&f=false] s.l. : IT Campus Academy, 2015. ISBN:978-1519620736.
- Echevarría, Dayma. 2010.** *Proceso de perfeccionamiento empresarial.* La Habana : Universidad de la Habana, 2010.
- EPSILONMAG. 2017.** epsilon. *Por que linux es considerado el mejor sistema operativo.* [En línea] 2017. [Citado el: 24 de septiembre de 2019.] [https://epsilonomag.com/pc/por-que-linux-es-considerado-el-mejor-sistema-operativo/.](https://epsilonomag.com/pc/por-que-linux-es-considerado-el-mejor-sistema-operativo/)
- Estela Raffino, María. 2018.** Concepto.de. *Software Libre.* [En línea] 23 de noviembre de 2018. [Citado el: 14 de noviembre de 2019.] [https://concepto.de/software-libre/.](https://concepto.de/software-libre/)
- FERGARCIA. 2013.** Entorno de desarrollo integrado. [En línea] 25 de enero de 2013. [Citado el: 1 de noviembre de 2019.] [https://fergarcia.wordpress.com/2013/01/25/entorno-de-desarrollo-integrado-ide/.](https://fergarcia.wordpress.com/2013/01/25/entorno-de-desarrollo-integrado-ide/)
- Fernández, Carlos Martínez. 2014.** *Metodología de desarrollo para la actividad productiva de la UCI.* Universidad de las Ciencias Informáticas. La Habana : UCI, 2014. Tesis de maestría.
- Ferrer, Laurato Cantínez. 2017.** *Propuesta de conceptualización de requisitos para proyectos software basados en formalismo de ingeniería de conocimiento.* Madrid : Evens Books, 2017. ISBN: 978-48-751-2.
- Ferrer, Vilma Rodríguez. 2016.** SQLite, sistema gestor de base de datos. [En línea] 29 de julio de 2016. [Citado el: 01 de 11 de 2018.] <https://blogs-engineering/Sytms-of-Date-Base.>
- FING. 2018.** Find Edu. *Testing basado en el cubrimiento.* [En línea] 2018. [Citado el: 22 de marzo de 2019.] [https://www.fing.edu.ut/inco/cursos/ingsoft/.](https://www.fing.edu.ut/inco/cursos/ingsoft/)
- Fuente Rodríguez, Juan Manuel. 2018.** Tesis de Maestría en Informática Avanzada. *Procedimiento de gestión de los Repositorios de la Distribución Cubana GNU/Linux Nova.* La Habana, Cuba : Universidad de las Ciencias Informática, 2018.
- Fuentes, Juan Manuel Rodríguez. 2018.** *Procedimiento de Gestión de los Repositorios de la Distribución Cubana de GNU/Linux Nova.* Universidad de Las Ciencias Informáticas. La Habana : s.n., 2018. Tesis de Maestría de Informática Avanzada.
- Gasterol, Yoandri. 2015.** [En línea] 17 de 09 de 2015.
- Gauchat, Juan Diego. 2013.** *Gran libro de html5, css3 y javascript.* España : CEDRO, 2013.
- GIT. 2016.** Git- Control de versiones. [En línea] 26 de marzo de 2016. [Citado el: 25 de marzo de 2019.] <https://git-scm.com/book/es/v1/Empezando-Acerca-del-control-de-versiones..>

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

- Gómez, Maria del Carmen. 2015.** *Notas del curso Base de Datos. MÉXICO, D.F : Casa abierta al tiempo.* 2015.
- González, Dra. Anaisa Hernández. 2013.** *Diagramas de Casos de Uso del Negocio y del Sistema.* Habana, Cuba : Instituto Superior Politécnico Jose Antonio Hechavarría (CUJAE), 2013.
- González, Jesus Ponce, Dominguez, Fransisco Jose, Rodriguez, Javier Gutierrez and Escalona, Maria Jose. 2014.** *Pruebas de aceptación orientadas al usuario: contexto ágil para un proyecto de gestión documental.* España : Universidad de Sevilla, Universidad de Sevilla, 2014. ISSN 1888-0967.
- GPARED. 2017.** gpared. *Manual de GParted.* [En línea] Septiembre de 2017. [Citado el: 1 de noviembre de 2019.] <https://gparted.org/display-doc.php?name=help-manual&lang=es>.
- Granda, Ailec. 2016.** Modelo didáctico para el uso de comunidades virtuales en el proceso de enseñanza aprendizaje de la Disciplina Ingeniería y Gestión de Software en la Universidad de las Ciencias Informáticas. Tesis para obtener el grado de doctora en tecnología educativa. 2016.
- Guerra, Arturo César. 2017.** Obtención de Requerimientos. Técnicas y Estrategia. [En línea] 2017. [Citado el: 12 de diciembre de 2019.] <https://sg.com.mx/revista/17/obtencion-requerimientos-tecnicas-y-estrategia>.
- Gutiérrez, Javier J. 2015.** framework Django. [En línea] 2015.
- Helmke, Matthew, Hudson, Andrew y Hudson, Paul. 2015.** *Ubuntu Unleashed 2015 Edition: Covering 14.10 and 15.04.* [https://books.google.com/cu/books?id=U1RwBQAAQBAJ&pg=PA161&dq=fdisk&hl=es&sa=X&ved=0ahUKEwjxreiz5pjmAhVkrIkKHTYCBBs4ChDoAQhMMAQ#v=onepage&q=fdisk&f=false] Indianapolis : Pearson Education. Inc, 2015. 978-0-672-33837-3.
- Hertzog, Raphael. 2015.** ¿Why a GNU/Linux Distribution? *The Debian Administrator's Handbook.* [En línea] 2015. [Citado el: 10 de octubre de 2019.] <https://debian-handbook.info/browse/stable/sect.why-gnu-linux.html>. ISBN: 979-10-91414-04-3.
- Hertzog, Raphael y Mas, Roland. 2015.** *The Debian Administrator's Handbook.* New York : Debian Free Software Guidelines, 2015. ISBN 979-10-91414-05-0.
- Heurtel, Olivier. 2011.** *Desarrollar un sitio Web dinámico e interactivo.* s.l. : ENI Ediciones, 2011. 497.
- Hidalgo, Asney Palmero. 2017.** *Herramienta para la gestión de perfiles de AppArmor para GNU/Linux Nova servidores.* Universidad de las Ciencias Informáticas. La Habana : s.n., 2017. Título para optar como ingeniero en Ciencias Informáticas.
- Hipertextual. 2015.** *Iniciativas a favor del software libre en América Latina.* [En línea] 2015. [Citado el: 2 de septiembre de 2019.] <https://hipertextual.com/2015/04/politicas-de-software-libre-en-latinoamerica>.
- Hugo, Jorge. 2014.** Analisis de perfiles de usuarios. [En línea] 28 de noviembre de 2014. [Citado el: 17 de octubre de 2018.] <https://www.googlebook.com/Análisis-de-perfiles-de-usuarios>.

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

IEEE. 2014. IEEE Recommended Practice for the Adoption of Computer-Aided Software Engineering (CASE) Tools. *Software Engineering*. 2da, 2014, Vol. 40, 24.

Jacobson, Ivar, Boocch, Grady and Rumbaugh, James. 2000. [aut. libro] Ivar Jacobson. *El proceso unificado de desarrollo de software*. Madrid, España : Pearsons Educacion S.A, 2000.

Jahongir Rahmonov. 2019. PyCharm for Productive Python Development (Guide). [En línea] 28 de agosto de 2019. [Citado el: 23 de diciembre de 2019.] <https://realpython.com/pycharm-guide/>.

Jiménez Toro, J.A. 2015. *UF1876- Atención a usuarios e instalación de aplicaciones cliente*.

[<https://books.google.com.cu/books?id=m7IWDwAAQBAJ&pg=PA72&dq=los+sisistemas+operativos+gen%C3%A9ricos&hl=es&sa=X&ved=0ahUKEwjL77fsxuDnAhXSVt8KHZL0BAQQ6AEIJzAA#v=onepage&q=los%20sisistemas%20operativos%20gen%C3%A9ricos&f=false>] Madrid : ELEARNING, 2015. ISBN 978-84-15199-14-3.

Jiménez, J.A Toro. 2015. *Uf-1876-Atención a usuarios e instalación de aplicaciones cliente*. [En línea] 22 de octubre de 2015. [Citado el: 1 de septiembre de 2019.]

https://books.google.com.cu/books?id=m7IWDwAAQBAJ&pg=PA76&dq=sistema+operativo+espec%C3%ADfico+jimenes+toro&hl=es&sa=X&ved=0ahUKEwjN8eeooZ_oAhWEmuAKHQWoD80Q6AEIKTAA#v=onepage&q=sistema%20operativo%20espec%C3%ADfico%20jimenex%20toro&f=false. ISBN 978-84-15199-14-3.

Jiménez, Jose. 2018. *Unidad 1. Principios del modelado del negocio*. Ciudad de México : Universidad Abierta y a Distancia de México, 2018.

Kaplan-Moss, Adrian Holovaty y Jacob. 2016. *The Definitive Guide to django*. 2016.

Kerlinger, Fred. 2016. perfiles de usuario en redes sociales. *perfiles de usuario en redes sociales*. México D.F : s.n., 2016.

Kili, Aaron. 2018. TecMint.com. *Top 6 Partition Managers for Linux*. [En línea] RHCSA and RHCE Certification Exam Study Book, 9 de febrero de 2018. [Citado el: 24 de octubre de 2019.] <https://www.tecmint.com/linux-partition-managers/>.

Larman, C. 2004. *UML y Patrones: Introducción al análisis y diseño orientado a objetos*. La Habana: Félix Varela. 2004.

—. 2004. *UML y Patrones: Introducción al análisis y diseño orientado a objetos*. La Habana: Félix Varela. 2004.

Larman, Craig. 2003. *UML y Patrones*. s.l. : Prentice Hall 2da Edición, 2003.

Lenguajes de programación. **Kuzko, Richard. 2016.** 2016.

León García, Aldy. 2017. Trabajo de diploma para optar por el título Ingeniero en Ciencias Informáticas. *Control de acceso a recursos compartidos en el Nautilus*. La Habana, Cuba : Universidad de las Ciencias Informáticas, 2017.

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

León, Aldy García. 2017. *Control de acceso a recursos compartidos en el Nautilus*. Universidad de las Ciencias Informáticas. La Habana : s.n., 2017. Trabajo de Diploma para optar por el título de ingeniero en ciencias informáticas.

León, Salvador Trujillo. 2018. *UF2213 - Modelos de datos y visión conceptual de una base de datos*. s.l. : Elearning, S.L, 2018. pág. 280. 978-84-16360-69-7.

Leontis, Neocles. 2012. RNA 3D Structure Analysis and Prediction. [En línea] 2012. [Citado el: 28 de abril de 2019.] <https://books.google.com.cu/books?id=4nVaim7HXmEC&pg=PA113&dq=Novo+Builder&hl=es&sa=X&ved=0ahUKewim1ZbKmpvmAhUBmlkKHcNkATEQ6AEIJzAA#v=onepage&q=Novo%20Builder&f=false>. ISBN 978-3-642-25739-1.

Lewis, William E. 2005. *Software testing and continuous quality improvement*. Estados Unidos : CRC Press LLC, 2005. ISBN: 0-8493-2524-2.

LinuxAdictos. 2017. *Como se compone la estructura del sistema de archivos de linux*. [En línea] 5 de octubre de 2017. [Citado el: 2 de noviembre de 2018.] <https://www.linuxadictos.com/como-se-compone-la-estructura-del-sistema-de-archivos-de-linux-parte-1.html>.

LLonch. 2016. Symphony. [En línea] 2016. [Citado el: 2017 de febrero de 2017.] <http://symfony.com>.

Loisel, Jérôme. 2016. JMeter Tutorial for Beginners. [En línea] 2016. [Citado el: 18 de 03 de 2019.]

Lopes-Martinez, Igor. 2012. 2012.

López, Dailyn Sosa. 2016. Que es un software Libre. [En línea] 22 de octubre de 2016. [Citado el: 2 de noviembre de 2018.] https://google_book/software_libre/. ISBN: 975-84-58-71-1.

López, David. 2014. Genbeta. [En línea] 9 de enero de 2014. <https://www.genbetadev.com/herramientas/netbeans-1>.

López, MSc. Dailyn Sosa. 2016. [En línea] 2016.

MAKETECHASIER. 2010. maketecheasier. *Build Your Own Ubuntu-based Distro With Novo Builder*. [En línea] 2 de Julio de 2010. [Citado el: 23 de octubre de 2019.] <https://www.maketecheasier.com/build-your-own-ubuntu-based-distro-with-novo-builder/>.

MarketShareReport. 2019. *Linux Market Share*. [En línea] 24 de enero de 2019. [Citado el: 19 de octubre de 2019.] <https://netmarketshare.com/linux-market-share>.

Martinez, Alberto. 2015. Las redes sociales en Internet. [En línea] 18 de 12 de 2015. [Citado el: 10 de 11 de 2018.]

Mateo, Sergi. 2015. Manual práctico de benchmarking. [En línea] 2015. [Citado el: 29 de 10 de 2018.]

Molinero Parra, Jose Manuel. 2018. *UF2218-Desarrollo de un CMS*. [<https://books.google.com.cu/books?id=cF5WDwAAQBAJ&pg=PA60&dq=tuber%C3%ADas+y+filtros+es+un+patr%C>]

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

3%B3n+arquitect%C3%B3nico&hl=es&sa=X&ved=0ahUKEwi31ZirwIToAhVdc98KHQjsDXMQ6AEIOjAC#v=onepage&q=tuber%C3%ADas%20y%20filtros%20es%20un%20patr%C3%B3n%20arquitect] 2018. 978-447-58-444-741-1.

Montero, Richard. 2014. Universidad de Antioquia. Escuela Interamericana de Bibliotecología. Seminario de Estudios de Usuarios. Unidad 2. [En línea] 25 de 06 de 2014. [Citado el: 08 de 12 de 2018.]

Naramore, Elizabeth. 2016. [En línea] 2016.

Negus, Christopher. 2014. *Live Linux CDs: Building and Customizing Bootables*. Boston : Safari Book online, 2014. ISBN 0-13-243274-9.

Nikkel, Bruce. 2016. *Practical Forensic Imaging: Securing Digital Evidence with Linux Tools*. [ed.] Alinson Law. San Francisco : Library of Congress Catalogy- in- Publication Data, 2016. ISBN 1-59327-793-8.

Noguera, Bulmaro. 2016. Culturación. Ventajas de utilizar HTML 5. [En línea] 16 de 08 de 2016. [Citado el: 2018 de 10 de 25.]

Nolasco, Jorge Santiago Valenzuela. 2018. *Python Aplicaciones prácticas*. Madrid : RA-MA Editorial, 2018. 978-849964-758-6.

NOVA. 2019. *Soluciones a la medida*. [En línea] Universidad de las Ciencias Informáticas, 2019. [Citado el: 1 de octubre de 2019.] <https://www.nova.cu/>.

Nuñez, Inter Hnos. 2014. importancia de las redes sociales en la sociedad actual. [En línea] 24 de junio de 2014. [Citado el: 05 de octubre de 2018.]

OBSBUSINESS. 2019. *¿Qué son las metodologías de desarrollo de software?* [En línea] Universidad de Barcelona, 26 de octubre de 2019. [Citado el: 22 de noviembre de 2019.] <https://obsbusiness.school/int/blog-project-management/metodologia-agile/que-son-las-metodologias-de-desarrollo-de-software>.

Pacheco, Juan Manuel Fernandez. 2017. Real Academia Española. [En línea] 2017. [Citado el: 07 de 10 de 2018.]

Palmero, Asney Hidalgo. 2017. Trabajo de diploma para optar por ingeniero en Ciencias Informáticas. *Herramienta para la gestión de perfiles de AppArmor para GNU/Linux Nova servidores*. La Habana : Universidad de las Ciencias Informáticas, 2017.

Pantaleo, Guillermo. 2016. *Ingeniería de Software*.

[<https://books.google.com/cu/books?id=a8j2DQAAQBAJ&pg=PT557&dq=modelo+conceptual+en+ingenieria+de+software&hl=es&sa=X&ved=0ahUKEwiv3sOioavnAhXmwVkkKHVg6C9AQ6AEIJzAA#v=onepage&q=modelo%20conceptual%20en%20ingenieria%20de%20software&f=false>] s.l. : Editorial- Ink, 2016. 978-987-1609-78-9.

Parrilla, Castillo y Antonio, José. 2018. *Bienes digitales. Una necesidad europea*.

[https://books.google.com/cu/books?id=ywSCDwAAQBAJ&pg=PA42&dq=importancia+de+los+ordenadores&hl=es&sa=X&ved=0ahUKEwiRx_bKyuDnAhWoc98KHe9HDH0Q6AEIMDAB#v=onepage&q=importancia%20de%20los%20ordenadores&f=false] Madrid : DYKINSON,SL, 2018. ISBN: 978-84-9148-987-0.

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

Patrones de Caso de Uso. **Cuesta, Saúl.** s.l. : SG.

Paz, Arturo Arias. 2017. Control de versiones de Software con GIT. [En línea] 2017. [Citado el: 5 de noviembre de 2019.]

https://books.google.com.cu/books?id=uvw0DgAAQBAJ&printsec=frontcover&dq=que+es+un+control+de+versiones&hl=es&sa=X&ved=0ahUKEwjy3t_pkKvnAhWSmlkKHa3TCgkQ6AEIjzAA#v=onepage&q=que%20es%20un%20control%20de%20versiones&f=false. 978-1544105536.

Peiso, Manuel Enrique Cruz. 2018. *Procedimiento de construcción de imágenes de la personalización de GNU/Linux.* Universidad de las Ciencias Informáticas. La Habana : s.n., 2018. Trabajo final presentado en opción al título de Máster en Informática Avanzada.

Peñalver, Gladys Marsi Romero. 2017. *Ingenierías Ágiles.* Universidad de las Ciencias Informáticas. La Habana : s.n., 2017. Conferencia.

PEP8. 2015. PEP 8-- Style Guide for Python Code. [En línea] 1 de agosto de 2015. [Citado el: 26 de marzo de 2019.] <https://www.python.org/dev/peps/pep-0008/>.

Pérez, Damián Alfonso. 2015. *Técnica para el diagnóstico de variantes de procesos de negocio.* Universidad de las Ciencias Informáticas. La Habana : s.n., 2015. Tesis presentada en opción al título de Máster en Informática Aplicada.

Pérez, Juan Fernández. 2015. *La oportunidad del software libre : capacidades, derechos e innovación.* [https://books.google.com.cu/books?id=2R-M68dD-foC&pg=PA11&dq=software+libre&hl=es&sa=X&ved=0ahUKEwjT3421i5zmAhWm1FkKHdAjAI4Q6AEINDAC#v=onepage&q=software%20libre&f=false] Valencia : Escuela de negocio, 2015. ISBN: 849-87-41-1.

Pilar, María Soler. 2011. programación web. [En línea] 21 de mayo de 2011. [Citado el: 22 de febrero de 2017.] <http://progpagweb.blogspot.com/2011/05/definicion-y-caracteristicas-de-php-con.html>.

Piñeiro, Javier Cárdenas. 2015. *Herramienta web para la creación de personalizaciones de Nova Servidores.* Universidad de las Ciencias Informáticas. La Habana : s.n., 2015. Tesis de fin de curso para optar como ingeniero en Ciencias Informáticas.

Pivotal. 2017. RabbitMQ is the most widely deployed open source message broker. [En línea] 23 de mayo de 2017. [Citado el: 27 de marzo de 2019.] [https:// rabbitmq/what-this/](https://rabbitmq/what-this/).

Polanco, Manuel Ponce. 2012. *Sistema de control de gestión.* Lima, Perú : s.n., 2012.

Pons, Nicolas. 2018. *Linux: principios b'asicos de uso del sistema.* Barcelona : Ediciones ENI, 2018. ISBN 978-2-409-00547-3.

Potencier, Fabien, Weaver, Ryan and Eguluz, Javier. 2014. *Buenas prácticas oficiales de Symfony.* 2014.

Pressman, Roger S. 2010. *INGENIERÍA DEL SOFTWARE. UN ENFOQUE PRÁCTICO.* s.l. : McGRAW-HILL, 2010. págs. 383 - 404. ISBN:978-607-15-0314-5.

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

—. 2010. *INGENIERÍA DEL SOFTWARE. UN ENFOQUE PRÁCTICO*. s.l. : McGRAW-HILL, 2010. págs. 383 - 404. 978-607-15-0314-5.

—. 2010. *Ingeniería del software. Un enfoque práctico*. s.l. : Editores S.A, 2010. 978-0-07-337597-7.

Pressman, Roger S. 2010. *Ingeniería de Software. Un enfoque práctico*. 2010.

Producción, Dirección Técnica de la Producción Dirección Técnica de la. 2010. *Programa de mejora. Estandar de codificación para Python. Universidad de las Ciencias Informáticas*. 2010.

PYTHONES. 2019. Qué es Python- Definición, características y ventajas. [En línea] 20 de julio de 2019. [Citado el: 1 de octubre de 2019.] <https://pythones.net/que-es-python-y-sus-caracteristicas/#>.

Queue., Celery: Distributed Task Queue. 2011. Homepage | Celery: Distributed Task. 2017. [En línea] 2017.

Quimera, Felipe. 2015. Patrones de diseño: qué son y por qué debes usarlos. [En línea] 2015.

<http://www.genbetadev.com/metodologias-de-programacion/patrones-de-diseno-que-son-y-por-que-debes-usarlos..>

Rafael Martinez. 2015. PostgreSQL . [En línea] 2 de octubre de 2015. [Citado el: 25 de octubre de 2018.] www.postgresql.org.

Ramírez, Jose Gabriel Espinosa. 2018. *Sistema para la detección de roles sobre colecciones de tuits*. Universidad de las Ciencias Informáticas. 2018. Título para optar por el título de Ingeniero en Ciencias Informáticas.

Ramos, César Vinuesa. 2015. *El auge y la importancia de las redes sociales en el mundo actual*. 2015.

Ramos, Frank Ernesto Sendiña. 2017. *Herramienta para consumir paquetes de repositorios comprimidos en la distribución cubana de GNU/Linux Nova*. Universidad de las Ciencias Informáticas. La Habana : s.n., 2017. Título para optar como Ingeniero en Ciencias Informáticas.

Raya Cabrera, José Luis, Raya González, Laura y Zurdo, Javier S. 2014. *Sistemas Informáticos (GRADO SUPERIOR)*. Madrid : Ra-Ma, 2014. ISBN 978-84-9964-349-6.

Richarte, Javier. 2018. Servicio Técnico 05: El BIOS: Curso visual y práctico: PCS • NOTEBOOKS ... [En línea] 2018. [Citado el: 3 de octubre de 2019.]

<https://books.google.com/cu/books?id=8jxLDwAAQBAJ&pg=PA14&dq=ventajas+de+utilizar+UEFI&hl=es&sa=X&ved=0ahUKewjWy4SZ1JmAhWhtVkkHbO7CicQ6AEIJzAA#v=onepage&q=ventajas%20de%20utilizar%20UEFI&f=false>. ISBN 949-4-571-852-1.

Rockcontent. 2019. *¿Qué es un lenguaje de programación y qué tipos existen?* . [En línea] 25 de julio de 2019. [Citado el: 12 de octubre de 2019.] <https://rockcontent.com/es/blog/que-es-un-lenguaje-de-programacion/>.

Rodríguez, Tamara Sánchez. 2015. *Metodología UCI*. Universidad de las Ciencias Informáticas. La Habana : s.n., 2015. Trabajo de diploma para optar por el título de Ingeniero en Ciencias Informáticas.

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

Rohaut, Sébastien. 2017. Linux: dominar la administración del sistema. [En línea] 3, 2017. [Citado el: 5 de diciembre de 2019.]

<https://books.google.com.cu/books?id=1P2BgluvLFsC&pg=PA284&dq=UEFI/GPT&hl=es&sa=X&ved=0ahUKEwi8sJ7J1pjmAhXIXvKkHXDwAkCQ6AEIKjAA#v=onepage&q=UEFI%2FGPT&f=false>. ISBN 978-2409-01222-8.

Rojas, Pablo Ramírez. 2012. Metodologías Tradicionales vs. Metodologías Ágiles. [En línea] 6 de febrero de 2012. [Citado el: 26 de noviembre de 2016.]

http://www.mygnet.net/manuales/software/metodologias_tradicionales_vs_dot_metodologias_agiles.

Rose, Heirald. 2014. [En línea] 2014.

Rothman, Michael y Zimmer, Vincent. 2017. *Harnessing the UEFI Shell: Moving the Platform Beyond DOS, Second Edition*. Berlin : CPI books GmbH, Leck, 2017. ISBN 978-1-5015-1480-7.

Salazar, Patricia. 2016. *El perfil de usuario de informacion*. 2016.

Sánchez, Manuel Alejandro del Campo. 2015. *PERSONALIZACIÓN DE GNU/LINUX NOVA PARA EL SISTEMA DE EDUCACIÓN PRIMARIA EN CUBA*. Universidad de las Ciencias Informáticas. La Habana : s.n., 2015. Trabajo de Diploma para Optar por el Título de Ingeniero en Ciencias Informáticas.

Sánchez, Manuel Alejandro. 2015. *Personalización de GNU/Linux Nova para el Sistema de Educación Primaria de Cuba*. Universidad de las Ciencias Informáticas. La Habana : s.n., 2015. Trabajo de diploma para optar por el título de Ingeniero en Ciencias Informáticas.

Santamaría, Pedro. 2015. Balsamiq mockup, una muy buen herramienta para esbozar tus futuras apps . *Applesfera*. [En línea] 25 de noviembre de 2015. [Citado el: 1 de noviembre de 2019.]

<https://www.applesfera.com/aplicaciones-os-x-1/balsamiq-mockup-una-muy-buen-herramienta-para-esbozar-tus-futuras-apps>.

Santana, Alberto. 2017. Sitios de redes sociales en Internet. [En línea] 20 de 07 de 2017. [Citado el: 05 de 11 de 2018.]

Sarmah, Harshajit. 2019. analyticsindiamag. *5 Tools that will help you create your on Linux Distro*. [En línea] 2019. [Citado el: 3 de diciembre de 2019.] <https://analyticsindiamag.com/5-tools-that-will-help-you-create-your-own-linux-distro/>.

Sarmiento, Johana. 2016. Vision General de los diagramas de despliegue. [En línea] 22 de 03 de 2016. [Citado el: 14 de 01 de 2019.] <http://umldiagramadespliegue.blogspot.com/>.

Schmuller, Joseph. 2015. 2015.

SIAC. 2018. Sistema Contable Administrativo. [En línea] 21 de octubre de 2018. [Citado el: 22 de octubre de 2019.] <http://sistemacontableec.com/sistema-contable-administrativo/>.

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

Silega, Nemury. 2017. *Método para la transformación automatizada del modelo de procesos de negocio a modelo de componentes para sistemas de gestión empresarial. Tesis presentada en opción al grado científico de Doctor en Ciencias Técnicas.* Universidad de las Ciencias Informáticas. 2017. Tesis de Maestría.

Smith, Roderick W. 2011. CompTIA Linux+ Complete Study Guide Authorized Courseware: Exams LX0-101 and ... [En línea] 2011. [Citado el: 4 de diciembre de 2018.] <https://books.google.com/cu/books?id=xa2-b3bsRMcC&pg=PA140&dq=GNU+pARTED&hl=es&sa=X&ved=0ahUKEwiI0Mfe55nmAhVlnlKkKHQZHCJEQ6AEIRDAD#v=onepage&q=GNU%20pARTED&f=false>. ISBN 978-0-470-88845-2.

Sobell, Mark G. 2014. A Practical Guide to Ubuntu Linux: A Practical Gui Ubun Linu_p4. [En línea] 2014. [Citado el: 24 de marzo de 2019.] https://books.google.com/cu/books?id=BOzIBQAAQBAJ&q=GNOME+DISK&dq=GNOME+DISK&hl=es&sa=X&ved=0ahUKEwi914u615nmAhWorFkKHXXC_gQ6AEIRDAD. ISBN 945-479-4-103-4.

SOFTWAREPROG. 2018. Entorno de Desarrollo Integrado. [En línea] 2018. [Citado el: 2 de septiembre de 2019.] <https://sites.google.com/site/softwaredeprogramacion2/entorno-de-desarrollo-integrado>.

Sommerville, Ian. 2011. *Ingeniería de Software.* Madrid : Pearson Addison Wesley, 2011. ISBN: 84-7829-074-5.

—. **2005.** *Ingeniería del Software.* Madrid : Pearson Addison Wesley, 2005. 84-7829-074-5.

SPRINGER. 2018. Visual Studio Code Distilled: Evolved Code Editing for Windows, macOS, and Linux. *Introducing Visual Studio Code.* [En línea] 2018. [Citado el: 10 de noviembre de 2019.] <http://link.springer.com/10.1007/978-1-4842-4224-7>. DOI 10.1007/978-1-4842-4224-7.

Thornton, J. 2013. *Bootstrap3 Manual Oficial, 2013* [Citado el: 28 de noviembre de 2018.] Disponible en: <http://conocimientoabierto.es>. 2013.

TICARTE. 2014. *Herramientas de compresión.* [En línea] 12 de septiembre de 2014. [Citado el: 28 de octubre de 2019.] <http://www.ticarte.com/contenido/herramientas-de-compresion>.

Torrecilla, Pablo. 2012. 2012.

TributosNet. 2016. Inventarios. [En línea] 11 de julio de 2016. [Citado el: 2019 de octubre de 15.] <http://www.tributos.net/definicion-de-gestion-de-inventarios-1013/>.

UBUNLOG. 2017. *Distroshare: un script que te ayuda a crea tu propia imagen de Ubuntu.* [En línea] 14 de noviembre de 2017. [Citado el: 5 de septiembre de 2019.] <https://ubunlog.com/author/darkcrist/>.

UCC. 2017. Visión general del Diagrama de despliegue. [En línea] 24 de septiembre de 2017. [Citado el: 1 de octubre de 2019.] <http://umldiagramadespliegue.blogspot.com/>.

UML. 2017. What is UML. [En línea] 2017. [Citado el: 24 de octubre de 2019.] <https://www.uml.org/what-is-uml.htm>.

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

Universidad de Mexico. 2015. Gestión de Software. [En línea] 1 de marzo de 2015. [Citado el: 2016 de 10 de 15.] <http://mejoratugestion.com/mejora-tu-gestion/que-es-un-sistema-de-gestion/>.

Urrutia, Angélica. 2015. Análisis de Información de Redes Sociales (Twitter). [En línea] 17 de 09 de 2015. [Citado el: 07 de 12 de 2018.]

Vázquez, Eduardo Carlo. 2018. *Ingeniería de Requisitos: Software Orientado al Negocio*. [https://books.google.com/cu/books?id=kDjMvwEACAAJ&dq=especificaci%C3%B3n+de+requisitos+del+software&hl=es&sa=X&ved=0ahUKEwjsIPDAwKznAhUro1kKHZvTDEEQ6AEIPDAD] 2018. ISMN:1729136680.

Velazquez, Luis Romero. 2015. *Análisis y diseño orientado a objetos*. Madrid : Lenguage Books, 2015. ISBN: 479-587-41-1.

Warsaw, Barry y Coghlan, Nick. 2013. Style Guide for Python Code. [En línea] 5 de julio de 2013. [Citado el: 22 de septiembre de 2019.] <https://www.python.org/dev/peps/pep-0008/>.

WOLF, Gunnar, y otros. 2015. *Fundamentos de sistemas operativos*. Mexico D.F : Círculo Mario de la Cueva, 2015. ISBN 978-607-02-5544-0.

ZAZO RODRÍGUEZ, Ángel F., ALONSO BERROCAL, José Luis y FIGUEROLA, Carlos G. 2014. *Herramientas de software libre para el trabajo científico colaborativo: EN ..*

[https://books.google.com/cu/books?id=78mMAwAAQBAJ&pg=PA149&dq=software+libre&hl=es&sa=X&ved=0ahUKEwjT3421i5zmAhWm1FkKHdAjAI4Q6AEITzAG#v=onepage&q=software%20libre&f=false] Salamanca : Universidad de Salamanca, 2014. 947-58-7-548-624-08225.

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

ANEXOS

Anexo 1: Entrevista realizada a especialistas del proyecto Nova

Tipo de entrevista: Estructurada porque se tiene un orden lógico de lo que se quiere preguntar, es decir se conoce el objetivo y finalidad de la entrevista.

Cantidad de personas entrevistadas: 4

Nombre de los entrevistados: Juan Manuel Fuentes Rodríguez, Aldy León García, Javier Piñeiro Cárdenas, Luis Daniel Sierra Corredera.

Función de los entrevistados en el proyecto: presentan más de 5 años de experiencia, han trabajado directamente con el proyecto.

Objetivo de la entrevista: conocer cómo se realiza el proceso de creación de personalizaciones de la distribución cubana GNU/Linux NOVA.

1. ¿Qué importancia tiene el surgimiento de las personalizaciones de Nova? ¿Por qué?
2. ¿Cómo se realiza el proceso actualmente en el centro de la creación de las personalizaciones?
3. ¿Cuáles son los pasos que se deben tener en cuenta para la realización de una personalización?
4. ¿Qué herramientas se utilizan en el centro para UEFI/GPT?
5. ¿Qué características tendría la herramienta a desarrollar?

Resultado de la entrevista: se evidencia que en la actualidad el proceso de creación de personalizaciones se realiza mediante un procedimiento manual, carecen de una herramienta informática que automatice el proceso.

Anexo 2: Guía de observación para el proceso de creación de personalizaciones de la distribución cubana GNU/Linux Nova.

Observador: Sulema Hernández Garrido

Lugar: Laboratorio del proyecto Nova

Objetivo: Identificar los requisitos fundamentales para la realización del proceso de creación de personalizaciones de la distribución cubana GNU/Linux Nova.

1. Datos de identificación del proceso

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

- Nombre del proceso
- Especialista que ejecuta el proceso

2. Características del espacio donde se desarrolló el proceso

- ¿Dónde se debe realizar?
- ¿Bajo qué condiciones se realiza?
- ¿Qué se necesita para iniciar el proceso?

3. Características del proceso

- ¿Cómo es el proceso?
- ¿Qué herramientas se emplean?
- ¿Cuáles son los pasos a seguir?
- ¿Cuál es el resultado?

Anexo 3: Diseños de Casos de Pruebas

Instalar Paquetes

Escenario	Descripción	Paquete de tipo Usuario	Paquete de tipo Estándar	Paquete de tipo Boot	Respuesta del Sistema	Flujo Central
EC 2.1 Instalar paquetes introducidos por el usuario.	El sistema permite instalar paquetes en caso de que no ocurra ningún error.	V <i>byobu</i>	V <i>casper</i>	V <i>Grub2</i>	Instala Paquetes con datos introducidos por el usuario.	1. Descomprimir el Debootstrap. 2. Montar el Sistema de Archivos. 3. Actualizar chroot. 4. Instalar paquetes con datos introducidos por el usuario.
EC 2.2 Instalar	El sistema solo	I	I	I	Instala paquetes válidos,	1. Descomprimir

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

paquetes insertando valores inválidos.	instala los paquetes válidos.	<i>isolinux</i>	<i>gpm</i>	<i>discover</i>	en caso de que no exista o contiene algún error dicho paquete el sistema no los ejecuta.	el Debootstrap. 2. Montar el Sistema de Archivos. 3. Actualizar chroot. 4. Instalar paquetes con datos introducidos por el usuario.
EC 2.3 Instalar paquetes dejando campos vacíos.	El sistema permite verificar si existen campos vacíos.	V	V	V	Crea Sistema Operativo Base con datos entrados por el usuario y datos por defecto entrados por el sistema en caso de que no hayan sido rellenados por el usuario.	1. Descomprimir el Debootstrap. 2. Montar el Sistema de Archivos. 3. Actualizar chroot. 4. Instalar paquetes con valores por defecto.
		<i>byobu</i>	<i>casper</i>	<i>grub2</i>		
		I	I	I		
		<i>lupin-casper</i>	<i>linux-image-generic</i>	<i>Vim</i>		

Ejecutar Hooks

Escenario	Descripción	<i>post-debootstrap</i>	<i>post-chroot</i>	<i>post-image</i>	Respuesta del Sistema	Flujo Central
EC 6.1 Ejecutar Hooks introducidos por el usuario	El sistema permite ejecutar Hook en caso de que no ocurra ningún error.	Válido	Válido	Válido	Ejecuta en la ruta especificada los Hooks que han sido elaborados por el usuario previamente.	1. Luego de terminar cada proceso se ejecutan los Hooks en la ruta en donde se encuentran los mismos.
		Código en Bash elaborado por el usuario.	Código en Bash elaborado por el usuario.	Código en Bash elaborado por el usuario.		
EC 6.2 Ejecutar	El sistema permite	Válido	Válido	Válido	Verifica en cada momento que se	1. Luego de terminar cada

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

Hooks con valores inválidos.	verificar si existen errores en los Hooks.	Código en Bash elaborado por el usuario que contiene errores.	Código en Bash elaborado por el usuario que contiene errores.	Código en Bash elaborado por el usuario que contiene errores.	necesite ejecutar Hooks, en caso de que estos contengan errores el Sistema los detecta como inválidos y envía un error al usuario para que el Hooks sea modificado por el usuario.	proceso se ejecutan los Hooks en la ruta en donde se encuentran los mismos. 2. Si existe algún error, el Sistema notifica al usuario con un error.
EC 6.3 Ejecutar Hooks que no existen	El sistema permite ejecutar Hook en caso de que existan.	Válido Vacío	Válido Vacío	Válido Vacío	Ejecuta en la ruta especificada los Hooks en caso de que existan.	1. Luego de terminar cada proceso se ejecutan los Hooks en caso de que hayan sido creado previamente.

Anexo 4: Casos de Pruebas.

Ruta Básica 2

Ruta Básica 2: 1-2-4-5-6-7-8	
Descripción	Proceso para correr todos los métodos del Debootstrap
Condición de ejecución	El usuario no haya creado un Debootstrap anteriormente; desea comprimir al final del proceso el Debootstrap.
Resultado esperado	La herramienta corre el Debootstrap, luego ejecuta los Hooks y finalmente comprime el Debootstrap.

Herramienta informática para la creación de personalizaciones de la distribución cubana GNU/Linux Nova

Resultado	Satisfactorio.
------------------	----------------

Ruta Básica 3

Ruta Básica 3: 1-2-3-5-6-7-8	
Descripción	Proceso para correr todos los métodos del Debootstrap
Condición de ejecución	El usuario haya creado un Debootstrap anteriormente; desea comprimir al final del proceso el Debootstrap
Resultado esperado	La herramienta descomprime el Debootstrap, ejecuta los Hooks y finalmente comprime el Debootstrap.
Resultado	Satisfactorio.