Universidad de las Ciencias Informáticas

Facultad 1

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

Título:

"Personalización de la distribución GNU/Linux Nova para el control de acceso a los comedores."

Autores:

Laura Elida Miranda Domínguez.

Leonel Barroso Bayón.

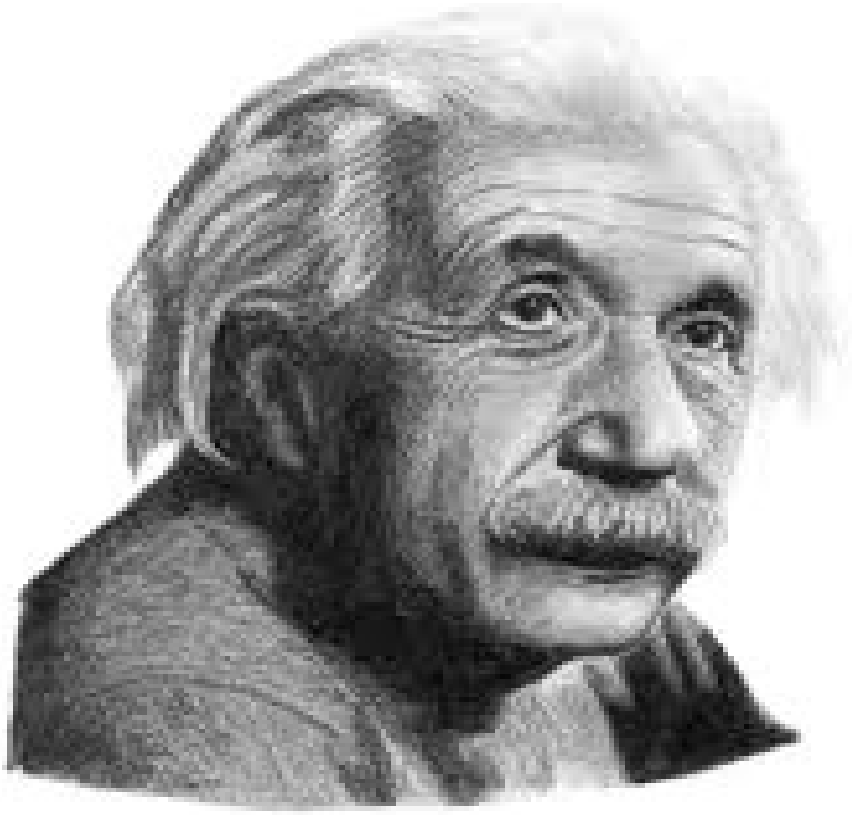
Tutores:

Ing. Edilberto Blez Doroncelé.

Ing. Juan Manuel Fuentes Rodríguez.

Junio del 2014

Pensamiento:



"Por más difícil que se nos presente una situación, nunca dejemos de buscar la salida, ni de luchar hasta el último momento. En momentos de crisis, sólo la imaginación es más importante que el conocimiento".

Albert Einstein.

Declaración de Autoría:

Declaramos ser los únicos autores de este trabajo y concedemos 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 2014.

_____	_____
Laura Elida Miranda Domínguez	Leonel Barroso Bayón
_____	_____
Ing. Edilberto Blez Doroncelé	Ing. Juan Manuel Fuentes Rodríguez

Agradecimientos:

- A nuestros ***padres*** por el apoyo y la confianza depositada en nosotros.
- A nuestros ***tutores*** por el apoyo brindado durante todo el proceso de investigación y desarrollo de la tesis.
- A nuestros ***hermanos y demás familiares*** por estar siempre ahí cuando los necesitamos.
- A nuestros ***amigos*** por hacernos pasar momentos tan agradables.
- A nuestros ***profesores*** por ayudar en nuestra formación como ingenieros informáticos.

Dedicatoria:

A nuestros padres, nosotros sólo somos un reflejo de la educación y el amor que nos han brindado a lo largo de nuestras vidas.

Resumen:

Con el presente trabajo de diploma titulado “Personalización de la distribución GNU/Linux Nova para el control de acceso a los comedores” se pretende contribuir al desarrollo de un quiosco para el sistema de gestión de acceso de personal a los comedores de la universidad, a través de la descripción del proceso de construcción de un quiosco de Nova. En el documento, se recogen los resultados del estudio del estado del arte, de las herramientas y técnicas empleadas, así como los diferentes elementos de la metodología de desarrollo utilizada. El resultado esperado será una personalización en modo quiosco de la distribución GNU/Linux Nova para el control de acceso a los comedores, además de la definición de un procedimiento que guíe a los miembros del proyecto Nova en la construcción de sistemas similares.

Palabras claves: Sistema operativo, personalización, quiosco.

Índice de contenido

Introducción....1

Capítulo 1: Fundamentación teórica....6

1.1 Introducción....6

1.2 Conceptos fundamentales....6

1.3 Estudio sobre quioscos informáticos....7

1.3.1 Software de quiosco....7

1.3.2 Navegador quiosco....9

1.3.3 Quioscos interactivos....10

1.3.4 Conclusiones acerca del estudio de los quioscos informáticos....11

1.4 Herramientas y técnicas de desarrollo....12

1.4.1 Herramienta para la creación del Sistema Operativo Base....12

1.4.2 IDE de desarrollo a utilizar....13

1.4.3 Lenguajes de desarrollo....14

1.4.4 Herramienta de modelado....15

1.5 Metodología de desarrollo de software....15

1.5.1 Metodología de desarrollo ágil....16

1.5.2 Metodología de desarrollo Nova OpenUp....16

1.6 Conclusiones del capítulo....17

Capítulo 2: Análisis y diseño de la solución propuesta....18

2.1 Introducción....18

2.2 Propuesta del sistema a desarrollar....18

2.3 Gestión del proceso de construcción del quiosco....19

2.4 Modelo del proceso de construcción del quiosco para el sistema de control de acceso....23

2.5 Requisitos de software....24

2.5.1 Requisitos Funcionales....24

2.5.2 Descripción de los requisitos ágiles....25

2.5.2 Requisitos no Funcionales....35

2.7 Conclusiones del capítulo....36
Capítulo 3: Implementación y Pruebas....37
3.1 Descripción del proceso de construcción....37
3.1.1 Fase 1: Construcción del Sistema Operativo Base....37
3.1.2 Fase 2: Construcción del Sistema Operativo Final....38
3.1.3 Fase 3: Empaquetamiento del sistema....45
3.2 Verificación Funcional....48
3.2.1 Pruebas de software....48
3.2.2 Casos de pruebas....49
3.2.3 Resultados obtenidos....52
3.3 Conclusiones del Capítulo....53
Conclusiones Generales....54
Recomendaciones....55
Referencias Bibliográficas....56
Bibliografía Consultada....58
Anexo I: Procesos que intervienen en el sistema....59
Anexo II: Casos de pruebas....63
CPR2: Iniciar el Sistema de Control de Acceso a los comedores (SCAC)....63
CPR3: Acceder al menú del sistema....63
CPR4: Configuración de distribución del teclado....64
CPR5: Apagar PC....64
CPR6: Cambiar contraseña del administrador....65
CPR7: Configurar la red....65
CPR10: Cancelar Menú del sistema....66
CPR11: Reiniciar PC....66
Anexo III: Acta de Aceptación del Producto....67
Acta de aceptación del cliente....67
Acta de aceptación de alta gerencia....68

Índice de Figuras

Figura 1: Fases de la metodología Nova OpenUp.....16

Figura 2: Relación entre componentes del sistema.....18

Figura 3: Diagrama de Gantt para las tareas de desarrollo del sistema.....19

Figura 4: Diagrama de proceso de construcción del quiosco para el control de acceso a los comedores 23

Figura 5: Menú del Instalador del sistema.....25

Figura 6: Prototipo de sistema de control de acceso a los comedores.....26

Figura 7: Menú del sistema.....27

Figura 8: Menú configurar teclado.....28

Figura 9: Opciones de cambio de idioma del sistema.....28

Figura 10: Menú apagar sistema.....29

Figura 11: Menú cambiar contraseña.....30

Figura 12: Cuadro de diálogo para verificar al administrador.....30

Figura 13: Consola para cambio de contraseña.....30

Figura 14: Menú configurar Red.....31

Figura 15: Interfaz para configurar la red.....31

Figura 16: Menú configurar USB.....32

Figura 17: Interfaz de configuración de medio USB.....32

Figura 18: Tty 2 para usuario administrador.....33

Figura 19: Menú cancelar.....34

Figura 20: Menú reiniciar sistema.....34

Figura 21: Porcentaje de aceptación y fallas de pruebas al sistema.....53

Índice de tablas

Tabla 1: Tareas para el proceso personalización de Nova en modo quiosco en función de riesgos. 23

Tabla 2: Descripción del RF1 Instalar Sistema Operativo. 25

Tabla 3: Descripción del RF2 Iniciar Sistema de control de acceso. 26

Tabla 4: Descripción del RF3 Acceder al menú del sistema. 27

Tabla 5: Descripción del RF4 Cambiar idioma. 28

Tabla 6: Descripción del RF5 Apagar PC. 29

Tabla 7: Descripción del RF6 Cambiar contraseña de administrador. 30

Tabla 8: Descripción del RF7 Configurar Red. 31

Tabla 9: Descripción del RF8 Configurar USB. 32

Tabla 10: Descripción del RF9 Iniciar consola de administrador. 33

Tabla 11: Descripción del RF10 Cancelar menú del sistema. 34

Tabla 12: Descripción del RF11 Reiniciar PC. 34

Tabla 13: Listado de aplicaciones a instalar 39

Tabla 14: CPR1 Instalar Sistema Operativo 50

Tabla 15: CPR8 Gestionar Dispositivo USB 51

Tabla 16: CPR9 Iniciar terminal para la administración 52

Tabla 17: Procesos que intervienen en el sistema, en ejecución. 59

Tabla 18: Procesos que intervienen en el sistema, detenidos. 62

Introducción

Constituye una injusticia pagar por obtener una aplicación, en la cual su desarrollador impone las condiciones de uso prohibiendo, por ejemplo, la copia a otras personas de la misma. Por otra parte no se puede adaptar, ni corregir errores en dicho programa, debido a la falta de acceso a su código fuente, teniendo que esperar a que salgan nuevas versiones mejoradas del mismo para pagarlas nuevamente. Estas dificultades las tiene el uso de *software* privativo y su solución está, en migrar hacia plataformas libres (1).

Estos últimos, entre los que se encuentran los sistemas GNU/Linux ofrecen a los usuarios la libertad de copiar, distribuir, cambiar y mejorar el *software*, respetando siempre la Licencia Pública General (GPL) por la cual se rigen (2). Dichas condiciones, sumadas a que estos sistemas son más potentes, seguros y estables, han hecho ganar al *software* libre, un lugar elevado en la comunidad informática mundial.

Cuba, en su necesidad por elevar el nivel tecnológico de sus instituciones, apuesta por el uso de estos sistemas que, además de brindar mayores beneficios a los desarrolladores y usuarios en costo, seguridad y rapidez, constituyen un ejemplo de solidaridad, ética y cooperación, pilares del ideal socialista de la Revolución Cubana.

Como entidad rectora en este proceso de sustitución de herramientas y plataformas privativas por otras alternativas libres en Cuba, media el Centro de *Software* Libre (CESOL) de la Universidad de Ciencias Informáticas (UCI) (4), el cual tiene como principal resultado en su desarrollo, la creación de la distribución GNU/Linux Nova, sistema operativo libre y puramente cubano, que pretende responder a las necesidades de las diferentes instituciones del país.

Sin embargo estas necesidades son muy diversas, puesto que pueden existir organizaciones donde se requiera una configuración específica del sistema a implantar. Es por eso que “Nova”, como *software* libre en fin, presenta la posibilidad a sus desarrolladores, de crear personalizaciones del mismo, en la medida que soliciten los clientes.

Existe un tipo de configuración para los sistemas operativos, donde el objetivo principal está en la restricción de los mismos, reduciendo considerablemente el número de sus funcionalidades (dejando habilitadas solo las necesarias), limitando permisos administrativos así como el acceso a los componentes de estos sistemas (aplicaciones, directorios y dispositivos de almacenamiento), siempre con el propósito de establecer su uso con un fin específico. A la conformación de un sistema con las características mencionadas se le denomina configuración en modo quiosco.

Estos sistemas restringidos pueden ser usados en cualquier ordenador, pero su necesidad está específicamente en las computadoras de acceso público, debido al nivel de seguridad, fiabilidad y rapidez que ofrecen los mismos (5).

Un ejemplo de ello es la propia UCI, donde existen instalaciones en las cuales sus ordenadores están previstos para un uso específico solamente, pero que no cuentan con una configuración en modo quiosco. Un caso más específico es el de los complejos comedores de la universidad, donde el control de acceso de personal se realiza mediante un programa de forma automatizada. Actualmente se construye, como proceso de evolución de *software*, una nueva versión de la aplicación incorporando nuevas funcionalidades y de esta manera poder perfeccionar el servicio del mismo. Sin embargo se han detectado dificultades en estas computadoras que por muy elevados que sean los niveles de mejoras en la aplicación, aún estarían presentes. Estos problemas son:

- El sistema operativo presente en estas computadoras es Windows XP, *software* privativo el cual, como se ha mencionado anteriormente no presenta la posibilidad de mejorarlo y adaptarlo, además de tener problemas de seguridad como la entrada de virus.

- El personal que interactúa con estos ordenadores tiene acceso al script que controla el plan diario de comensales y pueden modificarlo propiciando el desvío de recursos.

- El personal que interactúa con este programa no necesariamente tiene conocimientos avanzados sobre el funcionamiento de otras aplicaciones, por lo que puede accidentalmente borrar archivos importantes para el sistema operativo, dañando su funcionamiento.

- No existe un control sobre los dispositivos extraíbles, lo que propicia que los usuarios realicen acciones que no tienen que ver con su trabajo, provocando la introducción de virus y archivos no permitidos por las políticas de la universidad.

- Las aplicaciones de ocio (chat, música, video), pueden provocar el desvío de la atención del personal que interactúa con los ordenadores, provocando un mal desempeño en su puesto de trabajo y demorar el servicio.

- Las computadoras de estas instalaciones cuentan con hardware de pocas prestaciones, por lo que surge la necesidad de un sistema, en gran medida, ligero.

El problema se encuentra en la forma en el sistema operativo y su configuración, no en la aplicación para el control de acceso, por lo que se hace necesario establecer los correctos parámetros del mismo para lograr establecer un servicio con mayor calidad, rapidez, rigidez y control sobre las acciones de los usuarios de estas computadoras. Para ello se usará la distribución GNU/Linux Nova en modo quiosco que además de ser la desarrollada en la universidad, presenta la posibilidad de crear un sistema a la medida a través de su personalización.

Pero, lograr una configuración de este tipo no resulta sencillo, pues existen diversos elementos de las distribuciones GNU/Linux a tener en cuenta, además de que la experiencia del grupo de desarrolladores de CESOL no es muy amplia en este tipo de sistemas a la medida.

Partiendo de la problemática anterior se plantea como **problema a resolver:** ¿Cómo mejorar la seguridad en los ordenadores para el control de acceso a los comedores de la UCI de forma tal que los usuarios del mismo tengan acceso solo a las funcionalidades para las cuales está destinado dicho ordenador?, para orientar la investigación se identifica como **objeto de estudio:** el proceso de personalización del sistema operativo GNU/Linux en modo quiosco, seleccionándose como **campo de acción** la distribución cubana GNU/Linux Nova.

Para dar una solución efectiva al problema planteado anteriormente, se plantea como **objetivo general:** Desarrollar la personalización de la distribución GNU/Linux Nova en modo quiosco para el control de acceso a los comedores.

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

- Sistematizar en el estudio de las personalizaciones en GNU/Linux en modo quiosco y formas de ejecución.
- Analizar la propuesta de solución.
- Diseñar la propuesta de solución.
- Implementar la personalización de GNU/Linux Nova en modo quiosco.
- Validar la personalización de GNU/Linux Nova en modo quiosco.

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

- Estudio del estado del arte relacionado con los quioscos en la actualidad.
- Selección de herramientas necesarias para la personalización del sistema operativo GNU/Linux en modo quiosco.

- Selección de los paquetes y librerías necesarias para incluir en el sistema operativo personalizado.
- Elaboración del diseño para la configuración del sistema operativo personalizado.
- Implementación de la personalización de la distribución de GNU/Linux Nova en modo quiosco para las computadoras del control de acceso a los comedores de la universidad.
- Elaboración, ejecución y documentación de los casos de pruebas a la personalización de la distribución de GNU/Linux Nova en modo quiosco.

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

Con el desarrollo de la personalización en modo quiosco de la distribución GNU/Linux Nova, a implantar en las computadoras de los comedores, se logrará una mayor seguridad en las mismas, para el trabajo con el Sistema de Control de Acceso.

Para dar cumplimiento a los objetivos propuestos, se han combinado diferentes métodos teóricos y técnicas de la investigación científica, en la búsqueda y procesamiento de la información. Los escogidos son los siguientes:

Métodos Teórico:
El método Analítico-Sintético para descomponer el problema de investigación en varios elementos por separado y profundizar en el estudio de cada uno de ellos.

Como técnica de recogida de información se utilizaron las conversaciones Informales. Mediante estas se intercambió información con miembros del proyecto, necesaria tener en cuenta para construir la personalización de la distribución GNU/Linux Nova.

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

Capítulo 1: Fundamentación teórica:

La investigación se enfocará en los conceptos fundamentales para la comprensión del tema, el estudio acerca del uso de los quioscos, las herramientas para la configuración y personalización de los sistemas GNU/Linux, así como los lenguajes de programación necesarios para el desarrollo.

Capítulo 2: Análisis y diseño de la solución propuesta.

Se abordará con mayor detalle, los aspectos relacionados con los elementos de análisis y diseño de la solución a la problemática de la investigación. Tomando como bases fundamentales la propuesta de solución, levantamiento y descripción de requisitos, la arquitectura del *software*, así como los componentes principales a incluir en el sistema personalizado siguiendo los pasos de la metodología de software utilizada.

Capítulo 3: Implementación y pruebas.

Se implementará el sistema y documentarán los pasos definidos a partir de su desarrollo, para de esta manera llegar a una solución final, luego se realizarán las pruebas al producto con el fin de verificar su correcto funcionamiento. Como finalidad en este capítulo se tendrá un sistema funcional que podrá ser usado para el fin con que fue concebido.

Capítulo 1: Fundamentación teórica

1.1 Introducción

Para llevar a cabo la personalización de GNU/Linux Nova en modo quiosco para el sistema de gestión de acceso a los comedores de la universidad, se hace evidente la necesidad de realizar un estudio minucioso acerca de los principales conceptos y definiciones relacionados con dicha propuesta y de esta manera tener la seguridad de que la misma constituye, realmente, la solución que dará fin a los problemas encontrados. Es por ello que en este capítulo se mostrará el estudio del estado del arte relacionado al uso, importancia y aspectos más relevantes de los quioscos en la actualidad, seguido de la definición de las herramientas y técnicas a tener en cuenta, así como la elección de la metodología de desarrollo a utilizar.

1.2 Conceptos fundamentales

- **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 encima de la cual otros programas puedan funcionar (6).

- **Personalización de sistema operativo:**

Según el Diccionario de la Real Academia Española (DRAE), personalizar significa dar carácter personal a algo (7). De lo anteriormente planteado se puede establecer como personalización de un sistema operativo, a las acciones realizadas sobre el conjunto de aplicaciones que forman al mismo, con el objetivo de agregar características al sistema y adaptarlo a las necesidades de sus usuarios finales.

- **Sistema operativo base:**

Un sistema operativo Base (SOB en lo adelante) de GNU/Linux, es la combinación mínima necesaria de aplicaciones que permite la instalación de paquetes en un sistema GNU/Linux, que pueden o no soportar el arranque de la computadora (3).

- **Paquete de software:**

Un paquete 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 (8). Estos pueden ser binarios y de código fuente.

- **Paquetes binarios:**

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. Las utilidades para manejar este tipo de paquetes son apt y dpkg.
  - TGZ: Son utilizados por la distribución Linux Slackware.

- **Paquetes de código fuente:**

Contienen el código fuente del programa, estos 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). Normalmente cada aplicación tiene la información en el fichero README o INSTALL de como instalarlo.

1.3 Estudio sobre quioscos informáticos

Cuando se escucha la palabra quiosco, surge la idea de un local de pequeñas dimensiones donde se venden revistas y periódicos, o se puede comprar alimento elaborado para el consumo de las personas, pero en la informática este concepto va más allá que lo anteriormente mencionado (9).

Quiosco, es un término informático utilizado para describir una configuración realizada a un ordenador sobre su *software* con el objetivo de restringir la mayoría de las funcionalidades del mismo, dejando habilitadas sólo aquellas necesarias para la realización de las tareas por las que fue creado el dispositivo.

Existen diferentes conceptos asociados a quioscos como: *software* de quiosco, navegador de quiosco y quioscos interactivos:

1.3.1 *Software* de quiosco

Las herramientas que permiten restringir funcionalidades en los sistemas operativos o en las diferentes aplicaciones que componen los mismos se denominan *software* de quiosco. Mediante estas herramientas se pueden limitar diferentes elementos de los sistemas operativos, como por ejemplo: los menús, el acceso a la terminal, los permisos de usuarios y la utilización de aplicaciones que no sean necesarias.

A continuación se muestra una selección de las principales herramientas (*software* de quiosco) disponibles en las distribuciones GNU/Linux que permiten establecer una configuración en modo quiosco:

- **KDE Kiosk tool:**

  Es un *framework*[1] que ha sido incorporado al entorno de escritorio *KDE*, el cual permite a los administradores crear un ambiente controlado para sus usuarios, al personalizar y bloquear casi cualquier aspecto del escritorio, como por ejemplo el establecimiento del fondo de escritorio, deshabilitar las salidas del registro de usuarios y el acceso al sistema de impresión, además de la inhabilitación del acceso al *shell*[2], para garantizar la seguridad del sistema. Este se controla por medio de entradas en archivos de configuración o a través de la aplicación de interfaz gráfica de usuario, KDE *Kiosk tool* (10).

- **Pessulus**:

  Aplicación que permite al desarrollador limitar algunas de las funcionalidades del entorno de escritorio Gnome. Pessulus puede ser útil cuando se comparte el ordenador con otros usuarios y se necesita evitar que los mismos ejecuten ciertas acciones sobre el sistema (11). Entre sus funcionalidades se encuentran:

  - Evitar que el usuario use la consola del sistema, las impresoras y que ejecute “Guardar como…” en los diálogos.

  - Bloquear Paneles (no se permite hacer cambios en el panel de configuración, bloqueando la posición y los contenidos) y desactivar el cierre de sesión, apagar equipo o bloquear la pantalla.

  - Bloquear el navegador web (desactivando protocolos específicos, URL arbitrarias, obligando al usuario a estar en modo pantalla completa) (12).

- **Sabayon:**

  Es una herramienta administrativa que permite definir e implementar perfiles de escritorio en el entorno Gnome. Brinda el control de elementos del entorno como por ejemplo el panel, el menú de configuración de *gconf*[3] así como también provee de configuraciones predefinidas para aplicaciones que no son propias del entorno Gnome como el navegador *Firefox*. Se puede lograr todo esto gracias a que Sabayon tiene *"desktop within a window"* (escritorio completo en una ventana), permitiéndo

[1] Marco de desarrollo.

[2] Intérprete de comandos de GNU/Linux

[3] Utilizado por GNOME para almacenar las opciones de la configuración del entorno gráfico y de los distintos programas. Forma parte de política del entorno para mejorar y simplificar la interfaz gráfica de usuario.

personalizar el escritorio directamente en él mismo, entonces estos cambios son guardados, para ser aplicados a los perfiles de usuario con *sabayon-apply* cuando ellos inician sesión.

Sabayon es excelente para aquellos usuarios que desean proveer un entorno Gnome estandarizado a los diferentes usuarios del sistema. Maestros que administran laboratorios, librerías, y negocios como cibercafés, todos necesitan tener un escritorio "bloqueado", y hacer un bueno uso de Sabayon (13).

- **R-Kiosk:**

(Kiosk Real) es una extensión de *Firefox* que arranca el navegador en pantalla completa y desactiva todos los menús, barras de herramientas, comandos de teclado y el botón derecho del mouse. Para acceder a la página de inicio la extensión permite el uso de la combinación *Alt+Home*. Al combinar esta extensión con una página de inicio que contenga ciertos enlaces, es ideal para su uso en cibercafés cuyos usuarios deben poder navegar pero no modificar el funcionamiento del navegador. Al estar deshabilitadas todas esas características del navegador, para desinstalar la extensión se requiere que se arranque Firefox en modo seguro (14).

1.3.2 Navegador quiosco

En determinadas circunstancias, es importante mostrar una aplicación web en un navegador (browser) favorito, pero sin que se visualice este, o sea, con todas las barras ocultas (barra de direcciones, de herramientas y de estado), así como a pantalla completa; esto le permite al usuario enfocarse en el contenido, imposibilitándolo a usar las características típicas de navegación del propio *browser* (15). Estas características pertenecen a los navegadores quioscos.

En fin los navegadores quioscos no son más que navegadores web a los que se les ha bloqueado algunas de sus opciones con el propósito de ser instalados en computadoras de uso público y destinadas solamente a la navegación web.

Este modo es muy utilizado en *stands* (librerías, aeropuertos, bancos, centros comerciales, eventos como congresos y conferencias), así como en oficinas cuando un sistema corporativo está basado en un desarrollo web.

La mayoría de los navegadores web contienen herramientas que permiten su configuración en modo quiosco, para lograr las características de seguridad antes mencionadas, aunque se pueden brindar aportes desde la terminal de Linux que aumentarían dichos elementos.

1.3.3 Quioscos interactivos.

Un quiosco interactivo (o de autoservicio) consiste en una computadora situada en un lugar público que brinda determinados servicios, normalmente están compuestos de una *CPU*[4] y otros periféricos, que pueden ser pantallas (táctiles o no), teclados, lectores de todo tipo, así como impresoras. Estos generalmente están incrustados en estructuras de acero o madera. También se les conoce por el nombre de Punto de Información Multimedia o terminal informático.

El objetivo que persiguen estos quioscos en general es facilitar el uso por parte de cualquier tipo de usuario a partir de una interfaz amigable. Estos son utilizados en lugares tales como correos, bancos, aeropuertos, hospitales y grandes supermercados. La ventaja que tienen es que es el propio usuario es el que interactúa con la máquina, por eso se les llama interactivos, además de las posibilidades que ofrece internet como el acceso a todo tipo de servicios públicos, información personalizada, comercialización de servicios o productos. El objetivo principal de los mismos es liberar servicios de instituciones directamente a los clientes, de forma automatizada y sin la intervención de los integrantes de estas.

A continuación de muestran algunos ejemplos de los usos más comunes que pueden brindar los quioscos interactivos:

- **En Salas de Exposición:** En una gran exposición pueden haber varias cabinas con computadoras para ayudar a los asistentes en la ubicación de expositores específicos.

- **Información para visitantes:** En algunos hoteles suelen tener cabinas con computadoras para complementar el trabajo de un conserje, en estos quioscos se proporciona información acerca de eventos, restaurantes y locales específicos.

- **En aeropuertos:** En los aeropuertos se pueden encontrar cabinas con computadoras, donde los viajeros pueden comprar boletos, boletos de impresión y elegir asientos.

- **Muestra de productos:** Los quioscos también pueden ser utilizados para ayudar a los clientes a elegir un producto mostrando imágenes de las diferentes opciones con las que se cuenta.

- **En centros médicos:** En estos casos se pueden utilizar los quioscos para ayudar a explicar los procedimientos médicos complicados.

[4] Unidad Central de Procesamiento de un ordenador.

Es necesario decir que en la actualidad, esta peculiar configuración o uso del ordenador conocida como modo quiosco tiene su mayor explotación en los Quioscos Interactivos, y Cibercafés aunque también es ideal para entornos donde se requiera un nivel de seguridad elevado, dígase escuelas, hogares y sobre todo en empresas.

1.3.4 Conclusiones acerca del estudio de los quioscos informáticos

A pesar de que el estudio de los navegadores y quioscos de autoservicio poseen aspectos similares con respecto a la configuración necesaria para resolver los problemas anteriormente planteados en la introducción, no ofrecen los elementos necesarios para dar la mejor solución a los mismos, debido a que primeramente la aplicación para el control de acceso a los comedores a incluir en el sistema personalizado es de escritorio, por lo que el estudio de los navegadores quioscos no aporta mucho al desarrollo de la solución de la presente investigación, aunque constituye un punto de referencia a tener en cuenta para quien desee realizar un quiosco que cuente con un navegador web, por esta razón ha sido incluido en la investigación.

Otro elemento es que el quiosco a desarrollar, a diferencia de los quioscos interactivos, será utilizado solamente por un grupo reducido de personal capacitado y autorizado.

Por otra parte, es cierto que con las herramientas estudiadas se podría restringir un sistema, pero no constituiría la solución más óptima puesto que a pesar de que el uso de las mismas es automatizado, facilitando el trabajo y aumentando el ahorro de tiempo pueden existir elementos en los sistemas operativos que no puedan ser deshabilitados con estas herramientas, siendo necesario recurrir a la forma manual.

Pero ¿cómo llevar la información obtenida a la presente investigación y de esta manera lograr el desarrollo de la solución propuesta? Para ello se hace necesario definir de qué forma se va a proceder para lograr desarrollar la personalización del sistema operativo Nova.

El primer paso en la construcción de cualquier personalización de Nova consiste en la instalación del Sistema Operativo Base (SOB) (16), a partir del cual se realizarán las configuraciones necesarias para configurar el sistema en modo quiosco. Esta forma posee una elevada superioridad con respecto al uso de los *softwares* de quioscos debido a que no se restringen funcionalidades sino que simplemente no se incluyen en el nuevo sistema, logrando una mayor ligereza en el mismo.

A partir de este momento queda definido el punto de partida para el desarrollo del sistema personalizado. A continuación se muestran los elementos referentes a las herramientas y la metodología a usar para llevar a cabo el desarrollo de la investigación.

1.4 Herramientas y técnicas de desarrollo

En el presente epígrafe se expondrán las herramientas utilizadas para la creación y configuración del sistema personalizado en modo quiosco.

1.4.1 Herramienta para la creación del Sistema Operativo Base

En CESOL, actualmente se utiliza para la creación del SOB de Nova la herramienta debootstrap. 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*[5] o *apt*[6]. Para ello, descarga ficheros de extensión *deb* desde un repositorio especificado previamente, y los desempaqueta cuidadosamente en un directorio al que finalmente se puede entrar con *chroot*[7] (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.

[5] Es el gestor de paquetes Debian de medio nivel se utiliza para instalar, construir, borrar y gestionar los paquetes de Debian GNU/Linux.

[6] Sistema de gestión de paquetes, simplifica en gran medida la instalación y eliminación de programas en los sistemas GNU/Linux.

[7] Jaula de construcción del sistema montado.

- **--components=alpha,beta:** Utilizar paquetes de los componentes listados.
- **--variant=minbase | buildd | fakechroot | scratchbox:** Nombre de la variante a utilizar. Actualmente, la variantes admitidas son:
  - **minbase,** que sólo incluye paquetes esenciales y *apt;*
  - ***buildd,*** que instala el *build-essential*[8] en $MONTAJE.
  - ***fakechroot,*** que instala los paquetes sin privilegios de root.
  - ***Scratchbox*[9]**, 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.

1.4.2 IDE de desarrollo a utilizar

Geany: Es un editor de programación compatible con múltiples lenguajes, como C*, Java, PHP, HTML, Python, Perl* o *Pascal*. Es un pequeño y rápido *IDE*, que sólo tiene unas pocas dependencias de otros paquetes, sólo requiere de las bibliotecas de tiempo de ejecución *GTK*[10]. Las características básicas de Geany son (19)**:**

- El resaltado de sintaxis.
- Plegado de código.
- Nombre de símbolo de autocompletar.
- Auto-cierre de etiquetas *XML y HTML.*
- Consejos de llamadas.
- Muchas tipos de archivos soportados, incluyendo *C, Java, PHP, HTML, Python, Perl, Pascal.*
- Listas de símbolos.
- Navegación de Código.
- Sistema *Build* para compilar y ejecutar el código.

8 Es un paquete que contiene herramientas necesarias para la creación, compilación e instalación de programas [22].

9 Es un conjunto de herramientas de compilación cruzada diseñado para hacer que el desarrollo de aplicaciones Linux embebido más fácil. También proporciona un conjunto completo de herramientas para integrar y cross-compilar una distribución Linux completa (18).

10 "GIMP Tool Kit" es una biblioteca del equipo GTK+, la cual contiene los objetos y funciones para crear la interfaz gráfica de usuario. Maneja widgets como ventanas, botones, menús, etiquetas, deslizadores, pestañas, etc.

Geany se puede ejecutar en múltiples plataformas. El código está disponible bajo los términos de la Licencia Pública General GNU.

1.4.3 Lenguajes de desarrollo

- **Python:**

Es un lenguaje de programación de alto nivel creado por Guido van Rossum en el año 1991, la extensión de los proyectos creados en él es **.**py. Actualmente *Python* se desarrolla como un proyecto de código abierto, administrado por la *Python Software Fundation*. Posee una sintaxis muy limpia y un código legible. Es interpretado o de *script*, esto quiere decir que se ejecuta utilizando un programa intermedio llamado intérprete, pero tiene muchas características de los lenguajes *compilados*[11] por lo que se puede decir que es semi interpretado; con tipado dinámico, es decir, no es necesario declarar el tipo de dato que va a contener una determinada variable, sino que su tipo se determinará en tiempo de ejecución según el tipo del valor al que se asigne, y el tipo de esta variable puede cambiar si se le asigna un valor de otro tipo; fuertemente tipado, ya que no se permite tratar a una variable como si fuera de un tipo distinto al que tiene, es necesario primero convertir de forma explícita dicha variable al nuevo tipo; multiplataforma y orientado a objetos o sea que los elementos del problema se tratan como clases y objetos, y la ejecución del programa consiste en la interacción entre estos, también permite la programación imperativa, programación funcional y programación orientada a aspectos (20).

- **Bash:**

Bash *(Bourne again shell)* es un programa informático cuya función consiste en interpretar órdenes. Está basado en el shell de Unix y es compatible con *POSIX*[12]. 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 shell más populares de distribuciones de GNU/Linux, y la mayoría de los script 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 (21).

11 Compila el código a lenguaje máquina que pueda comprender y ejecutar directamente una computadora.

12 El acrónimo de Portable Operating System Interface; la X viene de UNIX. El término fue sugerido por Richard Stallman en respuesta a la demanda de la IEEE, que buscaba un nombre fácil de recordar. La traducción del acrónimo es "Interfaz portable de sistema operativo".

1.4.4 Herramienta de modelado

Visual Paradigm:

Es una herramienta *CASE*[13] para desarrollo de aplicaciones utilizando modelado *UML*[14] 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. Ente los elementos que ofrece Visual Paradigm se encuentran (22):

- Navegación intuitiva entre la escritura del código y su visualización.
- Potente generador de informes en formato PDF/HTML.
- Documentación automática *Ad-hoc*[15].
- Ambiente visualmente superior de modelado.
- Sofisticado diagramador automáticamente de *Layout*[16].

1.5 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 desarrollo de un *software*. Estas especifican (23):

- Cómo se debe dividir un proyecto en etapas.
- Qué tareas hay que realizar en cada etapa.
- Qué salidas se producen y cuándo.
- Qué herramientas se utilizan.
- Cómo se gestiona y controla un proyecto.

13 Computer Aided *Software* Engineering / Ingeniería de *Software* Asistida por Computadora.

14 Unified Modeling Language (Lenguaje de Modelado Unificado).

15 Que es apropiado o está dispuesto especialmente para un fin: solución ad hoc.

16 suele utilizarse para nombrar al esquema de distribución de los elementos dentro un diseño.

1.5.1 Metodología de desarrollo ágil

El desarrollo ágil de *software* sigue una filosofía en la que se busca la satisfacción del cliente y la entrega temprana de *software*, permitiendo la adaptabilidad a cualquier cambio, los equipos de desarrollo son pequeños y el desarrollo en general es simple. La agilidad además de la respuesta efectiva al cambio, también incluye la estimulación de estructuras y actitudes de los equipos para facilitar la comunicación entre los miembros del mismo, incluyendo al cliente como parte de este, resalta la entrega rápida del *software* operativo y resta importancia a los productos de trabajo intermedio. Este tipo de metodología surgió como concepto a mediados de 1990, como reacción a las metodologías robustas (24).

Se escogió una metodología de desarrollo de *software* ágil, debido a que el equipo de desarrollo es pequeño, además el período de tiempo de entrega del producto es relativamente corto. Específicamente se va a utilizar la metodología Nova OpenUp ya que es ideal para el desarrollo de una distribución GNU/Linux o una personalización de esta.

1.5.2 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[17], y otras que corresponden el nivel 2 de *CMMi*[18] *(25)*.

Se desarrolla en ***4 fases*** que se muestran en la figura 1:

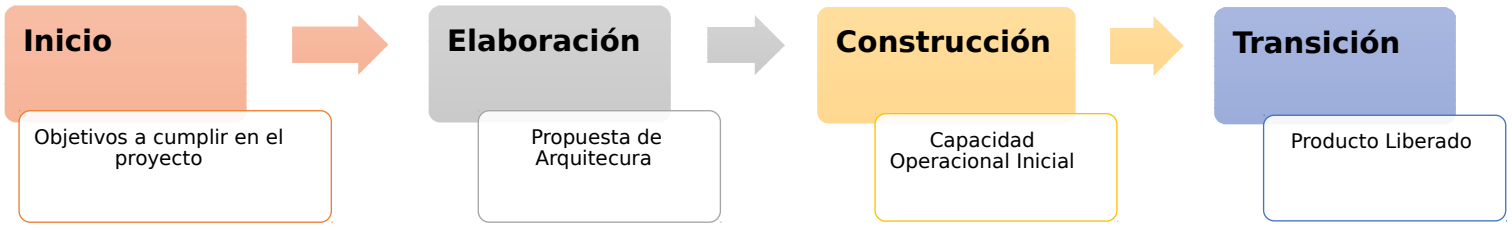


Figura 1: Fases de la metodología Nova OpenUp

[17] De las siglas en inglés Extreme Programming (Programación extrema), metodología ágil de desarrollo de *software*.

[18] De las siglas en inglés Capability Maturity 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*.

Fases de la Metodología Nova OpenUp

Inicio: En esta fase se debe entender el propósito y objetivos a cumplir, conocer los requisitos que debe cumplir el producto, y realizar un análisis para determinar costos y riesgos asociados al proyecto.

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 usarse 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.

Es necesario decir que esta metodología ofrece la libertad de ser configurada y de esta manera hacerla todo lo moderna o tradicional que se desee mediante la extracción de las tareas no obligatorias según el producto a construir. Por tanto, puede tener dos variantes, una para desarrollos pesados y otra para desarrollos ligeros. En el caso de la presente investigación se tomará en consecuencia con las características del producto, el desarrollo ligero (25).

1.6 Conclusiones del capítulo

En el concluido capítulo, con el objetivo de establecer las bases teóricas para el desarrollo de la personalización de GNU/Linux Nova en modo quiosco para el sistema de gestión de acceso a los comedores, se realizó un estudio de los diferentes conceptos asociados a este tipo de configuración , definiéndose que el sistema a personalizar no es un navegador quiosco ni un quiosco interactivo además de que las herramientas o software de quioscos disponibles en Linux no ofrecen la solución más óptima para el sistema a personalizar, por lo que se decidió desarrollar el quiosco a partir de un Sistema Operativo Base, utilizando para ello Debootstrap. En cuanto a los lenguajes de programación a utilizar se definieron Python en su versión 2.7 y Bash, mediante el *IDE* Geany. Como metodología de desarrollo de *software* se definió Nova OpenUp en su desarrollo ligero.

A partir de este momento 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: Análisis y diseño de la solución propuesta

2.1 Introducción

Definidos los elementos teóricos de la investigación, la metodología de desarrollo a utilizar y las herramientas para ejecutar la propuesta de solución, se procederá en el presente capítulo a la descripción de los elementos correspondientes al análisis y diseño de esta solución.

2.2 Propuesta del sistema a desarrollar

Con el objetivo de resolver los problemas planteados se tomó como solución a los mismos, la construcción de la personalización de GNU/Linux Nova en modo quiosco. Este sistema, con un número reducido de aplicaciones y permisos limitados para los usuarios que controlan estos servicios ofrece un nivel de seguridad elevado.

En el presente epígrafe se hará una descripción del funcionamiento del sistema a partir del modelo (Ver figura 2) del mismo, mostrando sus principales componentes y como es la relación entre estos.

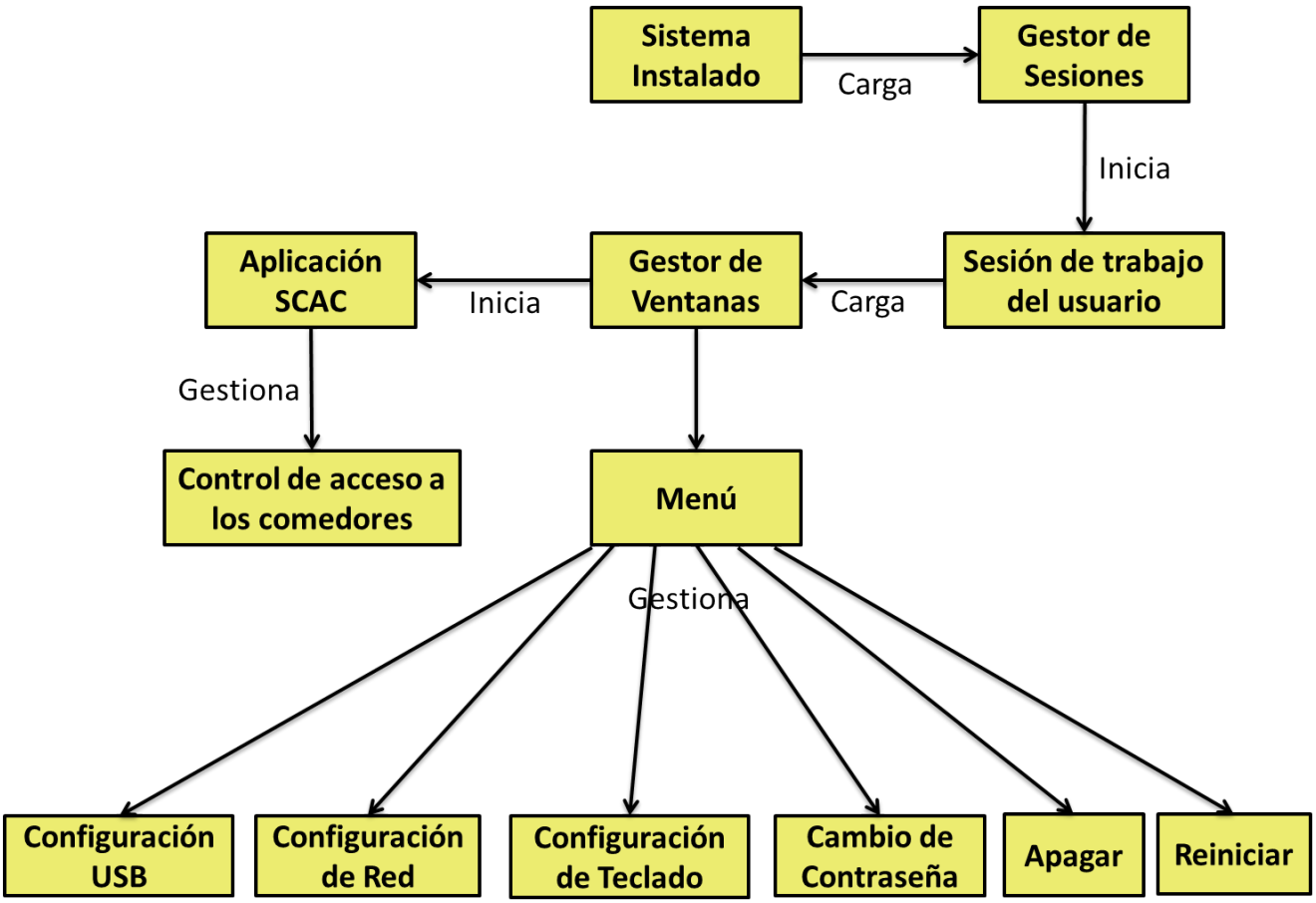


Figura 2: Relación entre componentes del sistema

2.3 Gestión del proceso de construcción del quiosco

Para gestionar la construcción de la personalización en modo quiosco de la distribución GNU/Linux Nova para el control de acceso a los comedores de la universidad, se realizó la planificación de dicho proceso (Ver figura 3). El tiempo disponible para llegar al resultado esperado es de 7 meses, por lo que ha sido repartido en tareas que abarcan todo el ciclo de vida del mismo.

ID	Tareas a realizar	Inicio	Fin	Duración
1	Proceso de creación de quiosco	01/10/2013	26/05/2014	170d
2	Recopilación de información	01/10/2013	28/04/2014	150d
3	Capturar requisitos del cliente	01/10/2013	14/10/2013	10d
4	Especificación y evaluación de requisitos	11/10/2013	24/10/2013	10d
5	Descripción de requisitos agiles	21/10/2013	01/11/2013	10d
6	Definición de vista del sistema	08/11/2013	21/11/2013	10d
7	Definición de vista de seguridad	21/11/2013	04/12/2013	10d
8	Definición de vista de infraestructura	04/12/2013	17/12/2013	10d
9	Definición de vista despliegue	17/12/2013	30/12/2013	10d
10	Creación de Sistema operativo base	30/12/2013	17/01/2014	15d
11	Modificación de paquetes	17/01/2014	30/01/2014	10d
12	Instalación de paquetes	30/01/2014	19/02/2014	15d
13	Configuración del sistema	19/02/2014	04/03/2014	10d
14	Elaboración de manual de usuario	04/03/2014	24/03/2014	15d
15	Empaquetamiento del sistema	02/04/2014	11/04/2014	8d
16	Pruebas de aceptación	22/04/2014	01/05/2014	8d
17	Plantilla de no conformidades	01/05/2014	12/05/2014	8d
18	Incorporar mejoras	12/05/2014	26/05/2014	11d

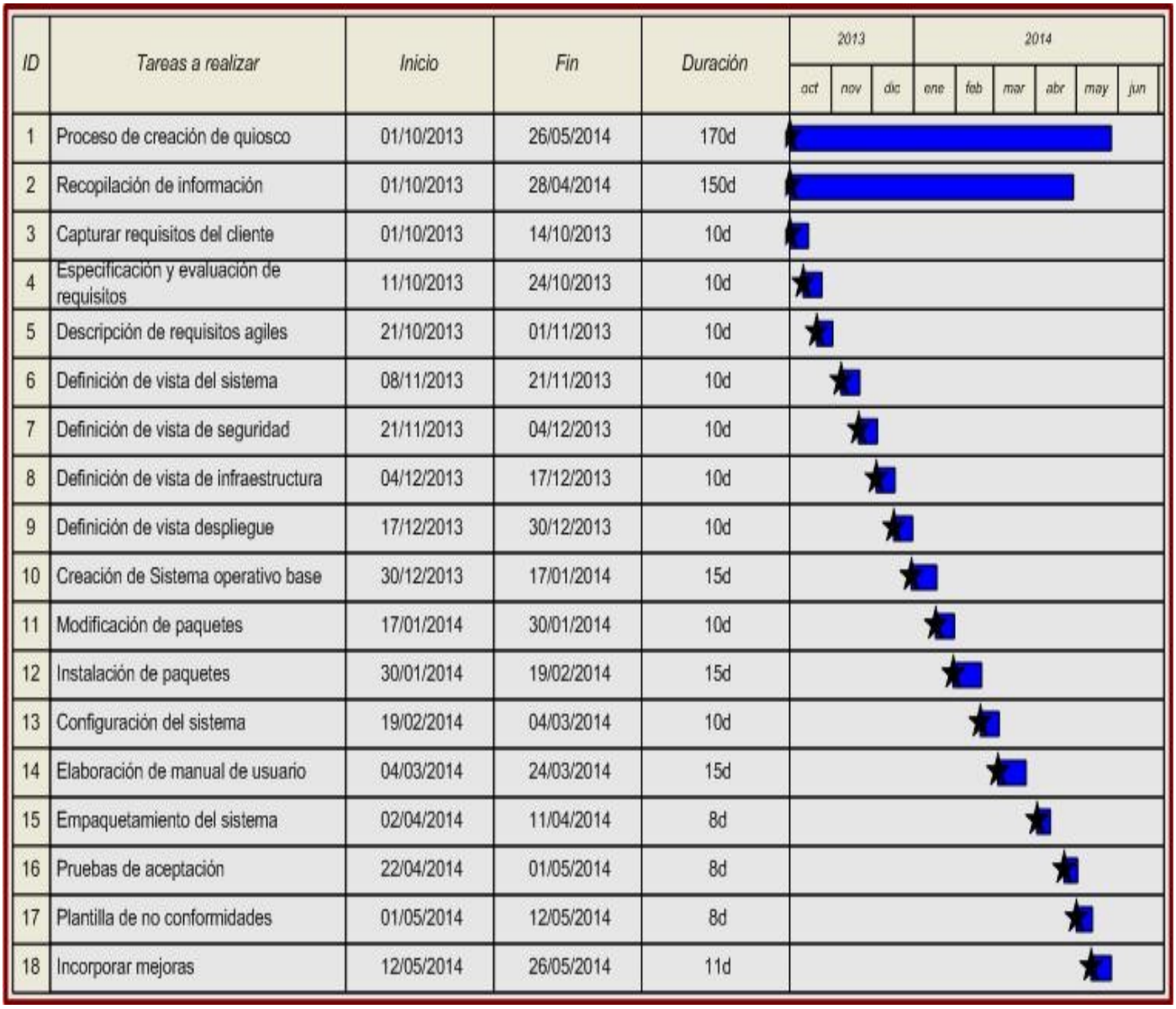


Figura 3: Diagrama de Gantt para las tareas de desarrollo del sistema

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 1***).

Asignadas a: Laura Elida Miranda Domínguez y Leonel Barroso Bayón.						
Nombre	**Descripción**	**Fecha-inicio**	**Fecha-fin**	**Riesgo**	**Mitigación**	**Estado**
Recopilación de información.	- Estudio del estado del arte. - Estudio de herramientas para personalización en modo quiosco. - Estudio de tecnologías para el desarrollo del sistema. - Estudio de Sistemas GNU/Linux.	1/10/2013	28/04/2014	- 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.	Hecho.
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.	1/10/2013	14/10/2013	Pueden obviarse requisitos fundamentales para el desarrollo del sistema. - Demora en cuanto a tiempo.	- Ser precisos en las preguntas al cliente. Elaborar un cuestionario. - Establecer un efectivo control de las tareas.	Hecho.
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_Especificacion_de_requisitos_de_*software*).	14/10/2013	24/10/2013	- 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.	Hecho.
Descripción de requisitos ágiles.	Se detallarán los requisitos en función del resultado a esperar, teniendo en cuenta su procesamiento interno (010114c_Descripcion_de_requisito_agil).	24/10/2013	01/11/2013	- Demora en la entrega.	- Establecer un efectivo control de las tareas.	Hecho.

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*).	01/11/2013	21/11/2013	- Demora en la entrega.	- Establecer un efectivo control de las tareas.	Hecho.
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).	21/11/2013	04/12/2013	- Demora en la entrega.	- Establecer un efectivo control de las tareas.	Hecho.
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).	04/12/2013	17/12/2013	- Demora en la entrega.	- Establecer un efectivo control de las tareas.	Hecho.
Definición de vista despliegue	Describirá las características que deben tener los nodos para que la distribución GNU/Linux pueda funcionar sobre ellos.	17/12/2013	30/12/2013	- Demora en la entrega.	- Establecer un efectivo control de las tareas.	Hecho.

Desarrollo del sistema						
Creación del SOB.	Se creará el Sistema Operativo Base, con el mínimo de paquetes requeridos para su posterior configuración.	30/12/2013	17/01/2014	- Demora en la entrega.	- Establecer un efectivo control de las tareas.	Hecho.
Instalación de paquetes.	Se instalarán los paquetes necesarios para el funcionamiento del sistema operativo final.	30/01/2014	19/02/2014	- No presenta.		Hecho.
Configuración de elementos del sistema.	Se modificará los archivos de configuración, para cambiar el inicio de sesión y hacer que la aplicación de los comedores se cargue al inicio.	19/02/2014	04/03/2014	- Demora en la entrega.	- Establecer un efectivo control de las tareas.	Hecho.
Elaboración de manual de usuario.	Constituye una ayuda a los usuarios del sistema para poder lograr un entendimiento del comportamiento de este.	04/03/2014	24/03/2014	- Demora en cuanto a tiempo.	- Establecer un efectivo control de las tareas.	Hecho.
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.	02/04/2014	11/04/2014	- No presenta.		Hecho.
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.	22/04/2014	01/05/2014	- No presenta.		Hecho.

Mejoras del sistema						
Plantilla de no conformidades	Los elementos que los clientes no consideren de calidad serán guardados en un documento para su posterior arreglo.	01/05/2014	12/05/2014	- No presenta.		Hecho.
Incorporar mejoras.	Se mejoran los elementos planteados en la plantilla de no conformidades.	12/05/2014	26/05/2014	- No presenta.		Hecho.

Tabla 1: Tareas para el proceso personalización de Nova en modo quiosco en función de riesgos.

2.4 Modelo del proceso de construcción del quiosco para el sistema de control de acceso.

Para modelar el proceso de construcción del quiosco de Nova para el sistema de control de acceso a los comedores (Ver ***figura 4***) se tuvieron en cuenta los roles desempeñados por los participantes, las actividades realizadas por cada uno, las decisiones a tomar y los artefactos generados por cada actividad.

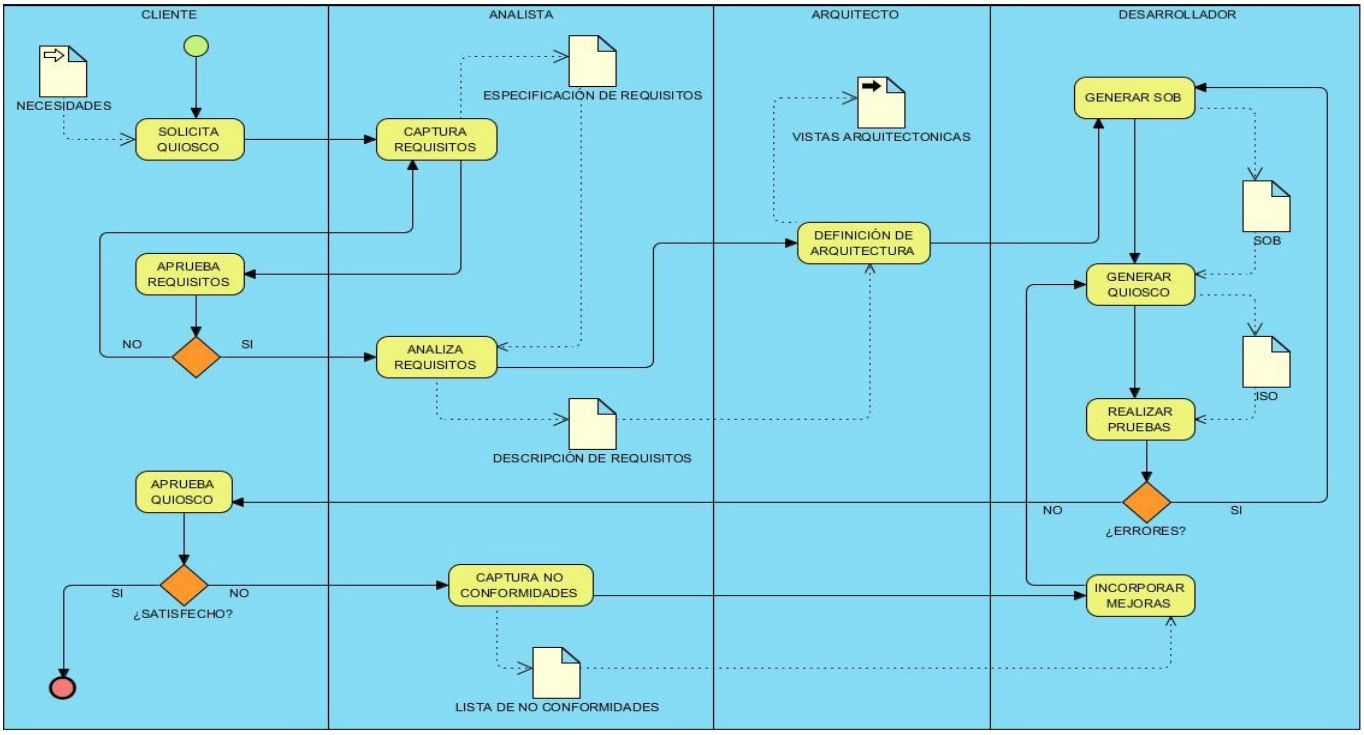


Figura 4: Diagrama de proceso de construcción del quiosco para el control de acceso a los comedores

2.5 Requisitos de *software*

Los requerimientos para un *software* son la descripción de los servicios proporcionados por el sistema y sus restricciones operativas. Los requerimientos reflejan las necesidades de los clientes de un sistema que ayude a resolver algún problema (23). A continuación se muestran los requisitos obtenidos para el desarrollo del quiosco de Nova.

2.5.1 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 (23). A continuación se muestran los requisitos funcionales definidos para la generación del sistema Nova en modo quiosco para el control de acceso a los comedores.

RF1: Instalar el sistema operativo..

RF2: Iniciar el Sistema de Control de Acceso a los comedores.

RF3: Acceder al menú del sistema.

RF4: Cambiar idioma del teclado.

RF5: Apagar PC.

RF6: Cambiar contraseña del administrador.

RF7: Configurar la red.

RF8: Autorizar dispositivos en la PC.

RF9: Iniciar Consola para la administración.

RF10: Cancelar menú.

RF11: Reiniciar sistema.

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 brinda una guía mejor elaborada de las mismas.

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

2.5.2 Descripción de los requisitos ágiles

La descripción de los requisitos ágiles posee gran importancia para el desarrollo del producto en la metodología Nova OpenUp, debido a que con un buen trabajo realizado en la ejecución de las mismas se podrán, a partir de estas, realizar las pruebas de funcionamiento al sistema creado.

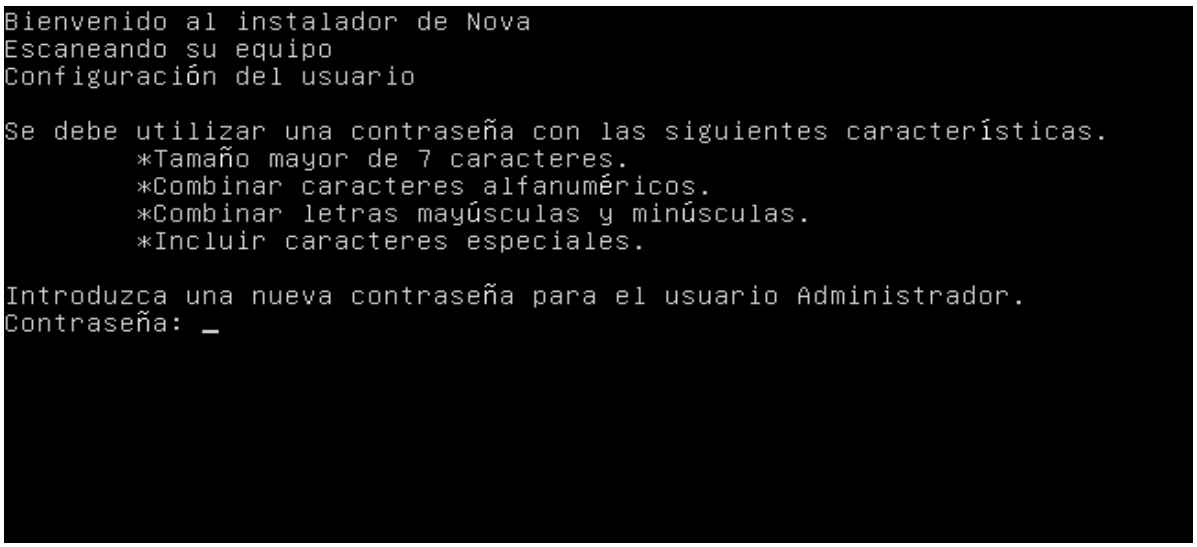
Número: RF1	**Nombre del requisito:** Instalar el sistema operativo.	
Programador: Leonel Barroso Bayón y Laura Elida Miranda Domínguez.		**Iteración Asignada:** 1
Prioridad: Alta		**Tiempo Estimado:** 1
Riesgo en Desarrollo: No posee riesgo.		**Tiempo Real:** 0,1
Descripción: Al encender la computadora, el usuario debe haber insertado algún dispositivo booteable con el sistema operativo dentro, luego al iniciar a cargar desde el sistema, este comenzará automáticamente con la instalación pidiéndole los datos de configuración al usuario que esté instalando, este los llena y el sistema instala (Figura 5).		
Prototipo de interfaz:  *Figura 5: Menú del Instalador del sistema*		

Tabla 2: Descripción del RF1 Instalar Sistema Operativo

Número: RF2	**Nombre del requisito:** Iniciar el Sistema de Control de Acceso a los comedores.
Programador: Leonel Barroso Bayón y Laura Elida Miranda Domínguez.	**Iteración Asignada:** 1
Prioridad: Alta	**Tiempo Estimado:** 1
Riesgo en Desarrollo: No posee riesgo.	**Tiempo Real:** 0,1
Descripción: Al encender la computadora, el sistema debe cargar los elementos necesarios para el quiosco. El acceso del usuario nova (creado por defecto en la instalación) entrará automáticamente en sesión. Posteriormente se iniciará el sistema de control de acceso a los comedores en pantalla completa. (Figura 6)	
Observaciones: La aplicación no poseerá botón de cerrar, ni de minimizar, por lo que siempre estará en pantalla.	

Prototipo de interfaz

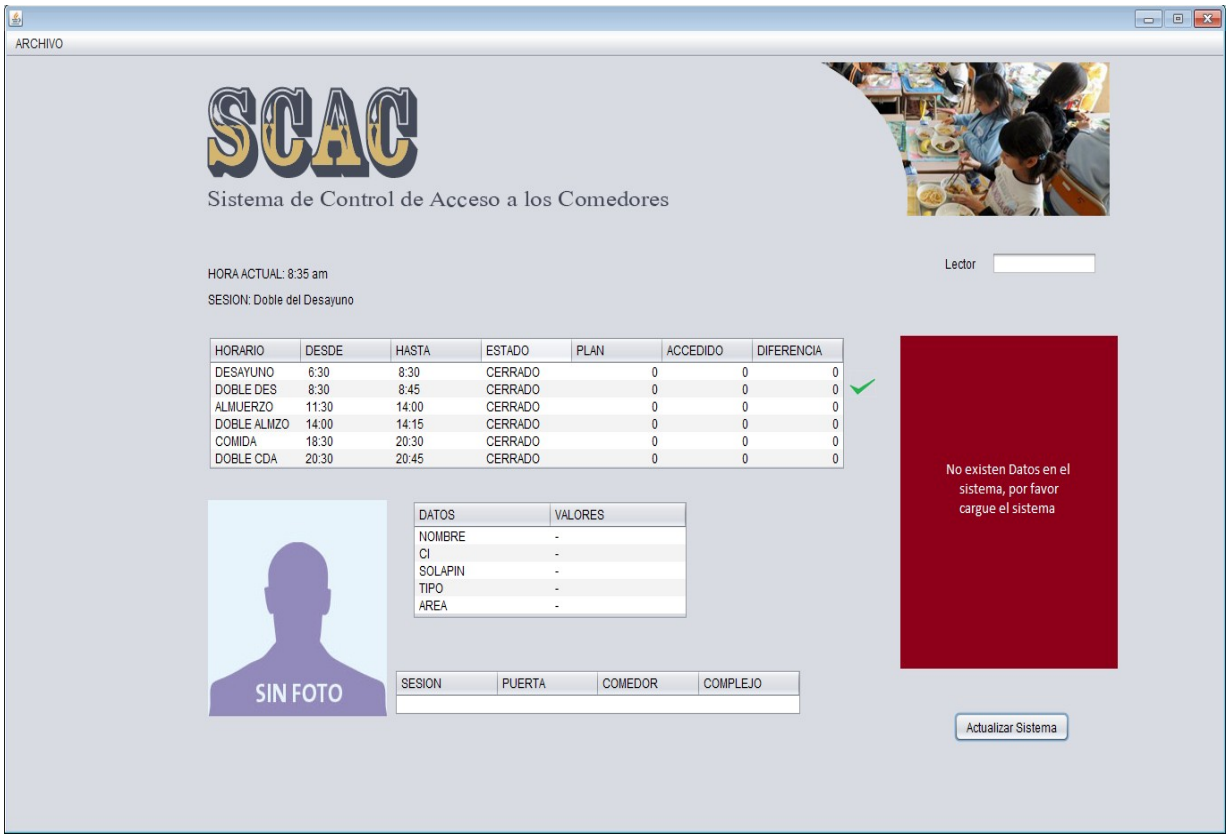


Figura 6: Prototipo de sistema de control de acceso a los comedores

Tabla 3: Descripción del RF2 Iniciar Sistema de control de acceso.

Número: RF3	**Nombre del requisito:** Acceder al menú del sistema	
Programador: Leonel Barroso Bayón y Laura Elida Miranda Domínguez.		**Iteración Asignada:** 1
Prioridad: Alta		**Tiempo Estimado:** 1
Riesgo en Desarrollo: No posee riesgos		**Tiempo Real:** 0,1

Descripción: Mediante la combinación de teclas “Ctrl+M”, se muestra el panel de control o menú del sistema, con las diferentes configuraciones del mismo. (Figura 7).

1. Cancelar
2. Configurar USB.
3. Cambiar contraseña.
4. Configurar teclado.
5. Configurar red.
6. Reiniciar
7. Apagar

Observaciones:

Prototipo de interfaz:

Figura 7: Menú del sistema

Tabla 4: Descripción del RF3 Acceder al menú del sistema

Número: RF4	**Nombre del requisito:** Cambiar idioma del teclado.	
Programador: Leonel Barroso Bayón y Laura Elida Miranda Domínguez.		**Iteración Asignada:** 1
Prioridad: Alta		**Tiempo Estimado:** 1
Riesgo en Desarrollo: No posee riesgos		**Tiempo Real:** 0,1
Descripción: Al elegir la opción de menú: Configurar teclado (Figura 8), se habilita un cuadro de diálogo (Figura 9), con las opciones de selección de idioma de teclado a seleccionar.		
Observaciones:		

Prototipo de interfaz:

Figura 8: Menú configurar teclado

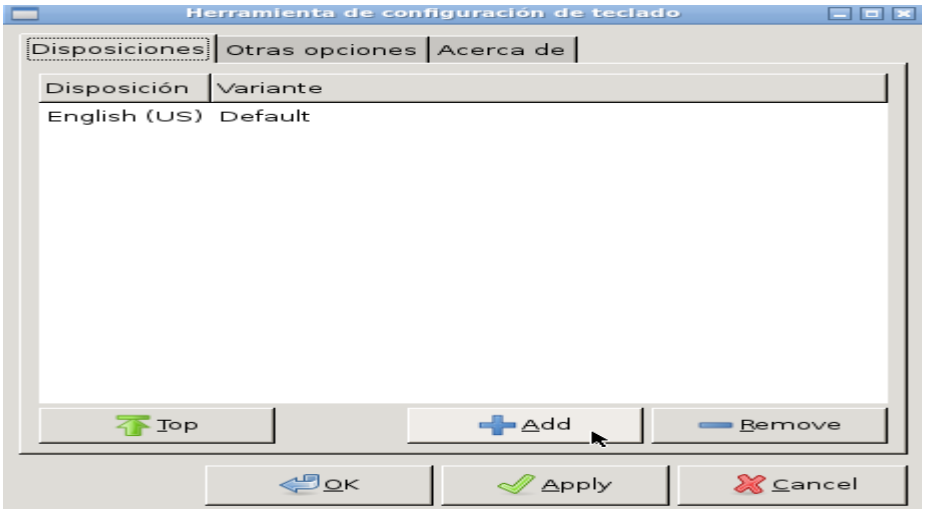


Figura 9: Opciones de cambio de idioma del sistema.

Tabla 5: Descripción del RF4 Cambiar idioma.

Número: RF5	**Nombre del requisito:** Apagar PC**.**	
Programador: Leonel Barroso Bayón y Laura Elida Miranda Domínguez.		**Iteración Asignada:** 1
Prioridad: Alta		**Tiempo Estimado:** 1
Riesgo en Desarrollo: No posee riesgos		**Tiempo Real:** 0,1
Descripción: Al elegir la opción de menú: Apagar, se apagará la pc.		
Observaciones: Se apaga el sistema (Figura 10).		
Prototipo de interfaz:  *Figura 10: Menú apagar sistema*		

Tabla 6: Descripción del RF5 Apagar PC.

Número: RF6	**Nombre del requisito:** Cambiar contraseña del administrador	
Programador: Leonel Barroso Bayón y Laura Elida Miranda Domínguez.		**Iteración Asignada:** 1
Prioridad: Alta		**Tiempo Estimado:** 1
Riesgo en Desarrollo: No posee riesgos		**Tiempo Real:** 0,1
Descripción: Al elegir la opción de menú: Cambiar Contraseña (Figura 11), se muestra un cuadro de diálogo pidiendo la contraseña de administrador (Figura 12), lanzando después una consola que solo permitirá el cambio de contraseña (Figura 13).		
Observaciones:		

Prototipo de interfaz:

Figura 11: Menú cambiar contraseña

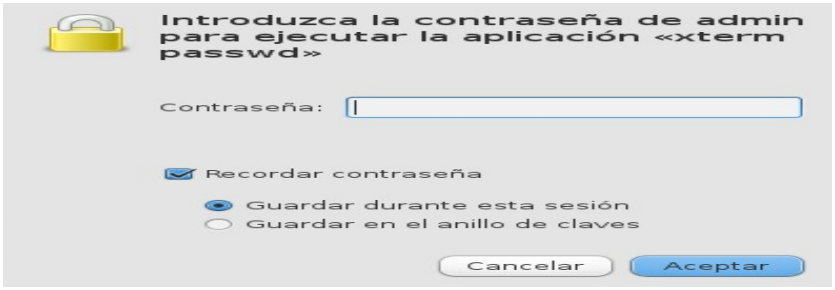


Figura 12: Cuadro de diálogo para verificar al administrador

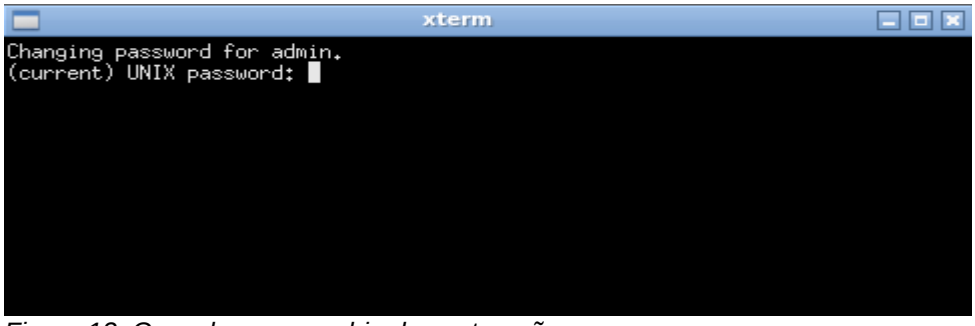


Figura 13: Consola para cambio de contraseña

Tabla 7: Descripción del RF6 Cambiar contraseña de administrador.

Número: RF7	**Nombre del requisito:** Configurar la red.	
Programador: Leonel Barroso Bayón y Laura Elida Miranda Domínguez.		**Iteración Asignada:** 1
Prioridad: Alta		**Tiempo Estimado:** 1
Riesgo en Desarrollo: No posee riesgos		**Tiempo Real:** 0,1
Descripción: Al elegir la opción de menú: Configurar la red (Figura 14), se muestra un cuadro de diálogo pidiendo la contraseña de administrador (Figura 12, tabla 7), lanzando después la interfaz para la configuración de la red (Figura 15).		
Observaciones: Solo podrá acceder el administrador del sistema		

Prototipo de interfaz:

Figura 14: Menú configurar Red

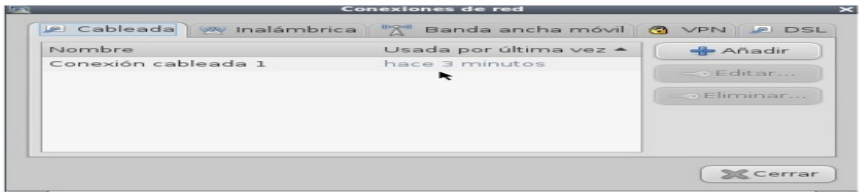


Figura 15: Interfaz para configurar la red

Tabla 8: Descripción del RF7 Configurar Red.

Número: RF8	**Nombre del requisito:** Configurar USB en la PC.	
Programador: Leonel Barroso Bayón y Laura Elida Miranda Domínguez.		**Iteración Asignada:** 1
Prioridad: Alta		**Tiempo Estimado:** 1
Riesgo en Desarrollo: No posee riesgos		**Tiempo Real:** 0,1
Descripción: Al usuario insertar un dispositivo USB, la PC no lo reconocerá y habrá que autorizarlo. Al elegir la opción de menú: Configurar USB (Figura 16), se muestra un cuadro de diálogo pidiendo la contraseña de administrador (Figura 12, tabla 7), lanzando después la interfaz para la configuración del dispositivo USB (Figura 17).		
Observaciones: Solo podrá acceder el administrador del sistema.		

Prototipo de interfaz:

Figura 16: Menú configurar USB

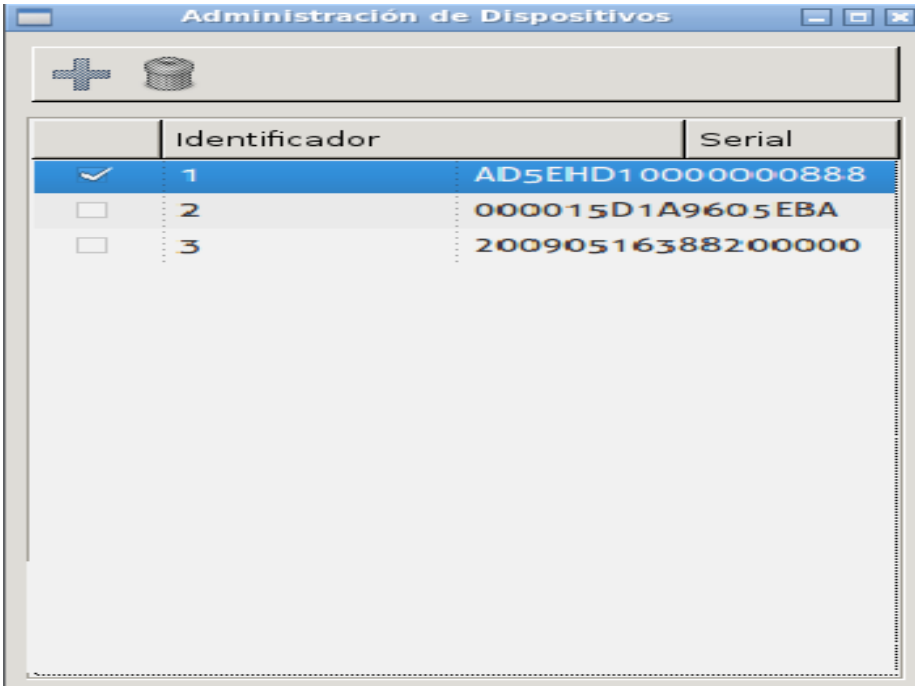


Figura 17: Interfaz de configuración de medio USB

Tabla 9: Descripción del RF8 Configurar USB.

Número: RF9	**Nombre del requisito:** Iniciar Consola para la administración.	
Programador: Leonel Barroso Bayón y Laura Elida Miranda Domínguez.		**Iteración Asignada:** 1
Prioridad: Alta		**Tiempo Estimado:** 1
Riesgo en Desarrollo: No posee riesgos		**Tiempo Real:** 0,1
Descripción: Existen configuraciones que desee realizar el administrador del sistema (por ejemplo unir la PC al dominio UCI.CU) que deberán ser realizadas por comandos de consola. Solo se podrá acceder a la tty2 del sistema (única habilitada) para realizar dichas configuraciones, mediante la combinación de teclas “Ctrl+Alt+F2”. Dentro se pedirá la contraseña de administrador (Figura 18).		
Observaciones: Solo podrá acceder el administrador del sistema. Para salir de la consola se usa la combinación de teclas “Cntrl+Alt+F7”.		

Prototipo de interfaz:

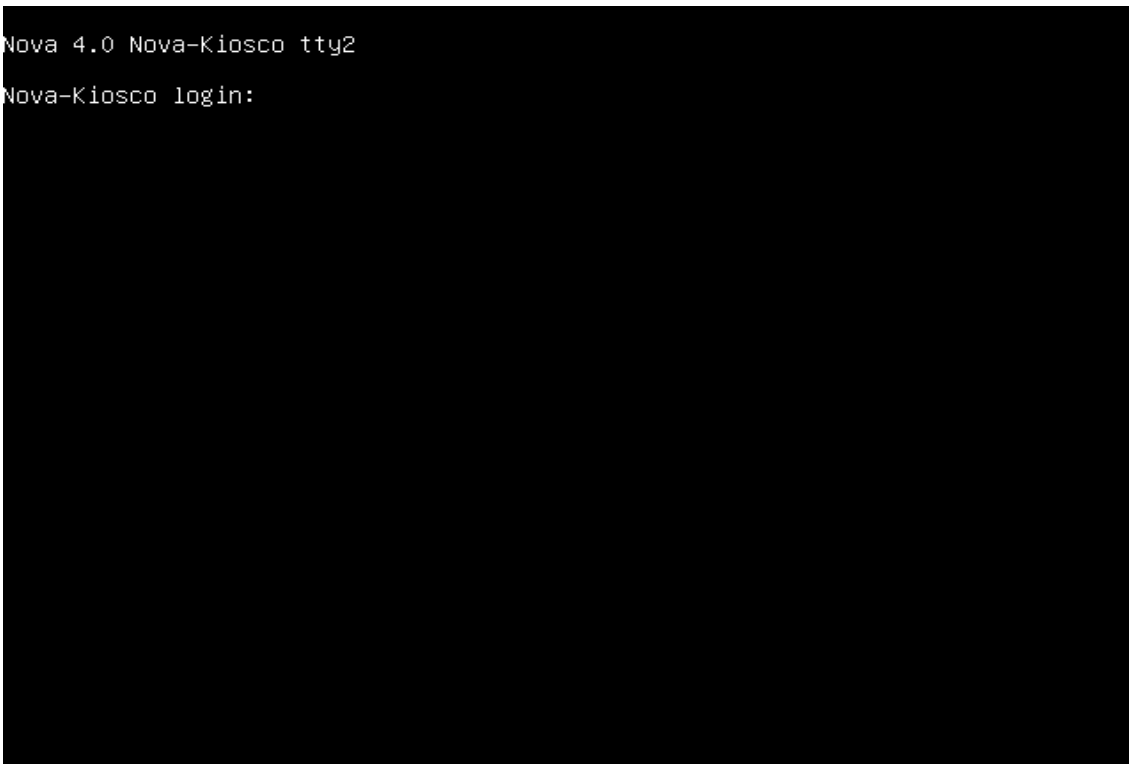


Figura 18: Tty 2 para usuario administrador

Tabla 10: Descripción del RF9 Iniciar consola de administrador.

Número: RF10	**Nombre del requisito:** Cancelar Menú del sistema	
Programador: Leonel Barroso Bayón y Laura Elida Miranda Domínguez.		**Iteración Asignada:** 1
Prioridad: Alta		**Tiempo Estimado:** 1
Riesgo en Desarrollo: No posee riesgos		**Tiempo Real:** 0,1
Descripción: Al elegir la opción de menú: Cancelar (Imagen 17), se cerrará el menú del sistema.		
Observaciones:		
Prototipo de interfaz:  *Figura 19: Menú cancelar*		

Tabla 11: Descripción del RF10 Cancelar menú del sistema.

Número: RF 11	**Nombre del requisito:** Reiniciar PC	
Programador: Leonel Barroso Bayón y Laura Elida Miranda Domínguez.		**Iteración Asignada:** 1
Prioridad: Alta		**Tiempo Estimado:** 1
Riesgo en Desarrollo: No posee riesgos		**Tiempo Real:** 0,1
Descripción: Al elegir la opción de menú: Reiniciar (Imagen 18), se Reiniciará la pc.		
Observaciones:		
Prototipo de interfaz:  *Figura 20: Menú reiniciar sistema*		

Tabla 12: Descripción del RF11 Reiniciar PC.

2.5.2 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 refiere más bien a las características del mismo como la fiabilidad, tiempo de respuesta y la capacidad de almacenamiento. En fin describen propiedades emergentes del sistema a implantar, aunque presentan gran importancia puesto que el no cumplimiento de un requisito funcional puede provocar la inutilización del sistema.

A continuación se muestran los requisitos no funcionales definidos para la generación del sistema Nova en modo quiosco para el sistema de gestión de acceso a los comedores.

Requerimientos de usabilidad:

- **RnF 1** tipo de usuario final: Un usuario medio (operador del sistema) sin permisos y un usuario avanzado (administrador) con permisos de *root.*
- **RnF 2** tipo de aplicación informática: Sistema operativo.
- **RnF 3** Finalidad: Establecer un sistema operativo seguro para realizar la gestión de acceso a los comedores de la UCI.

Requerimientos de confiabilidad:

- **RnF 4:** Si se pierde la energía en los comedores una vez que inicie el sistema, estará listo para iniciar la aplicación nuevamente.

Requerimientos de eficiencia:

- **RnF 5:** El sistema deberá instalarse como mínimo en 10 minutos.
- **RnF 6:** El disco duro de las computadoras deberá ser de 4GB como mínimo.
- **RnF 7:** La memoria RAM de las computadoras deberá ser de 512MB como mínimo.

Requerimientos de restricciones de diseño:

- **RnF 8:** Se usará el lenguaje de programación Python en su versión 2.7.
- **RnF 9:** Se usará el lenguaje de programación Bash.
- **RnF 10:** Como metodología de desarrollo se usará Nova OpenUp.
- **RnF 11:** Se usará como IDE de desarrollo la aplicación Geany.

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

- **RnF 12:** Se brindará un manual de usuario administrador, con la descripción de las acciones que se realizan en el sistema.

Requerimientos de licencia:

- **RnF 13:** Se implementará bajo la licencia GPL

Requerimientos de interfaces de software:

- **RnF 14:** Como interfaz de software o sistema Anfitrión del sistema a personalizar se usará Nova escritorio 4.0.
- **RnF 15:** El sistema deberá ser empaquetado en un ISO para su distribución a los usuarios.

Requerimientos de seguridad:

- **RnF 16:** Se instalará el antivirus Sav Unix en las computadoras, configurándolo para su actualización automática.
- **RnF 17:** El sistema tendrá integrada la aplicación Likewise, para permitir unir las computadoras al dominio.

2.7 Conclusiones del capítulo

A lo largo de este capítulo, para garantizar una mejor comprensión de las funcionalidades del sistema, se definió la solución propuesta así como el proceso de construcción de un quiosco. Además de los requisitos funcionales y no funcionales definidos por el cliente.

Capítulo 3: Implementación y Pruebas

En los capítulos anteriores se determinaron los elementos teóricos a tener en cuenta para crear una personalización en modo quiosco de la distribución GNU/Linux Nova. Se establecieron las herramientas que se utilizarán, así como los paquetes y aplicaciones con que contará el sistema operativo a crear. Este capítulo se centra en la descripción del proceso de construcción de un quiosco de Nova, además de las pruebas realizadas al mismo.

3.1 Descripción del proceso de construcción

Para mayor comprensión del proceso de construcción del quiosco, se dividirá el mismo en tres fases. Como se planteó en el primer capítulo un quiosco no es más que uno modo específico de personalización, y precisamente el primer paso en la construcción de cualquier personalización de Nova consiste en la instalación del SOB.

3.1.1 Fase 1: Construcción del Sistema Operativo Base

Antes de comenzar la construcción del SOB hay que tener en cuenta algunos requisitos (26):

- Disponer de una ubicación en el sistema de archivos (puede ser una partición o un directorio) en el Sistema Operativo Anfitrión (SOA) que se encuentre en un formato de archivos compatible (en el caso Nova el formato recomendado es Ext4) y cuente con espacio libre suficiente para hospedar el sistema operativo a construir.
- Contar con una conexión directa a un repositorio de aplicaciones informáticas compatibles con Nova, ya que el SOB se construirá a partir de los paquetes disponibles en dicho repositorio.

Primeramente se crea el directorio o partición donde se quiere alojar el SOB, en este caso se creó un directorio en /mnt con el nombre kiosk de la siguiente forma:

```
sudo mkdir /mnt/kiosk
```

Nota: La mayoría de los comandos utilizados requieren permisos administrativos por lo que serán ejecutados con la orden ***"sudo"*** al inicio de cada invocación.

El siguiente paso es instalar debootstrap si no se tiene, con el comando:

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

Luego, mediante la utilización de la herramienta debootstarp se procede a la creación del SOB como sigue:

```
sudo debootstrap --components=principal,extendido 2013 /mnt/kiosk/
```

```
http://nova.f10.uci.cu/nova
```

Se utilizó la opción ***--components*** para que a la hora de la instalación utilizara paquetes de los componentes del repositorio 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 segundos, y si todo sale bien ya se tiene en /mnt/kiosk una distribución básica instalada, solo queda agregar el software y las configuraciones deseadas.

3.1.2 Fase 2: Construcción del Sistema Operativo Final.

- **Configuraciones Iniciales:**

Primeramente se deben realizar algunas configuraciones y ajustes.

Lo primero será montar /dev en la carpeta donde se encuentra el SOB en este caso /mnt/kiosk. El directorio /dev contiene los archivos de dispositivos especiales para todos los dispositivos de hardware (27):

```
sudo mount --bind /dev /mnt/kiosk/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/kiosk/etc/
```

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:

```
sudo chroot /mnt/kiosk/
```

Una vez dentro de este entorno, se tienen privilegios de root, primeramente se procede a montar algunas unidades más:

Se montan los procesos:

```
mount -t proc none /proc
```

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, de aquí su nombre)

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

```
mount -t sysfs none /sys
```

Se monta /dev/pts, este sistema de ficheros vive exclusivamente en la memoria, las entradas en /dev/pts son pseudo-terminales[19] (pty).

```
mount -t devpts none /dev/pts
```

Se exportan algunas variables del sistema:

```
export HOME=/root export LC_ALL=C
```

- **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 específicas. Para la selección de los paquetes con que contará el sistema se tuvo en cuenta que se quiere que este sea ligero, por lo tanto se instaló solo el software necesario. A continuación se muestra el listado de aplicaciones que serán instaladas (Tabla 13), seleccionadas en el capítulo anterior, además de otras necesarias para la ejecución de la aplicación de control de acceso a los comedores:

Paquete	Descripción
xorg	Se encarga de la exhibición
openbox	Gestor de ventana
lightdm-gtk-greeter	Manejador de sesión
openjdk-7-jre	Entorno de ejecución de Java OpenJDK
oblogout	Menú del sistema
gksu	Interfaz gráfica para *su*, permite ejecutar programas como administrador.
lxkb-config	Interfaz para configuración de teclado
network-manager	Administración de red
usb-control-gui	Interfaz gráfica para gestionar dispositivos usb
sav-unix	antivirus
likewise-open	Servicio de autenticación de dominios
linux-image-3.8.0.35-generic	Kernel de Linux.
nova-base	Núcleo mínimo de nova.
casper	Ejecuta un sistema preinstalado *"live"* desde un dispositivo de sólo lectura.
serere-kiosk-terminal	Instalador de Nova en modo texto.

Tabla 13: Listado de aplicaciones a instalar

Para instalar estos paquetes se utiliza el comando siguiente:

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

Ejemplo:

```
apt-get install xorg
```

[19] 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.

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

```
aptitude clean
```

- **Configuraciones del Sistema:**

Esta parte depende mucho de los requerimientos de cada cual, en este caso en particular, se desea que un usuario interactúe únicamente con una aplicación y nada más por lo tanto se procede de la siguiente manera:

Se crea el usuario que interactuará con la aplicación, al cual se le llamó nova, para ello se usa el comando siguiente y se siguen las instrucciones:

```
adduser nova
```

Para garantizar que al iniciar el sistema cargue directamente la sesión del usuario creado, se modifica el fichero ***"lightdm.conf"*** que es el fichero de configuración del gestor de sesión *lightdm* que se encuentra en ***"/etc/lightdm/"***, para ello se procede como sigue:

```
nano /etc/lightdm/lightdm.conf
```

El fichero quedaría de la forma siguiente:

```
[SeatDefaults]

autologin-user=nova

autologin-user-timeout=0

user-session=openbox

greeter-session=lightdm-gtk-greeter
```

También se puede quitar la contraseña del usuario creado, modificando la entrada correspondiente a este usuario en el fichero ***"passwd"*** que se encuentra en **"/etc/"** de modo que quede así:

```
...

nova::1000:1000:Nova:/home/nova:/bin/Bash

...
```

Una vez que se tiene creado el usuario, se procede a la configuración de la sesión de trabajo del mismo, para ello primeramente se debe copiar la aplicación que utilizará el usuario en una ubicación en el sistema de archivos, en este caso se copió en ***"/mnt/kiosk/opt/"***, para ello se utiliza la orden *"cp"* que permite

copiar desde una ubicación origen hacia otra destino, esto se debe hacer fuera del entorno *chroot,* ya que es en el SOA donde se tendrá la aplicación, por lo que se puede hacer uso de una nueva pestaña en la terminal:

```
sudo cp -R <origen> <destino>
```

Ejemplo:

```
sudo cp -R /home/lmd/Escritorio/SCAC /mnt/kiosk/opt/
```

Seguidamente se procede a la configuración de los ficheros del gestor de ventana openbox, para ello se crean las carpetas que contendrán estos archivos, en la carpeta personal del usuario:

```
mkdir -p /home/nova/.config/openbox
```

Este comando crea la carpeta ***".config"*** en la carpeta personal ***"nova"*** y dentro de esta crea otra llamada ***"openbox"***, y es en esta última donde se deben copiar los ficheros de configuración ***"autostart"***, el cual contiene las aplicaciones que se ejecutarán automáticamente al iniciar la sesión y ***"rc.xml"*** que contiene las configuraciones para atajos de teclado, mouse, así como propiedades de ventana para aplicaciones, estos ficheros se encuentran en ***"/etc/xdg/openbox"***:

```
cp /etc/xdg/openbox/{autostart,rc.xml} /home/nova/.config/openbox/
```

Luego se procede a la edición de estos ficheros de configuración, para ello se utiliza la orden "*nano".*

Edición del fichero ***"autostart"***:

```
nano /home/nova/.config/openbox/autostart
```

Quedando así:

```
# DBUS message bus (automount removable devices)

dbus-launch --exit-with-session &

#Executing policykit

/usr/lib/policykit-1-gnome/polkit-gnome-authentication-agent-1 &

#Start nm-applet

nm-applet &

#Fix screen resolution

python /usr/bin/max-resolution.py
```

```
#Start aplication SCAC

java -jar /opt/SCAC/SistemaControlAcceso.jar &
```

Edición del fichero ***“rc.xml”***:

```
nano /home/nova/.config/openbox/rc.xml
```

Se especifica la cantidad de escritorios que estarán disponibles, en este caso solamente uno.

```
...

<desktops>

<number>1</number>

<firstdesk>1</firstdesk>

...

</desktops>

...
```

Se configuran los atajos de teclado para ejecutar el menú:

```
...

<keyboard>

...

<!-- Keybindings para ejecutar oblogout -->

    <keybind key="C-m">

      <action name="Execute">

        <command>menu</command>

      </action>

    </keybind>

</keyboard>

...
```

Se configuran las propiedades de ventana para la aplicación del control de acceso a los comedores, especificando que debe aparecer en pantalla completa:

```
...
<applications>
  <application name="sun-awt-X11-XFramePeer" class="SistemaControlAcceso">
    <fullscreen>yes</fullscreen>
  </application>
</applications>
...
```

Nota: Se deben eliminar de este fichero todas las configuraciones que no sean necesarias, de manera que la sesión esté lo más restringida posible.

"menu" es el nombre del script que ejecuta oblogout, que muestra el menú del sistema.

Existen opciones del menú que solo el administrador puede usar, por lo que para garantizar que se cumpla, se hace uso de "gksu". Para ello se editan algunos ficheros de la herramienta "*oblogout".* En ***"/usr/share/pyshared/oblogout/__init__.py"*** se modifican los comandos que requieren el uso del administrador como se muestra a continuación:

```
...
cmd_contrasena = "gksu -k -u root xterm passwd"
cmd_red = "gksu -k -u root nm-connection-editor"
...
```

Al igual que en la sesión de comandos del fichero de configuración ***"/etc/oblogout.conf"***

```
[commands]
...
Configurar Red = gksu -k -u root nm-connection-editor
Cambiar Contraseña = gksu -k -u root xterm passwd
...
```

Además hay que asegurarse de que el usuario nova tiene permisos para apagar la máquina, para ello se edita ***"/etc/sudoers":***

```
...

nova ALL=NOPASSWD:/sbin/shutdown

...
```

Hasta aquí se tiene preparada la sesión del usuario nova, de manera que inicie automáticamente, cargando al inicio la aplicación de control de acceso. Seguidamente se deben realizar algunas configuraciones más para garantizar que el sistema funcione correctamente.

Se debe modificar el fichero ***"tty1.conf"*** que se encuentra en ***"/etc/init/"***, quedando de la manera siguiente:

```
respawn

exec /bin/login -f root </dev/tty1 > /dev/tty1 2>&1
```

Esto garantiza el auto inicio de sesión del usuario en la primera consola de Terminal (tty1)

Luego se eliminan los ficheros ***"tty3.conf"***, ***"tty4.conf"***, ***"tty5.conf"***, ***"tty6.conf"***, también presentes en ***"/etc/init/"***. Ejemplo:

```
rm /etc/init/tty3.conf
```

Para garantizar que se ejecute el instalador del sistema al arrancar, se debe editar el fichero ***"./bashrc"*** de ***"/home/nova"***, agregando lo siguiente al final:

```
...

clear

sudo serere --terminal
```

Para garantizar que se ejecute el instalador se debe dar permisos en ***"/etc/sudoers"***:

```
...

nova ALL=NOPASSWD:/usr/bin/serere

...
```

Se debe configurar además el antivirus *"sav-unix"* para garantizar su correcta actualización, además de que realice periódicos análisis del disco.

Luego se procede a borrar algunos ficheros que son innecesarios:

```
rm /usr/share/initramfs-tools/scripts/casper-bottom/15autologin

rm /usr/share/initramfs-tools/scripts/casper-bottom/25adduser

rm /etc/udev/rules.d/{70-persistent-net.rules,70-persistent-cd.rules}

rm /etc/mtab
```

Con esto termina el proceso de configuración. Pero antes de pasar a la tercera y última fase, se realiza la restauración del sistema.

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

Para que el SOF creado funcione correctamente cuando se ejecute por sí mismo, se deben restaurar los valores adecuados así como borrar los archivos temporales para disminuir el tamaño del sistema:

```
rm -f /var/lib/dbus/machine-id

rm -f /etc/resolv.conf

rm -rf /tmp/*

rm -rf /tmo/.??*
```

Se desmontan las unidades montadas inicialmente y luego se sale del entorno *chroot* con el comando *"exit"*:

```
umount /proc

umount /sys

umount /dev/pts

exit
```

Y fuera de este entorno se desmonta /dev:

```
sudo umount -l /mnt/kiosk/dev/
```

3.1.3 Fase 3: Empaquetamiento del sistema.

Primeramente se crean las carpetas donde se instalarán los archivos necesarios para la creación de la imagen, 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/kiosk/boot/vmlinuz-3.8.0-35-generic /mnt/iso/casper/vmlinuz

sudo mv /mnt/kiosk/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, 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 su

chroot /mnt/kiosk/ dpkg-query -W --showformat='${Package} ${Version}\n' | tee /mnt/iso/casper/filesystem.manifest
```

Luego se necesita el ***"filesystem.squashfs"***, que no es otra cosa que el sistema en formato *squashfs*, esto puede tardar unos minutos:

```
sudo mksquashfs /mnt/kiosk/ /mnt/iso/casper/filesystem.squashfs

sudo printf $(du -sx --block-size=1 /mnt/kiosk | 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 SCAC 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 Kiosk" > /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 como se muestra a continuación:

```
cd /mnt/iso

sudo mkisofs -r -V "Nova Kiosk" -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-kiosk-2013-i386.iso .
```

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

Partiendo de la idea de que la vulnerabilidad de una aplicación, específicamente de un sistema operativo, se encuentra en el número de componentes que conforman el mismo, 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 en el quiosco con el objetivo de comprobar que se encuentran solamente 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, es decir 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 distribución 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.

3.2 Verificación Funcional

La calidad del software debe lograrse a lo largo de todo el ciclo de vida del desarrollo del sistema, su ejecución debe ser paralela al mismo, 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" (28).

3.2.1 Pruebas de software

La prueba 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 la personalización de GNU/Linux Nova en modo quiosco para el sistema de gestión de acceso a los comedores de la UCI, se definió el nivel de prueba de aceptación, así como el método de caja negra y la técnica manual, que permitieron la ejecución de casos de pruebas.

- **Pruebas de caja negra**

Las pruebas de caja negra son las que se llevan a cabo sobre la interfaz del software en desarrollo. Los métodos de caja negra se centran en los requisitos funcionales del mismo e intentan encontrar errores como funciones incorrectas o ausentes, errores de interfaz, errores en estructuras de datos o en acceso a base de datos externas, errores de rendimiento y errores de inicialización y terminación.

- **Pruebas 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. Son básicamente pruebas funcionales, sobre el sistema completo, y buscan una cobertura de la especificación de requisitos y del manual del usuario. Estas pruebas no se realizan durante el desarrollo, pues sería impresentable de cara al cliente (29).

La experiencia muestra que aún después del más cuidadoso proceso de pruebas por parte del desarrollador, quedan una serie de errores que sólo aparecen cuando el cliente usa el sistema. Es por eso que son necesarias para la correcta validación del producto que se presenta en la investigación.

3.2.2 Casos de pruebas

Los casos de prueba son una serie de pruebas de entrada, condiciones de ejecución y resultados esperados desarrollados para un objetivo en particular, tal como ejecutar una ruta particular de un programa o verificar el cumplimiento con un requerimiento en específico.

Los casos de pruebas forman el fundamento en el cual se diseña y desarrolla el guión de pruebas. La profundidad de probar es proporcional al número de casos de prueba. Existe mayor confianza en la calidad del producto y proceso de pruebas cuando el número de casos de pruebas se incrementa, ya que cada caso de prueba refleja un escenario diferente, una condición o flujo a través del producto (30).

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 1

CPR1: Instalar Sistema Operativo

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

Escenario	Descripción	V1	Respuesta del sistema	Flujo central
EC 1.1 Iniciar Instalador	Inicio automático del instalador al bootear el sistema.	N/A	Mostrar cartel de bienvenida al instalador, escanear el equipo y pedir al usuario entrar nueva contraseña.	Bootear el sistema desde un dispositivo extraíble.
EC 1.2 Insertar contraseña.	Pide una nueva contraseña para el usuario Administrador y valida que tenga la fortaleza requerida.	V	Pedir repetir contraseña.	Insertar la contraseña con la fuerza requerida.
		Fuerte		
		I	Mostrar mensaje de error y volver a pedir contraseña nuevamente desde el inicio.	
		Débil		
EC 1.3 Repetir contraseña	Pide repetir la contraseña y valida que coincida con la anteriormente introducida.	V	Pedir confirmación al usuario para instalar	Insertar contraseña nuevamente para ver si coincide con la anteriormente insertada.
		Coincide		
		I	Mostrar mensaje de error y vuelve a pedir contraseña.	
		No coincide		
EC 1.4 Pedir confirmación	Pide confirmación para proceder con la instalación.	V	Instalar el sistema operativo.	Insertar la letra ´s´ o ´n´ para continuar con la instalación.
		Inserta ´s´		
		I	Salir del programa de instalación.	
		inserta ´n´		

Tabla 14: CPR1 Instalar Sistema Operativo

Caso de Prueba 8:

CPR8: Gestionar dispositivos USB.

Condiciones de ejecución: Acceder al menú del sistema.

Escenario	Descripción	V1	Respuesta del sistema	Flujo central
EC 8.1 Acceder a la herramienta *usbgui*.	Interfaz gráfica para la configuración de dispositivos USB.	N/A	Ejecutar el paquete *usbgui*.	- Acceder al menú del sistema. - Seleccionar el botón configurar USB
EC 8.2 Configurar dispositivo USB.	Autoriza o deniega un dispositivo conectado al ordenador.	N/A	Pedir autenticación del administrador del sistema.	- Seleccionar el dispositivo a configurar. - Dar clic en autorizar o en denegar. - Asignar un identificador. - Dar clic en el botón aceptar
EC 8.3 Autenticación del administrador	Pide contraseña del administrador para verificar el rol del mismo.	V	Realiza la acción del usuario.	Insertar la contraseña de adimnistrador.
		Si		
		I	No realiza ninguna acción.	
		No		

Tabla 15: CPR8 Gestionar Dispositivo USB

Caso de Prueba 9:

CPR9: Iniciar Terminal para la administración.

Condiciones de ejecución: Tener instalado el sistema operativo.

Escenario	Descripción	V1	V2	Respuesta del sistema	Flujo central
EC 9.1 Acceder a la tty2	Terminal habilitada para la administración.	N/A	N/A	Pide autenticarse como administrador.	Presionar la combinación de teclas “Ctrl-Alt-F2”.
EC 9.2 Auntenticar como administrador	Se verifica usuario y contraseña del administrador.	I	I	Mostrar mensaje de error y vuelve a mostrar sección de autenticación.	- Insertar usuario admin. - Insertar contraseña de administrador.
		No usuario admin.	No contraseña admin.		
		I	V		
		No usuario admin.	Si contraseña admin.		
		V	I		
		Si usuario admin.	No contraseña admin.		
		V	V	Autenticación del administrador en la termininal para configurar sistema.	
		Si usuario admin.	Si contraseña admin.		

Tabla 16: CPR9 Iniciar terminal para la administración

3.2.3 Resultados obtenidos

Luego de diseñadas las pruebas del producto se procedió a la implementación y ejecución de las mismas. 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 fallo del 40% del total de pruebas diseñadas. Luego de trabajar durante un período en la solución de las mismas y efectuar la segunda iteración, se arrojó a un 18% de errores en el sistema, que fueron eliminados antes de concluir la aplicación.

Tal como se muestra en la Figura 21, la resolución de los errores detectados en esta etapa permitió el aseguramiento de una mayor calidad en el producto.

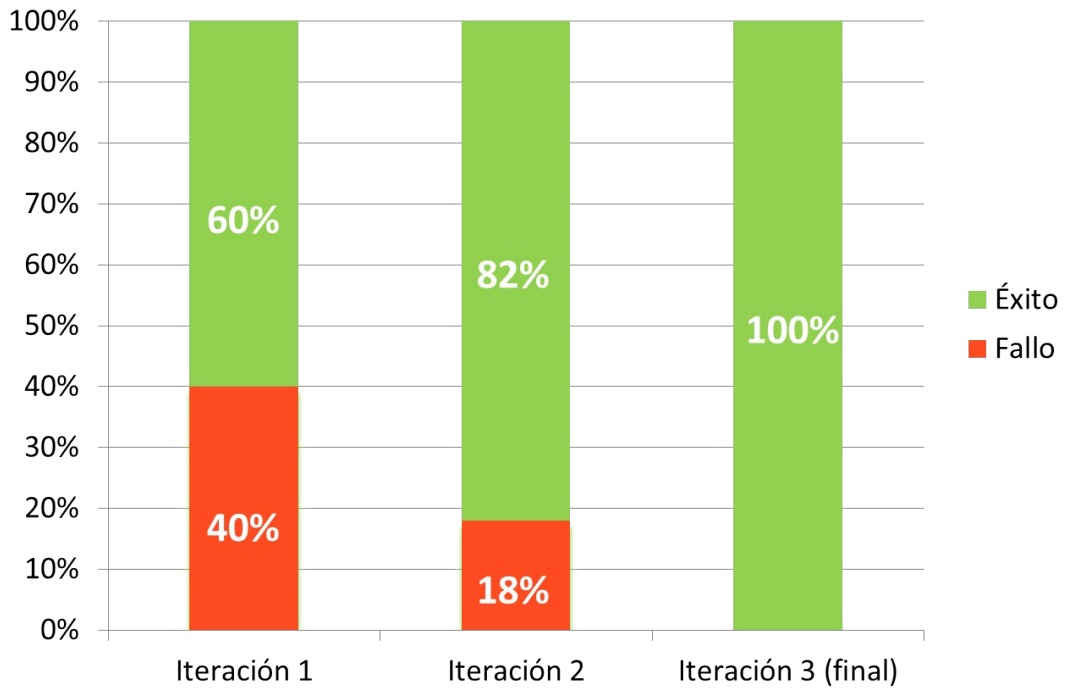


Figura 21: Porcentaje de aceptación y fallas de pruebas al sistema

Luego de concluido el desarrollo del sistema, este fue mostrado al cliente, quién realizó una minuciosa revisión de las funcionalidades del mismo definiendo que el producto estaba listo para ser entregado y puesto en funcionamiento, como consta en el acta de aceptación, reflejada en el anexo III de la presente investigación.

3.3 Conclusiones del Capítulo

En el trayecto del capítulo se describieron detalladamente los pasos llevados a cabo en la implementación de la personalización en modo quiosco de la distribución GNU/Linux Nova. Gracias al proceso seguido y la puesta en práctica de las pruebas realizadas a la solución, se obtuvo una personalización que satisface las necesidades planteadas inicialmente. Este resultado fue validado por el cliente donde se manifestó su satisfacción por el producto final.

Conclusiones Generales

Las personalizaciones en modo quiosco serán cada vez más importantes, por tanto es necesario definir conceptos y procesos por los cuales guiar a los futuros ingenieros informáticos que deseen realizar desarrollos similares. Con la realización de este trabajo se cumplieron los objetivos propuestos:

- Se sentaron las bases teóricas así como la definición de las herramientas, lenguajes de programación y metodología a utilizar, para proceder al análisis y diseño de la solución propuesta.
- Se analizaron los requisitos definidos por el cliente, y se diseñó la solución propuesta, realizando una minuciosa selección de las aplicaciones a incluir en el sistema.
- Al finalizar el ciclo de desarrollo de esta personalización se obtuvo un quiosco de Nova para el control de acceso a los comedores, que cumple con las funcionalidades requeridas, estas funcionalidades fueron validadas por el cliente, quien mostró su satisfacción con el producto.
- Se ejecutaron un conjunto de pruebas que demostraron la calidad del proceso.

También en el transcurso de esta investigación se generaron artefactos de la metodología Nova OpenUp que evidencian un proceso de ingeniería organizado que puede servir de base para investigaciones con características similares a esta.

Recomendaciones

Como resultado de la investigación los autores recomiendan:

- Utilizar el presente trabajo como una guía para realizar construcciones de quioscos similares.
- Continuar la investigación sobre aspectos que no fueron tratados profundamente en el trabajo.
- Los miembros del proyecto Nova pueden realizar cambios para enfocarlo a diferentes entornos de aplicación, utilizando otras tecnologías.

Referencias Bibliográficas

1. JESÚS GONZÁLEZ BARAHONA, Joaquín Seoane Pascual and GREGORIO ROBLES. *Introducción al software libre* [online]. Free Software Foundation, [no date]. 1.2. Available from: http://sunshine.prod.uci.cu/gridfs/sunshine/books/guia_maxima.pdf

2. *Guía cubana para la migración a SWL* [online]. Available from: http://sunshine.prod.uci.cu/gridfs/sunshine/books/guia-cubana-0.32.pdf

3. GERARD BEEKMANS. *Linux From Scratch*. Matthew Burgess and Bruce Dubbs. 2013. si, 7.4.

4. JUAN GUILLERMO LÓPEZ CASTELLANOS. *Serere 3.0. Instalador del sistema operativo Nova GNU/Linux.* [online]. 2012. [Accessed 20 February 2014]. Available from: http://repositorio_institucional.uci.cu/jspui/bitstream/ident/TD_05104_12/1/TD_05104_12.pdf

5. RAÚL GARCÍA NODARSE, Rafael Enrique Menéndez García . *Configuración del sistema operativo GNU/Linux en Modo Kiosko* [online]. 2009. [Accessed 20 February 2014]. Available from: http://repositorio_institucional.uci.cu/jspui/bitstream/ident/TD_2431_09/1/TD_2431_09.pdf

6. BUENAS TAREAS. *Sistemas operativos* [online]. Available from: http://www.buenastareas.com/ensayos/Sistemas-Operativos/3397096.html

7. DRAE. personalizar. *DRAE* [online]. [no date]. Available from: http://lema.rae.es/drae/srv/search?val=personalizar&submit.x=0&submit.y=0

8. OSCAR BERNAL. *Tipos de paquetes en linux/unix* [online]. 2014. Available from: http://www.oscarbernal.net/index.php?/content/view/36/20/

9. KIOSCOS INFORMATICOS. *GESTIÓN DE ENTRADAS, RESERVAS Y CONTROL DE ACCESO* [online]. 12 December 2013. [Accessed 8 February 2014]. Available from: http://www.kioskosinformaticos.com/kioskos/portfolio-items/sistemas-de-control-de-acceso/.

10. BARRY O'DONOVAN. *Administración del sistema de KDE* [online]. 13 July 2012. [Accessed 15 December 2013]. Available from: http://techbase.kde.org/KDE_System_Administration/Kiosk/.

11. UBUNTU LIFE. *Pessulus, Editor de restricciones para Gnome* [online]. 26 July 2008. Word press. [Accessed 15 December 2013]. Available from: http://ubuntulife.wordpress.com/2008/07/26/pessulus-editor-de-restricciones-para-gnome/.

12. UBUNTU. *lockdown editor for GNOME* [online]. 15 July 2013. Canonical LTD. [Accessed 15 December 2013]. Available from: https://launchpad.net/ubuntu/precise/+package/pessulus.

13. CLAUDIO CONCEPCIÓN CERTAD. *Administra perfiles de usuario con Sabayon User Profile Editor* [online]. 1 February 2011. [Accessed 15 December 2013]. Available from: http://fraterneo.blogspot.com/2011/02/administra-perfiles-de-usuario-con.html.

14. ZONA FIREFOX. *R-kiosk convierte a Firefox en el navegador ideal para cabinas y cybercafés* [online]. 10 July 2013. [Accessed 15 January 2013]. Available from: http://www.zonafirefox.net/r-kiosk-convierte-a-firefox-en-el-navegador-ideal-para-cabinas-y-cybercafes/.

15. GREGORIO ESPADAS. *Modo Kiosko en Firefox, Chrome, Opera, Safari e IE* [online]. 27 September 2010. [Accessed 20 February 2014]. Available from: http://gespadas.com/modo-kiosko.

16. ALLAN PIERRA FUENTES. *Conceptualización y Reestructuración Estratégica de la Distribución Cubana de GNU/Linux "Nova."*2011.

17. DEBOOTSTRAP. *Linux man page. Debootstrap* [online]. [Accessed 25 January 2014]. Available from: http://linux.die.net/man/8/debootstrap.

18. SCRATCHBOX. *Scratchbox* [online]. [Accessed 23 March 2014]. Available from: http://www.scratchbox.org/.

19. UNKNOW. *Geany* [online]. 10 July 2008. [Accessed 15 January 2014]. Available from: http://www.geany.org/.

20. DUQUE, Raúl González. *Python para todos.* España : Creative Commons Reconocimien- to 2.5, [no date].

21. GNU BASH. *Gnu Operating Sistem. GNU Bash* [online]. 2 February 2014. [Accessed 15 February 2014]. Available from: http://www.gnu.org/software/bash/bash.html.

22. SOFTWARE.COM.AR. *Visual Paradigm para UML* [online]. 2013. [Accessed 25 December 2014]. Available from: http://www.software.com.ar/visual-paradigm-para-uml.html.

23. SOMMERVILLE. *Ingenieria de Software* [online]. Available from: http://eva.uci.cu/mod/resource/view.php?id=9352

24. ECURED. metodologia agil. *ECURED* [online]. [no date]. [Accessed 4 April 2014]. Available from: http://www.ecured.cu/index.php/Metodolog%C3%ADa_%C3%A1gil#Fuente.

25. YUSLEYDI FERNÁNDEZ DEL MONTE. *Metodología Nova OpenUp*. June 2013. 4.0Metodología para desarrollar la distribución cubana de GNU/Linux Nova

26. DANIEL HERNANDEZ BAHR, Yunier Soler Franco and ALLAN PIERRA FUENTES, Belkis Soler Blanco. *PROCESO DE CREACIÓN DEL SISTEMA OPERATIVO NOVA. ESTUDIO DE CASO: QUIOSCO DE NAVEGACIÓN WEB.*

27. JOANNA OJA. *Guía Para Administradores de Sistemas GNU/Linux* [online]. 2003. [Accessed 15 March 2014]. Available from: http://www.tldp.org/pub/Linux/docs/ldp-archived/system-admin-guide/translations/es/html/

28. HECTOR PÉREZ BARANDA. *SISTEMA DE COMPILACIÓN DISTRIBUIDA DE NOVA*.

29. INGENIERÍA DE SOFTWARE II. *Flujo de trabajo de Prueba.*

30. *Aplicaciones de Ingeniería de Software* [online]. Available from: http://lsc.mxl.uabc.mx/~angelica/Casos%20de%20prueba.pdf

Bibliografía Consultada

- Acobson, I.; Booch, G. y Rumbaugh, J.; “El Proceso Unificado de Desarrollo de software”. 2000. Addison-Wesley.
- Booch, G.: Rumbaugh, J. y Jacobson, I.; “El Lenguaje Unificado de Modelado”. 2000. Addison-Wesley. Capítulo 11
- Pressman, Roger; Ingeniería de software. Un enfoque práctico. 2002. McGraw.Hill/Interamericana de España.
- GNU BASH. *Gnu Operating Sistem. GNU Bash* [online]. 2 February 2014. [Accessed 15 February 2014]. Available from: http://www.gnu.org/software/bash/bash.html
- SOFTWARE.COM.AR. *Visual Paradigm para UML* [online]. 2013. [Accessed 25 December 2014]. Available from: http://www.software.com.ar/visual-paradigm-para-uml.html.
- SOMMERVILLE. *Ingenieria de Software* [online]. Available from: http://eva.uci.cu/mod/resource/view.php?id=9352
- ECURED. metodologia agil. *ECURED* [online]. [no date]. [Accessed 4 April 2014]. Available from: http://www.ecured.cu/index.php/Metodolog%C3%ADa_%C3%A1gil#Fuente.
- YUSLEYDI FERNÁNDEZ DEL MONTE. *Metodología Nova OpenUp*. June 2013. 4.0Metodología para desarrollar la distribución cubana de GNU/Linux Nova
- DANIEL HERNANDEZ BAHR, Yunier Soler Franco and ALLAN PIERRA FUENTES, Belkis Soler Blanco. *PROCESO DE CREACIÓN DEL SISTEMA OPERATIVO NOVA. ESTUDIO DE CASO: QUIOSCO DE NAVEGACIÓN WEB.*
- JOANNA OJA. *Guía Para Administradores de Sistemas GNU/Linux* [online]. 2003. [Accessed 15 March 2014]. Available from: http://www.tldp.org/pub/Linux/docs/ldp-archived/system-admin-guide/translations/es/html/
- Curso de Introducción a GNU/Linux: Historia, Filosofía, Instalación y Conceptos Básicos. El sistema X- Window, entornos de escritorio gráficos. Entornos KDE y Gnomo, [Febrero, 2009]. Disponible en: http://www.ant.org.ar/cursos/curso_intro/x3359.html
- Guia Ubuntu. Entorno de escritorio Xfce (2007), [Febrero, 2009]. Disponible en: http://www.guia-ubuntu.org/index.php?title=XFce
- [Ingelan Networking, 2001] Concepto, solución y aplicaciones de Kioskos de autoservicio, [Enero, 2009]. Disponible en: http://www.kioskos.cl/quees.html
- [ICEWM: El Window Manager] Beneficios de Ice Window Manager, [Febrero, 2009]. Disponible en: http://www.sromero.org/wiki/doku.php/linux:sistema:icewm
- [MasterMagazine, 2004] Definición y significado de Escritorio. MasterMagazine archivo 2004, [Febrero, 2009]. Disponible en: http://www.mastermagazine.info/termino/4920.php
- Metacity. Capítulo 3. Tablero del Escritorio de la Base, tabla consultada, [Marzo, 2009]. Disponible en: http://www.gnome.org/~bmsmith/gconf-docs/es/metacity.html
- Proyecta LIVEBRAND. Uso y ventajas de los Kioskos Interactivos, [Enero, 2009]. Disponible en: http://www.kioscos.com.mx
- Red Hat Enterprise Linux 4: Manual de referencia Capítulo 7. El Sistema X Window. Archivos de configuración del servidor X, [Marzo, 2009]. Disponible en: http://web.mit.edu/rhel-doc/4/RH-DOCS/rhel-rg-es-4/s1-x-server-configuration.html

Anexo I: Procesos que intervienen en el sistema

Proceso	Descripción
rsyslog	Registra logs específicos del sistema de forma selectiva, ya sea en el equipo local o centralizado en un servidor. Este está preparado para leer todos los archivos ***.conf*** del directorio ***"/etc/rsyslog.d./".***
udev	Gestor de dispositivos para núcleos Linux. Se encarga de mantener actualizado el directorio ***"/dev"*** con los dispositivos presentes en el sistema. Gestiona los dispositivos a través de reglas.
upstart-udev-bridge	Demonio de ayuda que recibe que recibe eventos *udev* desde la red y emite equivalentes eventos *upstart.*
modemmanager	Administrador de conexión modem. Demonio que gestiona conexiones banda ancha móvil del sistema para *networkmanager*.
atd	Programa estándar de UNIX que ejecuta programas especificados por el usuario en momentos diferidos programados.
dbus	Bus de mensajes del sistema. Permite que los demonios del sistema y aplicaciones de usuario puedan ejecutarse.
resolvconf	Mantiene la información del sistema sobre nombres de servidores disponibles y gestiona el contenido del archive de configuración ***"resolv.conf"***.
network-manager	Administrador de conexión de red.
cron	Programa estándar de UNIX que ejecuta programas especificados por el usuario periódicamente a la hora programada.
lightdm	Servicio encargado de la exhibición que gestiona los servidores X que se ejecutan en el sistema, proporcionando servicios de logueo y autologueo.
ufw	Cortafuegos diseñado para ser de fácil uso. Utiliza líneas de comandos para configurar las *iptables*.
upstart-socket-bridge	Consulta al demonio *upstart* para la configuración de los trabajos que se inician o se detienen en el evento socket.
tty2	Servicio que mantiene una consola en tty2 desde que se inicia el sistema hasta que se apaga de nuevo.
network-interface	Configuración de dispositivos de red.
network-interface-security	Configuración de la seguridad de los dispositivos de red.

Tabla 17: Procesos que intervienen en el sistema, en ejecución.

Proceso	Descripción
mountall-net	Se encarga de montar sistemas de fichero de red.
passwd	Utilidad para cambiar contraseñas.
rc	Ejecuta el script *rc* del estilo del *Sistem V* (una de las versiones del Sistema UNIX) cuando se cambia entre los niveles de ejecución.
ureadahead-other	Corre un demonio de lectura anticipada que lee datos sobre archivos requeridos durante el arranque y los lee en la caché antes de su uso.
console-setup	Se encarga de ajustar la consola de teclado tan pronto como sea posible para que el administrador pueda interactuar con el sistema durante la comprobación de archivos del sistema.
hwclock-save	Salva el tiempo del reloj del sistema devolviendo al hardware de reloj en el apagado.
failsave	Retardo de arranque a prueba de fallos.
irqbalance	Demonio para equilibrar las interrupciones para el sistema SMP (Multiproceso Simétrico).
plymouth-log	Registro de inicio de descarga a disco. Amortigua mensajes de consola durante el arranque.
mounted-var	Rellena el sistema de archivos ***"/var"*** independiente (si los hay), creando la ejecución y bloqueo de enlaces.
plymouth	Proporciona un proceso de arranque gráfico sin parpadeo. Se basa en la configuración del modo *Kernel* para ajustar la resolución nativa de la pantalla tan pronto como sea posible.
udev-fallbak-graphics	Se encarga de tomar acciones para iniciar gráficos de retorno
control-alt-delete	Manejo de pulsación de tecla de emergencia. Esta tarea se ejecuta cada vez que se pulsa la combinación de teclas control-alt-delete, y lleva a cabo un reinicio seguro de la máquina.
hwclock	Ajusta el reloj del sistema y su zona horaria.
mounted-proc	Soluciona las permanentes en las entradas del sistema de archivos ***"/proc"*** sensibles. Algunos archivos en ***"/proc"*** tienen contenidos sensibles que pueden ser utilizados para ayudar a los atacantes a lanzar explotación del *kernel*. Hacer que estos archivos sólo estén accesibles por *root* reduce ligeramente las posibilidades de este tipo de ataques.
module-init-tools	Carga los módulos del *kernel* especificados en el archivo ***"/etc/modules"***.
setvtrgb	Este trabajo configura los colores de la consola de terminales virtuales.
shutdown	Desencadena un apagado inmediato cuando *upstart* recibe SIGPWR (Fallo de corriente eléctrica (*System V*)).
mountall	Monta el sistema de archivos en el arranque en el orden correcto.
mounted-debugfs	Soluciona permanentes en sistema de archivos ***"/sys/kernel/debug"***.

console	Este servicio mantiene una *getty* en consola desde que se inicia el sistema hasta que se apaga de nuevo.
mounted-run	Rellena el sistema de archivos ***"/run"*** y añade enlaces de compatibilidad a este.
plymouth-stop	Oculta la pantalla de bienvenida.
rcs	Modo de compatibilidad del *Sistem V single-user,* esta tarea maneja el viejo estilo del modo monousuario del SYSTEM V.
wait-for-state	Esperando por estado.
flush-early-job-log	Garantizar la salida del caché de trabajos que terminan antes de que el disco es escribible y se vacían del disco tan pronto como se convierte en escribible.
friendly-recovery	Script de inicio para la recuperación amigable.
rc-sysinit	Ejecuta el viejo script de compatibilidad de inicialización de *System V* y entra en el nivel de ejecución predeterminado cuando haya terminado.
udevtrigger	Dispositivos de conexión en frío. En el momento en que *udevd* comienza, se han perdido todos los eventos de los dispositivos pobladas en ***"/sys"***, esta tarea hace que el *kernel* los vuelva a enviar.
container-detect	Realiza un seguimiento de si *upstart* se está ejecutando en un contenedor.
mounted-dev	Rellena el sistema de archivos ***"/dev"*** de ***"/lib/udev/devices"*** una vez que el sistema de ficheros temporal *mount* está en su lugar.
udev-finish	Guarda los registros de *udev* y reglas de actualización.
hostname	Establece el nombre de *host* del sistema.
mountall-reboot	Si *mountall* existe para indicar que se requiere un reinicio, esto hace el reinicio necesario.
mountall-shell	*Shell* de recuperación para el fallo del sistema de archivos. Si *mountall* existe para indicar que se requiere la recuperación manual, éste inicia el *shell* necesario.
mounted-tmp	Limpia el directorio ***"/tmp"*** cuando no existe como un sistema de archivos temporal.
plymouth-ready	Envía un evento para indicar que *plymouth* esta levantado.
plymouth-splash	Muestra la pantalla de bienvenida.
plymouth-upstart-bridge	Este proceso recibe cambios de estado de *upstart* sobre D-Bus y envía mensajes correspondientes a *plymouth*.
udevmonitor	Inicia la creación inicial de un dispositivo.
dmesg	Esta tarea guarda el registro inicial de mensajes del núcleo.
networking	Configurar dispositivos de red virtuales, hace que los dispositivos de red virtuales que no tienen un objeto de núcleo asociado se inicien en el arranque.
procps	Establece las variables *sysctl* del *kernel* de ***"/etc/sysctl.conf"*** y ***"/etc/sysctl.d"***.

network-interface-container	Emite los eventos que faltan de redes de dispositivo agregado para contenedores. Es necesario en los casos en que se crean los dispositivos antes de que se inicien los contenedores y así no conseguirán un evento *udev*.
ureadahead	Corre un demonio lectura previa que lee los datos acerca de los archivos necesarios durante el arranque y los lee en la caché antes de su uso.

Tabla 18: Procesos que intervienen en el sistema, detenidos.

Anexo II: Casos de pruebas.

Caso de Prueba 2

CPR2: Iniciar el Sistema de Control de Acceso a los comedores (SCAC).

Condiciones de ejecución: Tener instalado el sistema operativo.

Escenario	Descripción	V1	Respuesta del sistema	Flujo central
EC 2.1 Iniciar sistema	Se prende la computadora, iniciándose el sistema operativo.	N/A	Mostrar cartel de sistema Nova, posteriormente se inicia sistema y aparece en pantalla la aplicación de control de Acceso.	- Encender la computadora. - Botear el sistema por el disco duro que lo contiene.
EC 2.2 Iniciar aplicación de control de Acceso	Pide una nueva contraseña para el usuario Administrador y valida que tenga la fortaleza requerida.	N/A	Mostrar panel de inicio de sistema de control de acceso a los comedores.	

Caso de Prueba 3:

CPR3: Acceder al menú del sistema

Condiciones de ejecución: Tener instalado el sistema operativo.

Escenario	Descripción	V1	Respuesta del sistema	Flujo central
EC 3.1 Mostrar menú.	Menú de opciones y configuraciones del sistema.	V	Muestra un menú con las opciones y configuraciones del sistema.	Presionar la combinación de teclas Ctrl+M.
		Presiona Ctrl+M		
		I	No muestra ningún menú en pantalla	
		NO presiona Ctrl+M		

Caso de Prueba 4:

CPR4: Configuración de distribución del teclado.

Condiciones de ejecución: Acceder al menú del sistema

Escenario	Descripción	V1	Respuesta del sistema	Flujo central
EC 4.1 Elegir menú idioma de teclado.	Luego de acceder al menú del sistema, aparecen las diferentes alternativas a elegir, entre las cuales se encuentra cambiar idioma de teclado.	N/A	Muestra panel de cambio de idioma de teclado	Acceder a la opción menú configurar teclado.
EC 4.2 Insertar contraseña de administrador.	Panel donde realizarán los cambios de idioma de teclado del sistema.	N/A	Se cambia a la configuración de teclado definida.	Realizar los cambios de configuración de idioma de teclado.

Caso de Prueba 5:

CPR5: Apagar PC.

Condiciones de ejecución: Acceder al menú del sistema.

Escenario	Descripción	V1	Respuesta del sistema	Flujo central
EC 5.1 Acceder a menú apagar PC.	Luego de acceder al menú del sistema, aparecen las diferentes alternativas a elegir, entre las cuales se encuentra Apagar la PC.	N/A	Se apaga el ordenador.	Acceder a la opción menú Apagar PC.

Caso de Prueba 6:

CPR6: Cambiar contraseña del administrador.

Condiciones de ejecución: Acceder al menú del sistema.

Escenario	Descripción	V1	Respuesta del sistema	Flujo central
EC 6.1 Acceder a menú cambiar contraseña.	Luego de acceder al menú del sistema, aparecen las diferentes alternativas a elegir, entre las cuales se encuentra cambiar contraseña.	N/A	Muestra cuadro de autenticación para verificar usuario administrador.	Acceder a la opción menú cambiar contraseña.
EC 6.1 Insertar contraseña de usuario administrador.	El sistema muestra un cuadro de autenticación de usuario administrador para verificar que sea este solamente.	V	Muestra la terminal donde se podrá cambiar la contraseña.	Insertar contraseña de usuario administrador.
		Contraseña válida		
		I	No muestra nada.	
		Contraseña inválida		

Caso de Prueba 7:

CPR7: Configurar la red.

Condiciones de ejecución: Acceder al menú del sistema.

Escenario	Descripción	V1	Respuesta del sistema	Flujo central
EC 7.1 Acceder a menú configurar red.	Luego de acceder al menú del sistema, aparecen las diferentes alternativas a elegir, entre las cuales se encuentra cambiar contraseña.	N/A	Mostrar panel de autenticación de usuario administrador.	Acceder a la opción menú configurar red.
EC 7.2 Insertar contraseña de administrador.	El sistema muestra un cuadro de autenticación de usuario administrador para verificar que sea este solamente.	V	Muestra la terminal donde se podrá configurar la red del sistema.	Insertar contraseña de usuario administrador.
		Contraseña válida		
		I	No muestra nada.	
		Contraseña inválida		

Caso de Prueba 10:

CPR10: Cancelar Menú del sistema.

Condiciones de ejecución: Acceder al menú del sistema.

Escenario	Descripción	V1	Respuesta del sistema	Flujo central
EC 5.1 Acceder a menú Cancelar.	Luego de acceder al menú del sistema, aparecen las diferentes alternativas a elegir, entre las cuales se encuentra Cancelar Manú.	N/A	Se sale del menú del sistema.	Acceder a la opción menú Cancelar.

Caso de Prueba 11:

CPR11: Reiniciar PC.

Condiciones de ejecución: Acceder al menú del sistema.

Escenario	Descripción	V1	Respuesta del sistema	Flujo central
EC 5.1 Acceder a menú reiniciar PC.	Luego de acceder al menú del sistema, aparecen las diferentes alternativas a elegir, entre las cuales se encuentra Reiniciar la PC.	N/A	Se reinicia el ordenador.	Acceder a la opción menú Reiniciar PC.

Anexo III: Acta de Aceptación del Producto.

Acta de aceptación del cliente.

UCI Universidad de las Ciencias Informáticas

CARTA DE ACEPTACIÓN DEL CLIENTE

El cliente de la investigación desarrollada certifica por este medio que el Trabajo de Diploma, titulado **"Personalización de la distribución GNU/Linux Nova para el control de acceso a los comedores"**, realizado en el **Departamento SO** cumple con los requerimientos trazados inicialmente. Ha pasado satisfactoriamente por las pruebas y está listo para ser utilizado.

Y para que así conste, se firma la presente a los 11 días del mes de junio de 2014.

Ing. César González Hernández
Cliente

Universidad de las Ciencias Informáticas
Carretera a San Antonio Km 2 ½. Torrens. Boyeros. La Habana. Cuba.

Acta de aceptación de alta gerencia.

UCI Universidad de las Ciencias Informáticas | **CARTA DE ACEPTACIÓN DE ALTA GERENCIA**

El Jefe del departamento de Sistemas Operativos (SO) certifica por este medio que el Trabajo de Diploma, titulado **"Personalización de la distribución GNU/Linux Nova para el control de acceso a los comedores"**, realizado en el **Departamento SO** cumple con los requerimientos trazados inicialmente. Ha pasado satisfactoriamente por las pruebas y está listo para ser utilizado.

Y para que así conste, se firma la presente a los 11 días del mes de junio de 2014.

Msc. Allan Pierra Fuentes
Jefe del departamento SO

Universidad de las Ciencias Informáticas
Carretera a San Antonio Km 2 ½. Torrens. Boyeros. La Habana. Cuba.