Universidad de las Ciencias Informáticas

Facultad 1

Módulo para la administración de clientes ligeros en Nova para Servidores

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

Autor: Yannier Peña Escalona

Tutor: Ing. Abel Fírvida Donestévez

Co-Tutor: Ing. Haniel Cáceres Navarro

Consultantes: Ing. Ernesto Puente Fuente

Ing. Eugenio Rosales Rosa

La Habana, junio del 2013

Declaración de autoría

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

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

_____

Firma del Autor

Yannier Peña Escalona

_____	_____
Firma del Tutor	Firma del Co-Tutor
Ing. Abel Fírvida Donestévez	Ing. Haniel Cáceres Navarro

"...¿Y qué juventud queremos? ¿Queremos, acaso, una juventud que simplemente se concrete a oír y a repetir? ¡No! Queremos una juventud que piense. ¿Una juventud, acaso, que sea revolucionaria por imitarnos a nosotros? ¡No!, sino una juventud que aprenda por sí misma a ser revolucionaria, una juventud que se convenza a sí misma, una juventud que desarrolle plenamente su pensamiento..."

Agradecimientos

El primero y el más grande de mis agradecimientos es para mis padres. Gracias por siempre confiar en mí y darme su apoyo incondicional en cada momento difícil. Gracias a ustedes hoy puedo decir.

A toda mi familia por el apoyo que me han brindado durante toda mi vida de estudiante, especialmente en estos años de universidad: mi hermana Aliatnis, mis abuelos, mis tíos y tías.

A mi prima Annia, que durante estos cinco años ha sido una segunda madre. Gracias por todos los consejos y por darme ánimo en cada momento en el que el mundo llegó a estar patas arriba.

A mis tutores Abel y Haniel, gracias por todas las horas que dedicaron a esta investigación, que no es solo mía, también es de ustedes.

A Mijail Hurtado, Ernesto Puente y Abel Fírvida por todo el conocimiento que me han trasmitido. A ustedes les debo mucho de lo que sé hoy en día, ustedes son mis padres si de informática se trata.

A todos mis amigos, los de la UCI y los de Las Tunas, en especial a: Yami, Yisel, René, Yasel, Ugarte, Yordanis, Keimer, Luis Daniel, Fernando, Omar, Salgado, Luis Carlos, Barban, Marlen, Alfonso, Jose Carlos, El chino, Javier, Tomás, Gerardo, Rolando, Arlen, Mailen, Yosleydi, Alex, Lazaro, Alfredo. Sin ustedes la vida en la universidad hubiese sido realmente difícil.

A todas aquellas personas que de una forma u otra han tenido que ver en algún momento con mi formación como profesional.

Al equipo de trabajo del proyecto Nova, el Migue, Daniel, Allan, Edito, Raidel, a todos de forma general.

Dedicatoria

A mis padres que día a día lo dieron todo porque este momento se hiciera realidad, a ustedes que siempre me apoyaron, que confiaron en mí, a ustedes va dedicado este trabajo.

A mi mamá, por comprenderme y darme apoyo incluso en los momentos más duros de toda la carrera universitaria. Te quiero.

A mi papá, por siempre haber sido un ejemplo que me ha ayudado a seguir. Te admiro y te quiero.

A mi hermana, por siempre haber sido la luz que me guía en cada paso que doy.

A mi abuelo, sé que donde quiera que estés en este momento te sientes mucho más orgulloso de mí, no sabes cuánto deseo en este momento que estés aquí.

A mis abuelas, siempre tan preocupadas y pendientes a mis estudios.

A mis tíos: Albertico y Daicy, por siempre estar presentes.

A mis primitas: Quelín y Elisha. Estudien mucho, para que un día yo me sienta orgulloso de ustedes.

Resumen

El presente trabajo de diploma titulado “Módulo para la administración de clientes ligeros en Nova para servidores” contiene un estudio realizado a aplicaciones de administración de clientes ligeros en distribuciones GNU/Linux, centrando la atención en la distribución cubana de GNU/Linux Nova, específicamente en la versión para servidores. Se describen las herramientas seleccionadas, así como la metodología utilizada para desarrollar la aplicación.

Con el desarrollo de la solución propuesta se obtuvo un módulo para la administración de clientes ligeros capaz de hacer funcionar correctamente a dichas computadoras, además se logró una mayor escalabilidad y estabilidad en el funcionamiento del servidor. De esta forma se integró a Nova para servidores el conocimiento obtenido durante años por el equipo de desarrollo de Nova en el ámbito de clientes ligeros.

La aplicación se encuentra funcional en una versión estable y está siendo probada en el Ministerio de la Informática y las Comunicaciones (MIC) con 56 clientes ligeros. Además fue expuesta en la feria internacional Informática Habana 2013. También alcanzó la categoría de relevante en la Jornada Científica Estudiantil a nivel de facultad.

Palabras clave: Clientes ligeros, Empresa cubana, Módulo, Nova para servidores, Sistema Operativo (SO).

Índice de Contenido

Índice de figuras

Índice de tablas

Introducción

Durante más de 50 años el bloqueo económico y financiero impuesto por los Estados Unidos de América a Cuba ha frenado su desarrollo económico y social. La industria de la informática y las comunicaciones no está a salvo de estas afectaciones. El proceso de informatización en Cuba ha sido afectado entre otras agravantes, por la carente y difícil adquisición de equipos y dispositivos informáticos de punta, el envejecimiento tecnológico, la poca planificación de la caducidad programada, los cortes de energía y el maltrato que reciben los activos informáticos en manos de los usuarios.

Las afectaciones existentes hasta el momento en el proceso de informatización del país provocaron que el gobierno cubano se lanzara en la búsqueda de nuevas alternativas para el mismo. Se puede decir que en ese entonces: *"El modelo de informatización de la sociedad cubana, basado en máquinas convencionales, mostró sus debilidades: por ser poco eficaz, limitado y sobre todo muy costoso."* [1] Convencidos de estas limitaciones se comenzaron a buscar alternativas, entre las que se puede encontrar el Proyecto de Desarrollo de Clientes Ligeros aplicados a la Informatización de la Sociedad.

Los clientes ligeros están diseñados para ser utilizados en una arquitectura *cliente-servidor* capaz de realizar el procesamiento de las estaciones de trabajo de forma centralizada; las cuales se convierten en una interfaz de entrada y salida de cara al usuario y en constante comunicación con el servidor.

Entre las principales características de los clientes ligeros destacan: el no poseer disco duro (HDD) local, hacer un menor uso de la memoria aleatoria y consumo de energía eléctrica, generar menos calor (en favor de los equipos de climatización) y ser más baratos que una máquina convencional.

Durante los últimos cinco años el uso de clientes ligeros en el país se ha convertido en una solución factible y económica. Estas computadoras son ensambladas y comercializadas por la empresa Gente de Mérito (GEDEME) en una relación de 10 estaciones de trabajo por servidor.

Desde el arribo a Cuba de la primera computadora sin disco, en la distribución cubana de GNU/Linux (Nova) se ha estado buscando una solución que permita la fácil administración e implementación de esta

tecnología.

En 2010 la versión para servidores de Nova adoptó a Zentyal[1] como su herramienta principal para la configuración de servicios. Esta provee una interfaz intuitiva para la administración, además brinda un entorno de desarrollo unificado y está soportado por una comunidad de desarrollo de la cual forma parte el equipo de desarrolladores de Nova para servidores.

Zentyal provee un módulo para la administración de clientes ligeros capaz de integrarse con otros servicios que brinda la plataforma como *Dynamic Host Configuration Protocol (DHCP)* y *Lightweight Directory Access Protocol (LDAP)*. Este módulo está diseñado para funcionar con una conexión directa a Internet y con un máximo de 20 estaciones por servidor, lo cual tiene sus limitantes en el ambiente de trabajo de la empresa cubana [2], en el que existe sobrecarga de estaciones por servidor y un escaso (o a veces inexistente) ancho de banda. [3] Partiendo de esta **situación problemática** se define como **problema científico** para la presente investigación: ¿Cómo adaptar el módulo de clientes ligeros de Zentyal a condiciones de sobrecarga de estaciones por servidor y escaso ancho de banda?

Planteando como **objeto de estudio** las herramientas de administración de clientes ligeros de distribuciones GNU/Linux, ubicando como **campo de acción** las herramientas de administración de clientes ligeros en Nova para servidores.

Idea a defender: proveer a Nova para servidores una herramienta para la administración de clientes ligeros, que tenga en cuenta las exigencias del entorno existente en la empresa cubana, podría aumentar la asimilación de este SO por los administradores de redes del país.

Se define como **objetivo general** de la presente investigación: adaptar el módulo de Zentyal para la administración de clientes ligeros a las condiciones de sobrecarga de estaciones por servidor y escaso ancho de banda presentes en las empresas cubanas.

Para dar cumplimiento al objetivo general se han definido los siguientes **objetivos específicos**:

1. Sistematizar el funcionamiento de las tecnologías para la administración de clientes ligeros.
2. Definir las tecnologías más adecuadas para la administración de clientes ligeros en el entorno informático de la empresa cubana.

[1] Herramienta de administración de servicios para pequeñas y medianas empresas.

3. Modificar una herramienta capaz de administrar computadoras sin disco para que utilice las tecnologías seleccionadas.

4. Realizar pruebas para validar la solución propuesta.

Entre los **posibles resultados** se espera proveer a Nova para servidores una interfaz para la administración de servidores que permita la conexión de al menos 40 clientes por cada servidor, que se pueda configurar desde cualquier empresa que tenga un repositorio de Nova o acceso a uno de los que están públicos nacionalmente.

Para dar solución a los objetivos trazados en la presente investigación se hará uso de los siguientes **métodos científicos**:

- **Métodos Teóricos**:
  - **Histórico-Lógico**: mediante este método se realiza una revisión histórica y análisis de la trayectoria del desarrollo de aplicaciones similares, fundamentalmente se tendrán en cuenta las soluciones desarrolladas por el proyecto Nova y la solución implementada por Zentyal, para tomar una posición al respecto en cuanto a características y funcionalidades.
  - **Inductivo-Deductivo**: este método se utiliza con la finalidad de retroalimentarse sobre el funcionamiento de las herramientas existentes (OSplugger y el módulo de Zentyal) y a partir de ahí llegar a conclusiones sobre cómo lograr el funcionamiento según las necesidades existentes en el entorno de la empresa cubana.
- **Métodos Empíricos**:
  - **Observación**: permitirá obtener un registro visual del comportamiento de algunas herramientas que auxilian el desarrollo de esta aplicación.
  - **Entrevista**: permitirá obtener información referente a las experiencias sobre el uso de clientes ligeros en el país, así como datos estadísticos en cuanto a la cantidad de clientes ligeros existentes en el territorio nacional.

El presente trabajo de diploma está compuesto por un resumen, una introducción, tres capítulos, conclusiones, recomendaciones, así como las referencias bibliográficas, bibliografía y un glosario de términos, donde se da cumplimiento a los objetivos planteados en el trabajo. A continuación se describe los principales aspectos abordados en cada uno de los capítulos:

Capítulo 1. Fundamentación teórica de la aplicación: se realiza un estudio de las herramientas referente al tema de investigación. De esto se deriva un estudio sobre los clientes ligeros en Cuba, la llegada al país de las primeras estaciones ligeras, cantidad existente en la actualidad, así como el software que se usa en la administración de las mismas. Se abordarán las primeras soluciones creadas en el seno del proyecto Nova, y además las ventajas y desventajas de OSplugger.

Capítulo 2. Propuesta de solución: se realiza la captura de requisitos del sistema y basado en estos, se crea un flujo de actividades para describir el proceso de funcionamiento de la aplicación. Se realiza un análisis de los componentes existentes para su utilización. Además de hacer la representación del modelo de clases del diseño, se describe la propuesta de solución.

Capítulo 3. Implementación y pruebas: se muestran los modelos de datos e implementación con sus respectivos diagramas de clases persistentes, de componentes y estándares de codificación a emplear en la solución propuesta. Se realizan las pruebas de aceptación de requisitos correspondientes a cada iteración durante el desarrollo de la herramienta propuesta y se muestran los resultados esperados de las mismas.

Capítulo 1

Capítulo I: Fundamentación teórica de la aplicación

En este capítulo se abordarán los elementos fundamentales para comprender el por qué de la necesidad de construir la aplicación propuesta. Se definirán algunos conceptos, se brindarán datos estadísticos y características que servirán de apoyo para el conocimiento del tema. Además se plasmará la metodología de desarrollo, se hará referencia al lenguaje de programación a utilizarse, herramienta de desarrollo y las herramientas de modelado.

1.1. Clientes ligeros en Cuba

El uso de las computadoras para muchas personas se limita a funcionalidades básicas tales como: abrir la cuenta de correo electrónico, navegar por Internet o tan solo escribir en un editor de texto. Por lo que en muchos sectores empresariales el uso de computadoras convencionales con configuraciones completas no son explotadas al máximo. Por tales motivos se hizo necesaria la simplificación, no solo de *software*, sino también de *hardware*, así se lograría que las máquinas fuesen menos costosas y más amigables a los usuarios.

Durante el año 2008, el proceso de informatización de la sociedad cubana arribó a una nueva fase con la llegada a la isla de los primeros clientes ligeros. En entrevista realizada al Ingeniero Fernando Fernández Fernández (Anexo I) especialista principal del departamento comercial de la empresa GEDEME brindaba los datos que se exponen en la Tabla 1.1 y que están representados gráficamente en la Figura 1.1.

Año	Cantidad	Configuración
2008	100 000	MB Intel DG201GYL
2011	1500	MB IPXV de Asus
2012	6000	MB IPXV de Asus
2013	3000	MB IPXV de Asus

Tabla 1.1: Cantidad de clientes ligeros importados y ensamblados en Cuba.

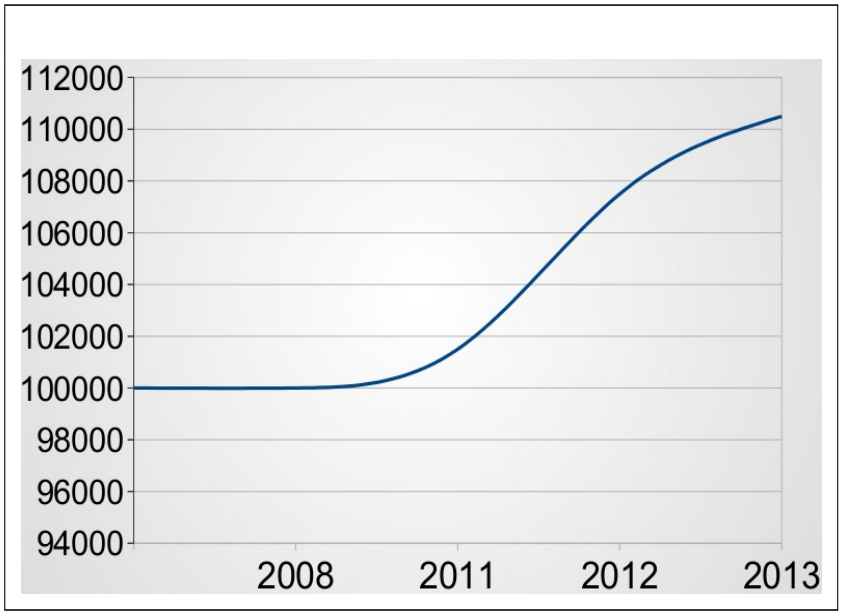

Figura 1.1: Gráfica que representa el aumento de clientes ligeros en Cuba.

Como se puede observar en la gráfica, existe una tendencia al aumento paulatino de la utilización de este tipo de ordenadores en el territorio nacional.

1.2. Software

Los clientes ligeros llegan para solucionar el problema de *hardware* existente en la pequeña y mediana empresa cubana, pero existía una dificultad: para hacer funcionar estas estaciones se hacía necesario el uso de un *software* que gestionara su arranque y posterior funcionamiento. En un inicio BXP-Ardence fue la

solución a la cual se acudió, pero este *software* es privativo y por causas del bloqueo económico y financiero impuesto por Estados Unidos de América al país su licencia no tiene vigencia; además, al haber sido desarrollado por la NASA trae consigo problemas de seguridad, lo que constituye uno de los aspectos fundamentales a resolver en el proceso de informatización de la sociedad. Sumado a lo anterior, no existía un equipo de soporte dedicado a solucionar los problemas que surgen durante el funcionamiento de la aplicación, lo cual obligó a salir en busca de nuevas soluciones que sustituyeran el uso de BXP-Ardence.

1.2.1. Linux Terminal Server Project. LTSP

En la búsqueda de nuevas alternativas para sustituir el uso de BXP-Ardence, LTSP se convierte en una solución factible para la administración de clientes ligeros en SO GNU/Linux. Su desarrollo fue iniciado en 1999 y su última versión estable es la 5.0 liberada el 10 de marzo del 2007. Es ideal para ser utilizado en entornos de trabajo tales como: aulas, gobierno y los negocios. En la actualidad distribuciones de GNU/Linux con mucho prestigio a nivel internacional usan este *software* para la administración de clientes ligeros, tales son los casos de Debian y Ubuntu. [4]

- **Características**

  Está constituido por un conjunto de aplicaciones servidoras desarrolladas en el lenguaje de programación *Bash*. Para garantizar el funcionamiento de los clientes ligeros hace uso de un grupo de tecnologías, entre las que se encuentran: *DHCP, PXE, TFTP* y el protocolo Network Block Device (NBD) para compartir las imágenes a los clientes, aunque también soporta el protocolo Network File System (NFS) para realizar esta acción, y hasta la salida de su última versión estable usaba este protocolo para la compartición de las imágenes. Consta de un administrador de pantalla de acceso en el servidor GNU/Linux específicamente desarrollado por dicho proyecto: *LTSP Display Manager (LDM).* La configuración de las sesiones en diferentes clientes ligeros está facilitada por el uso del fichero de configuración **lts.conf**, mediante el cual se establecen diferentes opciones que los clientes usan en el momento de inicio o durante el funcionamiento. [4]

- **Ventajas**

  1. **Instalación:** fácil de instalar, solo es necesario instalar **ltsp-server-standalone**.

2. **Administración:** hace uso de varias funcionalidades que garantizan una fácil administración de imágenes, entre las que se pueden encontrar: **ltsp-build-client**, **ltsp-chroot**, **ltsp-update-image** **ltsp-update-kernels**, **ltsp-update-sshkeys**.
3. **Licencia:** es una herramienta de código abierto, está distribuida bajo los términos de la *Licencia Pública General (GPL, por sus siglas en inglés)*. [4]

- **Desventajas**

1. **Sesión de usuario:** está diseñado para hacer uso de un servidor de sesiones para los usuarios, lo cual provoca sobrecarga si no se cuenta con un servidor de buenas prestaciones. Aunque es válido aclarar que en la versión 5.0 se incluyó una herramienta llamada **localapps** la cual permite ejecutar algunas aplicaciones de forma local.
2. **Clientes:** las prestaciones del servidor son explotadas al máximo, no así las de los clientes.
3. **Seguridad:** LTSP hace uso del protocolo NBD para la compartición de las imágenes, el uso de este protocolo trae consigo problemas de seguridad, al no contar con un mecanismo de autenticación, solo se limita a comprobar la dirección ip del cliente. Para una persona con conocimientos básicos de informática sería fácil montar los dispositivos compartidos por NBD.

- **Experiencias en Cuba**

El uso de LTSP en las instituciones cubanas, si bien no se puede decir que ha tenido grandes éxitos sí ha alcanzado resultados satisfactorios. A continuación se exponen algunas experiencias obtenidas en las que se puede observar la necesidad de realizar restricciones para lograr un funcionamiento aceptable:

En el Archivo Histórico de Camagüey el uso de LTSP ha tenido buenos resultados. En entrevista realizada al administrador de redes de esa institución (Anexo II), técnico medio en informática Mario Tomas Martínez Gómez, decía que ha llegado a administrar hasta 40 clientes ligeros haciendo uso de un servidor que presenta 4 GB de memoria RAM, un board Intel s3000ah, microprocesador Intel Xeon a 2.4 Ghz, y 2 HDD de 160 GB. Para lograrlo los clientes realizan procesamiento en el servidor y con restricciones que impiden ejecutar videos, emular aplicaciones de Windows[1] y velar por las páginas

[1] Se refiere a las aplicaciones nativas del SO Windows de Microsoft.

web que contengan flash cuando se navega por Internet, pues estas sobrecargan el procesamiento del servidor.

En la UCI, específicamente en la biblioteca, se cuenta con un total de 9 clientes ligeros funcionando con LTSP y un servidor que presenta 2 GB de memoria RAM, un microprocesador Intel Pentium4 CPU 3.00 GHz x2 y un HDD de 160 GB. Este pequeño entorno de clientes ligeros ha presentado inestabilidad en su funcionamiento, procesamiento lento y dificultades en el inicio del SO en los clientes.

Por lo anteriormente expuesto se puede apreciar que LTSP es un proyecto que cuenta con varias funcionalidades que facilitan a los administradores de una red de clientes ligeros gestionarlos de forma fácil. Por otro lado, su característica fundamental de realizar procesamiento en el servidor, trae consigo la necesidad de restringir en gran medida el uso de las estaciones para obtener un funcionamiento aceptable.

1.3. Herramientas de administración de clientes ligeros en Nova

La llegada al país de los primeros clientes ligeros y la necesidad de buscar una solución que fuese capaz de hacer uso de estos en una mayor proporción por servidores y elimine las restricciones que implica el uso de LTSP, abre una nueva línea de investigaciones, para la cual el proyecto Nova es designado. Esta nueva tarea dio inicio a un reto en la línea productiva del equipo de desarrollo: la búsqueda de nuevas soluciones que a la vez fuesen más factibles que las existentes en el mercado. A partir de entonces el trabajo duro y la dedicación del equipo de desarrolladores dio sus primeros frutos.

1.3.1. Uso de Clientes Ligeros en las Fuerzas Armadas Revolucionarias y en el Ministerio del Interior

Debido a la ausencia de una tecnología que cumpliera con los requerimientos necesarios para hacer uso de los clientes ligeros en las Fuerzas Armadas Revolucionarias y el Ministerio del Interior, se le encomienda la tarea a Nova de crear un sistema informático capaz de hacer funcionar, con *software* libre, este pequeño escenario de servidores y máquinas sin disco.

Ante tal desafío se logró una primera solución basada en *Gentoo*, la cual hacía uso del *Initramfs* de

Gentoo y se utilizaba el protocolo NFS con solo lectura para la compartición de las imágenes. Dicha solución fue difícil de implementar, pues se requería de pasos complejos en el momento de realizar el despliegue.

Esta situación conllevó a que se le agregara el soporte de NFS al *Initramfs* de Nova. Con este nuevo paso se le brindaba la posibilidad de hacer cambios persistentes de forma remota a los clientes. Pero el mismo no fue suficiente, una sola estación ligera podía guardar sus cambios en el servidor pero más de una a la vez provocaba conflictos en el SO. Luego se redirigió la escritura de los cambios a la RAM de las computadoras clientes, permitiendo así la utilización de más estaciones ligeras ligadas a un solo servidor.

Esta solución fue expuesta con máquinas sin disco en el Palacio de las Convenciones y PABEXPO en la Habana, mientras se desarrollaba la "Convención de Informática 2009". [1]

1.3.2. Osplugger

Después de lo expuesto en la "Convención de Informática 2009" se incrementó el número de instituciones interesadas en dicha solución pero aún estaba lejos de ser un producto terminado. Se hacía muy difícil y engorroso para una persona realizar cada paso de configuración en el servidor. Había llegado el momento de crear una herramienta informática que fuese capaz de automatizar las tareas de configuración. Un tiempo después surge Cacique, desarrollado sobre el lenguaje de programación Bash, el cual automatizaba parte del trabajo, pero continuaba estando lejos de ser usable.

Con Cacique se dieron los primeros pasos a lo que luego pasaría a ser OSplugger. Se pasó a una arquitectura basada en *Plug-in*, se agregaron nuevos soportes para NBD, *TFTP, GPXE, Syslinux*, COW[2] a nivel de archivos y de bloques. De esta forma se iniciaba el desarrollo de una aplicación que reuniría el conocimiento adquirido durante años.

Objetivos Generales

El desarrollo de OSplugger trazó como objetivos fundamentales:

1. Integrar los beneficios que ofrece tanto BXP-Ardence como LTSP. Además brindar soporte para los más de 100 000 clientes ligeros existentes en el país.

[2]Copias virtuales de la imagen.

2. Brindar un mejor aprovechamiento de los recursos de *hardware* existentes en la mayoría de las áreas estatales del país.

3. Asimilar el prototipo de cliente ligero cubano con la *arquitectura ARM* desarrollada conjuntamente por el Grupo de la Electrónica – GEDEME y el ICID[3]. [1]

Filosofía

OSplugger fue diseñado para lograr su máximo rendimiento en un entorno con más de 15 clientes ligeros, en dependencia de las prestaciones del servidor y la velocidad de la red en la que se implemente. Parte de la idea de mantener el mayor control sobre los clientes conectados a él y sobre las diferentes fases por las que transita el proceso de arranque de los mismos:

1. Virtualización de las imágenes, con el uso de las COW se optimiza el acceso a disco así como el ahorro de espacio en el mismo.

2. Uso de un gestor de arranque interactivo. Permitiendo al usuario seleccionar una imagen asignada.

3. La información de cada cliente registrada en el servidor es almacenada en una base de datos. (mac, ip, puerta de enlace, dirección de difusión).

4. El administrador puede determinar que SO será usado por un cliente. Además de poder determinar si un cliente puede o no iniciar. [5]

Estructura

Para realizar un trabajo más organizado y facilitar a la comunidad el desarrollo de futuros módulos, OSplugger usa una estructura de *Plug-in*. El núcleo de OSplugger se llama **osplugger**, el cual sin la presencia de los plugins está desprovisto de funcionamiento. Los plugins son nombrados con el formato **<nombre del pluggin>-osplugger-plugin**.

Hasta el momento la única forma de administrar OSplugger es mediante una interfaz visual desarrollada en *QT* llamada **osplugger-qt4-gui**.

[3] Instituto Central de Investigación Digital.

La información referente a los clientes, así como la de las imágenes que se comparten por la red es almacenada en una base de datos *postgresql*, que puede ser administrada de forma local o remota y su forma de acceso puede ser configurable mediante el archivo **"/etc/osplugger/service.conf"**. [5]

Despliegue. Ventajas y Desventajas

El primer entorno de pruebas de OSplugger fue un reto para los desarrolladores y para el proyecto en sí. El día 14 de noviembre de 2011 en el MIC, OSplugger llegó a registrar 50 clientes ligeros, cifra que luego se extendiera hasta 64. Haciendo uso de un servidor DELL con 2 HDD SCSI 74 GB en RAID-0, 2 GB de RAM y un microprocesador Intel Xeon a 2.4 Ghz. La velocidad de la red estaba regida por la utilización de un *Switch* a 1 Gbit. [5]

Aunque si bien el llegar a funcionar hasta con 64 clientes ligeros había superado las expectativas del equipo de desarrollo, muy pronto comenzaron a surgir los primeros problemas; no solo en el área de despliegue, en otras áreas e instituciones interesadas en el servicio de OSplugger era necesaria la presencia de especialistas para poder realizar la instalación y configuración. Las experiencias vividas en estos escenarios dieron origen a una lista de ventajas y desventajas a tener en cuenta en nuevas versiones.

- Ventajas

    1. **Administración:** pone en manos de los administradores la facilidad de mantener el control sobre el funcionamiento de los clientes ligeros.
    2. **Escalabilidad:** permite hacer uso de una mayor cantidad de clientes por servidor.
    3. **Registro automático de clientes:** realiza un registro automático de clientes, cuya información es almacenada en una base de datos postgresql, la cual puede ser administrada con facilidad.
    4. **Interacción con los usuarios:** muestra un menú interactivo en la primera fase de inicio de los clientes, permitiendo al usuario escoger el SO por el que desea iniciar.
    5. **Personalización:** los cambios realizados por los usuarios en sus sesiones son persistentes en el tiempo, lo cual permite la personalización del entorno de trabajo.
    6. **Estructura:** como se expuso anteriormente OSplugger presenta una estructura basada en plugin, lo cual propicia el desarrollo colaborativo.

- Desventajas

1. **Uso en disco:** el uso de las COW con permiso de rw[4] hace que un cliente pueda escribir cualquier cantidad de información en el disco duro del servidor. Por lo cual se necesita de gran cantidad de espacio en disco para un funcionamiento correcto y estable.
2. **Seguridad:** se hace uso del protocolo NBD para la compartición de las imágenes, la utilización de este protocolo trae consigo problemas de seguridad, al no poseer un mecanismo de autenticación, solo se limita a comprobar la dirección ip del cliente. Para una persona con conocimientos básicos de informática le sería fácil montar los dispositivos compartidos por NBD.
3. **Integración:** la interfaz gráfica de administración (y la única que existe hasta el momento) depende completamente de las librerías gráficas de *QT*. Lo cual obliga a instalar un gran número de dependencias en el servidor y limita la integración con Nova para servidores.
4. **Automatización:** el proceso de instalación aún es muy complejo. Además no existe un mecanismo de creación de las imágenes que son usadas por los clientes ligeros, lo cual crea una dependencia innecesaria de los especialistas.
5. **Usabilidad:** se hace necesaria la presencia de especialistas en las diferentes áreas de despliegue mientras se desarrolla la instalación.
6. **Estabilidad:** al conectarse un cliente al servidor de OSplugger, el mismo levanta un hilo de ejecución de NBD y así para cada uno de los que se conectan. Cuando uno de esos clientes es reiniciado el hilo que se levantó queda colgado o huérfano, esto ha causado que el servidor colapse.
7. **Actualización de imágenes:** cuando se realizan cambios en las imágenes exportadas y existe algún proceso huérfano la imagen se corrompe.
8. **Portabilidad:** aún no está portado a otras plataformas.

OSplugger logró realizar un mayor uso de clientes ligeros en proporción con un servidor. Brinda al administrador la posibilidad de mantener un control absoluto sobre los clientes que se conectan a él y

[4] Del inglés read/write - lectura/escritura

además permite al usuario realizar configuraciones persistentes en sus sesiones. Sin dudas son cualidades que agradan a la vista de los administradores y usuarios finales. Pero aun así se continúan presentando problemas en la configuración inicial del servidor, sigue siendo muy compleja y se necesitan realizar varios pasos para llegar a un correcto funcionamiento, el cual aún es inestable. Por ello fue necesaria la búsqueda de alternativas donde haciendo uso del conocimiento adquirido en estos primeros años diera al traste con una solución que fuese fácil de configurar, estable, y a la vez capaz de adaptarse al entorno tecnológico de la empresa cubana.

1.4. Nova Servidores

En el año 2010 la versión de Nova para servidores adopta a Zentyal como plataforma para la configuración y administración de servicios telemáticos. Lo que trae consigo ventajas, puesto que Zentyal cuenta con más de 30 módulos que garantizan un correcto funcionamiento de estos servicios. Es una herramienta de código abierto lo cual proporciona seguridad y brinda soberanía-tecnológica.

Dentro de los tantos módulos que brinda Zentyal se puede encontrar uno para la administración de clientes ligeros.

1.4.1. Zentyal

Zentyal es una plataforma de red unificada para las **PYMEs**[5], conocida anteriormente con el nombre de **eBox Platform**. Puede actuar gestionando la infraestructura de red, como puerta de enlace a Internet (Gateway), gestionando las amenazas de seguridad, como servidor de oficina, como servidor de comunicaciones unificadas o una combinación de estas. Incluye, además un marco de desarrollo para facilitar el desarrollo de nuevos servicios basados en Unix. Su propietario y patrocinador es la empresa española eBox Tecnologías S.L. [6]

Fue desarrollado con el objetivo de acercar Linux a las pequeñas y medianas empresas. Además de constituir una alternativa a los productos de Microsoft. [7]

Su desarrollo se inició en el año 2004 y actualmente cuenta con soporte para más de 30 herramientas de

[5] Acrónimo de pequeña y mediana empresa.

código abierto para la administración de sistemas y redes. Fue integrado a Ubuntu en el 2007 y desde el 2012 las ediciones comerciales son respaldadas por Canonical. El uso de Zentyal se extiende hasta instituciones de alto prestigio como la NASA. [7]

1.4.2. Módulo de administración de clientes ligeros en Zentyal

La inclusión de un nuevo módulo para la administración de clientes ligeros en la versión 3.0 de la plataforma Zentyal, abrió las posibilidades a Nova de llevar todo el conocimiento adquirido en el desarrollo de OSplugger a dicha tecnología. Zentyal ya provee módulos para la administración de servicios tales como *DHCP* y *LDAP*, los cuales unidos al nuevo servicio para clientes ligeros hacen más fácil la administración de los mismos.

Zentyal hace uso de LTSP para garantizar el funcionamiento de las estaciones ligeras. Entre las funcionalidades que brinda a los administradores se pueden encontrar:

1. Permite la creación de nuevas imágenes para clientes ligeros.
2. Monitorización del proceso de creación de imágenes.
3. Permite hacer actualizaciones e instalar nuevas aplicaciones en las imágenes creadas.
4. Permite hacer configuraciones generales dentro del servidor, tales como: limitar a una sesión por usuario, permitir a los clientes ejecutar sonido y dispositivos locales. [2]

Haciendo uso y tomando provecho de las facilidades que anteriormente se enumeraban, la incorporación a este módulo de los conocimientos que hoy en día están presentes en OSplugger brindará la posibilidad de corregir los errores y desventajas que están presentes en dicha herramienta. Se creará un módulo más robusto y con una mayor escalabilidad en entornos de trabajos reales.

El módulo de Zentyal hace uso del protocolo NBD para compartir las imágenes, creadas como un dispositivo de bloque, lo cual hace más rápido el arranque sobre la red. Pero el mismo presenta dificultades sobre las opciones de lectura y escritura, lo que hace necesaria la dependencia del protocolo *AUFS* para garantizar la escritura de los clientes. Estas copias virtuales en LTSP son realizadas en tiempo de

ejecución, lo cual ocupa espacio en disco, puesto que crea una copia virtual de la imagen para cada cliente que se conecta.

La incorporación de nuevas tecnologías como es el caso del protocolo NFS como una nueva opción para llevar a cabo la compartición de las imágenes en red hará del módulo de administración para clientes ligeros de Zentyal un producto más estable en su funcionamiento.

En la tabla que se muestra a continuación se comparan algunas de las características fundamentales de los protocolos NBD y NFS.

Características	NBD	NFS
Creación.	Fue desarrollado por Pavel Machek en 1998.	Fue creado en 1984 por la empresa Sun Microsystems.
Montaje como sistema de raíz.	Es posible, pero no recomendado hacer uso de NBD como un sistema de archivo raíz.	Es un sistema de archivo en red, que permite a los host remotos montar sistemas de archivos sobre la red.
Soporte para diferentes protocolos de red.	Solo hace uso del protocolo TCP para compartir los dispositivos en red.	Hace uso de los protocolos UDP y TCP. El uso de UDP minimiza el tráfico de red y lo hace más rápido, ya que proporciona una conexión de red sin supervisión. En el caso de TCP para mantener un mayor control y seguridad en la trasmisión de los datos.

Continúa en la próxima página. . .

Lectura/Escritura.	El uso de las opciones de lectura/escritura en NBD es extremadamente peligrosa. Si más de un cliente intenta escribir simultáneamente sobre un archivo compartido por NBD, se pueden perder datos.	Los directorios y archivos compartidos con NFS pueden ser compartidos con permiso de lectura y escritura (rw).
Seguridad.	Implementa su seguridad a través de los ficheros **/etc/nbd-server/allow** y **/etc/nbd-server/deny**, mediante ellos se deniegan y se permiten los clientes; haciendo uso de la dirección MAC de las máquinas clientes.	La versión 4 de NFS (NFSv4) incluye seguridad haciendo uso del protocolo *Kerberos*, realiza trabajo con cortafuegos y con *ACLs.*

Tabla 1.3: Tabla de comparación entre NBD y NFS.

El uso del protocolo NFS para compartir las imágenes a los clientes no obliga al usuario final a hacer uso del mismo sino que brinda una nueva opción. La cual corrige los errores presentes en OSplugger al hacer uso del protocolo NBD.

1.5. Lenguaje de Programación

Un **lenguaje de programación** es un idioma artificial para expresar procesos que pueden ser llevados a cabo por máquinas como las computadoras. Pueden usarse para crear programas que controlen el comportamiento físico y lógico de una máquina, para expresar algoritmos con precisión, o como modo de comunicación humana. Está formado por un conjunto de símbolos y reglas sintácticas y semánticas que definen su estructura y el significado de sus elementos y expresiones. Al proceso por el cual se escribe, se prueba, se depura, se compila y se mantiene el código fuente de un programa informático se le llama

programación. La implementación de un lenguaje es la que provee una manera de que se ejecute un programa para una determinada combinación de *software* y *hardware*. [6]

1.5.1. Perl

Para el desarrollo de la aplicación se hará uso del lenguaje de programación **PERL**, ya que el mismo es usado por Zentyal para desarrollar sus aplicaciones. PERL (por sus siglas en inglés de Practical Extraction and Report Language) es un sofisticado lenguaje de programación diseñado a finales de los años 80. Combina en forma concisa las mejores características de lenguajes como C, sed, awk y sh. En general, es posible reducir extensos programas escritos en C a pocas líneas de código de un programa PERL, con la ventaja adicional de que corren sin cambios sobre casi cualquier plataforma existente, lo que convierte a PERL en el lenguaje ideal para desarrollo de prototipos y aplicaciones robustas 100 % portables. [8]

Brinda facilidades para el desarrollo de aplicaciones web, PERL es útil en la resolución de cualquier tarea y es capaz de integrarse con SO, bases de datos, redes, protocolos, ambientes gráficos, otros lenguajes de programación como Java y C. Es versátil y eficiente en el manejo de texto, específicamente, haciendo uso de las “expresiones regulares”.

Este lenguaje dinámico de programación interpretado de alto nivel y de propósito general también está orientado a objetos aunque el programador no está forzado a programar con este esquema.

1.6. Herramientas de Desarrollo

Las **herramientas de desarrollo** son aquellos programas o aplicaciones que tienen un nivel de importancia en el desarrollo de un programa (en la programación). Pueden ser de importancia vital como un ensamblador, un compilador o un editor, o de importancia secundaria como un IDE (por sus siglas en inglés Integrated Development Environment).

1.6.1. Geany

Durante el desarrollo será usada la herramienta Geany para escribir el código fuente de la aplicación puesto que el mismo es un editor de texto pequeño y ligero que proporciona un entorno de desarrollo potente.

Brinda compatibilidad con múltiples lenguajes entre los que se pueden encontrar C/C++, Java, PHP, Python, Perl, entre otros. Geany incluye varias funcionalidades entre las que se puede resaltar: completamiento de código, resaltado de sintaxis para diferentes lenguajes, construye un entorno para compilar y ejecutar el código fuente. [9] También tiene pestañas en su interfaz para editar múltiples archivos a la vez. Otras características orientadas al código incluyen numeración de líneas, resaltado de la línea actual y además dispone de módulos entre los que se puede destacar: autoguardado, buscador de archivos y exportador. Es multiplataforma y su código fuente está disponible bajo los términos de la licencia GPL.

1.7. Metodología de Desarrollo

Las **metodologías de desarrollo** de *software* definen una serie de procedimientos, técnicas y herramientas para la realización de un producto de *software*. Guían todo el proceso de desarrollo del *software*, motivo por el que son muchos los especialistas que se han dedicado a estudiarlas y definirlas. Dada la variedad de características y necesidades de un proyecto, estas metodologías se han dividido en dos grandes grupos: Metodologías Ágiles/Ligeras y Metodologías Pesadas/Tradicionales. [6]

1.7.1. OpenUp

Esta metodología de desarrollo es un proceso unificado (de aplicación general) y ágil (se centra en el desarrollo rápido de sistemas) que involucra un conjunto mínimo de prácticas que ayudan a los equipos de trabajo a ser más efectivos en el desarrollo de *software* u otros sistemas de ingeniería.

El esfuerzo personal en un proyecto OpenUP se organiza en micro-pasos. Estos representan las unidades cortas de trabajo que producen una constante, el ritmo mensurable del progreso del proyecto (normalmente medido en horas o unos pocos días). [10] El proceso es aplicado en un ambiente de intensa colaboración con un equipo comprometido y auto-organizado.

Esta metodología de desarrollo se puede emplear en la realización de proyectos para los que se precisa un corto período de tiempo, ideal por su contenido fundamental y necesario incluido.

Está constituido por cuatro fases: Inicio, Elaboración, Construcción y Transición.

Fase de Inicio

Tiene como meta fundamental conocer y entender los objetivos a cumplir durante el proyecto, para esto se necesita identificar quiénes son los principales involucrados, además es en esta fase donde se conocen los requisitos que debe cumplir el producto. [11]

Los objetivos de esta fase son:

- Entender qué se va a construir.
- Identificar las funcionalidades claves del sistema.
- Hacer un estudio del arte de las distribuciones similares determinando al menos una solución.

Fase de Elaboración

Tiene por finalidad obtener una arquitectura lo más robusta posible tratando por tanto los riesgos significativos para la misma. Los objetivos de esta fase están dados por:

- Obtener una descripción más detallada de los requisitos.
- Diseñar, validar e implementar las diferentes vistas arquitectónicas.

Fase de Construcción

Logra dar al producto capacidad operacional cumpliendo con los objetivos:

- Crear de forma iterativa una versión beta del producto que cumpla con los requisitos especificados y que esté listo para ser utilizado por la comunidad de usuarios.

Fase de Transición

Se encarga de obtener una versión candidata del sistema que puede ser utilizado por los usuarios de la comunidad, los principales objetivos de esta fase son:

- Obtener una versión candidata.

- Validar la versión candidata para determinar si se cumplen las expectativas de los usuarios.
- Desplegar la distribución en entornos de los usuarios finales.

La duración de las fases depende del tamaño del equipo de trabajo, así como de la experiencia de los desarrolladores y el nivel de colaboración que se establezca con los contribuidores de las comunidades de usuarios. [11]

1.8. Herramientas de Modelado

Las **herramientas de modelado** de sistemas informáticos son herramientas que se emplean para la creación de modelos de sistemas que existen o que se desarrollan. Estas permiten crear un "simulacro" del sistema a bajo costo, pues son un conjunto de gráficos y textos que representan el sistema. Minimizan, además, los riesgos porque los cambios que se consideren necesarios hacer son más fáciles y rápidamente sobre el modelo del sistema ya implementado.

1.8.1. Lenguaje de modelado UML

Se hará uso del Lenguaje de Modelado Unificado (UML, por sus siglas en inglés), mediante el cual se representará todo el ciclo de vida del *software* en cuestión.

"UML es un lenguaje para visualizar, especificar, construir y documentar los artefactos de un sistema software. Utilizar herramientas de modelado visual facilita la gestión de dichos modelos, permitiendo ocultar o exponer detalles cuando sea necesario. El modelado visual también ayuda a mantener la consistencia entre los artefactos del sistema: requisitos, diseños e implementaciones. En resumen, el modelado visual ayuda a mejorar la capacidad del equipo para gestionar la complejidad del software." [12]

1.8.2. Visual Paradigm

Visual Paradigm es una herramienta CASE (del inglés Cumputer Aided *software* Engineering) de Lenguaje de Modelaje Unificado (UML, por sus siglas en inglés de Unified Modeling Language) profesional

que soporta el ciclo de vida completo del desarrollo de *software*, es decir, el análisis y diseño orientados a objetos, construcción, pruebas y despliegue. [6]

Esta herramienta posibilita el aumento de la calidad del *software*, a través de la mejora de la productividad en el desarrollo y mantenimiento del *software*. Permite representar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y documentación, además de la reutilización de *software*, portabilidad y estandarización de la documentación. Por estas razones ha sido seleccionada para el modelado de la aplicación.

1.9. Conclusiones del capítulo

Durante el desarrollo de este capítulo se ha explicado detalladamente el proceso de desarrollo e investigación que se ha realizado en el proyecto Nova para la administración de clientes ligeros. A partir del estudio realizado fue definido que: se trabajará sobre la plataforma de administración Zentyal, dirigiendo el conocimiento obtenido durante años al módulo de Zentyal+LTSP sobre el cual se realizarán las transformaciones necesarias para adaptarlo a las necesidades de la empresa cubana. Para la implementación se usará el editor de texto Geany y el lenguaje de programación Perl, como metodología de desarrollo se usará OpenUp y el Visual Paradigm como herramienta de modelado.

Capítulo 2

Capítulo 2: Propuesta de solución

A partir de los elementos a los que se hizo referencia en el capítulo anterior, en el presente se recogen los aspectos que, relacionados con el análisis y el diseño basado en la metodología mencionada, harán posible contar con una aplicación robusta. Para ello se definirá la arquitectura de la herramienta, así como su proceso de funcionamiento en virtud de cumplir con los requisitos funcionales de la misma. Se realizará además la modelación de los diagramas fundamentales.

2.1. Actores del sistema

Se define como **actor del sistema** a toda aquella entidad externa a él con la que mantiene una relación y que le solicita una funcionalidad. Pueden representar el rol que juega una o varias personas, un equipo o un sistema automatizado, además de entidades abstractas como el tiempo.

Actor del Sistema	Descripción
Administrador	El administrador es el encargado de hacer toda la gestión de la información que brinda el sistema.

Tabla 2.1: Actores del sistema.

2.2. Requisitos del sistema

La solución que se propone se desarrollará sobre un módulo de la plataforma Zentyal, aplicando en el mismo el conocimiento adquirido en el desarrollo de la herramienta OSplugger. El mismo permitirá la administración de clientes ligeros y máquinas sin disco. Dicho módulo debe de cumplir con los requisitos que a continuación se relacionan.

2.2.1. Requisitos Funcionales

Los **requisitos funcionales** son servicios que el sistema debe proporcionar, define como el sistema reacciona ante entradas particulares y como comportarse en determinadas situaciones. En algunos casos el requisito funcional puede también indicar explícitamente lo que el sistema no debe hacer. [13]

- RF1: Administrar módulo.

  RF3.1: Instalar módulo.

  RF3.1: Actualizar módulo.

  RF3.1: Eliminar módulo.

- RF2: Instalar aplicaciones.

- RF3: Crear imagen.

  RF3.1: Mostrar opción de seleccionar el tipo de imagen a crear.

  RF3.2: Mostrar opción de especificar el repositorio.

  RF3.3: Mostrar opción de especificar el nombre de la distribución.

- RF4: Actualizar imagen.

- RF5: Asignar imagen a cliente.

- RF6: Configurar opciones de clientes.

- RF7: Mostrar listado de imágenes.

2.2.2. Requisitos no Funcionales

Los **requisitos no funcionales** especifican criterios que pueden usarse para juzgar la operación de un sistema en lugar de sus comportamientos específicos. Se refieren a todos los requisitos que no describen información a guardar, ni funciones a realizar. En otras palabras son las propiedades o cualidades que el producto debe tener.

- **Apariencia o interfaz externa**

  RNF1: La aplicación deberá presentar y mantener una interfaz amigable, interactiva, intuitiva y entendible por el usuario.

  RNF2: Los mensajes mostrados al usuario deben seguir los patrones definidos por la plataforma Zentyal.

- **Accesibilidad**

  RNF3: La información y las funcionalidades estarán disponibles y el administrador podrá acceder a ella en todo momento.

- **Disponibilidad**

  RNF4: El sistema siempre estará disponible para el administrador, dependiendo solamente de la operatividad de la plataforma Zentyal.

- **Rendimiento**

  RNF5: La aplicación permitirá que múltiples usuarios estén conectados a la vez.

- **Legales**

  RNF6: Las herramientas seleccionadas para el desarrollo del producto están respaldadas por licencias libres, bajo las condiciones de software libre. En el caso de la herramienta Visual Paradigm que es propietaria, se utiliza la licencia con que cuenta la UCI.

  RNF7: La aplicación y toda la documentación generada pertenecen al grupo de proyecto Gestión Documental y Archivística y a la Universidad de las Ciencias Informáticas.

- **Software**

RNF8: La PC servidor debe contar con servidor web Apache 2.0.

RNF9: La PC servidor debe tener instalada la plataforma de desarrollo Zentyal.

- **Hardware**

  Procesamiento

  RNF10: El servidor de clientes ligeros requiere un CPU Intel Core 2 Dou a 2.20GHz o superior para su correcto funcionamiento.

  Memoria RAM

  RNF11: El servidor de clientes ligeros requiere 2 Gb de memoria RAM o superior para su correcto funcionamiento.

  Capacidad en disco

  RNF12: El servidor de clientes ligeros requiere 160 Gb disponibles para su correcto funcionamiento.

- **Confidencialidad**

  RNF13: La información manejada por el sistema está protegida de acceso no autorizado.

- **Restricciones de diseño e implementación**

  1. El sistema se desarrollará haciendo uso del lenguaje de programación Perl.
  2. El entorno de desarrollo integrado será Geany.
  3. Las interfaces de administración deben ser programadas en Perl.
  4. Para realizar el modelado del sistema se utilizará la herramienta Visual Paradigm 8.0.
  5. La metodología de desarrollo de software empleada será OpenUp, haciendo uso del Lenguaje de Modelado Unificado (UML).

2.3. Arquitectura de la aplicación

Al utilizar la plataforma de desarrollo Zentyal para desarrollar la solución, se hereda una arquitectura basada en el patrón Modelo-Vista-Controlador. [14] Este patrón permite separar los datos de la aplicación,

la interfaz y la lógica en tres componentes distintos, diseñando e implementando en la solución la Vista y el Controlador, y utilizando el acceso a datos definido por Zentyal.

2.4. Casos de uso del sistema

Un caso de uso describe lo que el sistema debe hacer para proporcionar un valor a los accionistas. Describe las interacciones entre uno o más actores y el sistema con el fin de proporcionar un resultado observable de valor para el actor. La funcionalidad del sistema se define por diferentes casos de uso, cada uno de ellos representa un objetivo concreto (para obtener el resultado observable de valor) de un actor en particular. [10] Por lo que una colección de casos de uso constituyen todas las posibles formas de utilización del sistema.

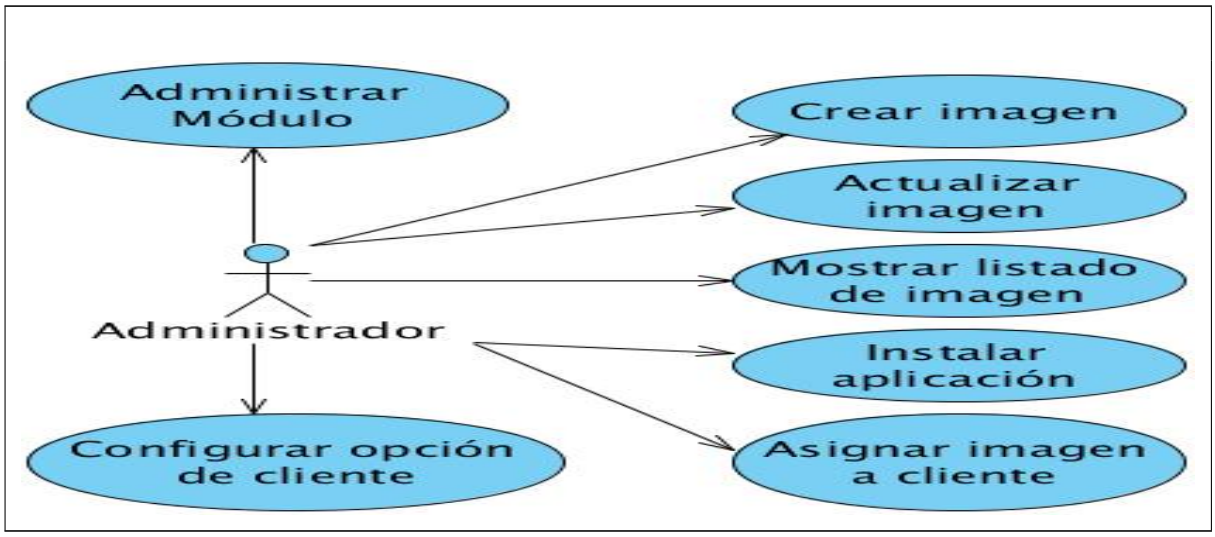

Figura 2.1: Diagrama de casos de uso del sistema.

2.4.1. Descripción de los casos de uso del sistema

A continuación se describen textualmente los casos de uso del sistema que fueron modelados en el diagrama anterior, se especificará su propósito y las condiciones de existencia de los mismos.

Descripción del caso de uso Administrar módulo

CU1: Descripción del caso de uso Administrar módulo	
Caso de Uso	**Administrar módulo**
Objetivos	Permitir al administrador administrar el módulo de clientes ligeros.
Resumen	El caso de uso inicia cuando el administrador selecciona la opción Componentes de Zentyal. El sistema muestra tres pestañas con las opciones Instalar, Actualizar y Eliminar dando la posibilidad de escoger la acción que se desee.
Complejidad	Alta
Prioridad	Crítica
Precondición	El administrador debe estar autenticado en la plataforma Zentyal y el módulo de clientes ligeros debe existir en el repositorio.
Postcondición	El módulo de clientes ligeros será instalado, actualizado o eliminado.
Flujo Normal de Eventos	
Acción del Actor	**Acción del Sistema**
1. El actor selecciona la opción Gestión de software/Componentes de Zentyal. 3. El actor selecciona una opción: a. Si selecciona la opción instalar véase la sección Instalar módulo. b. Si selecciona la opción actualizar véase la sección Actualizar módulo. c. Si selecciona la opción Eliminar véase la sección Eliminar módulo.	2. El sistema muestra la interfaz Componentes de Zentyal que contiene las opciones; Instalar, Actualizar y Eliminar.
Sección “Instalar módulo”	
Acción del Actor	**Acción del Sistema**

Continúa en la próxima página. . .

1. El actor selecciona la opción instalar módulo.	2. El sistema muestra un listado con los módulos disponibles para ser instalados.
3. El actor marca el cuadro de selección correspondiente al módulo clientes ligeros. 4. El actor da clic sobre el botón Instalar.	5. El sistema instala el módulo. 6. El sistema muestra un mensaje confirmando que el módulo ha sido instalado.
Sección "Actualizar módulo"	
Acción del Actor	**Acción del Sistema**
1. El actor selecciona la opción actualizar módulo.	2. El sistema muestra un listado con los módulos que pueden ser actualizados.
3. Marca el cuadro de selección correspondiente al módulo clientes ligeros. 4. Da clic sobre el botón Actualizar.	5. Actualiza el módulo. 6. Muestra un mensaje confirmando que el módulo ha sido actualizado.
Sección "Eliminar módulo"	
Acción del Actor	**Acción del Sistema**
1. El actor selecciona la opción Eliminar.	2. El sistema muestra un listado con los módulos instalados.
3. El actor marca el cuadro de selección correspondiente al módulo clientes ligeros. 4. Da clic sobre el botón Borrar.	5. Borra el módulo. 6. Muestra un mensaje confirmando que el módulo ha sido eliminado.

Tabla 2.3: Descripción del caso de uso Administrar módulo.

Descripción del caso de uso Crear imagen

CU1: Descripción del caso de uso Crear imagen	
Caso de Uso	**Crear imagen**
Objetivos	Permitir al administrador crear una nueva imagen para clientes ligeros.
Continúa en la próxima página. . .	

Resumen	El caso de uso inicia cuando el administrador selecciona la opción Clientes Ligeros de la sección Infraestructura. El sistema muestra el formulario que se debe completar para la creación de imagen.
Complejidad	Alta
Prioridad	Crítica
Precondición	El administrador debe estar autenticado en la plataforma Zentyal y el módulo de clientes ligeros debe estar instalado en el sistema.
Postcondición	Ninguna.

Flujo Normal de Eventos	
Acción del Actor	**Acción del Sistema**
1. El actor selecciona la opción Clientes Ligeros en la sección Infraestructura de la plataforma Zentyal. 3. Selecciona la arquitectura de la imagen a crear. 4. Selecciona el protocolo por el cual se compartirá la imagen a crear. 5. Seleccionar el tipo de imagen, de procesamiento en el cliente o en el servidor. 6. Especificar dirección del repositorio (Opcional). 7. Especificar el nombre de la distribución (Opcional). 8. Da clic en el botón Crear Imagen.	2. El sistema muestra la interfaz de creación de imágenes. 9. Muestra un mensaje informando que el proceso de creación de la imagen está siendo mostrado en el Panel de Control. 10. Mostrar mensaje de construcción satisfactoria de la imagen o error en caso de algún fallo.

Tabla 2.4: Descripción del caso de uso Crear imagen.

Descripción del caso de uso Mostrar listado de imagen

CU1: Descripción del caso de uso Mostrar listado de imagen	
Caso de Uso	**Mostrar listado de imagen**
Objetivos	Permitir al administrador ver las imágenes creadas.
Resumen	El caso de uso inicia cuando el administrador selecciona la opción Clientes Ligeros de la sección Infraestructura. El sistema muestra la vista Imágenes Disponibles.
Complejidad	Alta
Prioridad	Crítica
Precondición	El administrador debe estar autenticado en la plataforma Zentyal y deben existir imágenes creadas, sino el sistema mostrará un mensaje informando que no existen imágenes disponibles.
Postcondición	Ninguna.
Flujo Normal de Eventos	
Acción del Actor	**Acción del Sistema**
1. El actor selecciona la opción Clientes Ligeros de la sección Infraestructura de Zentyal.	2. El sistema muestra la interfaz de Imágenes disponibles.

Tabla 2.5: Descripción del caso de uso Mostrar listado de imagen.

Descripción del caso de uso Actualizar imagen

CU1: Descripción del caso de uso Actualizar imagen	
Caso de Uso	**Actualizar imagen**
Objetivos	Permitir al administrador actualizar una imagen ya creada.

Continúa en la próxima página. . .

Resumen	El caso de uso inicia cuando el administrador selecciona la opción Clientes Ligeros de la sección Infraestructura. El sistema muestra la vista de Imágenes Disponibles, dando la posibilidad de actualizar una imagen ya existente.
Complejidad	Alta
Prioridad	Media
Precondición	El administrador debe estar autenticado en la plataforma Zentyal y debe existir al menos una imagen creada.
Postcondición	Ninguna.
Flujo Normal de Eventos	
Acción del Actor	**Acción del Sistema**
1. El actor selecciona la opción Clientes Ligeros de la sección Infraestructura de Zentyal. 3. Da clic en la opción actualizar imagen de la sección Acción dentro de la tabla de imágenes.	2. El sistema muestra la tabla de Imágenes disponibles. 4. Muestra un mensaje informando que el proceso de actualización de la imagen se puede observar desde el Panel de Control. 5. Actualiza la imagen.

Tabla 2.6: Descripción del caso de uso Actualizar imagen.

Descripción del caso de uso Instalar aplicación

CU1: Descripción del caso de uso Instalar aplicación	
Caso de Uso	**Instalar aplicación**
Objetivos	Permitir al administrador instalar nuevas aplicaciones en las imágenes ya existentes.

Continúa en la próxima página. . .

Resumen	El caso de uso inicia cuando el administrador selecciona la opción Aplicaciones Locales en la vista de imágenes disponibles. El sistema muestra la interfaz de instalación de nuevas aplicaciones.
Complejidad	Media
Prioridad	Alta
Precondición	El administrador debe estar autenticado en la plataforma Zentyal y deben existir imágenes creadas.
Postcondición	Ninguna.
Flujo Normal de Eventos	
Acción del Actor	**Acción del Sistema**
1. El actor selecciona la opción Aplicaciones locales en la interfaz de Imágenes disponibles. 3. Introduce en el campo de texto el nombre de las aplicaciones a instalar separadas por espacio. 4. Da clic sobre el botón Instalar Aplicaciones.	2. El sistema muestra la interfaz de instalación de nuevas aplicaciones. 5. Muestra un mensaje informando que el proceso de instalación de aplicaciones será mostrado en el Panel de Control. 6. Muestra un mensaje de instalación satisfactoria o error en caso de algún fallo.

Tabla 2.7: Descripción del caso de uso Instalar aplicación.

Descripción del caso de uso Configurar opción de cliente

CU1: Descripción del caso de uso Configurar opción de cliente	
Caso de Uso	**Configurar opción de cliente**
Objetivos	Permitir al administrador configurar las opciones de los clientes.
Continúa en la próxima página. . .	

Resumen	El caso de uso inicia cuando el administrador selecciona la opción Clientes Ligeros de la sección Infraestructura. El sistema muestra la interfaz de administración del módulo para clientes ligeros, donde se puede observar cuatro pestañas con las opciones de: Imágenes, General, Inicio de sección automática y Perfiles dando la posibilidad de escoger la opción que se desea.
Complejidad	Media
Prioridad	Baja
Precondición	El administrador debe estar autenticado en la plataforma Zentyal.
Postcondición	Ninguna.

Flujo Normal de Eventos	
Acción del Actor	**Acción del Sistema**
1. El actor selecciona la opción Clientes Ligeros de la sección Infraestructura de Zentyal. 3. Selecciona la pestaña “General”. 5. Selecciona las opciones a ser deshabilitadas o habilitadas. 6. Da clic sobre el botón Cambiar.	2. El sistema muestra la interfaz de administración del módulo de Zentyal para clientes ligeros. 4. El sistema muestra una nueva vista con un conjunto de opciones. 7. Muestra un mensaje informando que los cambios han sido “Hechos”.

Tabla 2.8: Descripción del caso de uso Configurar opción de cliente.

Descripción del caso de uso Asignar imagen a cliente

CU1: Descripción del caso de uso Asignar imagen a cliente	
Caso de Uso	**Asignar imagen a cliente**

Continúa en la próxima página. . .

Objetivos	Permitir al administrador asignar a un cliente una imagen determinada.
Resumen	El caso de uso inicia cuando el administrador selecciona la opción Clientes Imágenes en la vista de imágenes disponibles. El sistema muestra la interfaz de asignación de imagen al cliente.
Complejidad	Alta
Prioridad	Baja
Precondición	El administrador debe estar autenticado en la plataforma Zentyal y debe existir al menos una imagen creada.
Postcondición	Ninguna.

Flujo Normal de Eventos	
Acción del Actor	**Acción del Sistema**
1. El actor selecciona la opción Clientes imágenes en la interfaz de Imágenes disponibles. 3. Introduce la dirección MAC del cliente al que se le asigna la imagen. 4. Da clic en el botón Adicionar Cliente.	2. El sistema muestra la interfaz de asignación de imágenes a clientes. 5. Muestra un mensaje informando que la imagen a sido asignada al cliente o error en caso de algún fallo.

Tabla 2.9: Descripción del caso de uso Asignar imagen a cliente.

2.5. Diagrama de clases del diseño

El diagrama de clases del diseño representa las clases que serán usadas dentro del sistema, así como las relaciones que existen entre ellas. Un diagrama de clases está compuesto por elementos tales como; Clase: la cual está compuesta por: atributos y métodos; y Relaciones: las cuales pueden ser de herencia, composición, agregación, asociación y uso. [15]

La siguiente figura muestra las clases que intervienen en la propuesta de solución. La clase **LTSP** constituye el inicio y es la encargada de proporcionarle la vista a la plataforma Zentyal. A esta clase se le integran las vistas **AvaibleImages** e **ImageCreation**. **AvaibleImages** es la encargada de proporcionar al usuario una vista de las imágenes creadas en el sistema, permitiendo realizar acciones sobre las mismas tales como: actualizar imagen, configurar opciones de los clientes a través de la clase **GeneralOpts**, instalar aplicaciones locales mediante la clase **LocalApps** y asignar imagen a un cliente haciendo uso de la clase **ClientImage**. La clase **ImageCreation** es la encargada de la creación de nuevas imágenes la cual delega parte de sus responsabilidades a la clase **BuildImage**.

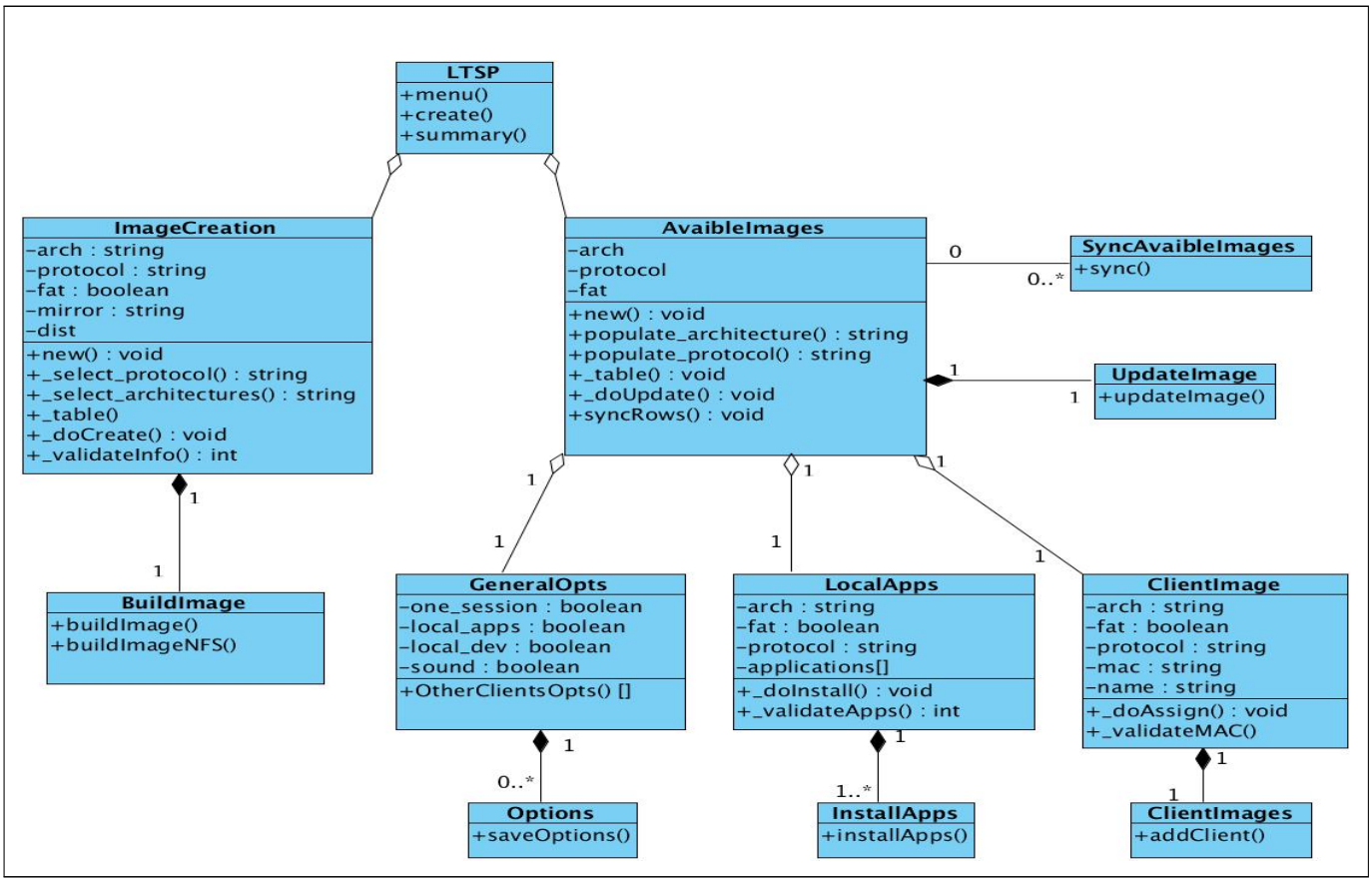

Figura 2.2: Diagrama de clases del diseño.

2.5.1. Diagrama de secuencia

En un diagrama de secuencia se muestra la interacción de los objetos en una aplicación a través del tiempo y se modela para cada caso de uso del sistema. Este tipo de diagrama contiene detalles de implementación del escenario expuesto en el caso de uso, incluye los objetos y clases que se usan para implementar el escenario, así como los mensajes intercambiados entre los objetos.

Se muestran los objetos que intervienen en el escenario con líneas discontinuas verticales, y los mensajes pasados entre los objetos como flechas horizontales.

Las siguientes figuras muestran los diagramas de secuencia para los requisitos funcionales más importantes del sistema.

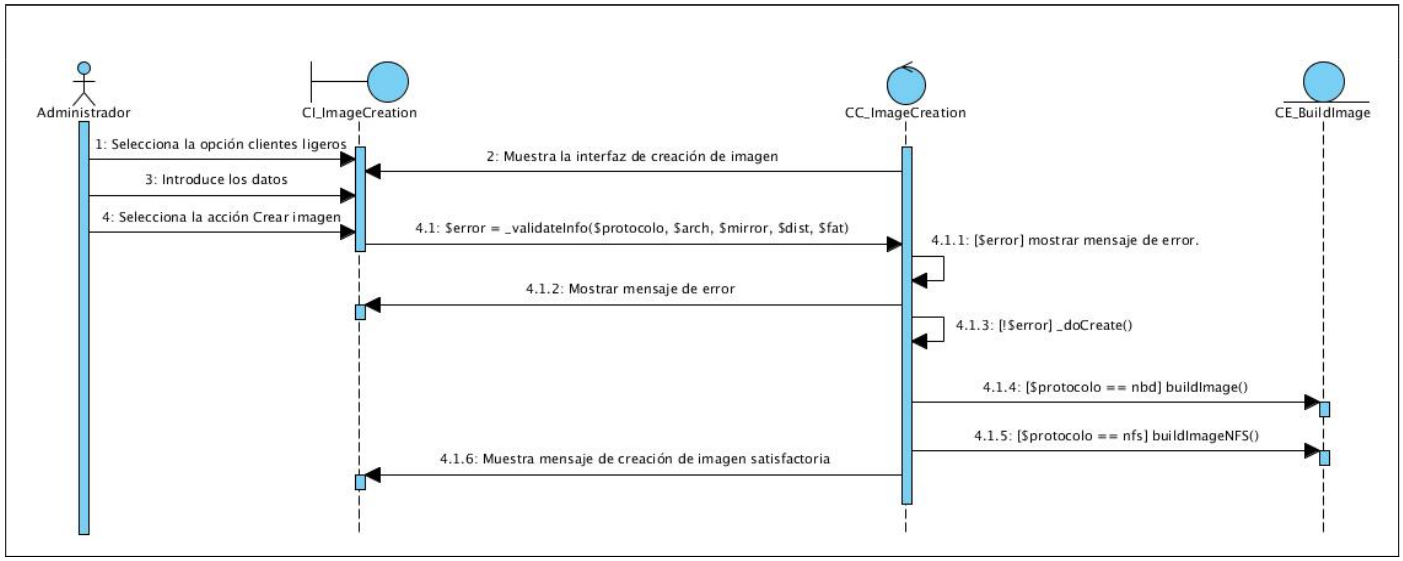

Figura 2.3: Diagrama de secuencia Crear imagen.

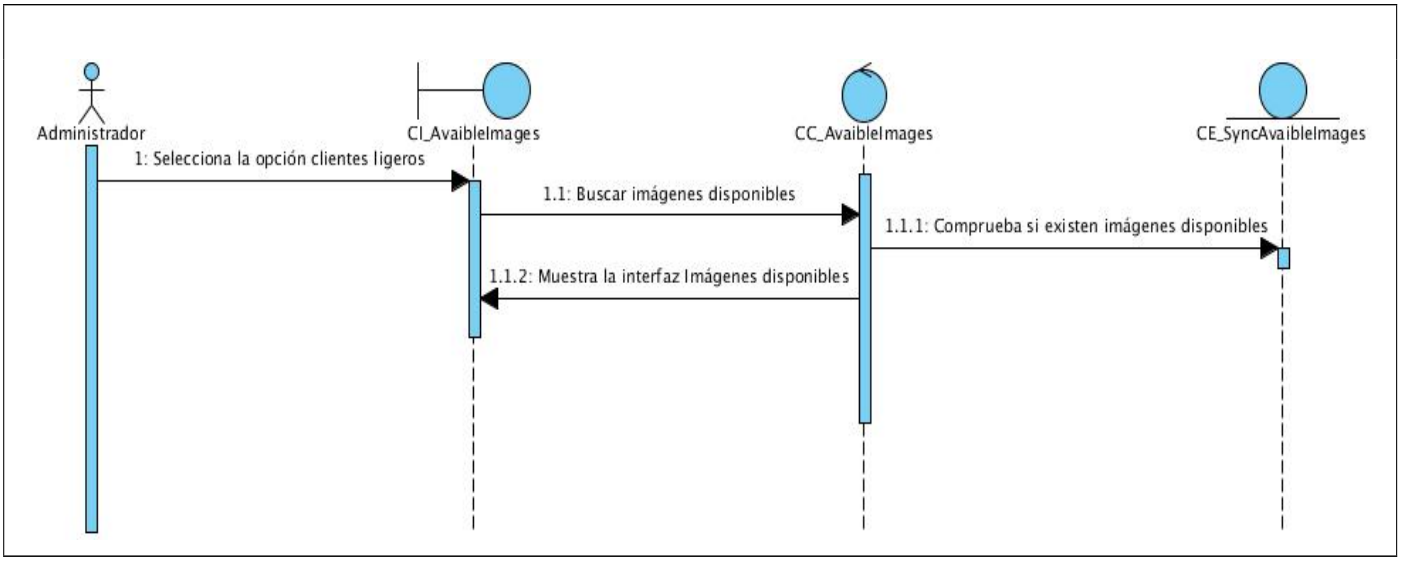

Figura 2.4: Diagrama de secuencia Mostrar listado de imagen.

En el Anexo III se pueden observar los diagramas de secuencia restantes.

2.6. Patrones de diseño

Un patrón describe un problema que ocurre una y otra vez en nuestro entorno y luego describe el núcleo de la solución a ese problema, de tal manera que se puede utilizar la solución un millón de veces sin tener que hacerlo de la misma manera dos veces. [16]

Un patrón de diseño puede ser caracterizado como una regla de tres partes que expresa una relación entre un cierto contexto, un problema, y una solución. [16]

A continuación se describen los patrones que fueron utilizados durante el desarrollo de la aplicación.

2.6.1. Patrones GRASP

GRASP es un acrónimo de General Responsibility Assignment Software Patterns o Patrones Generales de Software para Asignar Responsabilidades. Los patrones GRASP son utilizados para describir los principios fundamentales del diseño y la asignación de responsabilidades. [15]

- **Experto en información:** indica que la responsabilidad de la creación de un objeto o la implementación de un método, debe recaer sobre la clase que conoce toda la información necesaria para crearlo. De este modo se obtiene un diseño con mayor cohesión y así la información se mantiene encapsulada, posibilitando la disminución del acoplamiento.

  Este patrón es usado en todas las clases de la aplicación, pues cada clase tiene su responsabilidad definida. Un ejemplo de ello lo representa la clase **ImageCreation**, la cual es responsable de dirigir el proceso de creación de las imágenes, y velar por las configuraciones que se realizan en cada una de ellas.

- **Alta cohesión:** la cohesión es la medida de la fuerza que une a las responsabilidades de una clase. Una clase con baja cohesión hace muchas cosas no relacionadas, o hace demasiado trabajo. Tales clases no son convenientes pues son difíciles de mantener, de reutilizar y de entender. Una clase con alta cohesión mejora la claridad y la facilidad de su uso, su mantenimiento se simplifica y es fácil de reutilizar.

  Durante el diseño de la aplicación se realizó asignación de responsabilidades a cada clase presente en la aplicación de manera tal que la cohesión se mantuviese alta. Un ejemplo del uso de este patrón lo constituye la clase **LocalApps** que es la encargada de instalar nuevas aplicaciones sobre una imagen, después de realizar las validaciones correspondientes delega el trabajo de creación de la imagen a la clase **InstallApps**. De esta forma el código se hace mucho más fácil de mantener.

- **Bajo acoplamiento:** su principio es tener las clases lo menos ligadas posibles entre sí. De tal forma que en caso de producirse una modificación en alguna de ellas, se tenga la mínima repercusión posible en el resto de clases, potenciando la reutilización, y disminuyendo la dependencia entre las clases.

  El uso de este patrón se evidencia en todas las clases de la aplicación. Solo existe dependencia necesaria entre clases encargadas de realizas una acción.

2.7. Conclusiones del capítulo

Durante el desarrollo de este capítulo se definieron los aspectos relacionados con el análisis y diseño de la aplicación. Se especificaron los requisitos funcionales y no funcionales, en aras de lograr una aplicación con buen funcionamiento. Se incluyen algunos diagramas importantes para favorecer un mejor entendimiento del sistema y su funcionamiento y se explican los patrones de diseño GRASP que fueron utilizados para la asignación de responsabilidades en cada clase del sistema.

Capítulo 3

Capítulo 3: Implementación y Pruebas

El presente capítulo se centra en los flujos de trabajo, implementación y pruebas. Se describe la implementación de la herramienta, así como sus principales métodos. Se realizan pruebas de caja negra, para validar la solución propuesta.

3.1. Implementación

El flujo de trabajo implementación enmarca el inicio de la fase de Construcción del software. El propósito general de la implementación es desarrollar la arquitectura y el sistema como un todo. La aplicación que se desarrolle debe tener la calidad requerida para su uso y cumplir con los requisitos de software determinados en el capítulo anterior.

3.1.1. Diagrama de componentes

La vista la proporciona la plataforma Zentyal que funcionará como un subsistema, que cuenta con el módulo clientes ligeros desde el que se establece comunicación con el sistema de fichero del SO para realizar la configuración de los ficheros necesarios y gestionar los servicios. Inicia el proceso de creación de la imagen, el cual también actúa como un subsistema en el que se hace necesario configurar ficheros que garantizan un correcto funcionamiento. En el caso de los ficheros con extensión **pm** constituyen las vistas de la aplicación. El siguiente diagrama muestra lo anteriormente descrito.

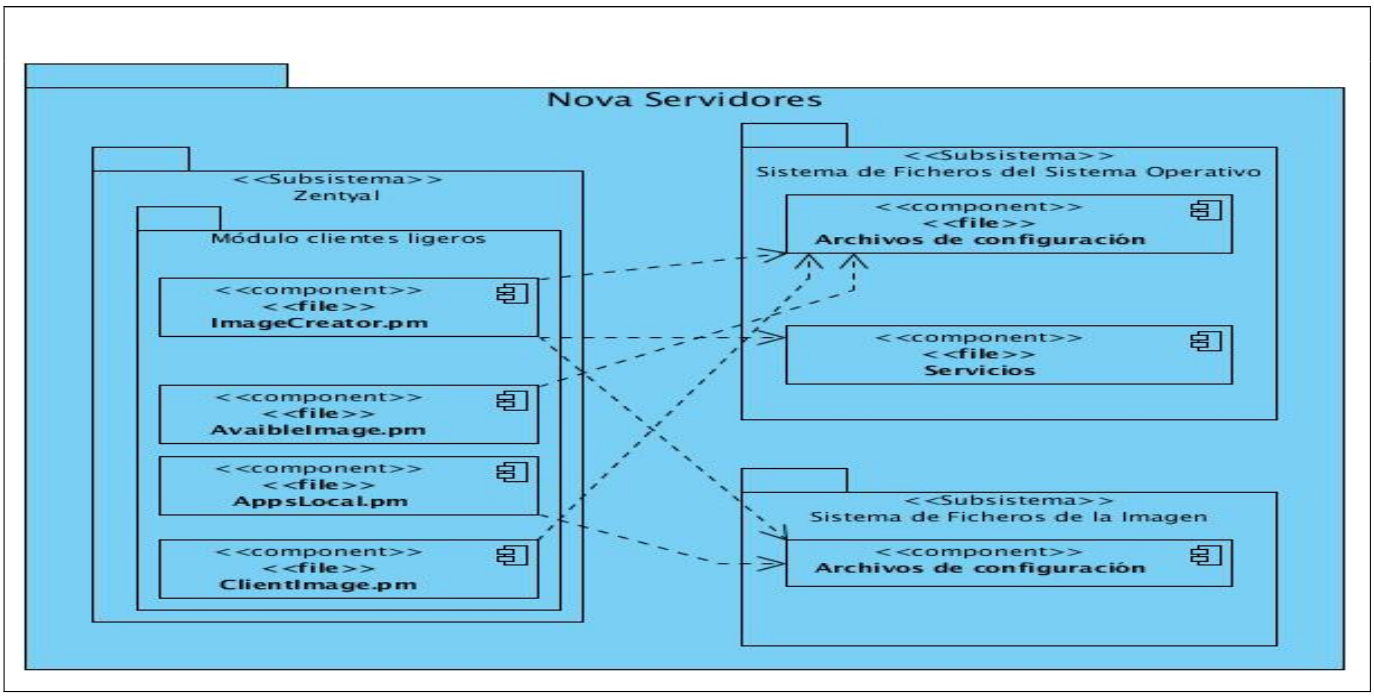

Figura 3.1: Diagrama de componentes

3.1.2. Diagrama de despliegue

El **diagrama de despliegue** se utiliza para modelar el hardware utilizado en las implementaciones de sistemas así como las relaciones entre sus componentes. Muestra el cómo y el dónde se desplegará el sistema. Las máquinas físicas y los procesadores se representan como nodos, y la construcción interna puede ser representada por nodos o artefactos embebidos. Los estereotipos permiten precisar la naturaleza del equipo: dispositivos, procesadores y memoria. [17]

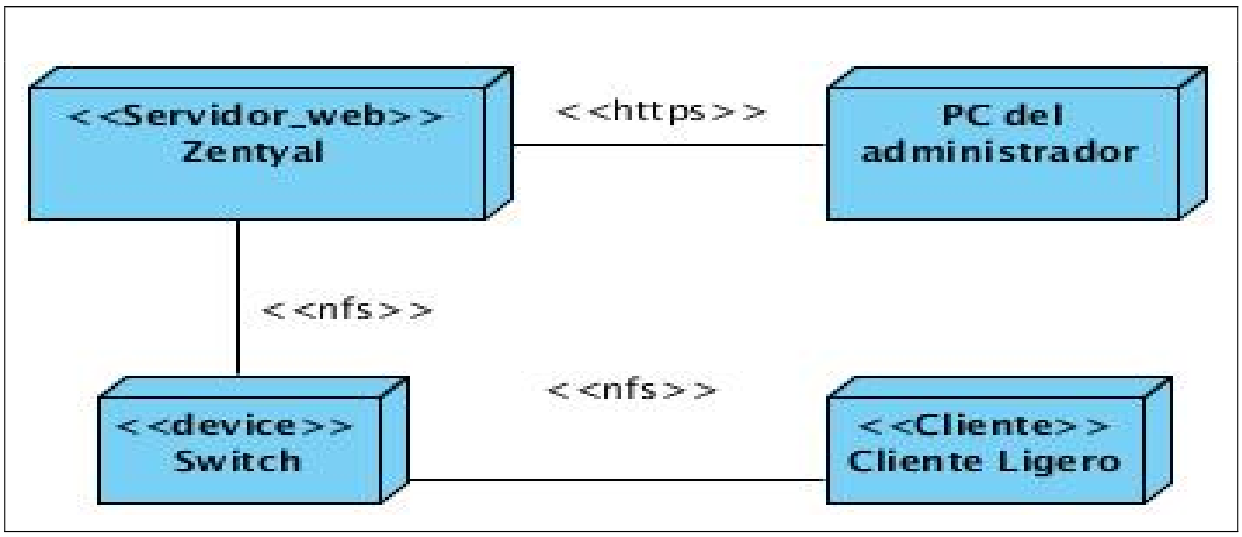

Figura 3.2: Diagrama de despliegue

Descripción del diagrama de despliegue

El diagrama de despliegue mostrado anteriormente está distribuido de la siguiente forma:

1. **Servidor:** servidor en el cual se encuentra el servidor web apache, sobre el cual está montada la plataforma de desarrollo Zentyal. En este servidor se almacenarán las imágenes que serán compartidas por NFS. El mismo establecerá comunicación con los ordenadores clientes mediante el protocolo DHCP (no necesariamente debe de estar montado sobre este servidor).

2. **PC del administrador:** es una PC cliente, la cual es capaz de conectarse a la interfaz de administración de Zentyal mediante el protocolo HTTPS.

3. **Switch:** dispositivo de red que permite la interconexión entre redes de computadoras, el cual permitirá conectar varias Clientes Ligeros a un mismo servidor.

4. **Cliente Ligero:** ordenador cliente, capaz de conectarse al servidor mediante el protocolo NFS para iniciar un SO en red.

3.2. Resultados obtenidos

Para obtener los resultados propuestos en el desarrollo de la aplicación se trazaron como requerimientos básicos:

1. Lograr una familiarización con el lenguaje de programación Perl.
2. Lograr una familiarización con el módulo de administración para clientes ligeros desarrollado por Zentyal; estudiar la estructura, principales métodos y su principio de funcionamiento.
3. Configurar el entorno de desarrollo con las herramientas necesarias.

3.2.1. Entorno de desarrollo

La configuración del entorno de desarrollo es la parte fundamental del proceso de construcción del software. Las herramientas y aplicaciones necesarias pueden ser encontradas en los repositorios oficiales de las distribuciones de Ubuntu o Nova.

Durante la siguiente explicación serán tomados como ejemplo los repositorios de aplicaciones de la distribución cubana de GNU/Linux Nova.

Instalación y acceso a Zentyal

La instalación de la plataforma de desarrollo Zentyal puede llevarse a cabo con la siguiente línea de comando:

```
$ sudo apt-get install zentyal
```

Después de concluido el proceso de instalación para acceder a la interfaz de administración de Zentyal es suficiente escribir en el navegador web la dirección:

```
https://localhost
```

En caso de usarse una máquina remota para la administración del servidor escribir en el navegador:

```
https://dirección_ip_servidor
```

En ambos casos es necesario usar el protocolo HTTPS para conexiones seguras.

Como el objetivo final del presente trabajo es realizar modificaciones al módulo para clientes ligeros de Zentyal, se hace necesaria la instalación de la herramienta para la compilación y empaquetado zbuildtools[1], mediante la cual se compilará y empaquetará la nueva versión del módulo de clientes ligeros.

```
$ sudo apt-get install zbuildtools
```

Obtener código fuente del módulo zentyal-ltsp

La instalación de la plataforma Zentyal y la herramienta de compilación *zbuildtools* es tan solo el primer paso. Para obtener el código fuente del módulo para clientes ligeros de Zentyal es necesario agregar en el fichero **/etc/apt/sources.list** la dirección de los repositorios de código fuente, para lo que se ejecuta:

```
$ sudo nano /etc/apt/sources.list
```

Para abrir el fichero y escribir:

```
deb-src http://nova.f10.uci.cu/nova 2013 principal extendido
```

Después de salvados los cambios se actualiza el sistema, ejecutando:

```
$ sudo apt-get update
```

Luego se procede a descargar el código fuente haciendo uso del comando:

```
$ apt-get source zentyal-ltsp
```

El código descargado puede ser encontrado en el directorio del usuario[2].

[1] Herramienta usada por Zentyal para facilitar la compilación de código fuente.

[2] Se refiere al usuario que está ejecutando los comandos.

Modificación del módulo de Zentyal

Después que el código fuente está descargado se accede al mismo y se procede a realizar las modificaciones pertinentes, para ello se hace uso de un editor de texto o algún IDE de programación. En este caso se usará el editor de texto Geany tal y como se expuso en el Capítulo 1.

Para lograr el funcionamiento que se desea, se hace necesario realizar modificaciones en los modelos **ImageCreator** y **AvaibleImage**.

En el método **_table** del modelo **ImageCreator** se definen los parámetros o valores con sus tipos de datos. En la siguiente figura se muestran los elementos por los que se compone dicha tabla, en la cual aparecerán señaladas las opciones que han sido añadidas a dicha tabla para dar cumplimiento a los requisitos trazados.

```
sub _table
{
    my ($self) = @_;

    my @fields =
    (
        new EBox::Types::Select(
            'fieldName'     => 'architecture',
            'printableName' => __('Architecture'),
            'populate'      => \&_select_architectures,
            'editable'      => 1,
        ),
        new EBox::Types::Select(
            'fieldName'     => 'protocol',
            'printableName' => __('LTSP Root Sharing Protocol'),
            'populate'      => \&_select_protocol,
            'editable'      => 1,
        ),
        new EBox::Types::Boolean(
            'fieldName'     => 'fat',
            'printableName' => __('Fat Image'),
            'dafaultValue'  => 0,
            'editable'      => 1,
        ),
        new EBox::Types::Text(
            'fieldName'     => 'mirror',
            'printableName' => __('Mirror'),
            'size' => '20',
            'unique' => 1,
            'editable' => 1,
            'help' => __('Specify the repository\'s address, eg http://my.mirror.com/ubuntu.'),
            'optional' => 1,
        ),
        new EBox::Types::Text(
            'fieldName'     => 'dist',
            'printableName' => __('Dist'),
            'editable' => 1,
            'help' => __('Specify distribution\'s name, eg: precise.'),
            'optional' => 1,
        ),
    );
```

Figura 3.3: Método _table de la clase ImageCreator

Durante el desarrollo de las modificaciones también se corrigieron errores al código escrito por los desarrolladores de Zentyal, específicamente en la opción de seleccionar la arquitectura de la imagen a crear. La corrección de dicho error fue enviado al equipo de desarrolladores de Zentyal, siendo esta la primera de las colaboraciones. El código fuente del parche realizado se puede ver en la siguiente figura:

```
sub _select_architectures
{
        #Nova patch for 32 bits servers
        my $sys_arch = `getconf LONG_BIT`;

        my $i386 = {
                value => 'i386',
                printableValue => __('32 bits'),
        };

        my $amd64 = {
                value => 'amd64',
                printableValue => __('64 bits'),
        };

        if ($sys_arch == 64){
                return [$i386, $amd64,];
        }

    return [$i386,];
}
```

Figura 3.4: Método _select_architectures de la clase ImageCreator

El método **_select_protocol** permitirá que el administrador seleccione el tipo de imagen que desea compartir o sea el protocolo por el cual se va a compartir la imagen que se desea crear. Está opción hará que la solución que se desarrolla pueda ser usada por el administrador.

```
sub _select_protocol
{
    return [
        {
            value => 'nbd',
            printableValue => __('nbd'),
        },
        {
            value => 'nfs',
            printableValue => __('nfs'),
        },
    ];
}
```

Figura 3.5: Método _select_protocol de la clase ImageCreator

Hasta este momento solo se ha hecho una descripción de las vistas, aún no está funcional. Para lograr el funcionamiento del módulo con la inclusión de la solución propuesta se requiere implementar los métodos que harán posible un servicio completamente funcional.

Compilación e instalación

El proceso de compilación e instalación puede realizarse con facilidad haciendo uso de las siguientes herramientas:

1. **cd:** mediante este comando se procede a entrar en el directorio donde se encuentra el código fuente del módulo; ejemplo:

   ```
   $ cd zentyal-ltsp-3.0.5
   ```

   Si se está en el directorio donde se descargó dicho paquete.

2. **dpkg-buildpackage:** realiza el proceso de empaquetado, tiene como salida un archivo **.deb**, el cual será utilizado para la instalación.

   ```
   $ sudo dpkg-buildpackage
   ```

3. **dpkg:** toma la salida de la acción anterior y realiza la instalación en el sistema. La opción **-i** indica que lo que se hará será instalar; ejemplo:

```
$ sudo dpkg -i zentyal-ltsp.deb
```

Después de haber realizado los pasos anteriores, el módulo queda listo para ser sometido a pruebas.

3.3. Pruebas

Durante todo el ciclo de desarrollo del software es preciso velar, controlar y garantizar su correcta calidad, para así hacer posible el cumplimiento de los requerimientos que precisamente satisfacen las necesidades de la parte interesada; los clientes o stakeholder. El objetivo de un proceso de pruebas es encontrar errores de funcionamiento en determinadas situaciones a las que es expuesta la aplicación. En otras palabras, un proceso de pruebas está orientado a detectar. Este aspecto debe estar presente de forma paralela desde la concepción del producto hasta la fase de producción del mismo. Para verificar lo antes mencionado se recurre a la realización de las pruebas de software.

Las pruebas de software es un concepto que a menudo es conocido como verificación y validación. 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. Entre las técnicas que se llevan a cabo para validar el correcto funcionamiento de la aplicación se encuentra la de caja negra. [18]

3.3.1. Pruebas de caja negra

Este tipo de pruebas no están basadas en el conocimiento del diseño interior del programa sino que se enfocan en los requerimientos establecidos y en el funcionamiento del sistema. Estas pruebas se limitan a que el probador verifique los datos de entrada y estudie las salidas sin importar lo que ocurre en el interior.

Cuando se habla de un software, la prueba de caja negra se refiere a las pruebas que se llevan a cabo sobre la interfaz del mismo. Los métodos de prueba de caja negra se centran en los requisitos funcionales e intentan encontrar errores de las siguientes categorías: 1) funciones incorrectas o ausentes; 2) errores de

interfaz; 3) errores en estructuras de datos o en acceso a bases de datos externas; 4) errores de rendimiento y 5) errores de inicialización y terminación. [18]

A continuación se muestran los casos de prueba para los casos de uso fundamentales, representados a través de la matriz de datos, posibilitando de esta manera comprobar el correcto funcionamiento de la aplicación. Para la misma la V significa que es válido, la I que es inválido y NA indica que no es necesario proporcionar un valor del dato en este caso, ya que es irrelevante.

Matriz parcial del escenario para el caso de uso Crear imagen

Nombre del escenario	**Flujo donde empieza**
Escenario 1. Crear imagen.	Flujo básico

Tabla 3.6: Matriz parcial del escenario para el caso de uso Crear imagen.

Matriz de caso de prueba para el caso de uso Crear imagen

ID del caso de prueba	**Escenario/ Condición**	**Arquitectura**	**Protocolo**	**Imagen de cliente pesado**	**Dirección del repositorio**	**Resultado esperado**
Continúa en la próxima página...						

RC1	Escenario 1. Crear imagen.	V	V	V	N/A	Muestra un mensaje informando que el proceso será mostrado en el Panel de Control. Visualiza una nueva imagen en la lista de imágenes.

Tabla 3.7: Matriz de caso de prueba para el caso de uso Crear imagen.

Matriz de caso de prueba con valores para los datos del caso de uso Crear imagen

ID del caso de prueba	Escenario/ Condición	Arquitectura	Protocolo	Imagen de cliente pesado	Dirección del repositorio	Resultado esperado
Continúa en la próxima página...						

RC1	Escenario 1. Crear imagen.	i386	NFS	0	http:// novarepo.uci .cu/nova	Visualiza una nueva imagen en la lista de imágenes.

Tabla 3.8: Matriz de caso de prueba con valores para los datos del caso de uso Crear imagen.

Matriz parcial del escenario para el caso de uso Actualizar imagen

Nombre del escenario	**Flujo donde empieza**
Escenario 1. Actualizar imagen.	Flujo básico

Tabla 3.9: Matriz parcial del escenario para el caso de uso Actualizar imagen.

Matriz de caso de prueba para el caso de uso Actualizar imagen

ID del caso de prueba	**Escenario/ Condición**	**Arquitectura**	**Protocolo**	**Imagen de cliente pesado**	**Dirección del repositorio**	**Resultado esperado**
Continúa en la próxima página...						

RC1	Escenario 1. Actualizar imagen.	N/A	N/A	N/A	N/A	Muestra un mensaje informando que el proceso será mostrado en el Panel de Control. Actualiza la imagen seleccionada por el administrador.

Tabla 3.10: Matriz de caso de prueba para el caso de uso Actualizar imagen.

Matriz de caso de prueba con los valores para los datos del caso de uso Actualizar imagen

ID del caso de prueba	Escenario/ Condición	Arquitectura	Protocolo	Imagen de cliente pesado	Dirección del repositorio	Resultado esperado
Continúa en la próxima página...						

RC1	Escenario 1. Actualizar imagen.					Actualiza la imagen seleccionada por el administrador.

Tabla 3.11: Matriz de caso de prueba con los valores para los datos del caso de uso Actualizar imagen.

Matriz parcial del escenario para el caso de uso Instalar aplicación

Nombre del escenario	**Flujo donde empieza**
Escenario 1. Instalar aplicación.	Flujo básico
Escenario 2. No se puede instalar una nueva aplicación.	Flujo básico

Tabla 3.12: Matriz parcial del escenario para el caso de uso Instalar aplicación.

Matriz de caso de prueba para el caso de uso Instalar aplicación

ID del caso de prueba	**Escenario/ Condición**	**Listado de aplicaciones**	**Resultado esperado**
Continúa en la próxima página. . .			

RC1	Escenario 1. Instalar aplicación.	V	Muestra un mensaje informando que el proceso de instalación se puede observar en el Panel de Control. Instala las aplicaciones.
RC2	Escenario 2. No se puede instalar una nueva aplicación.	I	Muestra un mensaje de error, vuelta al escenario 1.

Tabla 3.13: Matriz de caso de prueba para el caso de uso Instalar aplicación.

Matriz de caso de prueba con valores para los datos del caso de uso Instalar aplicación

ID del caso de prueba	**Escenario/ Condición**	**Listado de aplicaciones**	**Resultado esperado**
RC1	Escenario 1. Instalar aplicación.	firefox geany	Muestra un mensaje informando que el proceso de instalación se puede observar en el Panel de Control. Instala las aplicaciones.
Continúa en la próxima página...			

RC2	Escenario 2. No se puede instalar una nueva aplicación.	La entrada contiene caracteres inválidos. Todos los caracteres alfanuméricos más los caracteres \ & . @ ? ' : + y los espacios están permitidos.	Muestra un mensaje de error, regresa al escenario 1.

Tabla 3.14: Matriz de caso de prueba con valores para los datos del caso de uso Instalar aplicación.

Al realizar las pruebas y teniendo en cuenta el nivel de complejidad de la solución con respecto a los requerimientos especificados para la misma, se obtuvo como resultado una aplicación donde no existen no conformidades.

El módulo fue liberado por CALISOFT[3] durante las pruebas realizadas a la versión de Nova para servidores 2013.

Conclusiones del capítulo

En el presente capítulo se realizó la implementación de la aplicación, para lo cual se llevó a cabo el diseño de los componentes del sistema y las relaciones entre los mismos, realizando una modelación del Diagrama de Despliegue del Sistema. Se muestra el código fuente de los principales métodos presentes en la aplicación y se resalta la corrección de errores en el código escrito por los desarrolladores de la aplicación inicial. En el caso de las pruebas realizadas al sistema se definió el uso de la técnica de caja negra para validar el correcto funcionamiento de la solución desarrollada, mostrando así las entradas y salidas válidas de la aplicación según los requerimientos definidos, sin arrojar no conformidades. Se obtuvo como resultado un sistema que funciona de la manera esperada.

[3]Se refiere al Centro Nacional de Calidad de Software.

Conclusiones

Una vez concluida la adaptación del módulo para clientes ligeros de la plataforma de desarrollo Zentyal, al entorno de la empresa cubana se puede concluir lo siguiente:

- Se sistematizó el funcionamiento y las tecnologías para la administración de las computadoras sin disco y clientes ligeros.
- Se desarrolló un módulo capaz de adaptarse a condiciones de baja conectividad a internet y sobrecarga de clientes por servidor. Para ello se reutilizó el conocimiento obtenido en el desarrollo de OSplugger.
- Se validó el correcto funcionamiento del módulo propuesto mediante casos de pruebas de caja negra, obteniéndose buenos resultados y la liberación de la herramienta por la empresa CALISOF.

Dando así cumplimiento a los objetivos trazados en esta investigación, durante la cual se logró establecer comunicación con el equipo de desarrollo de Zentyal e insertar el equipo de desarrolladores de Nova para servidores a la comunidad internacional de Zentyal realizando una primera colaboración durante el desarrollo de la presente investigación.

Recomendaciones

El desarrollo de la presente investigación se efectuó de forma satisfactoria, sin embargo, para lograr alcanzar mejores resultados se recomienda:

- Realizar un estudio sobre otras aplicaciones especializadas en la administración de clientes ligeros existentes en la actualidad, para tomar de ellas las mejores características.
- Desarrollar una opción que sea capaz de importar imágenes creadas con anterioridad.
- Desarrollar una opción que sea capaz de crear imágenes para clientes ligeros a partir de una imagen para liveCD (iso).

Glosario de Términos

ACLs Lista de Control de Acceso. Una ACL especifica qué usuarios o procesos del sistema tienen acceso a los objetos, así como las operaciones que se permiten en los objetos dados.

Arquitectura ARM ARM. Es una arquitectura RISC (Reduced Instruction Set Computer=Ordenador con Conjunto de Instrucciones Reducidas) de 32 bits desarrollada por ARM Holdings. La arquitectura ARM es el conjunto de instrucciones de 32 bits más ampliamente utilizado en unidades producidas. Es la arquitectura en la cual está basado el prototipo de cliente ligero cubano desarrollado por el ICID.

AUFS AUFS. Es un sistema de archivo unificado, que apila un conjunto de directorios y los muestra como si fuese uno solo.

Bash Bash. 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) — haciendo un juego de palabras (born-again significa renacimiento) sobre el Bourne shell (sh), que fue uno de los primeros intérpretes importantes de Unix.

DHCP Dynamic Host Configuration Protocol. Protocolo de red que permite a los clientes de una red IP obtener sus parámetros de configuración automáticamente. Se trata de un protocolo de tipo cliente/servidor en el que generalmente un servidor posee una lista de direcciones IP dinámicas y las

va asignando a los clientes cuando están libres, sabiendo en todo momento quién ha estado en posesión de esa IP, cuánto tiempo la ha tenido y a quién se la ha asignado después.

Cliente-Servidor Arquitectura Cliente-Servidor. La arquitectura Cliente/Servidor es un modelo basado en la idea del servicio, en el que el cliente es un proceso consumidor de servicios y el servidor es un proceso proveedor de servicios. Además esta relación está establecida en función del intercambio de mensajes que es el único elemento de acoplamiento entre ambos. Existen dos elementos fundamentales sobre la implementación de esta arquitectura; el cliente que inicia la conexión al realizar una petición al servidor y el servidor que es quien le provee al cliente el servicio que se desea consultar.

Gentoo Gentoo Linux. Distribución de Linux orientada a usuarios con cierta experiencia en el uso de SO de este tipo.

GPL Licencia Pública General. Llamada comúnmente GPL de GNU, se usa para la mayoría de los programas de GNU y para más de la mitad de los paquetes de software libre. La Licencia Pública General de GNU es una licencia libre, sin derechos para software y otro tipo de obras. Pretende garantizar la libertad de compartir y modificar todas las versiones de un programa, para asegurarse de que sigue siendo software libre.

gPXE antes Etherboot. Es una implementación de fuente abierta del Preboot Execution Environment (PXE) y cargador de arranque. Puede ser usado para habilitar a los computadores que no tienen soporte para el PXE para que puedan cargar desde la red.

Hardware Hardware. Son todas las partes tangibles de un sistema informático; sus componentes son: eléctricos, electrónicos, electromecánicos y mecánicos. Son cables, gabinetes o cajas, periféricos de todo tipo y cualquier otro elemento físico involucrado; contrariamente, el soporte lógico es intangible y es llamado software.

Initramfs Initramfs. Es una solución introducida por el núcleo de Linux en su versión 2.6. Se utiliza como el primer sistema de ficheros raíz que la máquina inicia.

Kerberos Network Authentication Protocol. Está diseñado para proporcionar una autenticación fuerte para aplicaciones cliente/servidor utilizando criptografía de clave secreta. Kerberos fue creado por el MIT (Instituto Tecnológico de Massachusetts) para dar solución a los problemas de seguridad en la red.

LDAP Lightweight Directory Access Protocol. Protocolo a nivel de aplicación que permite el acceso a un servicio de directorio ordenado y distribuido para buscar diversa información en un entorno de red. LDAP también se considera una base de datos (aunque su sistema de almacenamiento puede ser diferente) a la que pueden realizarse consultas.

LDM LTSP Display Manager. Administrador de pantalla de acceso desarrollado por el proyecto LTSP para gestionar el acceso a sesión de los usuarios.

Plug-in Plugin. Un plug-in (o complemento) es un componente de software que añade una característica específica a una aplicación de software ya existente. Cuando una aplicación es compatible con plug-ins, permite la personalización.

Postgresql Postgresql. PostgreSQL es un potente sistema de base de datos objeto-relacional de código abierto. Cuenta con más de 15 años de desarrollo activo y una arquitectura probada que se ha ganado una sólida reputación de fiabilidad e integridad de datos. Se ejecuta en los principales sistemas operativos que existen en la actualidad.

PXE Preboot eXecution Environment. Entorno para arrancar e instalar el SO en ordenadores a través de una red, de manera independiente de los dispositivos de almacenamiento de datos disponibles (como discos duros) o de los sistemas operativos instalados.

QT QT4. Qt es una biblioteca multiplataforma ampliamente usada para desarrollar aplicaciones con interfaz gráfica de usuario, así como también para el desarrollo de programas sin interfaz gráfica, como herramientas para la línea de comandos y consolas para servidores.

Software Software. Equipamiento lógico o soporte lógico de un sistema informático, que comprende el conjunto de los componentes lógicos necesarios que hacen posible la realización de tareas específicas, en contraposición a los componentes físicos que son llamados hardware.

Switch Switch. Un conmutador o switch es un dispositivo digital lógico de interconexión de redes de computadoras que opera en la capa de enlace de datos del modelo OSI.

Syslinux Syslinux. Abarca un conjunto de gestores de arranque ligeros, para arrancar ordenadores en el SO Linux. Entre los que se puede encontrar PXELINUX, usado para arrancar desde un servidor de red con el sistema entorno de ejecución de Pre-arranque (PXE).

TFTP Trivial file transfer Protocol. Protocolo de transferencia muy simple semejante a una versión básica de FTP. TFTP a menudo se utiliza para transferir pequeños archivos entre ordenadores en una red, como cuando un terminal X Window o cualquier otro cliente ligero arranca desde un servidor de red.

Referencias bibliográficas

[1] M. Hurtado Fedoróvich, "*Nova al servicio de los clientes ligeros y máquinas sin disco*." [En línea] Disponible en la red de redes, 2010. [Citado: Noviembre 2012].

[2] Zentyal-LTSP, "*Configuración de un servidor de clientes ligeros con Zentyal*." [En línea] Disponible en http://doc.zentyal.org/es/ltsp.html, 2012. [Citado: Febrero 2013].

[3] S. Arango Winograd, "*Optimización de las descargas de repositorios en la distribución GNU/Linux Nova*." [En línea] Disponible en http://repositorio_institucional.uci.cu/jspui/handle/ident/JCE-2012-F318-P131-Ponencia-2701, 2012. [Citado: Noviembre 2012].

[4] LTSP, "*Welcome the LTSP Ubuntu page*." [En línea] Disponible en http://wiki.ltsp.org/wiki/Category:Ubuntu, 2012. [Citado: Enero 2013].

[5] M. Hurtado Fedoróvich, "*Manual de usuario OSPLUGGER - 1.0*," 2012. [Citado: Enero 2013].

[6] E. Rosales Rosa, "*Módulo para la administración de NAS en Nova para Servidores*." [En línea] Disponible en http://repositorio_institucional.uci.cu/jspui/handle/ident/TD_05162_12, 2012. [Citado: Enero 2013].

[7] Zentyal, "*Zentyal, servidor linux para Pymes*." [En línea] Disponible en http://doc.zentyal.org/es/presentation.html, 2012. [Citado: Febrero 2013].

[8] Perl, "*The Perl Programming Language*." [En línea] Disponible en http://www.perl.org/. [Citado: Febrero 2013].

[9] J. M. López, "*Entorno de desarrollo multilenguaje ligero pero potente*." [En línea] Disponible en http://geany.softonic.com/linux, 2011. [Citado: Febrero 2013].

[10] OpenUp, "*Concepto de Caso de Uso*." [En línea] Disponible en http://epf.eclipse.org/wikis/openup/, 2010. [Citado: Marzo 2013].

[11] Y. Fernández del Monte, "*Metodología Nova OpenUp para el desarrollo de distribuciones GNU/Linux*," 2012. [Citado: Febrero 2013].

[12] E. Puente Fuente, "*Soporte para la Arquitectura de Computadora ARM en Nova para Cliente Ligero*." [En línea] Disponible en http://repositorio_institucional.uci.cu/jspui/handle/ident/TD_05034_11, 2011. [Citado: Febrero 2013].

[13] I. F. Sommerville, "*Overview*." Disponible en la red de redes, octava edición. [Citado: Abril 2013].

[14] Zentyal, "*Personalización avanzada de servicios*." [En línea] Disponible en http://doc.zentyal.org/es/develop.html, 2012. [Citado: Abril 2013].

[15] L. Craig, "*UML y Patrones*." Disponible en la red de redes, 2004. [Citado: Marzo 2013].

[16] R. S. Pressman, "*Software engineering a practitioner's approach*." Disponible en la red de redes, séptima edición. [Citado: Abril 2013].

[17] H. Cáceres Navarro, "*Propuesta de integración del Gestor de Documentos Administrativos eXcriba con una Infraestructura de Clave Pública*." [En línea] Disponible en http://repositorio_institucional.uci.cu/jspui/handle/ident/TD_05027_11, 2011. [Citado: Marzo 2013].

[18] J. L. Esperanza Fernández, "*Sistema para el Análisis de Sensibilidad en la plataforma BioSyS*." [En línea] Disponible en http://repositorio_institucional.uci.cu/jspui/handle/ident/TD_04414_11, 2012. [Citado: Abril 2013].

Bibliografía

1. “AUFS” [En línea], Abril 2013, [Consultado: Mayo 2013]. Disponible en la dirección web: http://aufs.sourceforge.net/

2. “Commands” [En línea], 2012, [Consultado: Octubre 2012]. Disponible en la dirección web: http://wiki.ltsp.org/wiki/Commands

3. “Concepts” [En línea], 2012, [Consultado: Octubre 2012]. Disponible en la dirección web: http://wiki.ltsp.org/wiki/Concepts

4. “Configuration and Maintenance” [En línea], 2012, [Consultado: Octubre 2012]. Disponible en la dirección web: http://wiki.ltsp.org/wiki/Configuration

5. “Debian+NFS” [En línea], 18 de Septiembre de 2012, [Consultado: Noviembre 2012]. Disponible en la dirección web: http://www.debian.org/distrib/netinst.es.html

6. “DisklessUbuntuHowto” [En línea], 2011, [Consultado: Septiembre 2012]. Disponible en la dirección web: https://help.ubuntu.com/community/DisklessUbuntuHowto

7. “Entorno de desarrollo de nuevos módulos” [En línea], 2012, [Consultado: Octubre 2012]. Disponible en la dirección web: http://doc.zentyal.org/es/develop.html

8. “Entorno de desarrollo multilenguaje ligero pero potente” [En línea], 2011, [Consultado: Febrero 2013]. Disponible en la dirección web: http://geany.softonic.com/linux

9. “Geany” [En línea], 2012, [Consultado: Febrero 2013]. Disponible en la dirección web: http://www.geany.org/

10. “Importación de datos de configuración” [En línea], 2012, [Consultado: Octubre 2012]. Disponible en la dirección web: http://doc.zentyal.org/es/develop.html

11. “Informatización” [En línea], 2010, [Consultado: Noviembre 2012]. Disponible en la dirección web: http://www.hlg.jovenclub.cu/mic/web/index.php/Principal/informatizacion

12. “Introduction to OpenUP” [En línea], 2010, [Consultado: Marzo 2013]. Disponible en la dirección web: http://epf.eclipse.org/wikis/openup/index.htm

13. “Instalar Debian a través de Internet” [En línea], 2012, [Consultado: Septiembre 2012]. Disponible en la dirección web: http://www.debian.org/releases/stable/powerpc/ch04s05.html.es

14. “Installation” [En línea], 2012, [Consultado: Octubre 2012]. Disponible en la dirección web: http://wiki.ltsp.org/wiki/Installation

15. “Integration” [En línea], 2012, [Consultado: Octubre 2012]. Disponible en la dirección web: http://wiki.ltsp.org/wiki/Integration

16. “Kerberos: The Network Authentication Protocol” [En línea], 06 Febrero 2012, [Consultado: Abril 2013]. Disponible en la dirección web: http://web.mit.edu/kerberos/

17. “Linux NFS faq” [En línea], 2012, [Consultado: Abril 2013]. Disponible en la dirección web: http://nfs.sourceforge.net/

18. “LTSP. Configuración de un servidor de clientes ligeros con zentyal” [En línea], 2012, [Consultado: Octubre 2012]. Disponible en la dirección web: http://doc.zentyal.org/es/ltsp.html

19. “MIC. Informatización de la sociedad” [En línea], 2010, [Consultado: Noviembre 2012]. Disponible en la dirección web: http://www.mic.gov.cu/

20. “NFS” [En línea], 2008, [Consultado: Febrero 2013]. Disponible en la dirección web: http://doc.ubuntu-es.org/Network_File_System

21. “NFS Configuration” [En línea], 2007, [Consultado: Octubre 2012]. Disponible en la dirección web: http://ltsp4.2.revamp-it.ch/twiki/bin/view/ltsp/NFS

22. “Network Block Device” [En línea], 2012, [Consultado: Febrero 2013]. Disponible en la dirección web: http://nbd.sourceforge.net/

23. “Perl array exec() and system() function” [En línea], 2011 [Consultado: Octubre 2012]. Disponible en la dirección web: http://perl.about.com/od/programmingperl/qt/perlexecsystem.htm

24. “Perl Doc” [En línea], 2010, [Consultado: Octubre 2012]. Disponible en la dirección web: http://www.ayni.com/perldoc/functions/system.html

25. “Personalización avanzada de servicios” [En línea], 2012, [Consultado: Abril 2013]. Disponible en la dirección web: http://doc.zentyal.org/es/develop.html

26. “Primeros pasos con Zentyal” [En línea], 2012, [Consultado: Septiembre 2012]. Disponible en la dirección web: http://doc.zentyal.org/es/firststeps.html

27. “Primeros pasos con Zentyal” [En línea], 2012, [Consultado: Octubre 2012]. Disponible en la dirección web: https://help.ubuntu.com/community/DisklessUbuntuHowto

28. “Protocolo Network Block Device (servidor)” [En línea], 2009, [Consultado: Marzo 2013]. Disponible en la dirección web: http://packages.trisquel.info/es/taranis-updates/nbd-server

29. “Servicio de clientes ligeros” [En línea], 2012 [Consultado: Octubre 2012]. Disponible en la dirección web: http://doc.zentyal.org/es/ltsp.html

30. “UbuntuLTSP/LTSPWithoutNFS” [En línea], 2011, [Consultado: Octubre 2012]. Disponible en la dirección web: https://help.ubuntu.com/community/UbuntuLTSP/LTSPWithoutNFS

31. “Zentyal: servidor Linux para pymes” [En línea], 2010, [Consultado: Noviembre 2012]. Disponible en la dirección web: http://doc.zentyal.org/es/presentation.html

Anexo I

----- Original Message -----
From: "Yannier P. Escalona" <yannierp@estudiantes.uci.cu>
To: <jar@gedeme.co.cu>
Sent: Tuesday, February 12, 2013 11:31 AM
Subject: buenos días

Buenos días: soy estudiante de 5to año en la UCI,
miembro del equipo de desarrollo de la distribución cubana
de GNU/Linux Nova, específicamente de la versión para servidores
y estoy desarrollando la Tesis sobre clientes ligeros.
La solución que brindamos está trabajando en la actualidad en el MIC
con un total de 56 clientes realizando peticiones de forma concurrentes
en el servidor.
Para el desarrollo de la documentación necesito información referente a:
el año de entrada al país de los primeros clientes ligeros, la cantidad,
el software que se
utiliza para la administración de los mismos, así como la cantidad
existente actualmente en el país. Si está al alcance de usted y me puede
facilitar más información se lo agradecería.
Estoy siendo tutorado por los ingenieros Abel Firvida Donestévez y Haniel Cáceres Navarro.

Gracias de antemano.
Saludos atentamente Yannier.

Figura 6: Entrevista al Ingeniero Fernando Fernández Fernández

Yannier.
Es muy interesante tu trabajo, GEDEME en el año 2009 comenzó las cooperaciones con la UCI y uno de los temas que se nos quedo pendientes fue precisamente el software de gestión para las máquinas sin disco y clientes ligeros sobre plataforma Linux, específicamente con la versión Nova que la UCI nos entrego, en estos momentos estamos coordinando con la vice-rectora de producción de la UCI para retomar los acuerdos de cooperación y este precisamente será uno de nuestros puntos de interés, espero poder intercambiar un poco al respecto con ustedes.
De la información que nos pides.
Las clientes ligeros arribaron a Cuba en el 2008 con un total de 100 000U de la configuración MB Intel DG201GYL, en el 2011 se ensamblaron 1500 con MB IPXV de Asus, en el 2012 se ensamblaron y distribuyeron 6000 de la misma configuración, en el 1er trimestre del 2013 se terminaran otras 3000.
Espero estos datos te sirvan.
Salu2

Ing. Fernando Fernandez Fdez.
Esp. Princ. Dpto Comercial.
UEB GEDEME Sevimatica.
Telf: 260 0510 o 260 3985.
email: fernando@gedeme.co.cu

Figura 7: Respuesta del Ingeniero Fernando Fernández Fernández

Anexo II

Soy el Administrador de Redes del Archivo Histórico Provincial de Camagüey, mi
nombre completo es Mario Tomas Martínez Gómez.
En cuanto a la cantidad de clientes que he logrado conectar es alrededor de 40, el servidor tiene las siguientes
características: una board intel s3000ah, micro procesador intel xeon a 2.13 GHz (tiene 2 núcleos),
con 4gb de memoria RAM y 2 HDD de 160 GB. Soy técnico medio en informática.
Para lograr un funcionamiento aceptable se le restringe a los clientes ejecutar los juegos de GNOME,
ni videos, el decoding del video hace consumir mucho a los clientes,
tampoco se pueden emulador aplicaciones de windows, y en cuanto a la
navegación se debe velar por las páginas web que contengan flash pues van directo a
trabajar en el servidor.

Saludos
Tomy

Figura 8: Información obtenida del administrador de redes del Archivo Histórico Provincial de Camagüey.

Anexo III

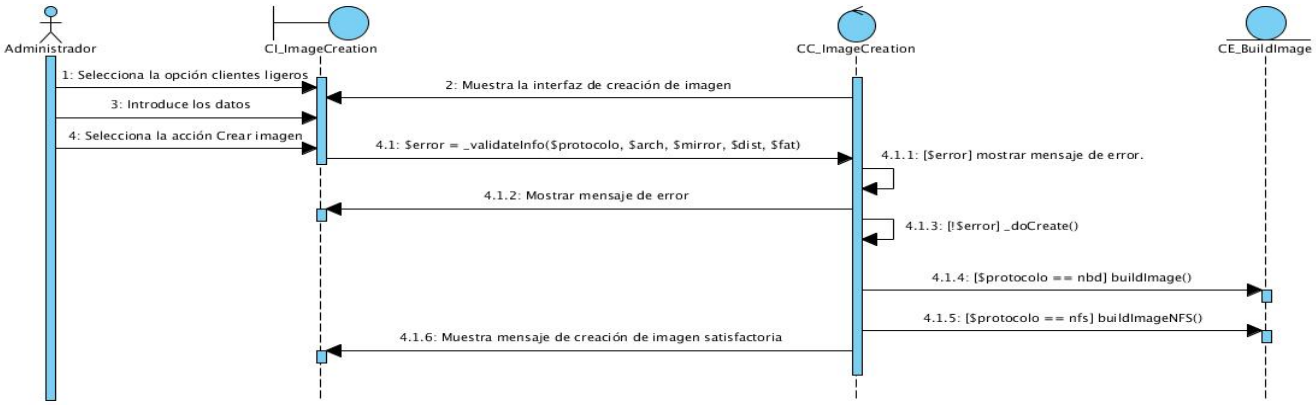

Figura 9: Diagrama de secuencia caso de uso Crear imagen.

Figura 10: Crear nueva imagen.

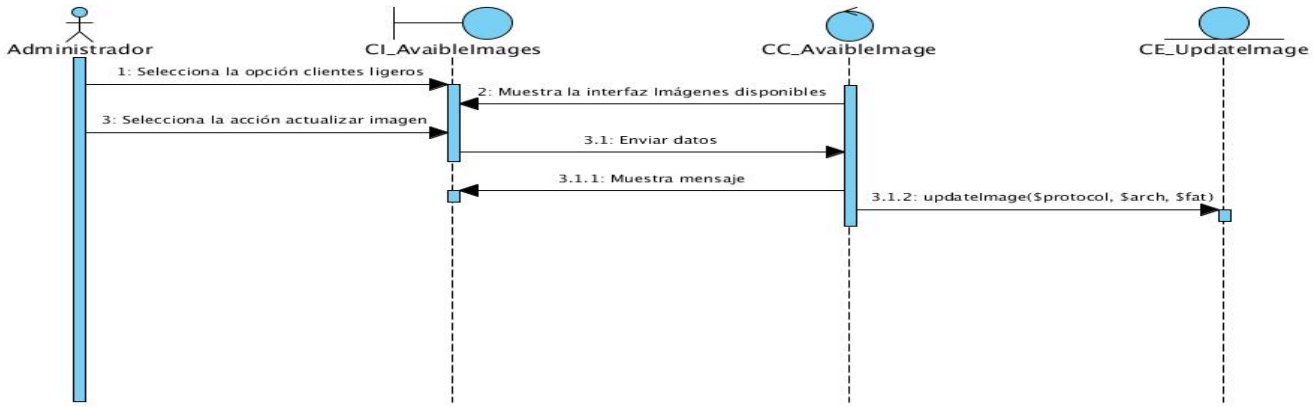

Figura 11: Diagrama de secuencia caso de uso Actualizar imagen.

Figura 12: Listado de imágenes disponibles. Observar la opción actualizar, dentro de la sesión Acción.

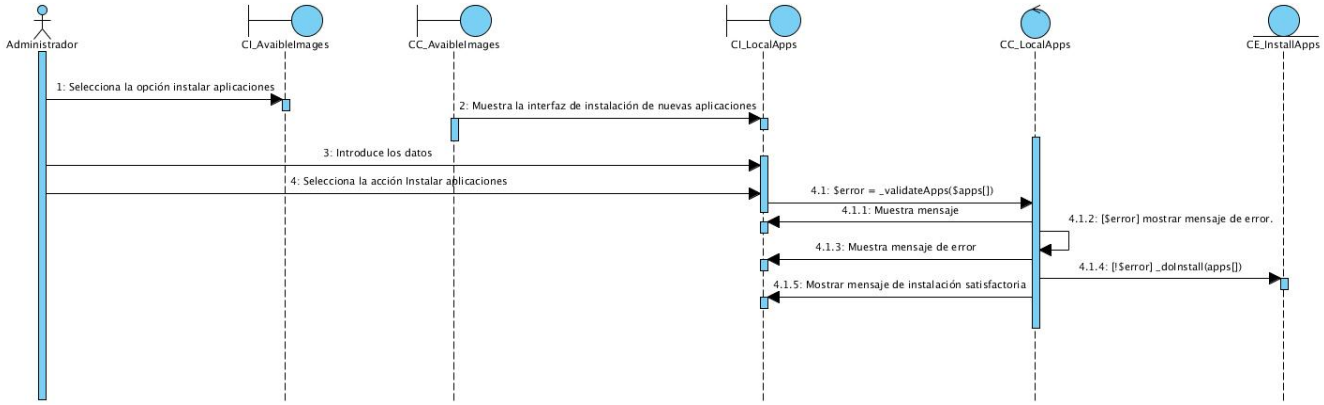

Figura 13: Diagrama de secuencia caso de uso Instalar aplicaciones locales.

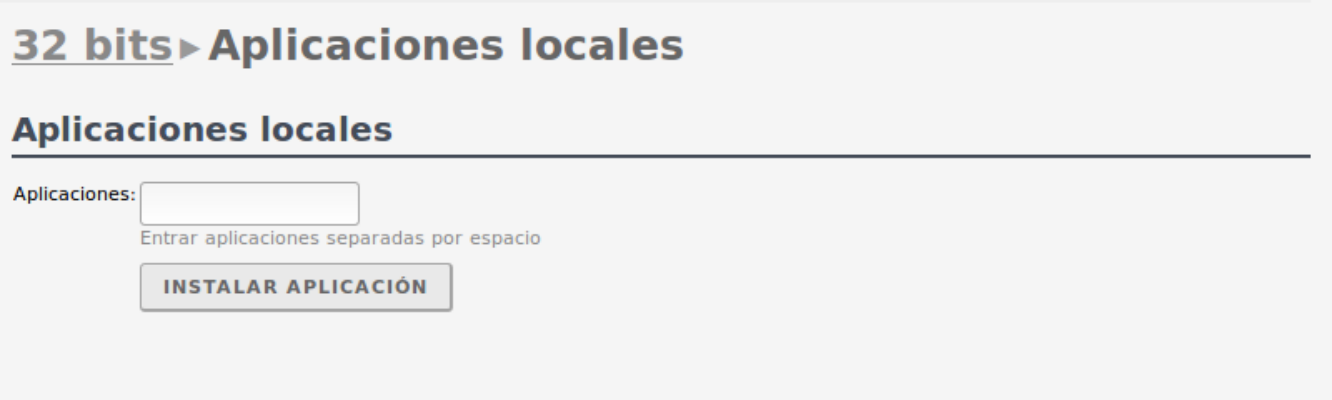

Figura 14: Interfaz instalar Aplicaciones locales.

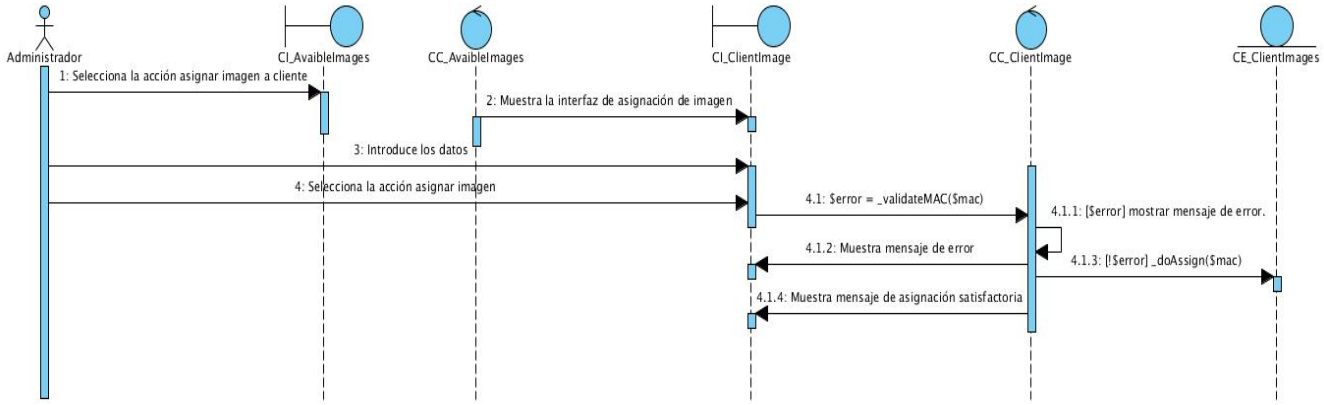

Figura 15: Diagrama de secuencia caso de uso Asignar imagen a cliente.

Figura 16: Interfaz de Asignación de imágenes.