Universidad de las Ciencias Informáticas
Facultad 1

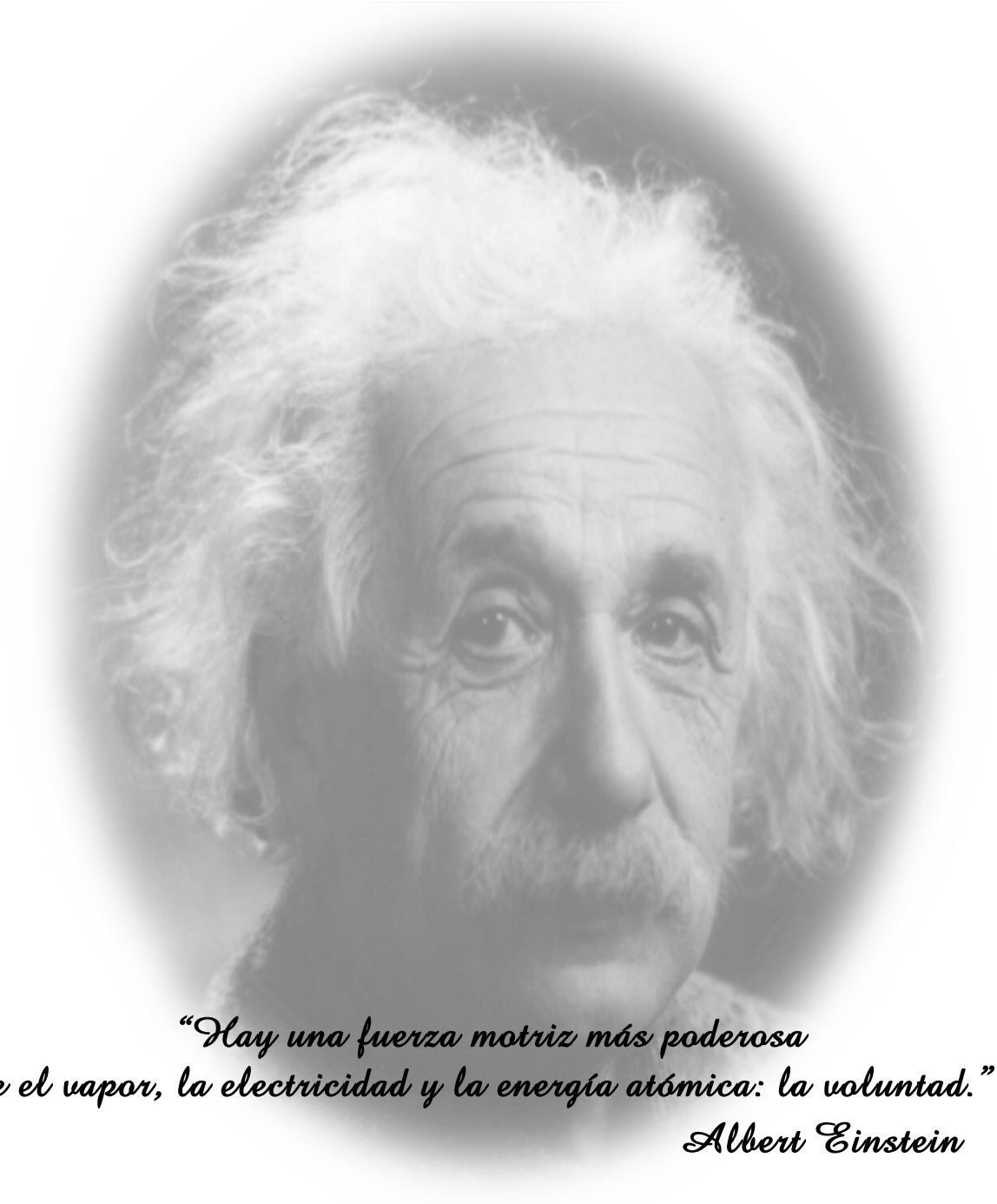
“Nova Bonita”

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

Autores Yisel Obiol González
Dayrol Lázaro Ferrer Durruthy
Tutores Mtr. Allan Pierra Fuentes
Ing. Hector Pérez Baranda

La Habana

Junio 2015



"Hay una fuerza motriz más poderosa
que el vapor, la electricidad y la energía atómica: la voluntad."

Albert Einstein

Declaración de Autoría

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

Para que así conste firmamos la presente a los ___ días del mes de ___ del año 2015.

_____	_____
Yisel Obiol González	Dayrol Lázaro Ferrer Durruthy
_____	_____
Mtr. Allan Pierra Fuentes	Ing. Hector Pérez Baranda

Dedicatoria

Yisel:

A mi mamá por siempre apoyarme, por todo su amor y preocupación.

A mi papá por toda su amor,comprensión y paciencia.

A mi abuela, por siempre estar pendiente de mi y por todo el amor que me ha dado.

Dayrol:

A mi abuela, mi mamá y a toda mi familia.

Yisel:

Son muchas las personas especiales a las que me gustaría agradecer su amistad, apoyo y compañía en las diferentes etapas de mi vida. Sin importar en dónde estén o si alguna vez llegan a leer estos agradecimientos quiero darles las gracias por formar parte de mí y por todo lo que me han brindado.

A mis padres, por estar conmigo en cada paso que doy, por todo el esfuerzo realizado por ellos. Por apoyarme en todo momento y por ser la fuerza que me impulsa cada día a tratar de superarme como persona.

A mi familia, por todo su apoyo y confianza.

A mi tutor y amigo Hector, por haberme apoyado en cada momento que lo he necesitado y por su ayuda brindada.

A mis amigos que siempre estaban en los momentos que más los necesite, especialmente: Neyvis, Dvito, Yennis, Michel, Manuel, Arian y Danilo. Gracias a todos por estar siempre para mi, por soportarme en mis peores momentos y darme ánimos cuando creía que no podía. De todos tengo muy bonitos recuerdos y gracias por ser mis más grandes amigos.

A mis amistades Lisdy, Yanet, Anita, Ilo, Yenifer , Karen , Nathaly, Yelaysi, Yileni,Roger y Leonard por tantos momentos que pasamos juntos y por hacerme reír en los momentos de estrés.

A mi grupo, por ser parte de mis mejores días y juntos pasar los buenos y malos momentos. Sin ellos no hubiera disfrutado tanto estos 5 años.

A mi compañero de tesis, Dayrol, por ser mi amigo durante tanto tiempo y por todos los buenos momentos que pasamos juntos.

A todos mis amigos de la UCI por cada uno de los momentos vividos.

A las personas del proyecto, en especial Yaiselis y Haniel, por toda su ayuda y comprensión.

Dayrol:

A mi compañera de tesis por ayudarme en todo en los 5 años de carrera, a los tutores.

A mi madre materializadora de esta obra y responsable de hacerme un hombre de valores morales y sentimientos puros.

A mi abuela Martha.

Agradezco a mis familiares, a los que ya no están; artífices de este sueño .

A los profesores que fueron una parte importante en mi formación como ingeniero.

A mis hermanos Yoandri, Raúl y Dennis, a mis amigos del Politécnico . A la gente del grupo por todas las experiencias vividas juntos en los 5 años y las veces que me ayudaron.

A todos aquellos que de una compartieron conmigo buenos y malos momentos.

A todos gracias.

Resumen

La discapacidad física de cualquier índole está afectando a más de 100 millones de personas en todo el mundo y es necesaria la realización de acciones que ayuden a la inserción de estas personas a la sociedad; en este sentido los avances tecnológicos pueden ser de vital importancia. Un ejemplo claro lo es la distribución cubana de GNU/Linux Nova, surgida para el apoyo al proceso de migración a software libre en el cual se encuentra inmersa Cuba, la cual incluye en su propuesta el soporte para su uso por discapacitados. Aunque este soporte es limitado ya que las aplicaciones predefinidas y sus configuraciones, en Nova 4.0, no estaban diseñadas para brindar servicio a todas las discapacidades visuales y motoras. Tomando como punto de partida esta problemática, se define como objetivo en la presente investigación desarrollar una personalización de Nova que incluya funcionalidades y aplicaciones orientadas a personas con discapacidades visuales y motoras. Para la confección del sistema se utilizó el lenguaje Bash, mediante la herramienta Debootstrap y se tuvieron en cuenta todos los pasos y métodos que ofrece la metodología Nova-OpenUp. Con la realización de la propuesta de solución, que se sometió a pruebas de funcionalidad y de aceptación, se dio cumplimiento a la premisa de que un mundo mejor es posible, ya que se trabaja en la mejora de la calidad de vida de las personas discapacitadas y en la integración social en igualdad de derechos de las mismas.

Palabras clave: discapacidad, distribución, personalización, sistema operativo, Nova.

Índice de contenido

Introducción

Uno de los retos para la sociedad actual constituye la atención a personas que presentan algún tipo de discapacidad, enfocado a la mejora de su calidad de vida y a la integración social en igualdad de derechos.

Según la Organización Mundial de la Salud (O.M.S) el 15% de la población mundial está afectada por alguna discapacidad física, psíquica o sensorial que dificulta su desarrollo personal y su integración social, educativa o laboral. Tal porcentaje equivale a 900 millones de personas, casi el doble de la población de Latinoamérica (1). Es por ello que existe una creciente preocupación mundial por eliminar, hasta donde sea posible, las desventajas que estas discapacidades conllevan por medio de acciones específicas; como el recuperar las funciones faltantes y cuando no sea posible la completa recuperación, compensarla con la rehabilitación. Este nivel de compensación puede tomar como punto de partida el desarrollo de habilidades y destrezas necesarias en los discapacitados, o dotar a los mismos de elementos compensatorios.

Las acciones llevadas a cabo con este fin, permitirán que las personas con discapacidades creen habilidades y puedan tener acceso, entre otros campos, al desarrollo tecnológico actual como parte del proceso de inserción de los mismos en la sociedad. La aparición de las Tecnologías de la Información y las Comunicaciones (TIC) pretende solucionar, en parte, el problema de inserción social al cual se enfrentan los discapacitados y con ello el desarrollo de importantes labores de formación e interacción como ciudadanos (2).

El desarrollo de sistemas que faciliten la relación en grupos, el trabajo cooperativo y la creación de nuevas formas de interacción social, son algunas de las herramientas procedentes de las TIC que han ido ganando espacio en el mundo de los usuarios con algún tipo de discapacidad.

El acelerado crecimiento y actualización de las tecnologías ha traído consigo que a estos usuarios se les dificulte el acceso a los ordenadores en cierta medida, pues el surgimiento de los mismos fue pensado para personas sin discapacidades. A pesar de ello, existen alternativas, así como aplicaciones

informáticas para que puedan acceder como cualquier otro ciudadano a la información. Se pueden agrupar las tecnologías de ayuda al discapacitado en cinco grupos: Sistemas Alternativos y Aumentativos de Acceso a la Información, Sistemas de Acceso, Sistemas Alternativos y Aumentativos de comunicación, Sistemas de Movilidad y Sistemas de Control de Entornos, los cuales intentan compensar las carencias que poseen estos individuos (1).

Cuba es uno de los países que ha identificado, desde muy temprano, la conveniencia y necesidad de dominar e introducir en la práctica social las TIC, para lograr la completa informatización de la sociedad y ha dado pasos de avance en ese sentido. La migración a plataformas libres y de código abierto forma parte de dicho proceso de informatización de la sociedad cubana. La distribución cubana de GNU/Linux Nova (en lo adelante Nova), es la propuesta de sistema operativo a utilizar en el país. Según Allan Pierra Fuentes, *"Nova surge como una distribución de GNU/Linux orientada a un ambiente escritorio desarrollada en la Universidad de las Ciencias Informáticas (UCI) a través del proyecto del mismo nombre"* (3). Su desarrollo, uso y aplicación está basado en los principios de Seguridad, Soberanía Tecnológica, Socio-Adaptabilidad y Sostenibilidad, definidos como las 4S, los cuales pretenden materializar las razones que llevan al país a ejecutar el necesario proceso de migración tecnológica (3).

El uso de sistemas operativos de código abierto en Cuba aumentará la productividad y la agilidad para responder a cambios en el mercado, se reducirá la dependencia del software privativo sobre el cual se articulan en el país, transacciones y desarrollos de inmensos volúmenes de información de gran valor, se mantendrá una solución efectiva en costos y se ofrecerán mejores servicios o productos. Por otra parte, disminuirá la necesidad de adquirir las licencias privativas de software que ha obligado a operar la mayoría de las computadoras del país con programas que no se pagan y por los cuales, en caso de no existir el bloqueo económico impuesto por Estados Unidos a la isla, se deberían pagar cientos de millones de dólares (4).

Además de lo antes expresado, como uno de los pilares de la distribución Nova, se encuentra la Socio-Adaptabilidad lo cual plantea, entre otros elementos, que el sistema operativo debe cumplir con todas las normas establecidas para el uso de todas las personas del país; donde se debe tener en consideración las

personas con discapacidad.

El lanzamiento de Nova 4.0, en marzo de 2013, trajo como propuesta el soporte, aunque limitado, para su uso por discapacitados. Las aplicaciones predefinidas y sus configuraciones, en esta versión, no estaban diseñadas para brindar servicio a todas las discapacidades visuales y motoras. Esta deficiencia limitaba las posibilidades de garantizar la inclusión de este grupo de personas, en el escenario actual de la migración a plataformas libres en que se encuentra el país.

Tomando como punto de partida la problemática planteada se define como **problema de la investigación**: ¿Cómo aumentar la accesibilidad de usuarios con discapacidad visual y/o motora a la distribución cubana de GNU/Linux Nova? Para orientar el curso de la investigación se identifica como **objeto de estudio**: personalizaciones de GNU/Linux enfocadas a usuarios discapacitados centrando el **campo de acción** de la misma en la distribución cubana de GNU/Linux Nova.

Se traza como **objetivo general**, desarrollar una personalización de GNU/Linux Nova que incluya aplicaciones para la accesibilidad universal.

Para alcanzar este objetivo general se plantean los siguientes **objetivos específicos:**

- Analizar las propuestas de solución existentes de apoyo a los discapacitados visuales y motores, con el propósito de identificar las aplicables a la personalización de la distribución cubana de GNU/Linux Nova.
- Construir una personalización de la distribución cubana de GNU/Linux Nova con aplicaciones destinadas al apoyo de la mayor cantidad posible de discapacitados visuales y motores en su inserción en el proceso de migración del país.
- Evaluar la propuesta de solución para garantizar la calidad de la misma.

Se propone como **idea a defender** que si se desarrolla una personalización de Nova orientada a usuarios con discapacidad visual y/o motora se contribuye a aumentar su accesibilidad.

El proceso de investigación está orientado por las siguientes **tareas de investigación**:

1. Análisis del marco teórico relacionado con las discapacidades visuales y motoras para identificar

las necesidades de los usuarios finales.

2. Análisis de las soluciones existentes para el apoyo a discapacitados visuales y motores.
3. Selección de las aplicaciones y herramientas necesarias para la construcción de la personalización de Nova.
4. Construcción de la personalización de Nova para darle respuesta al objetivo propuesto.
5. Probar la propuesta de solución para garantizar su calidad y funcionamiento.

Para un mejor desarrollo de la investigación se usaron los siguientes **métodos científicos:**

Métodos teóricos:

Analítico-Sintético: se utilizó este método en la identificación de conceptos y definiciones relevantes relacionados con las discapacidades visuales y motoras, así como las aplicaciones informáticas existentes para dichas personas, posibilitando la extracción de los elementos más significativos que sustentan la investigación.

Histórico-Lógico: se realizó un estudio del desarrollo y evolución de las aplicaciones informáticas para personas con discapacidades visuales y motoras, que permite tener una visión tanto de los principales problemas como de los avances obtenidos en la materia para así poder tomar ideas y técnicas que se utilizarán en la solución a desarrollar.

Métodos Empíricos:

Análisis documental: es utilizada en la consulta de la literatura especializada publicada a nivel nacional e internacional, para extraer la información necesaria que permitió diseñar la solución.

Método experimental: el experimento permitió estudiar el objeto y crear las condiciones para verificar la idea a defender.

El presente trabajo cuenta con tres capítulos en los cuales se recogen los resultados de la investigación realizada:

- **Capítulo 1 Sistemas Operativos Personalizados:** se definirán una serie de conceptos fundamentales relacionados con el objeto de estudio. Se realiza un estudio del estado del arte

sobre las experiencias similares y aplicaciones existentes para usuarios con discapacidad visual y motora.

- **Capítulo 2 Análisis y diseño del sistema:** consiste en la especificación de las aplicaciones informáticas que se seleccionaron. Se abordará con mayor detalle los aspectos relacionados con los elementos de análisis y diseño de la solución a la problemática de la investigación.
- **Capítulo 3 Proceso de construcción del sistema:** se implementará el sistema, para de esta manera llegar a una solución final. Luego, se efectuará un análisis de los resultados mediante la realización de las pruebas de aceptación a la personalización así como las aplicaciones que en ella se encuentran, basado en el criterio de un grupo de usuarios finales.

Capítulo 1. Sistemas Operativos Personalizados

Introducción

Para lograr una personalización de GNU/Linux Nova para el apoyo a personas discapacitadas visuales y motoras es necesario realizar un estudio acerca de los conceptos fundamentales tratados en el proceso de creación de una personalización de la distribución Nova.

En este capítulo se analizan aspectos teóricos que servirán de soporte para la elaboración del sistema. Se mostrará un estudio del estado del arte sobre distribuciones y herramientas de apoyo a personas discapacitadas en el área de la informática. Por último, se presentan tanto la metodología como las herramientas y tecnologías a utilizar en la implementación del sistema.

1.1 Conceptos fundamentales

A continuación se definirán los principales términos que en lo adelante serán utilizados, o que de alguna manera será necesario conocer su significado durante el proceso de construcción de la propuesta de solución.

Sistema Operativo: es un programa o conjunto de programas que en un sistema informático gestiona los recursos de hardware y provee servicios a los programas de aplicación, o sea, proporciona una plataforma de software sobre la cual otros programas puedan funcionar (5).

Distribución de GNU/Linux: una distribución de GNU/Linux puede ser descrita 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 aplicaciones o controladores privativos (3).

Personalización de una distribución de GNU/Linux: el uso de 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, cambios y transformaciones que se le realizan a una distribución determinada con el fin de suplir necesidades

particulares se le denominan: personalizaciones (6).

Repositorio: un repositorio, depósito o archivo es un sitio centralizado donde se almacena y mantiene información digital, habitualmente bases de datos o archivos informáticos. Pueden contener los archivos en su servidor o referenciar desde su web al alojamiento originario. Los repositorios pueden ser de acceso público, o pueden estar protegidos y necesitar de una autenticación previa (7).

Paquete de software: un paquete de software es un conjunto de ficheros relacionados con una aplicación, que contiene los objetos ejecutables, los archivos de configuración, información acerca del uso e instalación de la aplicación, todo ello agrupado en un mismo contenedor. Estos pueden ser binarios y de código fuente (5).

Paquetes binarios (5): contienen código máquina, y no código fuente, por lo que cada tipo de arquitectura (X86(i386-i686), AMD64, ARM, MIPSEL), necesita su propio paquete. Estos pueden ser:

- RPM: estos paquetes son utilizados por distribuciones como Red Hat, Suse, Mandrake, Conectiva, Caldera.
- DEB: estos paquetes son utilizados por distribuciones como Debian, y las basadas en ella, como Ubuntu.
- TGZ: son utilizados por la distribución Linux Slackware.

Paquetes de código fuente (5): contienen el código fuente del programa y vienen con los archivos necesarios para compilar e instalar el programa manualmente. Suelen presentarse en formato .tar.gz o tar.bz2 (o sea, compactado con tar y comprimido con gzip o bzip).

Deficiencia: una deficiencia es una falla o un desperfecto. El término, que proviene del vocablo latino *deficientia*, también puede referirse a la carencia de una cierta propiedad que es característica de algo (8).

Discapacidad: desde el punto de vista semántico, el término discapacidad está formado por el prefijo *dis* que significa separación y la palabra *capacidad* que significa inteligencia, talento y estado óptimo. Restricción o ausencia (causada por una deficiencia) de la capacidad de realizar una actividad en la forma o dentro del margen que se considera normal por el ser humano (9).

Discapacidad visual: la discapacidad visual está relacionada con una deficiencia del sistema de la visión que afecta la agudeza visual, campo visual, motilidad ocular, visión de los colores o profundidad, afectando la capacidad de una persona para ver. Al hablar de discapacidad visual se puede hacer referencia a la persona que presenta ceguera o baja visión (10).

Discapacidad motriz: la discapacidad motriz es una condición de vida que afecta el control y movimiento del cuerpo, generando alteraciones en el desplazamiento, equilibrio, manipulación, habla y respiración de las personas que la padecen, limitando su desarrollo personal y social. Esta discapacidad se presenta cuando existen alteraciones en los músculos, huesos, articulaciones o médula espinal, así como por alguna afectación del cerebro en el área motriz impactando en la movilidad de la persona (11).

1.2 Distribuciones para el apoyo a discapacitados

En el mundo existen un conjunto de distribuciones que fueron desarrolladas para ayudar a las personas con discapacidades en su desarrollo personal. Estas distribuciones tienen como objetivo establecer las bases que permitan la inclusión de estas personas dentro del marco de la igualdad en todos los ámbitos de la vida.

1.2.1 Lazarux

Lazarux es un proyecto surgido en la Universidad de La Coruña (España), a partir de la necesidad de mejorar la accesibilidad en la informática. Consiste en un *Live-CD*[1] adaptado a deficientes visuales de habla hispana, que incluye un amplio conjunto de aplicaciones accesibles con una configuración diseñada para que una persona con este tipo de deficiencia las pueda utilizar desde el inicio (12).

Entre las aplicaciones que proporciona se encuentran un lector en voz alta de textos, una lupa que amplía la imagen, y un sintetizador de voz con el que la máquina se comunica con el usuario. Entre las principales ventajas de esta distribución se encuentran el idioma y su facilidad de acceso (13).

Esta distribución viene con las aplicaciones instaladas y por defecto: el sintetizador de voz Festival, empaquetado y configurado para que utilice la voz en castellano; el servidor de voz Eflite, para lectores de pantalla; el lector de pantalla y amplificador Gnopernicus, configurado para que utilice la voz en castellano;

1 Sistema operativo almacenado en un medio extraíble.

el demonio BRLTTY; la lupa Xzoom; el teclado en pantalla Gok; la interfaz de texto predictivo Dasher; el sintetizador de voz Emacspeak; Yasr, un lector de pantalla para la consola de texto; la lupa Gnome-mag y Orca, el lector de pantalla para el escritorio GNOME (14) .

Desventajas:

Dentro de las desventajas de esta distribución se encuentran que solo está adaptada para deficientes visuales; además de que en el momento de la investigación sus enlaces de descargas no se encontraban disponibles, haciendo imposible obtenerla y hacer un análisis más profundo de dicha distribución.

1.2.2 Oralux

Oralux es una distribución GNU/Linux basada en Knoppix[2]. Consiste en un *Live-CD* para personas con deficiencias visuales, trabaja en modo texto (no tiene entorno gráfico) y es multilingüe, por lo que brinda la posibilidad de trabajar en idiomas como el español, inglés, francés, ruso, portugués y alemán, proporcionando una síntesis de voz para cada uno de los idiomas mencionados (15).

Oralux cubre un gran número de usuarios, desde aquel que sólo desea desempeñar las tareas básicas en una PC, hasta el que desea administrar sistemas o desarrollar.

Oralux ofrece tres modos de trabajo:

- Usando el entorno Emacs con Emacspeak.
- Usando el lector de pantalla Yasr.
- Usando el lector de pantalla Speakup.

Y todos estos entornos trabajando con las voces de Mbrola (15).

Desventajas:

Dentro de las desventajas de esta distribución se encuentran que solo está adaptado para deficientes visuales y que es una distribución que se encuentra descontinuada desde abril de 2007, por lo cual no es factible su utilización ya que implicaría gastos en tiempo y recursos, debido a que está basada en una distribución de Ubuntu ya obsoleta.

2 Distribución de GNU/Linux basada en Debian.

1.2.3 Tiflobuntu

Tiflobuntu es una distribución Ubuntu (Hardy Heron) personalizada para hacer más fácil el acceso al sistema operativo GNU/Linux a las personas que sufren alguna discapacidad visual. Utilizando las herramientas de accesibilidad disponibles ya en Ubuntu, Tiflobuntu ofrece una configuración de las mismas para que el usuario no tenga que realizar ningún paso adicional para acceder al sistema. El idioma predeterminado de esta distribución es español, teniendo la posibilidad de fijarlo en inglés.

Tiflobuntu se inicia con soporte para síntesis de voz y Braille[3] en el arranque sin tener el usuario que preocuparse en iniciar las ayudas técnicas manualmente.

Esta distribución viene con OpenOffice.org para la creación de documentos y hojas de cálculo, Evolution como gestor de correo electrónico, Firefox 3.0 de navegador web, Pidgin, mensajería instantánea MSN; Yahoo; IRC Jabber; Gwget, gestor de descargas; gnome-mud, cliente para jugar; Rhythmbox, reproductor de música con soporte para escuchar mp3; ogg; wma; Totem, reproductor de películas; Sound juicer, extractor de sonido de CD; grabadora de sonido; Mplayer para reproducir audio/vídeo y DVD; Brasero, programa de grabación de cd.

Tiflobuntu viene con las últimas versiones de Orca (el lector de pantalla para el escritorio GNOME(*GNU Network Object Model Environment)*); Yasr, un lector de pantalla para la consola de texto; sintetizador de voz Espeak; sistemas de voz de GNOME (gnome-speech) y Speech dispatcher, además del demonio BRLTTY.

Se podrá actualizar el sistema cuando haya nuevas actualizaciones de Ubuntu, y además Tiflobuntu incluye un repositorio de paquetes de software mantenido por la comunidad de *tiflolinux.org* con las últimas versiones de los programas de accesibilidad (16).

Desventajas:

Dentro de las desventajas de esta distribución se encuentran que solo está adaptado para deficientes visuales y que es una distribución que surgió en el año 2008; pero se encuentra descontinuada desde el año 2009 por lo cual tampoco es factible su utilización ya que implicaría gastos en tiempo y recursos

3 Sistema de lectura y escritura táctil.

debido a que está basada en una distribución de Ubuntu ya obsoleta y la actualización de sus paquetes traería conflictos.

1.2.4 EVuntu

EVuntu es una distribución Linux, basada en Ubuntu 8.04, Hardy Heron y en Tiflobuntu configurada para hacer más fácil el acceso al sistema operativo GNU/Linux a usuarios que sufran algún grado de discapacidad.

Utiliza herramientas de accesibilidad disponibles en Ubuntu y Tiflobuntu, y ofrece una configuración automática orientada a la simplicidad de uso por parte del usuario, para que así no tenga que preocuparse por realizar ningún paso adicional para acceder al sistema y utilizar el computador para navegar, estudiar, trabajar o entretenerse.

Usa núcleo monolítico y su entorno gráfico por defecto es GNOME. Es posible configurar en el arranque las ayudas técnicas para que se inicien automáticamente con el sistema operativo. Esta versión de Ubuntu cuenta con ayudas técnicas como Orca (el lector de pantalla para el escritorio GNOME); Yasr, un lector de pantalla para la consola de texto; sintetizador de voz Espeak; sistemas de voz de GNOME (gnome-speech) y Speech dispatcher; BRLTTY y el lector de libros Daisy[4] DBR. Además cuenta con los mismos software preinstalados que Tiflobuntu (17).

Desventajas:

Como desventajas de esta distribución se pueden plantear que no es una versión estable ya que solamente está en las versiones alpha[5]. Está basado en Hardy Heron, en su momento fue una versión LTS[6] de Ubuntu; pero ya está descontinuada.

1.2.5 Vinux 4.0

Vinux es una distribución de GNU/Linux derivada de Ubuntu 12.04.2, desarrollada por la comunidad de usuarios ciegos y deficientes visuales, para este tipo de personas (18).

Los desarrolladores se han esforzado en mejorar la accesibilidad del sistema operativo para personas con

4 Sistema de Información Digital Accesible.
5 Primera versión del programa la cual es enviada a los verificadores para probarla. Se refiere a una fase donde el producto todavía es inestable.
6 *Long Term Support* (Soporte de Término Largo).

problemas visuales que van desde herramientas para convertir el texto en voz, lupas dinámicas para visualizar mejor el contenido de la pantalla, y soporte para teclados Braille USB (18).

Esta distribución está disponible en *Live-CD*, aunque sus desarrolladores han habilitado repositorios específicos desde donde es posible conseguir todas las aplicaciones si se prefiere trabajar con otras distribuciones. De igual forma, esas mismas aplicaciones se han compilado en paquetes **.deb** facilitando el acceso a ellas para poder instalarlas y utilizarlas fácilmente en Ubuntu (19).

Hay disponibles versiones *Live* en DVD, CD y USB, basadas tanto en *Intrepid Ibex*[7] como en *Lucid Lynx*.[8] Además hay versiones para arquitecturas de 32 y 64 bits. Incluye numerosas aplicaciones de trabajo, menús accesibles semejantes a GNOME 2.32, Firefox 21 y Thunderbird 17 (19).

Desventajas:

Dentro de las desventajas de esta distribución se encuentran que solo está adaptada para deficientes visuales y el idioma que trae defecto es inglés; por tanto todas las configuraciones de sus herramientas así como su sintetizador de voz son para dicho idioma.

1.3 Valoración acerca de las distribuciones para el apoyo a discapacitados

Las distribuciones estudiadas poseen aspectos en su configuración que pueden ser reutilizados en la propuesta de solución, pero ninguna de ellas ofrece en sí, todos los elementos necesarios para dar la mejor solución. La mayoría de estas distribuciones fueron diseñadas solo para personas con discapacidades visuales; no obstante, constituyen un punto de referencia a tener en cuenta para la personalización de la distribución cubana de GNU/Linux Nova, donde se abarquen las discapacidades visuales y motoras. Por otra parte algunas de estas distribuciones como son los casos de Evuntu, Oralux y Tiflobuntu, están descontinuadas u obsoletas, por lo que no es factible dedicar tiempo y esfuerzo en la adopción de alguna de estas distribuciones, aún cuando pudiera valorarse la incorporación de elementos a las mismas, puesto que en un final sería improductivo a la vez que implicaría un mayor gasto en tiempo y recursos.

7 Distribución de Ubuntu 8.10.
8 Distribución de Ubuntu 10.04.

También existe la condición que estas distribuciones al estar basadas en Ubuntu no se adaptan al contexto del país (Lazarux) y junto a otras que son desarrolladas por comunidades (Vinux 4.0), no permiten poseer una soberanía tecnológica sobre estas herramientas, así como se dificulta la sostenibilidad en el tiempo y no se garantiza la seguridad de las mismas.

1.4 Estudio sobre aplicaciones para discapacitados

Las aplicaciones usadas por las personas discapacitadas son sistemas técnicos especiales, disponibles en el mercado para posibilitar el acceso a las aplicaciones (20). A continuación se profundizará en las aplicaciones para discapacitados más utilizadas y presentes, en su mayoría, en las distribuciones analizadas anteriormente.

1.4.1 Aplicaciones para ayudar la discapacidad visual

Lectores de pantalla

Los lectores de pantalla verbalizan y producen el sonido de cada elemento que se encuentre en la pantalla, así como de cada tecla o función que se ejecute en el teclado. La lectura de la información se realiza secuencialmente, es decir, hacen un barrido de izquierda a derecha y de arriba hacia abajo.

Los lectores de pantalla son utilizados principalmente por discapacitados que presentan ceguera total, aunque puede servir de ayuda y apoyo a cualquier persona que presente algún tipo de discapacidad visual.

- **Orca:**

Dentro de los distintos lectores existentes el más difundido es el Orca debido a que se destaca por sus posibilidades de configuración destinadas a las necesidades y preferencias de cada usuario. Orca es una ayuda técnica libre/*open source*[9], flexible, extensible y potente para las personas ciegas y deficientes visuales que utilizan el sistema operativo Linux. Se incluye en septiembre del 2006 en la versión 2.16 de GNOME. Usando varias combinaciones de voz, Braille y ampliación, Orca ayuda a proporcionar accesibilidad a las aplicaciones. El desarrollo de Orca ha sido liderado por el *Accessibility Program Office*

9 Software de código abierto.

de *Sun Microsystems, Inc.* con las contribuciones de muchos miembros de la comunidad (20).

Orca no es una sola aplicación, sino un conjunto de aplicaciones, controladores y voces que permiten que una persona con discapacidad visual pueda usar programas como OpenOffice, reproductores de música y programas de chat. De esta manera, la persona discapacitada usa las mismas herramientas que una persona sin discapacidad; pero con Orca actuando como intermediario. El proyecto Orca combina herramientas de síntesis de voz, para que el ordenador lea en voz alta lo que aparece en la pantalla, con la posibilidad de trabajar con Braille y ampliación de pantalla (21).

- **Emacspeak:**

Emacspeak es un lector de pantalla de código abierto bajo la Licencia Pública General GNU, desarrollado por T. V. Raman. Se obtiene de forma gratuita para plataforma Linux. Comenzó siendo un lector de pantalla para el editor Emacs, de las plataformas UNIX[10] y Linux. Actualmente es un "paquete parlante de aplicaciones" que incluye u ofrece soporte para editor de texto, clientes de correo electrónico, calendario, edición de archivos, ambiente de programación, hojas de cálculo, multimedia, bases de datos, chat, FTP, juegos, navegadores web (W3, Lynx, W3M, *Links, Home Page Reader*) y soporte para el sistema DAISY de lectura de libros electrónicos. Trabaja con los motores de voz ViaVoice y DECtalk (20).

Contiene un conjunto de herramientas organizadas en torno a diferentes tareas que permiten transformar datos en voz. Al combinarse con Linux trabajando en equipos de bajo costo, Emacspeak proporciona una solución confiable y estable de recursos de voz, y permite que Internet sea utilizada por personas con discapacidad visual en todo el mundo (22).

- **Gnopernicus:**

Gnopernicus es parte del Proyecto de Accesibilidad de GNOME. Permite a los usuarios con visión limitada, o sin visión, usar el escritorio y las aplicaciones GNOME. Es un paquete de utilidades compuesto de una lupa ampliadora de pantalla, lectura de pantalla con voz mediante el sintetizador Festival, y uso de un teclado Braille para mostrar la salida texto. El lector de pantalla Gnopernicus permite a los usuarios utilizar, basándose en la información recibida a través de la síntesis de voz y Braille, las aplicaciones cuya

10 Sistema Operativo Portable.

interfaz de usuario estándar está basado en Java y GTK 2[11]. Este sistema ha quedado obsoleto y ha sido desbancado por Orca (23).

- **Gnome-Speech:**

El paquete Gnome-Speech proporciona una API[12] general que facilita la programación de software basado en librerías Gnome con funciones para producir voz a partir de texto. La librería Gnome-Speech soporta diversas interfaces; pero actualmente sólo está activada en este paquete la interfaz Festival, el resto requiere Java o software propietario. Gnome-Speech puede utilizar a través de voz, Festival, eSpeak, FreeTTS, DecTalk y Cepstral (23).

- **BRLTTY:**

Demonio que permite usar la terminal o consola de Unix/Linux a través de un teclado Braille conectado al puerto de serie. Es un proceso informático y no interactivo que se ejecuta en segundo plano en vez de ser controlado directamente por el usuario. Lee en voz alta los mensajes de la consola para facilitar la interacción con el usuario. El demonio BRLTTY puede ser lanzado al inicio del sistema, para ayudar a iniciar la sesión. BRLTTY permite el uso completo de la función de terminal virtual del controlador de consola. Es adecuado para realizar las tareas de administración del sistema. Es igual de eficaz, fiable y más flexible que los programas de MS-DOS. Contiene soporte para múltiples códigos Braille. Ofrece una utilización opcional de parpadeo (velocidades individualmente ajustables) para el cursor, remarcado especial de subrayado, y/o letras en mayúsculas, además congelado de la pantalla para una revisión sin prisa. BRLTTY es utilizado en sistemas como Linux, Solaris, OpenBSD, FreeBSD, NetBSD, y Microsoft Windows. Actualmente no está planeado el portarlo a otros sistemas operativos similares a Unix (24).

- **Festival:**

Es un sistema de síntesis de voz de propósito general para múltiples lenguajes, desarrollado originalmente por el Centro de Investigación de Tecnologías del Lenguaje de la Universidad de Edimburgo. Festival ofrece un marco general para la construcción de sistemas de síntesis de voz. En su conjunto se ofrece el

11 Biblioteca la cual contiene los objetos y funciones para crear la interfaz de usuario.
12 Interfaz Visual de Aplicación.

texto completo a voz a través de una API de números: desde el nivel del depósito, a través de un intérprete de comandos Scheme, como una biblioteca de C++, desde Java, y una interfaz de Emacs. Esta herramienta es software libre y multilingüe (actualmente inglés tanto británico como americano, y español), aunque inglés es el más avanzado. Festival y las herramientas de voz se distribuyen bajo una licencia de tipo X11[13] que permite el uso comercial y no comercial sin restricciones por igual (23).

- **Gnome-Shell:**

Es la interfaz de usuario básica del entorno de escritorio GNOME. Proporciona funciones de interfaz como el cambio de ventanas, lanzar aplicaciones o ver sus notificaciones. Se aprovecha de las capacidades del hardware de gráficos moderno e introduce conceptos de interfaz de usuario fáciles de usar (25). El escritorio GNOME incluye las tecnologías de asistencia para apoyar a los usuarios con diferentes discapacidades y necesidades especiales, y para interactuar con dispositivos de ayuda comunes. Ofrece la posibilidad de agregar un menú de accesibilidad a la barra superior, dando un acceso más fácil a muchas de las funciones de accesibilidad. Incluye funcionalidades como (26):

- Leer pantalla en voz alta utilizando el lector de pantalla Orca.
- Ajustar el contraste haciendo que las ventanas y botones que aparecen en la pantalla sean más fáciles de ver.
- Cambiar tamaño del texto en la pantalla utilizando las fuentes más grandes para hacer el texto más fácil de leer así como ampliar una zona de la pantalla.
- Hacer el parpadeo del cursor del teclado, o sea, hacer el punto de inserción de parpadeo y controlar la rapidez con que parpadea.
- Ajustar la velocidad del ratón, permite cambiar la rapidez con la que el puntero se mueve al utilizar el ratón.
- Activar las teclas del ratón para controlar el ratón con el teclado numérico.
- Ajustar la velocidad de doble clic.

13 Sistema de ventanas transparente a la red informática para presentaciones de mapas de bits.

- Simular un clic derecho del ratón manteniendo pulsado el botón izquierdo del ratón para hacer clic derecho.
- Navegación por teclado lo que permite utilizar aplicaciones y el escritorio sin un ratón.
- Hacer que el teclado no repita las letras cuando se mantiene pulsada una tecla, o cambiar el retraso y la velocidad de las teclas de repetición.
- Ignorar pulsaciones de teclas rápidamente repetidas de la misma clave.
- Tipo de atajos de teclado, lo que permite pulsar una tecla a la vez en lugar de tener que mantener pulsado todas las teclas a la vez.
- Utilizar un teclado en pantalla para introducir texto pulsando los botones con el ratón o una pantalla táctil.

Sistemas de ampliación de pantalla

Un sistema de ampliación de pantalla es un software que amplifica cierta zona de la pantalla, generalmente la que tiene el foco del cursor, como si fuera una lupa. Permite a personas con baja visión aumentar el tamaño de los contenidos; además posibilita determinar la escala para los ampliadores, así como un tipo de ampliación entre los tipos disponibles.

- **Kmagnifier:**

Programa que incrementa considerablemente algunas áreas de la pantalla, de modo que puedan ser leídas con mayor facilidad. Kmagnifier, o Kmag, aumenta el área alrededor del cursor, o de un área definida por los usuarios. Ofrece la posibilidad de guardar las porciones aumentadas de la pantalla en el disco. Es usado por personas con discapacidad visual, por aquellos que trabajan en el campo del análisis de imágenes y desarrolladores web (23).

- **GNOME Magnifier:**

Amplificador de la visión de pantalla que puede usarse desde la línea de comandos; pero más comúnmente es accedido por otras aplicaciones y tecnologías de asistencia. GNOME Magnifier puede ser habilitado independientemente, o junto con Orca (23). Su principal utilidad es dotar a las aplicaciones de

asistencia basadas en AT-SPI[14] de servicios de ampliación de pantalla, tanto parcial, como a pantalla completa. Este último modo se ha conseguido mejorar considerablemente en las recientes versiones, así como ofrecer unos filtros de color para daltonismo (27).

- **Xzoom:**

Amplificador de pantalla disponible para cualquier distribución con servidor gráfico, continuamente está actualizando el área aumentada para mostrar los cambios. Agranda porciones específicas del escritorio (seleccionadas con el ratón) y es lo suficientemente rápido y ligero como para agrandar vídeos y animaciones (23). Produce aumentos continuos que permiten, por ejemplo, desplazar un documento arriba y abajo, mientras se mantiene aumentada la sección que se está leyendo. Como alternativa, se puede mover una pequeña caja por la pantalla, aumentando su contenido y permitiéndole buscar la zona que desea ver. Las mayores desventajas de Xzoom son que no puede agrandarse a sí mismo, que algunos de los controles de teclado son incompatibles con ***fvwm*** (el manejador de ventanas de Linux) y que su configuración por defecto no tiene soporte de red (esta desventaja se puede arreglar a cambio de perder algo de velocidad) (28).

1.4.2 Aplicaciones para ayudar a la discapacidad motora

Herramientas para facilitar el uso del ratón

Las herramientas del ratón permiten usar este dispositivo de diversas maneras, y así ofrecer soluciones alternativas a personas con movilidad reducida. Dentro de las herramientas disponibles que cumplen esta función se encuentran:

- **KmouseTool:**

Programa de KDE, ofrece un método alternativo para hacer clic con el ratón, al hacer que éste haga un clic cada vez que el cursor se detenga, y también ofrece la posibilidad de arrastrar lo que haya sido seleccionado. KMouseTool funciona con cualquier ratón o dispositivo apuntador. Fue diseñado para ayudar a aquellos que por padecer algún tipo de lesión sienten dolor al pulsar los botones del ratón.

14 Interfaz de proveedor de servicios de Tecnología de Asistencia.

KMouseTool también puede configurarse para esperar un tiempo específico para arrastrar al principio antes de soltar el ratón (29).

Su función se basa en controlar el movimiento del ratón, y se encarga de generar una pulsación cuando este último se detiene por algunos segundos. En el modo de arrastre inteligente, KMouseTool también permite arrastrar los elementos de un lugar a otro de la pantalla (29).

- **MouseTweaks:**

Su desarrollo se inició durante el evento *Google Summer of Code 2007.* Es un conjunto de mejoras de la accesibilidad para dispositivos apuntadores. El paquete de MouseTweaks proporciona las funciones que incluye la pestaña *Accessibility* (Accesibilidad) de la herramienta de preferencias del ratón. Ofrece funciones para simular clics secundarios, clics anidados, o retención del puntero. Esta aplicación provee el llamado *dwell-clicking*, lo cual significa la posibilidad de hacer clic tan solo con colocar el cursor sobre un botón o enlace por un determinado tiempo. Esto es útil para aquellas personas que no pueden de ningún modo presionar dispositivos. También permite abrir un menú contextual al mantener el botón izquierdo del ratón activado por un espacio de tiempo. Además, contiene dos miniaplicaciones de panel relacionadas con la accesibilidad del ratón (30):

- Captura de puntero: provee un área en el panel en el cual es posible bloquear el puntero.
- *Dwell-click*: una funcionalidad simple que permite cambiar de tipos de clic.

La presente herramienta tiene dos características particulares (31):

- Permite a los usuarios realizar un clic secundario manteniendo presionado el botón principal del ratón.
- Permite hacer los distintos clics sin utilizar ningún botón de hardware.

- **Eviacam:**

Es una aplicación libre (licencia GNU/GPL), que permite realizar movimientos del cursor del ratón con la cabeza, y permite especificar la acción a realizar (no clic, izquierdo, derecho, arrastre, doble clic, mostrar ventana), que se ejecutará en el momento que se ubique el cursor y se mantenga sin movimiento (32).

Mover la cabeza hará que el puntero del ratón también se desplace. Si el usuario se mantiene quieto, Eviacam lo interpretará como un clic. Funciona en un ordenador con una *webcam* sin elementos adicionales. El rendimiento de Eviacam es óptimo. Con un poco de práctica, permite manejar por completo el ordenador con sólo mover la cabeza (33).

Teclados en pantalla

Los teclados en pantalla han sido creados para entornos en los cuales no existe un teclado físico. Este tipo de teclados es muy útil para mejorar la accesibilidad cuando la computadora tampoco tiene conectado un ratón, y en su lugar hay otro dispositivo apuntador para el que no se usen las manos. Algunos de los teclados en pantalla existentes son:

- **Gok:**

Siglas de *GNOME Onscreen Keyboard* (Teclado en pantalla de GNOME), es una aplicación creada para el escritorio GNOME del sistema operativo GNU/Linux, que permite interactuar con la computadora a través del ratón o dispositivos alternativos. Consiste en un teclado alfanumérico virtual, que realiza entrada de texto e interfaz gráfico de usuario, controla todas las funciones de las aplicaciones Gnome, permite insertar caracteres especiales o escribir documentos de textos con el ratón. También permite crear teclados personalizados. La ventaja principal de GOK es que este panel virtual y todas sus funciones, se pueden acceder utilizando el clic directo con un ratón convencional o cualquier otro método alternativo de selección (34).

- **Florence:**

Florence es un teclado virtual extensible y escalable, cuyo único requerimiento es un dispositivo apuntador. En el caso de que los usuarios tengan dificultades para pulsar sobre los botones del ratón, cuenta con un temporizador, que se activa en el momento en que el puntero se posiciona sobre una tecla. Entre las ventajas de Florence, destaca el hecho de contar con extensiones de teclas, y de ser altamente configurable (35).

- **Dasher:**

Es una interfaz de entrada de texto de tipo predictivo, impulsada por gestos naturales del ratón, permitiendo escribir a través de un sofisticado sistema basado en el movimiento del puntero del ratón, o sea, permite sustituir la escritura del teclado por el movimiento de ratón. Dasher no es en realidad un teclado, en su lugar utiliza una interfaz de *zoom* y un modelo de lenguaje predictivo para la construcción de palabras. Esta aplicación logra que el ingreso de datos sea más sencillo para personas que con solo una mano pueden utilizar una palanca de juegos, pantalla táctil, *trackball*, o un ratón especial. Es muy útil para aquellas personas que se vean obligadas a manejar la computadora con una sola mano o con ninguna (a través de un *eyetracker*) (23).

1.5 Valoración acerca del estudio de las herramientas para discapacitados

Luego del análisis de las herramientas para personas discapacitadas se propone integrar las siguientes aplicaciones al sistema por las características expuestas en el epígrafe 1.4 que las hacen resaltar por encima de las otras:

- Lector de pantalla Orca.
- El entorno de escritorio GNOME-Shell, por incluir entre sus configuraciones un conjunto de aplicaciones para los discapacitados visuales.
- Para apoyar el uso del ratón las herramientas MouseTweaks y Eviacam.

1.6 Tecnologías a utilizar en el desarrollo del sistema

1.6.1 Herramienta para la creación del Sistema Operativo Base

En el Centro de Software Libre (CESOL), actualmente se utiliza para la creación del Sistema Operativo Base de Nova la herramienta Debootstrap en su versión 1.0.40. En los inicios del sistema, este proceso se realizaba manualmente; pero gracias a la automatización que ofrece esta herramienta, ha sido implantada en los proyectos de este centro desde algún tiempo.

Debootstrap constituye el arranque de un sistema Debian básico, se utiliza para crear un sistema base de

Debian, o derivado de este, desde cero, sin requerir la disponibilidad de dpkg[15] o apt[16]. Para ello, descarga ficheros de extensión **.deb** desde un repositorio especificado previamente, y los desempaqueta en un directorio al que finalmente se puede entrar con chroot[17].

La sintaxis de este comando es la siguiente:

debootstrap [OPCIONES...] $DISTRO $MONTAJE $MIRROR

- $DISTRO se refiere a la versión de Debian, Ubuntu, o en este caso de Nova, que se desea instalar.
- $MONTAJE hace referencia al punto de montaje o directorio donde se quiere instalar la distribución.
- $MIRROR se refiere al repositorio o espejo (*mirror* en inglés), de donde se van a descargar los paquetes para la instalación.

A continuación se explican de manera breve las principales opciones de Debootstrap:

- --arch=ARCH: selecciona la arquitectura que poseerá el nuevo sistema, se sustituye normalmente ARCH por i386 o amd64.
- --include=alpha,beta: lista de paquetes separados por coma que se van a agregar a la descarga.
- --exclude=alpha,beta: lista de paquetes separados por coma que serán retirados de la descarga. Hay que tener mucho cuidado con esta opción, ya que se podrían excluir paquetes esenciales.
- --components=alpha,beta: utilizar paquetes de los componentes listados.
- --variant=minbase | buildd | fakechroot | scratchbox: nombre de la variante a utilizar. Actualmente, las variantes admitidas son:
  - minbase, que sólo incluye paquetes esenciales y apt.
  - buildd, que instala el build-essential[18] en $MONTAJE.

15 Es el gestor de paquetes Debian de medio nivel. Se utiliza para instalar, construir, borrar y gestionar los paquetes de Debian GNU/Linux.
16 Sistema de gestión de paquetes, simplifica en gran medida la instalación y eliminación de programas en los sistemas GNU/Linux.
17 Jaula de construcción del sistema montado.
18 Es un paquete que contiene herramientas necesarias para la creación, compilación e instalación de programas.

- fakechroot, que instala los paquetes sin privilegios de root.
- scratchbox[19], es para la creación de destinos para el uso scratchbox. De forma predeterminada, sin un argumento --variante = X, se crea una instalación de Debian base en $MONTAJE.

• --verbose: producir más información acerca de la descarga.

1.6.2 Herramienta para la creación del ISO

Genisoimage 1.1.11:

Antes llamado mkisofs; cambió de nombre a partir de la versión 4 Etch de Debian. Acrónimo de *Generate ISO Image*, es un programa para crear imágenes de sistemas de archivos ISO-9660 para CD-ROM, que pueden grabarse en CD o DVD usando un programa específico para quemado. Es capaz de generar el protocolo de sistema de uso compartido de registros (SUSP), especificados por el protocolo de intercambio *Rock Ridge*. Esto se utiliza para describir aún más los ficheros en el sistema de archivos ISO-9660 a un Host UNIX y proporciona información como nombres de archivos largos, permisos POSIX[20], enlaces simbólicos y dispositivo de bloque. Genisoimage incluye herramientas útiles para trabajar con las imágenes ISO:

- **mkzftree**, crea la imagen ISO-9660 con contenidos comprimidos.
- **dirsplit**, separa fácilmente grandes contenidos de un directorio en discos de un tamaño predefinido.

Características:

• Puede generar un verdadero (o compartido) sistema de ficheros híbrido HFS[21]. Se observan los mismos archivos cuando se accede desde un Macintosh y los archivos en formato ISO-9660 cuando se accede desde otras máquinas.

19 Es un conjunto de herramientas de compilación cruzada diseñado para hacer que el desarrollo de aplicaciones Linux embebido sea más fácil. También proporciona un conjunto completo de herramientas para integrar y realizar una compilación cruzada de una distribución Linux completa.

20 Interfaz Portable de Sistema Operativo. Permite asignar permisos, o derechos de acceso, a los archivos para determinados usuarios o grupos.

21 Sistema de Archivos Jerárquico o *Hierarchical File System* (HFS), es un sistema de archivos desarrollado por *Apple Inc*. para su uso en computadores que corren Mac OS.

• Como alternativa, Genisoimage puede generar las extensiones de Apple ISO-9660 para cada archivo. Estas extensiones proporcionan a cada fichero algunas banderas Finder[22] cuando se accede desde un Macintosh.

• Genisoimage toma lo que se encuentra en el directorio seleccionado y genera una imagen binaria que corresponderá a un sistema de archivos ISO-9660 y/o sistema de ficheros HFS mientras escribe en un dispositivo de bloque.

• Cuando se utilizan varias opciones de HFS, Genisoimage intentará un reconocimiento de archivos almacenados en un número de formatos de Apple/Unix y copiará los datos y las bifurcaciones de recursos, así como cualquier información relevante del Finder.

• Se debe tener en cuenta que Genisoimage no está diseñado para comunicarse directamente con el hardware. La mayoría de los quemadores tienen un conjunto de comandos propietarios que pueden variar de un fabricante a otro, y necesita una herramienta especializada para grabar el disco.

1.6.3 Editor de texto

Nano 2.2.6 (36):

Es un sencillo editor de textos para el terminal que viene instalado por defecto en Nova. No es tan potente como los editores de texto *Vim* o *Emacs* pero es mucho más fácil de manejar que estos. Al ser un editor en modo texto, se suele usar sobretodo en entornos sin interfaz gráfica como *Ubuntu Server*, pero eso no impide que se utilice en *Ubuntu Desktop*. Para editar un archivo con Nano se ejecuta el siguiente comando:

nano nombre_archivo*;* donde nombre_archivo es el nombre del archivo que se quiere editar. En caso de que el archivo no existiera, se creará uno vacío con ese nombre.

Nano está pensado para usarse con el teclado, no con el ratón, por lo que tiene asociadas multitud de acciones a combinaciones de teclas.

22 Finder es la aplicación ejecutada en el sistema operativo Mac OS X responsable de la gestión total de los archivos de usuario, discos, red y el lanzamiento de otras aplicaciones. Finder actúa como el *shell* en otros sistemas operativos, pero usando una interfaz gráfica de usuario (GUI).

1.7 Metodología de desarrollo de software

Una metodología no es más que un conjunto de métodos, estas se elaboran a partir del marco definido por uno o varios modelos del ciclo de vida. Una metodología de desarrollo consiste en un conjunto de procedimientos, técnicas, herramientas y soporte documental, que deben seguirse para el progreso de un software (37).

Actualmente existen varias metodologías, cada una de ellas con características particulares que las hacen diferenciarse, son clasificadas en dos grupos: metodologías ágiles y tradicionales. Las metodologías tradicionales se basan en la idea de que el éxito del producto se puede lograr si se tiene todo correctamente documentado, mientras que, las ágiles defienden la idea de que el proceso de desarrollo del software, se centra en el software como tal y no en la documentación alrededor de este, sino que se toma en cuenta sólo la documentación necesaria y de forma muy sencilla.

Se escogió una metodología de desarrollo de software ágil, por las características del desarrollo del producto, que se basa principalmente en un equipo de desarrollo pequeño y con poca experiencia en el desarrollo de ese tipo de software, además el período de tiempo de entrega del producto es relativamente corto y los requisitos son cambiantes. Específicamente se va a utilizar la metodología Nova-OpenUp ya que es la más adecuada para el desarrollo de una distribución GNU/Linux o una personalización de esta.

1.7.1 Metodología de desarrollo Nova-OpenUp

Nova-OpenUp es una metodología de desarrollo de distribuciones GNU/Linux, abarca todo el ciclo de vida de proyectos que hacen productos de este tipo mediante la ejecución de un conjunto de disciplinas compuestas por actividades que se relacionan entre sí, roles y artefactos. En ella se tienen en cuenta buenas prácticas que sugieren metodologías como OpenUp, XP[23], y otras que corresponden al nivel 2 de CMMi[24]. Esta metodología se desarrolla en 4 fases (5):

- **Inicio:** en esta fase se debe entender el objetivo a cumplir, conocer los requisitos que debe tener el producto, y realizar un análisis para determinar costos y riesgos asociados al proyecto.

23 De las siglas en inglés *Extreme Programming* (Programación Extrema), metodología ágil de desarrollo de software.
24 De las siglas en inglés *Capability Maturaty Model Integration* (Integración de modelos de madurez de capacidades), es un modelo para la mejora y evaluación de procesos para el desarrollo, mantenimiento y operación de sistemas de software.

- **Elaboración:** los objetivos fundamentales de esta fase son obtener los requisitos de manera más detallada, diseñar, validar e implementar la arquitectura de las distribuciones GNU/Linux así como mitigar los riesgos necesarios.
- **Construcción:** esta fase logra dar al producto capacidad operacional cumpliendo con los objetivos, su finalidad es completar el desarrollo del sistema, para ello se debe crear una versión del producto que cumpla con los requisitos especificados y listo para utilizarse por los usuarios, así como optimizar los recursos necesarios y reutilizar componentes para minimizar los costos de desarrollo.
- **Transición:** en esta fase es donde se obtiene una liberación candidata del sistema, la cual se debe validar para saber si cumple con las expectativas y desplegar en los entornos de los usuarios finales.

1.8 Lenguaje de desarrollo

Bash 4.2 (5):

Bash (*Bourne again shell*) es un programa informático cuya función consiste en interpretar órdenes. Está basado en el *shell*[25] de Unix y es compatible con POSIX. Fue escrito para el proyecto GNU y es el intérprete de comandos por defecto en la mayoría de las distribuciones de Linux. Su nombre es un acrónimo de *Bourne-Again Shell* (otro *shell bourne*) el *Bourne shell* (sh), fue uno de los primeros intérpretes importantes de Unix, la mayoría de los *scripts sh* pueden ser ejecutados por Bash y sin modificación.

Bash es una de las *shells* más populares de distribuciones de GNU/Linux, y la mayoría de los *scripts* de configuración de estas están programados en este lenguaje, por lo que para personalizar las mismas se necesita tener un gran conocimiento acerca de su uso.

1.9 Herramienta de modelado

Visual Paradigm 8.0:

25 Intérprete de órdenes.

Es una herramienta CASE[26] para desarrollo de aplicaciones utilizando el lenguaje de modelado UML[27] que permite la representación de todo tipo de diagramas (Procesos de Negocio, Decisión, Actor de negocio, Documento). Fue diseñado para una amplia gama de usuarios interesados en la construcción de sistemas de software de forma fiable a través de la utilización de un enfoque Orientado a Objetos. Se encuentra disponible para varias plataformas y cuenta con múltiples versiones con diferentes especificaciones. Entre los elementos que ofrece Visual Paradigm se encuentran (5):

- Navegación intuitiva entre la escritura del código y su visualización.
- Potente generador de informes en formato PDF/HTML.
- Documentación automática Ad-hoc[28].
- Ambiente visualmente superior de modelado.
- Sofisticado diagramador automático de Layout[29].

Conclusiones del capítulo

Como resultado de la investigación y el análisis bibliográfico realizado a lo largo de este capítulo, han sido expuestos los principales puntos de interés con el objetivo de establecer las bases teóricas para el desarrollo de la personalización de GNU/Linux Nova para personas discapacitadas visuales y motoras.

El estudio realizado sobre las distribuciones existentes para el apoyo a personas discapacitadas disponibles en Linux arrojó que ninguna ofrece la solución mas óptima a la problemática planteada, por lo que se decidió desarrollar una personalización de GNU/Linux Nova tomando como referencia las distribuciones estudiadas.

Se realizó el estudio de algunas aplicaciones destinadas para el apoyo a personas con discapacidades visuales y motoras, lo que permitió determinar las que se desean integrar en la distribución cubana de GNU/Linux Nova.

En el presente capítulo fueron definidos, como lenguaje de programación a utilizar, Bash, como editor de

26 Ingeniería de Software Asistida por Computadora.
27 Lenguaje de Modelado Unificado.
28 Que es apropiado o está dispuesto especialmente para un fin: solución ad hoc.
29 Suele utilizarse para nombrar al esquema de distribución de los elementos dentro un diseño.

textos Nano, y como metodología de desarrollo de software para guiar el proceso de desarrollo, Nova-OpenUp en su variante ligera.

En este punto y luego de haber definido las principales características de la propuesta de solución, así como los elementos que serán utilizados en su concepción, están creadas las bases teóricas para proceder al análisis y diseño de la solución propuesta; elementos que serán tratados en el próximo capítulo.

Capítulo 2. Diseño y modelación del sistema

Introducción

El diseño es la parte del proceso de desarrollo de software cuyo propósito es decidir cómo el sistema se llevará a cabo. Durante el diseño, se toman decisiones estratégicas y tácticas para cumplir los requerimientos funcionales y de calidad de un sistema.

En este capítulo se plasman los resultados de la etapa de diseño del sistema, siguiendo la metodología Nova-OpenUp, para llegar a tener un producto con buena calidad teniendo en cuenta las prácticas y los principios que contribuyan a agilizar el proceso.

2.1 Propuesta del sistema a desarrollar

Para dar solución a la problemática planteada se propone la realización de una personalización de GNU/Linux Nova para discapacitados visuales y motores. Esta personalización utilizará el núcleo de Linux e incluirá un paquete de aplicaciones informáticas que faciliten el acceso de usuarios discapacitados visuales y motores al mismo. Al ser una personalización de la distribución cubana, la misma será parte del proceso de la migración a plataformas de código abierto que experimenta Cuba en la actualidad. La construcción y distribución de esta personalización de GNU/Linux Nova están enfocadas a alcanzar niveles de excelencia y a la obtención de un producto seguro.

A continuación se describe el funcionamiento del sistema a partir del modelo de relación entre componentes del mismo (*Ver Figura 1*). En dicho modelo se detallan los principales componentes del sistema y las relaciones que se establecen entre ellos.

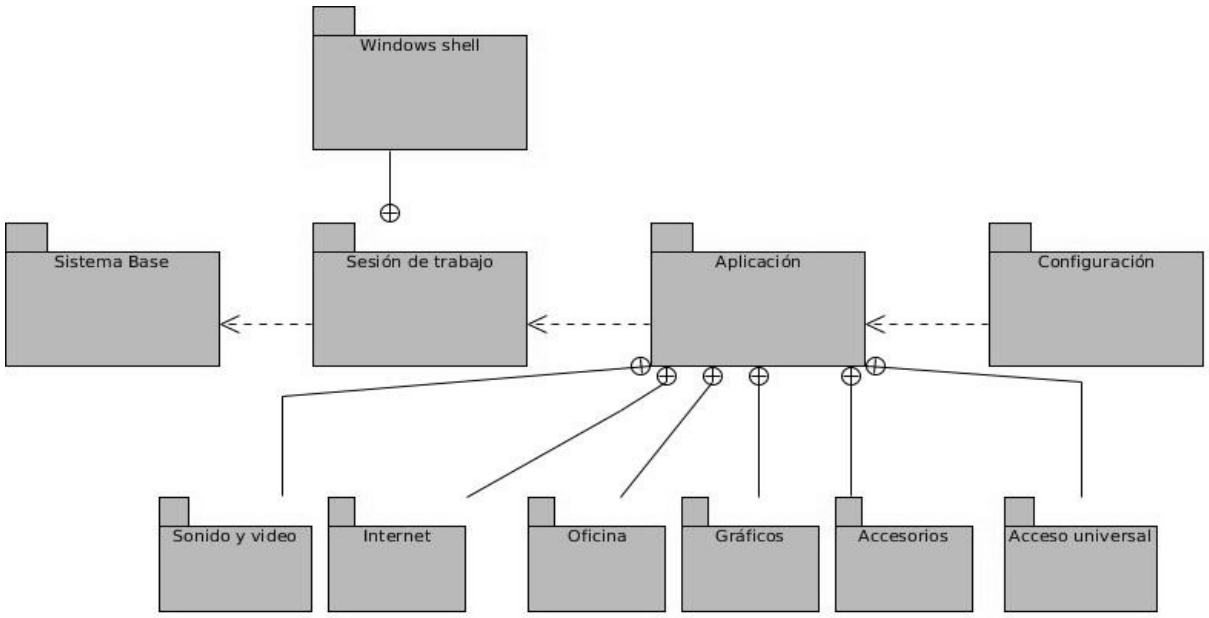


Figura 1: Relación entre componentes del sistema.

Descripción de los componentes del sistema:

- Sesión de Trabajo: Nova Bonita es la variante del Sistema Operativo Nova para personas discapacitadas visuales y motoras. Se trata de un sistema operativo que utiliza el núcleo Linux e incluye determinados paquetes de aplicaciones informáticas para satisfacer las necesidades de usuarios discapacitados.

- Windows-shell: esta es la sesión dedicada a suavizar el cambio de los usuarios que migran al sistema operativo Nova desde Microsoft Windows, por lo que su diseño se parece mucho al de este último.

- Sistema Operativo Base: se dedicará al Desarrollo de las Capas Tecnológicas Básicas del Sistema Operativo, la definición de la Arquitectura, la Persistencia o Gestión de los Paquetes de Software, así como el desarrollo de tecnologías y servicios para el soporte de los productos asociados a las Líneas de Producción y Sistemas a la medida.

- Sonido y vídeo: aplicaciones para la grabación y reproducción de archivos de vídeo y audio.

- Internet: paquete de herramientas que permiten la comunicación del usuario con la Internet.
- Oficina: paquete de herramientas ofimáticas para el trabajo de oficina.
- Gráficos: paquete de herramientas para el procesamiento de datos con representación gráfica.
- Accesorios: paquete de herramientas auxiliares utilizadas para realizar cierto trabajo o que permiten un funcionamiento complementario de una máquina.
- Acceso Universal: se refiere a los métodos de interacción humano-computadora para usuarios con limitaciones físico-motoras.
- Configuración: es el gestor de configuración del sistema. Permite configurar: la apariencia del sistema (fondo de escritorio y tema de ventanas), dispositivos Bluetooth, resolución y rotación de monitores así como las configuraciones con varios monitores, red, soporte de impresión, soporte de idiomas, usuarios y grupos, hora y fecha, controladores de hardware, sonido, distribución de teclado y teclas de acceso rápido, ratón, pantalla de bloqueo y brillo, opciones de privacidad de la información y acceso universal.

2.2 Gestión del proceso de construcción de la personalización

Para gestionar la construcción del sistema, se realizó la planificación de dicho proceso, dando como resultado un tiempo estimado de 7 meses para obtener el resultado esperado. A continuación se detallan las tareas que abarcan todo el ciclo de vida del proyecto (*Ver Tabla 1*).

Tabla 1: Tareas para el desarrollo del sistema.

No	Nombre	Duración	Inicio	Terminado
1	Proceso de creación de Nova Bonita	170 días	01/10/14 08:00	26/05/15 17:00
2	Recopilación de información	150 días	01/10/14 08:00	28/04/15 17:00
3	Capturar requisitos del cliente	10 días	01/10/14 07:00	14/10/14 17:00
4	Especificación y evaluación de requisitos	10 días	11/10/14 08:00	24/10/14 17:00

5	Descripción de requisitos ágiles	9 días	21/10/14 08:00	31/10/14 17:00
6	Definición de vista del sistema	10 días	08/11/14 08:00	21/11/14 17:00
7	Definición de vista de seguridad	10 días	21/11/14 08:00	04/12/14 17:00
8	Definición de vista de infraestructura	10 días	04/12/14 08:00	17/12/14 17:00
9	Definición de vista despliegue	10 días	17/12/14 09:00	31/12/14 09:00
10	Creación de sistema operativo base	15 días	30/12/14 09:00	20/01/15 09:00
11	Modificación de paquetes	10 días	17/01/15 09:00	30/01/15 17:00
12	Instalación de paquetes	15 días	30/01/15 09:00	20/02/15 09:00
13	Configuración del sistema	10 días	19/02/15 09:00	05/03/15 09:00
14	Elaboración de manual de usuario	15 días	04/03/15 09:00	25/03/15 09:00
15	Pruebas de aceptación	8 días	22/04/15 08:00	01/05/15 17:00
16	Plantilla de no conformidades	8 días	01/05/15 08:00	12/05/15 17:00

A continuación se encuentran las tareas correspondientes a la planificación realizada y se muestran estas en función a los riesgos asociados y las acciones para mitigarlos (*Ver Tabla 2*).

Tabla 2: Tareas para el proceso personalización de Nova Bonita en función de riesgos.

Asignadas a: Yisel Obiol González y Dayrol Lázaro Ferrer Durruthy					
Nombre	**Descripción**	**Fecha-inicio**	**Fecha-fin**	**Riesgo**	**Mitigación**
Recopilación de información.	- Estudio del estado del arte. - Estudio de tecnologías de apoyo a personas discapacitadas visuales y motores. - Estudio de herramientas para la creación de un SO.	1/10/2014	28/04/2015	- Poca información a consultar. - Mal uso de la información.	- Definir métodos de búsqueda de información. - Desarrollar correcta gestión de la información. - Establecer un efectivo control de las tareas.
Levantamiento de requisitos					
Capturar requisitos del cliente.	Se hará una reunión con el cliente, recopilando información sobre las necesidades del mismo para con el sistema a desarrollar.	1/10/2014	14/10/2014	- Pueden obviarse requisitos fundamentales para el desarrollo del sistema. - Demora en cuanto a tiempo.	- Ser precisos en las preguntas al cliente. - Establecer un efectivo control de las tareas.
Especificación y evaluación de requisitos del sistema.	Como artefacto de expediente de proyecto, según la metodología Nova-OpenUp, se redactará el documento oficial que contiene los requisitos del sistema (010113_Especificación_de_requisitos_de_software).	14/10/2014	24/10/2014	- Poca comprensión por parte del cliente. - El cliente se puede mostrar en desacuerdo. -Demora en cuanto a tiempo.	- Detallar correctamente las definiciones y acciones expuestas. - Establecer un efectivo control de las tareas.
Descripción de	Se detallarán los requisitos en función del resultado a	24/10/2014	01/11/2014	- Demora en la entrega.	- Establecer un efectivo control de las

requisitos ágiles.	esperar, teniendo en cuenta su procesamiento interno (010114c_Descripción_de_requisito_ágil).				tareas.
Definición de Arquitectura					
Definición de vista del sistema.	Es una descripción del sistema en cuanto a los componentes del mismo. Se establecerá la relación de prioridad entre los requisitos, se muestran los paquetes y componentes y los escenarios presentes en el sistema (010216_0_Arquitectura_de_software).	01/11/2014	21/11/2014	- Demora en la entrega. - Puede obviarse información imprescindible.	- Establecer un efectivo control de las tareas.
Definición de vista de seguridad.	Permitirá presentar los diferentes métodos para garantizar la seguridad de las distribuciones GNU/Linux a partir de los requisitos a cumplir y teniendo en cuenta los diferentes niveles del producto (010216_7_Arquitectura_vista_de_seguridad).	21/11/2014	04/12/2014	- Demora en la entrega. - Puede obviarse información imprescindible.	- Establecer un efectivo control de las tareas.
Definición de vista de infraestructura.	Definir las tecnologías (herramientas) a utilizar en el desarrollo del sistema y como deben ser configuradas (010216_8_Arquitectura_vista_de_infraestructura).	04/12/2014	17/12/2014	- Demora en la entrega. - Pueden obviarse herramientas fundamentales para el desarrollo del sistema.	- Establecer un efectivo control de las tareas.
Definición de vista despliegue.	Describirá las características que deben tener los nodos para que la distribución GNU/Linux pueda funcionar sobre ellos.	17/12/2014	30/12/2014	- Demora en la entrega.	- Establecer un efectivo control de las tareas.
Desarrollo del sistema					
Creación del SOB.	Se creará el Sistema Operativo Base, con el mínimo de paquetes requeridos para su posterior configuración.	30/12/2014	17/01/2015	- Demora en la entrega. - Puede pasarse por alto algún paquete requerido.	- Establecer un efectivo control de las tareas.
Instalación de	Se instalarán los paquetes necesarios para el	30/01/2015	19/02/2015	- No presenta.	

paquetes.	funcionamiento del sistema operativo final.				
Configuración de elementos del sistema.	Se modificará los archivos de configuración, para hacer que algunas aplicaciones se carguen al inicio del sistema.	19/02/2015	04/03/2015	- Demora en la entrega.	- Establecer un efectivo control de las tareas.
Elaboración de manual de usuario.	Constituye una ayuda a los usuarios del sistema para poder lograr un entendimiento del comportamiento de este.	04/03/2015	24/03/2015	- Demora en cuanto a tiempo.	- Establecer un efectivo control de las tareas.
Empaquetar el sistema.	Se empaqueta el sistema en un ISO que estará listo para guardar en usb, cd o dvd y estará listo para instalarse.	02/04/2015	11/04/2015	- No presenta.	
Pruebas al sistema					
Pruebas de aceptación.	El cliente analiza el producto, así como los implicados y estos deciden si el producto satisface o no sus necesidades y si no existe ningún elemento que disminuya la calidad del producto.	22/04/2015	01/05/2015	- No presenta.	

2.3 Modelo del proceso de construcción de la personalización de GNU/Linux Nova

Para modelar el proceso de construcción de la distribución de Nova para discapacitados visuales y motores fueron definidos los roles desempeñados por los miembros del equipo de desarrollo, las actividades realizadas por cada uno de ellos, las decisiones a tomar y los artefactos generados durante cada actividad (*Ver Figura 2*).

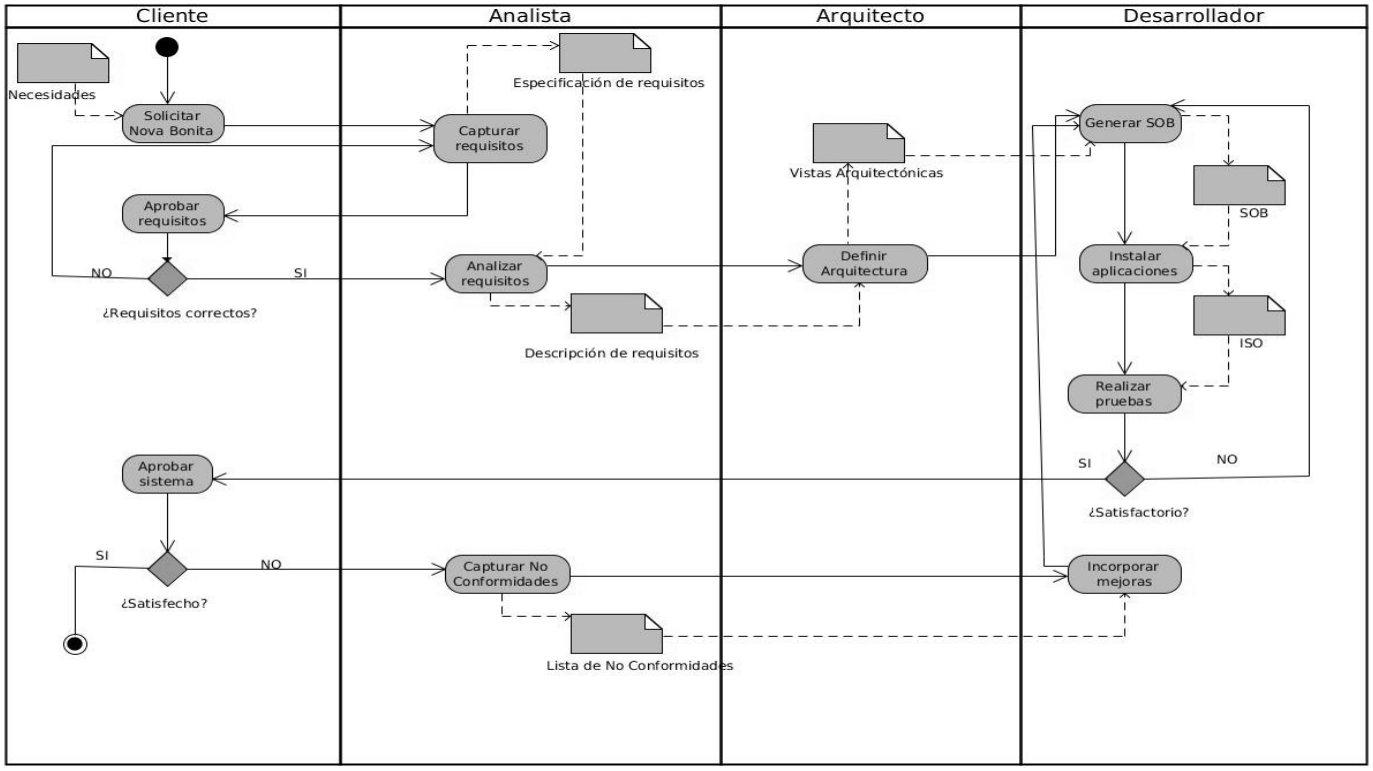


Figura 2: Diagrama del proceso de construcción de Nova Bonita.

2.4 Requisitos de software

Los requerimientos o requisitos para un software son la descripción de los servicios proporcionados por el sistema y sus restricciones operativas. Reflejan las necesidades de los clientes de un sistema que ayude a resolver algún problema (37). A continuación se muestran los requisitos obtenidos para el desarrollo de la personalización de Nova.

2.4.1 Requisitos Funcionales

Los requisitos funcionales de un software indican lo que el sistema debe hacer, dependiendo de la acción del usuario será la respuesta del mismo (37). A continuación se muestran los requisitos funcionales definidos para la personalización de GNU/Linux Nova para discapacitados visuales y motores.

RF1: Proveer aplicaciones para la categoría Accesorios.

RF2: Proveer aplicaciones para la categoría de Acceso Universal.

RF3: Proveer aplicaciones para la categoría de Gráficos.

RF4: Proveer aplicaciones para la categoría de Herramientas del sistema.

RF5: Proveer aplicaciones para la categoría de Internet.

RF6: Proveer aplicaciones para la categoría de Juegos.

RF7: Proveer aplicaciones para la categoría de Oficina.

RF8: Proveer aplicaciones para la categoría de Sonido y vídeo.

RF9: Proveer aplicaciones para la categoría de Configuración del sistema.

RF10: Instalar el sistema desde un dispositivo extraíble.

La metodología escogida, Nova-OpenUp, en su desarrollo genera diferentes artefactos pertenecientes al expediente de proyecto, que constituyen los elementos de documentación y apoyo en los diferentes procesos por los que el mismo transita. Entre estos se encuentra la descripción de requisitos ágiles, los cuales sirven para establecer una visión más clara de lo que se quiere de cada funcionalidad del sistema y brindar una guía mejor elaborada de las mismas.

En la siguiente sección se mostrará esta descripción para cada uno de los requisitos mencionados:

Tabla 3: Descripción del RF1 Proveer aplicaciones para la categoría de Accesorios.

Número: RF1	**Nombre del requisito:** Proveer aplicaciones para la categoría de Accesorios.
Programador: Yisel Obiol González y Dayrol Lázaro Ferrer Durruthy	**Iteración Asignada:** 1
Prioridad: Alta	**Tiempo Estimado:** 15 días
Riesgo en Desarrollo: Ausencia de desarrolladores por enfermedad. Pérdida de información imprescindible. Falta de personal calificado.	**Tiempo Real:** 3 semanas
Descripción: El sistema debe permitir a los usuarios navegar por sus archivos teniendo en cuenta sus permisos. Debe mostrar una ayuda que asista a los usuarios en sus actividades en el sistema. Permitir al usuario interactuar con el sistema a través de líneas de comandos, debe mostrar información sobre los dispositivos de almacenamiento conectados al terminal y comprimir y descomprimir archivos.	
Prototipo de interfaz	

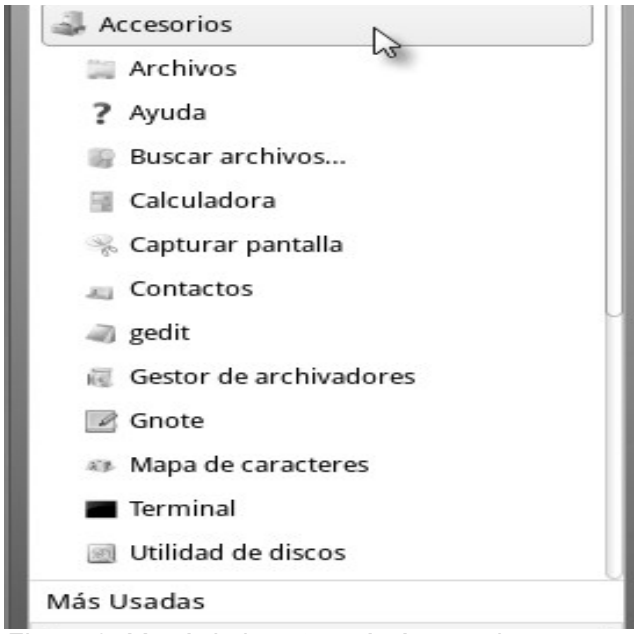


Figura 3: Menú de la categoría Accesorios.

Tabla 4: Descripción del RF2 Proveer aplicaciones para la categoría de Acceso Universal.

Número: RF2	**Nombre del requisito:** Proveer aplicaciones para la categoría de Acceso Universal.
Programador: Yisel Obiol González y Dayrol Lázaro Ferrer Durruthy	**Iteración Asignada:** 1
Prioridad: Alta	**Tiempo Estimado:** 15 días
Riesgo en Desarrollo: Ausencia de desarrolladores por enfermedad. Pérdida de información imprescindible. Falta de personal calificado.	**Tiempo Real:** 3 semanas
Descripción: El sistema debe tener aplicaciones y configuraciones que le faciliten la interacción a los usuarios discapacitados con el sistema: ampliadores de pantalla, lectores de pantalla, teclado en pantalla y sintetizadores de voz.	

Prototipo de interfaz

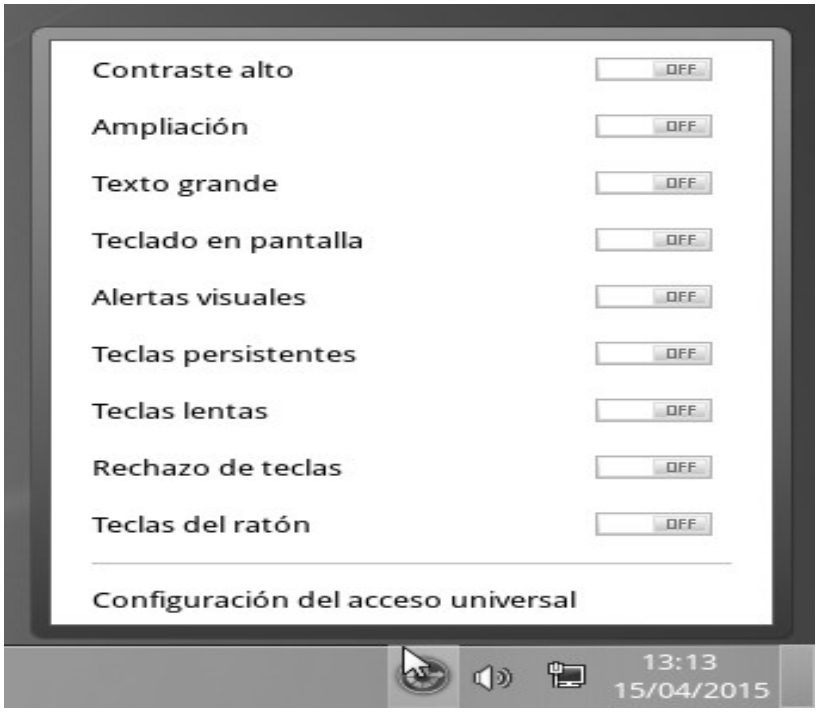


Figura 4: Menú de la categoría Acceso Universal.

Tabla 5: Descripción del RF3 Proveer aplicaciones para la categoría de Gráficos.

Número: RF3	**Nombre del requisito:** Proveer aplicaciones para la categoría de Gráficos.
Programador: Yisel Obiol González y Dayrol Lázaro Ferrer Durruthy	**Iteración Asignada:** 1
Prioridad: Alta	**Tiempo Estimado:** 21 días
Riesgo en Desarrollo: Ausencia de desarrolladores por enfermedad. Pérdida de información imprescindible. Falta de personal calificado.	**Tiempo Real:** 4 semanas
Descripción: El sistema debe contener la categoría de Gráficos donde estén las aplicaciones correspondientes a la misma. Se debe permitir la edición y visualización de imágenes, teniendo en cuenta la modificación parcial o completa de las mismas.	

Prototipo de interfaz

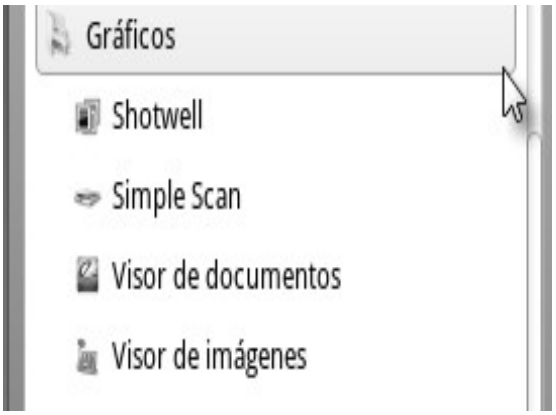


Figura 5: Menú de la categoría Gráficos.

Tabla 6: Descripción del RF4 Proveer aplicaciones para la categoría de Herramientas del sistema.

Número: RF4	**Nombre del requisito:** Proveer aplicaciones para la categoría de Herramientas del sistema.
Programador: Yisel Obiol González y Dayrol Lázaro Ferrer Durruthy	**Iteración Asignada:** 1
Prioridad: Alta	**Tiempo Estimado:** 21 días
Riesgo en Desarrollo: Ausencia de desarrolladores por enfermedad. Pérdida de información imprescindible. Falta de personal calificado.	**Tiempo Real:** 3 semanas
Descripción: El sistema debe tener un centro que le facilite al usuario la administración del sistema. En el se puede encontrar: Configuración del sistema, Centro de software, Cuentas de usuarios, Actualizaciones del sistema y Conexiones de red.	

Prototipo de interfaz

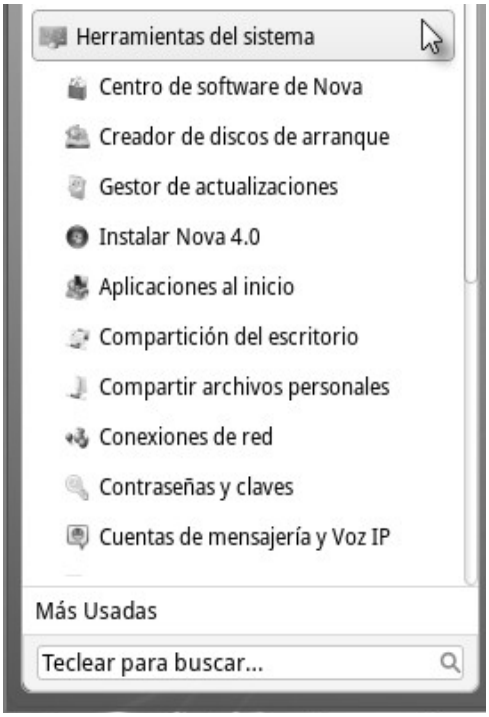


Figura 6: Menú de la categoría Herramientas del sistema.

Tabla 7: Descripción del RF5 Proveer aplicaciones para la categoría de Internet.

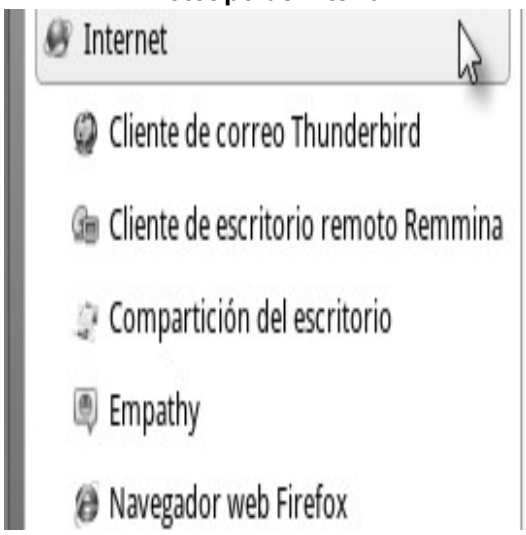
Número: RF5	**Nombre del requisito:** Proveer aplicaciones para la categoría Internet.	
Programador: Yisel Obiol González y Dayrol Lázaro Ferrer Durruthy		**Iteración Asignada:** 1
Prioridad: Alta		**Tiempo Estimado:** 21 días
Riesgo en Desarrollo: Ausencia de desarrolladores por enfermedad. Pérdida de información imprescindible. Falta de personal calificado.		**Tiempo Real:** 4 semanas
Descripción: El sistema debe permitir a los usuarios desde su PC administrar sus cuentas personales de correo, permitir la conexión desde la estación de trabajo actual a uno o más escritorios remotos a la vez, utilizar la mayoría de los estándares de mensajería instantánea y permitir la conexión a otros sistemas a través de la web. El sistema debe tener aplicaciones para trabajar conectado a la red: Cliente de correo. Navegador web. Cliente de escritorio remoto. Cliente de mensajería instantánea.		
Prototipo de interfaz  		

Figura 7: Menú de la categoría Internet.

Tabla 8: Descripción del RF6 Proveer aplicaciones para la categoría Juegos.

Número: RF6	**Nombre del requisito:** Proveer aplicaciones para la categoría Juegos.	
Programador: Yisel Obiol González y Dayrol Lázaro Ferrer Durruthy		**Iteración Asignada:** 1
Prioridad: media		**Tiempo Estimado:** 15 días
Riesgo en Desarrollo: Ausencia de desarrolladores por enfermedad. Pérdida de información imprescindible. Falta de personal calificado.		**Tiempo Real:** 3 semanas
Descripción: El sistema debe tener juegos para permitir a los usuarios jugar en su tiempo de ocio.		
Prototipo de interfaz		

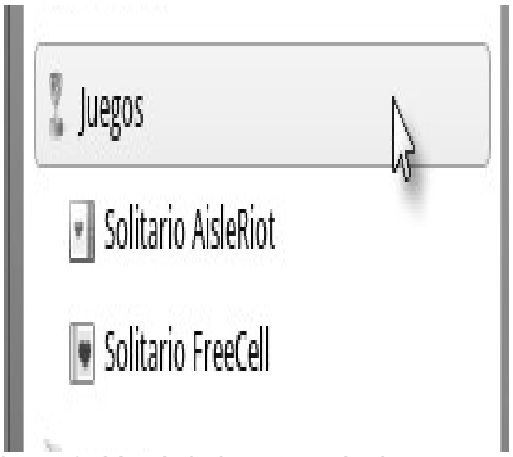


Figura 8: Menú de la categoría Juegos.

Tabla 9: Descripción del RF7 Proveer aplicaciones para la categoría Oficina.

Número: RF7	**Nombre del requisito:** Proveer aplicaciones para la categoría Oficina.
Programador: Yisel Obiol González y Dayrol Lázaro Ferrer Durruthy	**Iteración Asignada:** 1
Prioridad: Alta	**Tiempo Estimado:** 15 días
Riesgo en Desarrollo: Ausencia de desarrolladores por enfermedad. Pérdida de información imprescindible. Falta de personal calificado.	**Tiempo Real:** 3 semanas
Descripción: El sistema debe tener aplicaciones que permitan la interacción con documentos que cumplan con el estándar de Microsoft y el de OpenDocument. Además debe permitir crear, modificar, eliminar y mostrar documentos con formato pdf. Incluye procesador de texto, editor de hojas de cálculo, gestor de presentaciones, gestor de base de datos, editor de gráficos vectoriales y un editor de fórmulas matemáticas.	
Prototipo de interfaz	

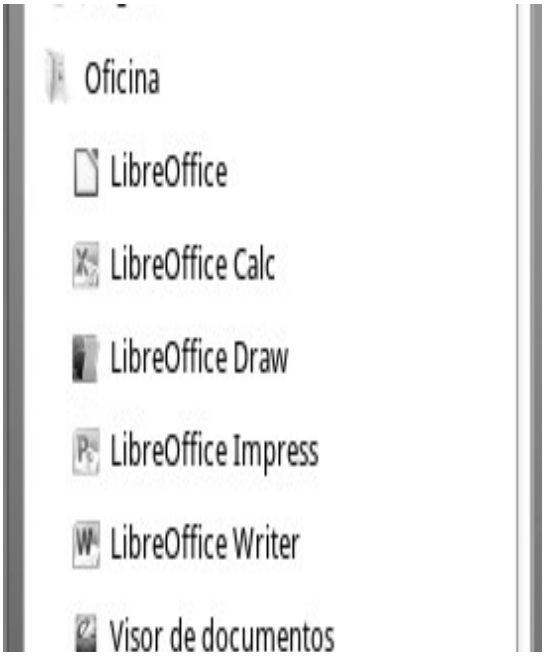


Figura 9: Menú de la categoría Oficina.

Tabla 10: Descripción del RF8 Proveer aplicaciones para la categoría Sonido y vídeo.

Número: RF8	**Nombre del requisito:** Proveer aplicaciones para la categoría Sonido y vídeo.
Programador: Yisel Obiol González y Dayrol Lázaro Ferrer Durruthy	**Iteración Asignada:** 1
Prioridad: Alta	**Tiempo Estimado:** 15 días
Riesgo en Desarrollo: Ausencia de desarrolladores por enfermedad. Pérdida de información imprescindible. Falta de personal calificado.	**Tiempo Real:** 3 semanas
Descripción: El sistema debe permitir ajustar el sonido y el vídeo según las prestaciones de la máquina y las preferencias del usuario. Se debe poder grabar información en dispositivos CD y DVD, reproducir música y vídeo en los formatos más utilizados actualmente así como poder grabar sonido y almacenarlos en la carpeta personal. Deben quedar integradas las funcionalidades relacionadas con la reproducción y gestión de música, vídeos y grabación de archivos de estos tipos.	
Prototipo de interfaz  *Figura 10: Menú de la categoría Sonido y vídeo.*	

Tabla 11: Descripción del RF9 Proveer aplicaciones para la categoría de Configuración del sistema.

Número: RF9	**Nombre del requisito:** Proveer aplicaciones para la categoría de Configuración del sistema.
Programador: Yisel Obiol González y Dayrol Lázaro Ferrer Durruthy	**Iteración Asignada:** 1
Prioridad: Alta	**Tiempo Estimado:** 15 días
Riesgo en Desarrollo: Ausencia de desarrolladores por enfermedad. Pérdida de información imprescindible. Falta de personal calificado.	**Tiempo Real:** 3 semanas
Descripción: El sistema debe tener aplicaciones que le permitan a los usuarios personalizar el sistema según sus preferencias. Configurar el brillo permitiendo aumentarlo o disminuirlo según la necesidad de los usuarios, brindar opciones de apagado para el sistema inactivo, configurar el idioma para el teclado, configurar el teclado y el ratón teniendo en cuenta la velocidad pudiendo aumentarla o disminuirla, permitir cambiar el tamaño del monitor del sistema y mostrar opciones de configuración para discapacitados visuales, auditivos o motoras.	
Prototipo de interfaz	

Figura 11: Menú de la categoría Configuración del sistema.

Tabla 12: Descripción del RF10 Instalar el sistema desde un dispositivo extraíble.

Número: RF10	**Nombre del requisito:** Instalar el sistema desde un dispositivo extraíble.	
Programador: Yisel Obiol González y Dayrol Lázaro Ferrer Durruthy		**Iteración Asignada:** 1
Prioridad: Alta		**Tiempo Estimado:** 15 días
Riesgo en Desarrollo: Ausencia de desarrolladores por enfermedad. Pérdida de información imprescindible. Falta de personal calificado.		**Tiempo Real:** 3 semanas
Descripción: Debe ser posible la instalación del sistema desde un dispositivo extraíble.		

2.4.2 Requisitos no Funcionales

Los requisitos no funcionales describen aspectos del sistema que son visibles por el usuario, pero que no incluyen una relación directa con el comportamiento funcional del sistema. Se refieren más bien a las características del mismo como la fiabilidad, tiempo de respuesta y la capacidad de almacenamiento. Tienen como función describir propiedades emergentes del sistema a implantar, aunque presentan gran importancia puesto que el no cumplimiento de un requisito no funcional puede provocar la inutilización del sistema.

A continuación se muestran los requisitos no funcionales definidos para la personalización de GNU/Linux Nova para discapacitados visuales y motores:

Requerimientos de confiabilidad:

RnF 1: Si se desconectan dispositivos externos al sistema, este brinda una notificación sobre el suceso ocurrido, tratando de guardar el estado final existente.

RnF 2: Si el funcionamiento del sistema se ve interrumpido por la ausencia del fluido eléctrico en el área donde están alojadas las PC, una vez que vuelva este, el sistema debe de reanudar y realizar todas las operaciones normalmente.

RnF 3: El sistema debe ser capaz de realizar las operaciones si el funcionamiento de las PC se ve afectado por la ausencia de conexión a través de la red.

RnF 4: Al desconectarse algún dispositivo como impresora o escáner mientras se realiza operaciones con ellos, el sistema debe mostrar un mensaje al usuario para indicar el incidente.

Requerimientos de usabilidad:

RnF 5: El sistema va dirigido a cualquier usuario, que comprenda la edad de 10 años en lo adelante, tenga sexo tanto masculino como femenino, cualquier nivel de escolaridad y cualquier experiencia en la utilización del sistema. Además está diseñado específicamente para el apoyo a personas discapacitadas visuales y motoras.

RnF 6: La finalidad de las aplicaciones y sus configuraciones es permitir a cualquier persona discapacitada visual y motora realizar las tareas necesarias en un sistema operativo, teniendo como meta fundamental satisfacer sus necesidades.

RnF 7: Se requiere hardware con más de 1GB de memoria RAM. Se debe disponer de 5GB para instalar el sistema, aunque la variante ideal es 10GB y CPU Pentium IV.

RnF8: Con el objetivo de hacer más fácil la interacción de los usuarios con el sistema y minimizar la resistencia al cambio a software libre, se definió que el entorno de trabajo a utilizar sea el de Windows-shell.

Requerimientos de eficiencia:

RnF 9: El sistema debe iniciar en menos de 30 segundos en una configuración de hardware con los requisitos mínimos. Se requiere hardware con más de 1GB de memoria RAM. Se debe disponer de 5GB para instalar el sistema.

Requerimientos para la documentación de usuarios en línea y ayuda del sistema:

RnF 10: El sistema debe mostrar la ayuda, para que los usuarios puedan aclarar sus dudas en cuanto surjan, potenciando así el aprovechamiento de cada una de las funcionalidades brindadas.

RnF 11: La aplicación debe estar provista de un manual de usuarios. Parte de la aplicación debe ser un manual de usuarios que explique detalladamente los procedimientos para ejecutar cada funcionalidad del sistema.

Requerimientos de interfaz:

RnF 12: El sistema utilizará las bibliotecas de interfaz gráfica de usuario Gtk+[30] en sus versiones 2.24 y superior a 3.4.

RnF 13: El sistema utilizará las interfaces provistas por el kernel Linux en su Versión 3.8.

RnF 14: El sistema utilizará Dbus[31] para la comunicación entre procesos, y ssh, sftp, smb, rdp, vnc para la comunicación con otros sistemas en la red.

Requerimientos de licencia:

RnF 15: Se implementará bajo la licencia GPL v3.

Requerimientos de estándares aplicables:

RnF 16: *Linux Standard Base*[32].

Requerimientos legales, de derecho de autor y otros:

RnF 17: Registrar el producto cuando culmine su desarrollo. La UCI debe tener los derechos patrimoniales del producto registrado.

Conclusiones del capítulo

La descripción de la propuesta de solución a través de un conjunto de artefactos permitió sentar las bases prácticas y teóricas para la implementación y entendimiento de la misma.

Los artefactos obtenidos durante la fase de diseño y modelación permitieron identificar los roles y tareas que deben cumplir los miembros del equipo de desarrollo durante el ciclo de vida del proyecto.

La definición y descripción de los requisitos funcionales y no funcionales trajeron como resultado las pautas a seguir para la implementación de los mismos durante la fase de desarrollo.

30 GTK+ o *The GIMP Toolkit* es un conjunto de bibliotecas multiplataforma para desarrollar interfaces gráficas de usuario (GUI), principalmente para los entornos gráficos GNOME, XFCE y ROX aunque también se puede usar en el escritorio de Windows, Mac OS y otros.

31 Un sistema de comunicación entre procesos (IPC), para aplicaciones de software con el fin de comunicarse entre sí.

32 La Base Estándar para Linux es un proyecto conjunto de varias Distribuciones de Linux bajo la estructura organizativa del F*ree Standards Group* con el objeto de crear y normalizar la estructura interna de los sistemas operativos derivados de Linux.

Capítulo 3. Proceso de construcción del sistema

Introducción

Con el objetivo de verificar y demostrar la confiabilidad de la propuesta de solución en el presente capítulo se dará paso al proceso de validación de la misma. En este capítulo se describe la puesta en práctica de la construcción de la propuesta de solución. Los elementos teóricos definidos en capítulos anteriores constituyen el punto de partida para construir una personalización de la distribución GNU/Linux Nova destinada a usuarios discapacitados visuales y motores. Como parte del proceso de validación de la propuesta en el presente capítulo se describen las pruebas realizadas al sistema para validar su funcionamiento.

3.1 Descripción del proceso de construcción

El proceso de construcción del sistema se dividió en tres fases principales:

- Primera fase: Construcción del Sistema Operativo Base.
- Segunda fase: Construcción del Sistema Operativo Final.
- Tercera fase: Empaquetamiento del sistema.

3.1.1 Primera fase: Construcción del Sistema Operativo Base

El primer paso en la construcción de cualquier personalización de Nova consiste en la instalación del Sistema Operativo Base (SOB).

La mayoría de los comandos utilizados en esta primera fase de construcción requieren privilegios administrativos, por lo que serán ejecutados con la orden **"sudo"** al inicio de cada invocación.[33]

El primer paso consiste en crear el directorio o partición donde se quiere alojar el SOB, en el caso de estudio se creó un directorio en /mnt con el nombre **nova** de la siguiente forma:

```
$mkdir /mnt/nova
```

33 Los comandos con privilegios administrativos también pueden ejecutarse iniciando sesión en la terminal como administrador del sistema o *"root"*.

El siguiente paso es instalar la herramienta **debootstrap** con el comando:

```
$sudo apt-get install debootstrap
```

Utilizando dicha herramienta se procede a la creación del SOB utilizando el comando siguiente:

```
$sudo debootstrap --components=principal,extendido 2013 /mnt/nova/ http://nova.f10.uci.cu/nova
```

Se utilizó la opción **--components** para indicar las ramas del repositorio de las que serán utilizados paquetes durante el proceso de creación del SOB, en el caso de estudio **"principal"** y **"extendido"**. El nombre de la distribución a instalar es **2013** y la dirección del repositorio de donde se descargarán los paquetes es **http://nova.f10.uci.cu/nova**. Este proceso puede tardar unos minutos, y una vez concluido en **/mnt/nova** se encontrará un SOB instalado.

3.1.2 Segunda fase: Construcción del Sistema Operativo Final (SOF)

En esta fase serán incorporadas al SOB las aplicaciones y configuraciones necesarias para dar paso al Sistema Operativo Final.

- **Configuraciones iniciales:**

Primeramente se debe montar **/dev** en la carpeta donde se encuentra el SOB, en este caso **/mnt/nova**. El directorio **/dev** contiene los archivos de dispositivos especiales para todos los dispositivos de hardware (38).

```
$sudo mount --bind /dev /mnt/nova/dev
```

Para poder conectarse y descargar de los repositorios se debe copiar el fichero **resolv.conf** del equipo anfitrión, este contiene las direcciones de los servidores de DNS del dominio:

```
$sudo cp /etc/resolv.conf /mnt/nova/etc/
```

Se inicializa al fichero **initctl** en una constante *true* para que una vez que las aplicaciones que se instalen en el sistema que está en proceso de creación, soliciten el estado de un servicio, puedan comprobar que este está activo:

```
$sudo mv /mnt/nova/sbin/initctl /mnt/nova/sbin/initctl.nim_blocked
$sudo  ln -s /mnt/nova/bin/true /sbin/initctl
```

Se montan los procesos. El directorio **/proc** contiene un sistema de archivos imaginario o virtual. Este no existe físicamente en disco, sino que el núcleo lo crea en memoria. Se utiliza para ofrecer información relacionada con el sistema (originalmente acerca de procesos, por eso su nombre).

```
$sudo mount -t proc none /mnt/nova/proc
```

Se monta **/sys**, el cual contiene parámetros de configuración del sistema que se está ejecutando:

```
$sudo mount -t sysfs none /mnt/nova/sys
```

Se monta **/dev/pts**, este sistema de ficheros vive exclusivamente en la memoria, las entradas en **/dev/pts** son seudo-terminales[34](pty).

```
$sudo mount -t devpts none /mnt/nova/dev/pts
```

Se exportan algunas variables del sistema:

```
$sudo chroot /mnt/nova export LC_ALL=C
```

Se actualizan los índices de la caché del sistema para luego poder instalar paquetes que se necesitan en el sistema operativo a construir:

```
$sudo chroot /mnt/nova apt-get update
```

Se instala el protocolo **dbus** para permitir la comunicación entre los procesos de las aplicaciones a instalar:

```
$sudo chroot /mnt/nova apt-get install --yes dbus
```

Luego se monta el entorno **chroot**, este permite interactuar con el sistema que se está creando como si se tratara de un sistema en marcha. Una vez dentro de este entorno, se tienen privilegios de root:

```
$sudo chroot /mnt/nova/
```

- **Instalación de paquetes:**

Una vez que se tienen hechas las configuraciones iniciales se procede a la instalación de los paquetes que tendrá el SOF, así como las configuraciones que se deseen realizar para adaptarlo a necesidades

34 Es un par de pseudo-dispositivos, uno de los cuales, el esclavo, emula una verdadera terminal de texto dispositivo, el otro de los cuales, el maestro, proporciona los medios por los que un proceso emulador de terminal controla el esclavo.

específicas. A continuación se muestra el listado de aplicaciones que serán instaladas y seleccionadas en el capítulo anterior, además de otras necesarias para la ejecución del sistema (*Ver Tabla 13*).

Tabla 13: Listado de aplicaciones a instalar.

Paquete	Descripción
Nova-desktop	Es un metapaquete que incluye a todos los paquetes que conforman al SO.
Casper	Ejecuta un sistema preinstalado desde un soporte de solo lectura.
linux-image-3.8.0-35-generic	Es la imagen del kernel de Linux para la versión 3.8.0 en 64 bits x86 SMP.
Serere3-gtk3	Es el instalador del sistema GNU/Linux Nova.
squashfs-tools	Herramienta para crear y añadir sistemas de archivos squashfs[35].
Speech-dispatcher	Interfaz común para sintetizadores de voz.
Eviacam	Permite realizar movimientos del cursor del ratón con la cabeza.

Para instalar estos paquetes se utiliza el comando siguiente:

```
#apt-get install <paquete>
```

Ejemplo:

```
#apt-get install speech-dispatcher
```

Una vez que se tengan instalados todos los paquetes deseados se procede a limpiar la caché:

```
#aptitude clean
```

- **Configuraciones del sistema:**

Para obtener el sistema deseado no solo se depende de los requerimientos establecidos o la instalación de aplicaciones, sino de su configuración, para que la interacción de los usuarios con las mismas sea lo más fácil posible. Basándose en lo anterior se procede de la siguiente manera para realizar la configuración de las aplicaciones instaladas:

Lector de pantalla Orca:

Se necesita que al iniciar el sistema, el lector de pantalla Orca cargue directamente y con una

35 Squashfs (.sfs) es un sistema de archivos comprimido de sólo lectura para Linux.

configuración que sea lo más fluida posible. Para esto se ejecuta en consola el comando spd-conf:

```
#spd-conf
```

Luego se establecen los parámetros de configuración quedando de la siguiente manera:

```
Módulo de salida predeterminado: espeak
Sistema de voz: Speech-Dispatcher
Persona: Spanish-latin-american(es)
Velocidad de la voz por defecto: 30
Tono de voz por defecto: 5.0
Volumen: 10
```

Para que el lector se ejecute de forma automática una vez que el usuario inicie la sesión es necesario copiar su ejecutable en la carpeta **autostart**:

```
#cp /usr/share/applications/orca.desktop /etc/skel/.config/autostart/
```

Todas las configuraciones realizadas al Orca son almacenadas en:

```
/etc/skel/.local/share/orca/
```

Eviacam:

En el Eviacam se utilizaron configuraciones que vienen predefinidas por defecto y se establecieron solo los siguientes valores:

Detección de rostro: Localización automática.

Modo clic: Pulsación al posarse.

Para que este programa se ejecute de forma automática una vez que el usuario inicie la sesión es necesario copiar su ejecutable en la carpeta **autostart:**

```
#cp /usr/share/applications/eviacam.desktop /etc/skel/.config/autostart
```

Todas las configuraciones realizadas al Eviacam se realizan en el fichero **.eviacam** y son almacenadas en:

```
/etc/skel/
```

Mousetweaks:

Se utiliza esta herramienta para definir los distintos tipos de pulsación del ratón, complementándose con el Eviacam permitiendo un completo funcionamiento del puntero en el sistema operativo. En el MouseTweaks se utilizaron las configuraciones que vienen predefinidas por defecto, las mismas son almacenadas en:

```
/etc/skel/.config/dconf/user
```

Para que este programa se ejecute de forma automática una vez que el usuario inicie la sesión es necesario copiar su ejecutable en la carpeta **autostart**:

```
#cp /usr/share/applications/mousetweaks.desktop /etc/skel/.config/autostart
```

Windows-shell:

Debido a que el sistema operativo Windows es uno de los sistemas operativos más extendidos en Cuba y en el mundo (39), se define que la vista a integrar en el sistema fuera la de Windows-shell. Esto se realiza con el objetivo de hacer más fácil la interacción de los usuarios con el sistema y evitar que el cambio a software libre no se produzca de forma brusca. Las configuraciones del entorno de escritorio Windows-shell que se proponen para el sistema son las siguientes:

```
icons: true
gtk: true
metacity: true
background: true
overlay: true
cursor: true
```

Las configuraciones son almacenadas en:

```
/etc/skel/.local/share/windows-shell/first-run
```

Otras de las configuraciones necesarias es lograr que aparezca el icono de accesibilidad en el área de notificación de la barra de menú, ya que la vista de Windows-shell por defecto no la trae. Esto se logra añadiendo funcionalidades al fichero **/usr/share/windows-shell/js/ui/panel.js**:

```
const STANDARD_STATUS_AREA_ORDER = ['ally', 'keyboard', 'volume', 'bluetooth', 'network', 'battery',
'userMenu'];
const STANDARD_STATUS_AREA_SHELL_IMPLEMENTATION = {
'ally': imports.ui.status.accessibility.ATIndicator,
'volume': imports.ui.status.volume.Indicator,
'battery': imports.ui.status.power.Indicator,
'keyboard': imports.ui.status.keyboard.XKBIndicator,
'userMenu': imports.ui.userMenu.UserMenuButton
};
```

Es necesario eliminar los elementos del menú que se encuentren vacíos para evitar confundir al usuario. Para esto se realizan modificaciones al fichero **/usr/share/windows-shell/js/ui/userMenu.js**:

```
loadCategory: function() {
    this.loadCategoryAux(this.dir);
    if (this.apps.length == 0) {
        this.actor.visible = false;
    }
},
```

Con esto termina el proceso de configuración, antes de pasar a la tercera fase se debe realizar la restauración del sistema.

- **Restauración de las configuraciones del sistema:**

Primeramente se debe salir del entorno virtual **chroot** con el comando *exit*:

```
#exit
```

Para que el SOF creado funcione correctamente, se deben restaurar los valores adecuados así como

borrar los archivos temporales para disminuir el tamaño del sistema:

```
$sudo rm -f /mnt/nova/etc/resolv.conf
$sudo rm -rf /mnt/nova/tmp/*
$sudo rm -f /mnt/nova/sbin/initctl
$sudo mv /mnt/nova/initctl.nim_blocked /mnt/nova/sbin/initctl/sbin/
```

Luego se desmontan las unidades montadas inicialmente:

```
$sudo umount /mnt/nova/proc
$sudo umount /mnt/nova/sys
$sudo umount /mnt/nova/dev/pts
$sudo umount /mnt/nova/dev/
```

3.1.3 Tercera fase: Empaquetamiento del sistema

Primeramente se crean las carpetas donde se instalarán los archivos necesarios para la creación de la imagen ISO-9660, en este caso se creó en **"*/mnt*"** la carpeta **"*iso*"** la cual contendrá las subcarpetas **"*casper*"**, **"*isolinux*"**, e **"*install*"**.

```
$sudo mkdir -p /mnt/iso/{casper,isolinux,install}
```

Se mueven los archivos del núcleo para la carpeta **"*casper*"**:

```
$sudo mv /mnt/nova/boot/vmlinuz-3.8.0-35-generic /mnt/iso/casper/vmlinuz
$sudo mv /mnt/nova/boot/initrd.img-3.8.0-35-generic /mnt/iso/casper/initrd.lz
```

Se copia el contenido de la carpeta **"*isolinux*"** presente en cualquier ISO de Nova, que ya está previamente configurada con un *splash* y un menú en concordancia con el patrón de diseño de Nova, para la carpeta creada con el mismo nombre.

Se copia la lista de los paquetes instalados en el sistema dentro de la carpeta ***"casper"*** en el fichero ***"filesystem.manifest"***:

```
$sudo chroot /mnt/nova/ dpkg-query -W --showformat ='${Package} ${Version}\n' | tee
```

```
/mnt/iso/casper/filesystem.manifest
```

De este grupo de paquetes instalados para el sistema *Live* no todos son necesarios una vez instalado el mismo en una terminal cliente, por lo que se deben excluir estos paquetes en aras de tener un sistema limpio de elementos innecesarios.

Los paquetes a excluir son:

- El instalador (serere).

- Todas las herramientas que proveen el funcionamiento de un sistema *Live* (casper, live-initramfs, user-setup).

```
REMOVE='serere3-slideshow serere3-gtk3 casper user-setup live-initramfs'
for i in $REMOVE;
do
sudo sed -i "/${i}/d" image/casper/filesystem.manifest-desktop;
done
```

Luego se necesita el ***"filesystem.squashfs"***, que no es otra cosa que el sistema comprimido en el formato *squashfs:*

```
$sudo mksquashfs /mnt/nova/ /mnt/iso/casper/filesystem.squashfs -comp xz
$sudo printf $(du -sx --block-size=1 /mnt/nova | cut -f1) >/mnt/iso/casper/filesystem.size
```

Se escriben las definiciones del sistema *Live* en el archivo ***"README.diskdefines"***, donde se especifican la arquitectura de computadora utilizada, la numeración del disco y una descripción del sistema:

```
$sudo touch /mnt/iso/README.diskdefines
$sudo nano /mnt/iso/ README.diskdefines
```

Quedando de la manera siguiente:

```
#define DISKNAME Nova Bonita i386
#define TYPE binary
```

```
#define TYPE binary 1
#define ARCH i386
#define ARCH i386 1
#define DISKNUM 1
#define DISKNUM1 1
#define TOTALNUM 0
#define TOTALNUM0 1
```

Se crea la carpeta ***".dist"*** y se escriben en ella otros datos necesarios.

```
$sudo mkdir /mnt/iso/.disk
$sudo touch /mnt/iso/.disk/base_installable
$sudo echo "full_cd/single" > /mnt/iso/.disk/cd_type
$sudo echo "Nova Bonita" > /mnt/iso/.disk/info
$sudo echo "http://www.nova.cu" > /mnt/iso/.disk/release_notes_url
```

Se escribe la suma de control MD5:

```
$cd /mnt/iso
$find . -type f -print0 | xargs -0 md5sum | grep -v "\./md5sum.txt" > md5sum.txt
```

Por último se crea la imagen de instalación ISO-9660 como se muestra a continuación:

```
$cd /mnt/iso
$sudo mkisofs -r -V "Nova Bonita" -cache-inodes -J -l -b isolinux/isolinux.bin -c isolinux/boot.cat -no-emul-boot -boot-load-size 4 -boot-info-table -o /mnt/nova-bonita-$(date+"%Y%m%d").iso
```

Como resultado se obtiene en la carpeta ***'/mnt'*** la imagen **iso** del sistema.

3.2 Pruebas al sistema

Durante y después del proceso de implementación, el sistema que se está desarrollando debe ser comprobado para asegurar que satisface los requerimientos del cliente, por tanto se hace necesario

establecer un control estricto de los elementos del sistema a desarrollar en la presente investigación. Por esta razón se realizó un análisis de los procesos existentes con el objetivo de comprobar que se encuentran todos los necesarios para su correcto funcionamiento.

Para la realización del análisis de procesos se listaron los mismos y a partir de su descripción, se determinó la función que realizan dentro del sistema así como el momento en que son ejecutados, ya sea al iniciar el sistema, durante su funcionamiento o en el apagado. En el Anexo I se muestran los procesos definidos como necesarios para el correcto funcionamiento del sistema.

Una vez que se tiene concebido el sistema, es necesario saber si este se encuentra apto para ser liberado, por lo que hay que realizar pruebas, y de este modo verificar que la personalización creada funciona acorde a lo especificado y poder medir hasta qué punto se cumplieron los objetivos trazados inicialmente. Para ello, en el siguiente epígrafe se muestran los casos de prueba y sus resultados.

3.3 Verificación Funcional

La calidad del software debe implementarse a lo largo de todo el ciclo de vida de un software, debe correr paralelamente desde la planificación del producto hasta la fase de producción de este como actividad de protección. El aspecto a considerar en el Control de la Calidad del Software es la Prueba de Software (40).

3.3.1 Pruebas de software

Las pruebas de software es un concepto que a menudo es conocido como verificación y validación del software. Integra las técnicas de diseño de casos de prueba en una serie de pasos bien planificados que dan como resultado una correcta construcción del software.

Para la ejecución de las pruebas a Nova Bonita se definió el método de caja negra, definiendo como tipo de prueba la prueba de funcionalidad, realizada para detectar no conformidades; las pruebas son aplicadas en diferentes tipos de niveles, de los cuales se utiliza el nivel de prueba de aceptación, que permitieron la ejecución de casos de pruebas.

- **Pruebas de caja negra**

Las pruebas de caja negra se llevan a cabo sobre la interfaz del software, obviando el comportamiento

interno y la estructura del sistema. Los métodos de caja negra se centran en encontrar tipos de errores como funciones incorrectas o ausentes, errores en la interfaz, errores en estructuras de datos o en accesos a bases de datos externas, errores de rendimiento y errores de inicialización y terminación (41).

- **Prueba de aceptación**

Las pruebas de aceptación son pruebas a gran escala, que pueden llegar a dimensiones industriales cuando el número de módulos es muy elevado, o la funcionalidad que se espera del programa es muy compleja.

En las pruebas de aceptación el objetivo es la evaluación del producto y la realización de una revisión de la documentación final. Estas pruebas las realiza el cliente y son básicamente pruebas funcionales, sobre el sistema completo y buscan una cobertura de la especificación de requisitos. Estas pruebas no se realizan durante el desarrollo, pues sería impresentable de cara al cliente (41).

3.3.2 Casos de Prueba

Un caso de prueba es una serie de pruebas de entrada, condiciones de ejecución y resultados esperados desarrollados para un objetivo definido, tal como ejecutar una ruta particular de un programa o verificar el cumplimiento con un requerimiento en específico (41).

A continuación se muestran los casos de pruebas planificados para el desarrollo de la validación del sistema a desarrollar. Se mostrarán algunos de estos casos de pruebas, el resto se encuentra en el Anexo II.

Caso de Prueba 2

CPR2: Proveer aplicaciones para la categoría Acceso Universal.

Condiciones de ejecución: Acceder al menú de Acceso Universal.

Tabla 14: CPR2 Proveer aplicaciones para la categoría Acceso Universal.

Escenario	Descripción	V1	Respuesta del sistema	Flujo central
EC 2.1 Mostrar menú.	Luego de acceder al menú de aplicaciones para el apoyo a las personas discapacitadas aparecen las diferentes alternativas a elegir.	V	Muestra un menú con las aplicaciones para el apoyo a personas discapacitadas.	Presionar el menú de aplicaciones.
		Presiona menú de aplicaciones.		
		I	No muestra ningún menú en pantalla.	
		No presiona menú de aplicaciones.		

Caso de prueba 10

CPR10: Instalar el sistema desde un dispositivo extraíble.

Condiciones de ejecución: Se debe contar con el sistema en un dispositivo booteable.

Tabla 15: CPR10 Instalar el sistema desde un dispositivo extraíble.

Escenario	Descripción	V1	Respuesta del sistema	Flujo central
EC 1.1 Seleccionar partición.	Seleccionar partición a instalar.	V	Instala en la partición seleccionada.	Instalación del sistema operativo en la partición seleccionada.
		Seleccionar partición a instalar.		
		I	Instala en todo el disco duro.	
		No selecciona partición a instalar.		
EC 1.2 Insertar contraseña.	Pide una nueva contraseña para el usuario Administrador y valida que tenga la fuerza requerida.	V	Pedir repetir contraseña.	Insertar la contraseña con la fuerza requerida.
		Fuerte		
		I	Mostrar mensaje de error y volver a pedir contraseña nuevamente.	
		Débil		
EC 1.3 Repetir contraseña.	Pide repetir la contraseña y valida que coincida con la anteriormente introducida.	V	Pedir información al usuario para instalar.	Insertar contraseña nuevamente para ver si coincide con la anteriormente insertada.
		Coincide		
		I	Mostrar mensaje de error y volver a pedir la contraseña.	
		No coincide		
EC 1.4 Pedir confirmación.	Pide confirmación para proceder con la instalación.	V	Instalar el sistema operativo.	Pulsar botón continuar para seguir con la instalación.
		Pulsar botón continuar.		
		I	Salir del programa de instalación.	
		No pulsar botón continuar.		

3.3.3 Resultados esperados

Luego de diseñadas las pruebas del producto se procedió a su ejecución. Los resultados fueron obtenidos luego de tres iteraciones de ejecución, guiadas por las actividades del flujo de Prueba de la metodología Nova-OpenUp.

La primera iteración arrojó como resultado un total de 20 no conformidades en las pruebas realizadas. Luego de trabajar durante un período en su solución y efectuar la segunda iteración se redujo a 9 errores en el sistema, los cuales fueron eliminados antes de concluir la aplicación.

Tal como se muestra en la siguiente figura, la resolución de los errores detectados en esta etapa permitió el aseguramiento de una mayor calidad en el producto (*Ver Figura 12*).

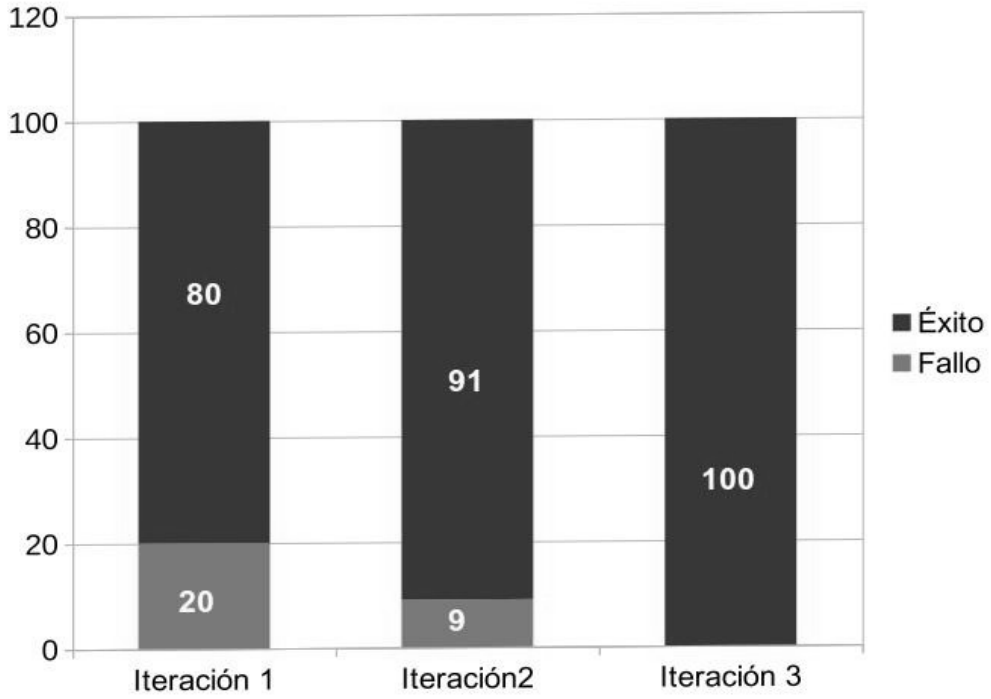


Figura 12: Resultados de las pruebas funcionales

Después de ejecutadas las pruebas y para evaluar el aumento de usabilidad de la propuesta de solución se muestra una comparación entre un antes y un después de la distribución de GNU/Linux Nova y la personalización Nova Bonita respectivamente. Esta comparación se realiza en función de las aplicaciones

integradas y las funcionalidades adquiridas: sistema ampliador de pantalla, teclado en pantalla, lector de pantalla configurado para que su inicio sea automático y el sintetizador de voz sea lo más fluido posible, uso de la *webcam* para el control del mouse, selección del tipo de clic y mejoras en la vista Windows-shell (*Ver Tabla 16*).

Tabla 16: Comparación entre la distribución de GNU/Linux Nova y Nova Bonita

Funcionalidades	Nova 4.0	Nova Bonita
Sistema ampliador de pantalla.	X	X
Teclado en pantalla.	X	X
Lector de pantalla configurado.		X
Uso de otros periféricos para el control del mouse.		X
Alternativas de control del puntero.		X
Mejoras estéticas.		X

Conclusiones del capítulo

La puesta en práctica de las tres fases definidas en el presente capítulo para la construcción de una personalización de Nova, dieron como resultado la imagen de instalación ISO-9660 de la personalización Nova Bonita, siendo la misma el producto que se propone como solución de la presente investigación.

Los diseños de casos de prueba creados a partir de los requisitos definidos del sistema permitieron validar la propuesta de solución luego de tres iteraciones de pruebas que dieron como resultado un sistema libre de no conformidades.

Las pruebas de aceptación realizadas validaron la propuesta de solución por parte del cliente y su satisfacción con el producto final.

Conclusiones generales

Como resultado de la investigación se obtuvo Nova Bonita, una personalización de la distribución cubana de GNU/Linux Nova enfocada al apoyo de usuarios con discapacidades visuales y motoras. Entre los principales resultados obtenidos durante el desarrollo de la investigación y obtención del producto final se pueden destacar:

- El estudio realizado de las distribuciones basadas en GNU/Linux utilizadas en el apoyo a usuarios discapacitados visuales y motores permitió identificar las vulnerabilidades existentes, demostrando la necesidad de crear una nueva personalización que integrara los elementos más significativos de algunas de ellas e incluyera las configuraciones necesarias para dar soporte a la mayor cantidad de usuarios.
- El análisis y diseño de la propuesta de solución permitió el desarrollo de la personalización Nova Bonita dirigida al apoyo a los discapacitados visuales y motores, garantizando la inserción de los mismos en el proceso de migración del país.
- Las pruebas de software realizadas permitieron evaluar el funcionamiento de la personalización, la conformidad con los requisitos definidos y la eliminación de las no conformidades encontradas en las iteraciones, validando de esta forma el correcto funcionamiento de la propuesta de solución.

Recomendaciones

Partiendo de que con el desarrollo del presente trabajo se dio cumplimiento a su objetivo general se recomienda lo siguiente:

- Utilizar el presente trabajo como una guía para realizar construcciones de distribuciones similares.
- Incluir el proceso de construcción de una personalización de la distribución cubana de GNU/Linux Nova dentro de la metodología Nova-OpenUp.
- Considerar las nuevas actualizaciones de las tecnologías para el apoyo a personas discapacitadas y tener en cuenta para futuros cambios de la propuesta de solución.

Bibliografía

1. Grupo Galgano. *Aplicaciones Informáticas para discapacitados.* [online]. 2010. Available from: http://www.ssreyes.org/acces/recursos/doc/Sansenet/288802359_642009173818.pdf

2. KOON RICARDO A. and DE LA VEGA MARIA EUGENIA. *El impacto tecnológico en las personas con discapacidad.* [online]. Available from: http://diversidad.murciaeduca.es/tecnoneet/docs/2000/14-2000.pdf

3. PIERRA FUENTES ALAN. *Nova, Distribución Cubana de GNU/Linux. Reestructuración estratégica de su proceso de desarrollo.* [online]. La Habana, 2011. Available from: http://catalogoenlinea.uci.cu/cgi-bin/koha/opac-detail.pl?biblionumber=11161

4. FERNÁNDEZ DEL MONTE YUSLEYDI, PIERRA FUENTES ALAN and FÍRVIDA DONÉSTEVEZ ABEL ALFONSO. *Nova 3.0, avances y expectativas de la distribución cubana de GNU/Linux*. 12 July 2011.

5. MIRANDA DOMÍNGUEZ LAURA ELIDA and BARROSO LEYÓN LEONEL. *Personalización de la distribución GNU/Linux Nova para el control de acceso a los comedores.* Universidad de las Ciencias Informáticas, 2014.

6. MOSQUEDA, Ariannis Lafita and GUZMÁN, Ángel Camilo Guillén. *NOVAMEDIA: Personalización de Nova orientada al diseño gráfico y la animación para FreeViUX.* [online]. La Habana : Universidad de las Ciencias Informáticas, 2012. [Accessed 8 May 2015]. Available from: http://catalogoenlinea.uci.cu/cgi-bin/koha/opac-detail.pl?biblionumber=11684

7. Repositorios digitales. [online]. [Accessed 8 March 2015]. Available from: http://bibliotecabiologia.usal.es/tutoriales/catalogos-repositorios-bibliosvirtuales/repositorios_digitales.html

8. Definición de deficiencia - Qué es, Significado y Concepto. [online]. 2008 2015. [Accessed 8 March 2015]. Available from: http://definicion.de/deficiencia/

9. Revista Cubana de Salud Pública - Caracterización epidemiológica de las personas con discapacidad en Cuba. [online]. December 2010. [Accessed 8 March 2015]. Available from: http://scielo.sld.cu/scielo.php?script=sci_arttext&pid=S0864-34662010000400004

10. Dirección General de Educación Especial. [online]. [Accessed 8 March 2015]. Available from: http://eespecial.sev.gob.mx/difusion/visual.php

11. Dirección General de Educación Especial. [online]. [Accessed 8 March 2015]. Available from:

http://eespecial.sev.gob.mx/difusion/motriz.php

12. Entrevista a desarrolladores de distribución Lazarux orientada a discapacitados - Software Libre. [online]. 2011. [Accessed 8 March 2015]. Available from: http://www.somoslibres.org/modules.php?name=News&file=article&sid=755

13. Lazarux: la versión de Linux para discapacitados visuales. [online]. [Accessed 8 March 2015]. Available from: http://www.soft-tzalan.org/joomla/index.php?option=com_content&view=article&id=50:lazarux-la-version-de-linux-para-discapacitados-visuales&catid=4:general&Itemid=4

14. Lazarux - Wikipedia, la enciclopedia libre. [online]. 1 April 2013. [Accessed 8 March 2015]. Available from: http://es.wikipedia.org/wiki/Lazarux

15. Oralux, distribución GNU/Linux para ciegos y deficientes visuales. | tiflolinux.org. [online]. [Accessed 8 March 2015]. Available from: http://www.tiflolinux.org/node/45

16. Tiflobuntu: GNU/Linux más fácil a personas ciegas y deficientes visuales. | tiflolinux.org. [online]. [Accessed 8 March 2015]. Available from: http://www.tiflolinux.org/node/89

17. EVuntu - EcuRed. [online]. [Accessed 8 March 2015]. Available from: http://www.ecured.cu/index.php/EVuntu

18. El blog de Jabba: Vinux: distribución Linux para discapacitados visuales. [online]. 2007 2015. [Accessed 8 March 2015]. Available from: http://www.elblogdejabba.com/2010/06/vinux-distribucion-linux-para.html

19. Vinux 4.0. Distribución GNU/Linux para ciegos y deficientes visuales, ya disponible | tiflolinux.org. [online]. [Accessed 8 March 2015]. Available from: http://www.tiflolinux.org/node/464

20. CASTILLO HERNÁNDEZ SUSANA. *Propuesta de accesibilidad en el catálogo en línea para discapacitados visuales* [online]. La Habana : Universidad de las Ciencias Informáticas, 2012. Available from: http://catalogoenlinea.uci.cu/cgi-bin/koha/opac-detail.pl?biblionumber=11765

21. Los lectores de pantalla en el escritorio GNOME de GNULinux: Orca, Gnopernicus y LSR. [online]. 2010. [Accessed 9 March 2015]. Available from: http://www.nodo50.org/utlai/joomla/index.php/softlibre/56-apuntes-sobre-linux-2-xwindow/143-los-lectores-de-pantalla-en-el-escritorio-gnome-de-gnulinux-orca-gnopernicus-y-lsr.html

22. Guía de accesibilidad. [online]. 2011. [Accessed 9 March 2015]. Available from: http://docs.fedoraproject.org/es-ES/Fedora/13/html-single/Accessibility_Guide/index.html#Emacspeak

23. SÁNCHEZ CABALLERO MATÍAS. *Software libre y accesibilidad a las TIC.* [online]. December 2010. Available from: www.coitt.es/res/revistas/07c_Filosofia_del_Software.pdf

24. BRLTTY - Official Home. [online]. 6 November 2014. [Accessed 9 March 2015]. Available from: http://mielke.cc/brltty/index.html

25. Projects/GnomeShell - GNOME Wiki! [online]. 2005 2015. [Accessed 11 March 2015]. Available from: https://wiki.gnome.org/Projects/GnomeShell

26. Universal access. [online]. 2005 2014. [Accessed 11 March 2015]. Available from: https://help.gnome.org/users/gnome-help/stable/a11y.html

27. 5.2. GNOME Magnifier. [online]. 2011. [Accessed 9 March 2015]. Available from: http://docs.fedoraproject.org/es-ES/Fedora/14/html/Accessibility_Guide/ar01s05s02.html

28. Gnome-mag - Por un Mundo Accesible. [online]. [Accessed 9 March 2015]. Available from: http://wiki.mundoaccesible.org.ve/index.php?title=Gnome-mag

29. KMouseTool - EcuRed. [online]. [Accessed 9 March 2015]. Available from: http://www.ecured.cu/index.php/KMouseTool

30. Configuración del ratón - Guía de accesibilidad del escritorio GNOME de Oracle Solaris 11. [online]. December 2011. [Accessed 9 March 2015]. Available from: http://docs.oracle.com/cd/E26921_01/html/E26300/dtconfig-1.html

31. MouseTweaks - Por un Mundo Accesible. [online]. [Accessed 9 March 2015]. Available from: http://wiki.mundoaccesible.org.ve/index.php?title=MouseTweaks

32. EViacam - Sitio oficial de Lihuen. [online]. [Accessed 9 March 2015]. Available from: http://lihuen.linti.unlp.edu.ar/index.php?title=EViacam

33. eViacam - Descargar. [online]. 2015. [Accessed 9 March 2015]. Available from: http://eviacam.softonic.com/

34. GOK: GNOME Onscreen Keyboard | CODIGO LINUX. [online]. [Accessed 9 March 2015]. Available from: http://www.codigo-linux.com/site/content/gok-gnome-onscreen-keyboard

35. 7.3. Florence. [online]. 2011. [Accessed 9 March 2015]. Available from: http://docs.fedoraproject.org/es-ES/Fedora/16/html/Accessibility_Guide/ar01s07s03.html

36. Nano: sencillo editor de textos para el terminal | Slice of Linux. [online]. 2010.

[Accessed 23 February 2015]. Available from: https://sliceoflinux.wordpress.com/2010/02/26/nano-sencillo-editor-de-textos-para-el-terminal/

37. SOMMERVILLE IAN. *Ingeniería del software* [online]. 7ma Edición. 2005. Available from: http://eva.uci.cu/mod/resource/view.php?id=9352&subdir=/Sommerville

38. El directorio /dev. [online]. 2003. [Accessed 26 March 2015]. Available from: http://www.tldp.org/pub/Linux/docs/ldp-archived/system-admin-guide/translations/es/html/ch04s04.html

39. Refuerza Microsoft medidas contra uso de versiones ilegales de sus sistemas operativos - Cuba - Juventud Rebelde - Diario de la juventud cubana. [online]. 22 February 2007. [Accessed 28 April 2015]. Available from: http://www.juventudrebelde.cu/cuba/2007-02-22/refuerza-microsoft-medidas-contra-uso-de-versiones-ilegales-de-sus-sistemas-operativos-/

40. BARANDA, Hector Pérez and MEJÍAS, Ricardo Quevedo. *Sistema de compilación distribuida de Nova.* [online]. La Habana : Universidad de las Ciencias Informáticas, 2012. Available from: http://catalogoenlinea.uci.cu/cgi-bin/koha/opac-detail.pl?biblionumber=11743

41. NATALIA JURISTO, ANA M. MORENO and SIRA VEGAS. *Técnicas de Evaluación de Software* [online]. 17 October 2005. Available from: http://www.google.com.cu/url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&sqi=2&ved=0CBwQFjAA&url=http%3A%2F%2Fwww.grise.upm.es%2Fsites%2Fextras%2F12%2Fpdf%2FDocumentacion_Evaluacion_7.pdf&ei=4gFdVdyzDcjZsATY2oHQBw&usg=AFQjCNFQq1YCIEX4LCA1_ZdckCMaaII1Wg&bvm=bv.93756505,d.cWc