Universidad de las Ciencias Informáticas

Facultad 2

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

Instalación automática de Gclient para el sistema GRHS

Autores:

Yoel Hernández Camejo

Arianna Martínez Sánchez

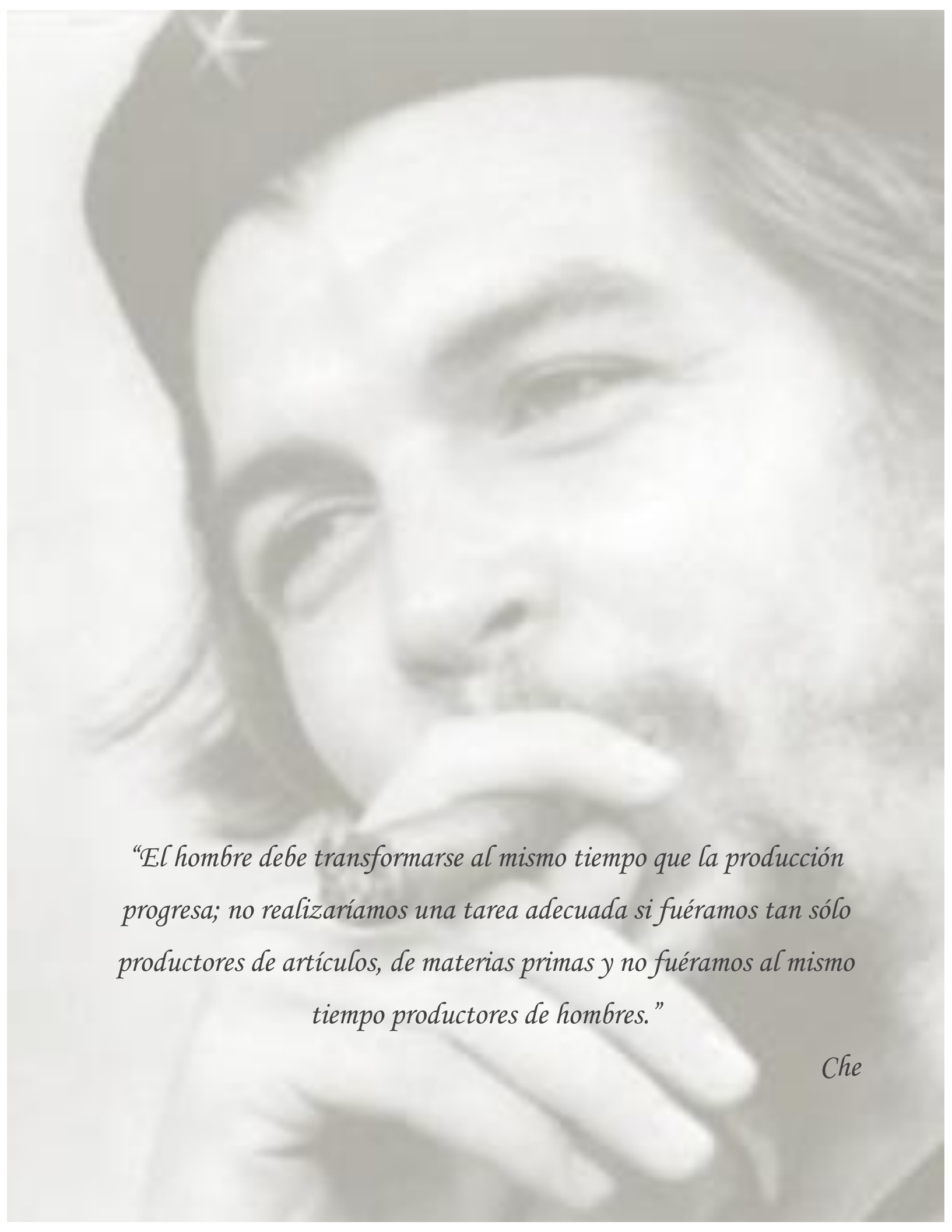
Tutores:

MSc. Ramón Alexander Anglada Martínez

Ing. Yenlys Guerra Dávila

La Habana, junio de 2015

"Año 57 de la Revolución"



"El hombre debe transformarse al mismo tiempo que la producción progresa; no realizaríamos una tarea adecuada si fuéramos tan sólo productores de artículos, de materias primas y no fuéramos al mismo tiempo productores de hombres."

Che

DECLARACIÓN DE AUTORÍA

Declaramos ser autores del trabajo de diploma titulado “Instalación automática de Gclient para el sistema GRHS” y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales sobre ésta, con carácter exclusivo.

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

Yoel Hernández Camejo
Autor

Arianna Martínez Sánchez
Autor

Ing. Yenlys Guerra Dávila
Tutor

MSc. Ramón Alexander Anglada Martínez
Tutor

AGRADECIMIENTOS

A mi mami María y mi papi Osvaldo por todo su amor, cariño, apoyo y comprensión; por inculcarme sus valores, ser mis guías, mi ejemplo a seguir, por demostrarme siempre que el que persevera triunfa, los amo.

A mí adorado hermano Alejandro por ser mi confidente, mi amigo, el pintor de mis días y por dejarme ser su ejemplo a seguir.

A mis tíos Ana y Ballester por ser mis segundos papás, por enseñarme a luchar por la vida y darme tanto cariño, son mi inspiración, los adoro.

A mi novio y compañero de tesis por todos esos momentos que hemos vivido juntos, por las dificultades y alegrías que hemos compartido. Por apoyarme en los momentos más difíciles de la tesis haciéndome reír en los momentos de estrés. Gracias por acompañarme en todas mis locuras y desvelos, gracias por completar mi vida de una forma tan linda y tan única.

A Zove, Leo, Lipi, Nataly y Tamy por acogerme en su casa como un miembro más de la familia, gracias por brindarme su apoyo en todo momento.

A mis grandes amigos Eblis, Tito, Yarissel, Aymara, Roxana, Lorena, Lisvet, Pedro, Lester, Eduardo, Idalmis, Jorge, el Chino, el Mamao, Madelis, Zailet y Carlos que han estado conmigo en las buenas y en las malas, me permitieron formar parte de su vida durante estos años y con los cuales he pasado excelentes e inolvidables momentos. Los quiero mucho.

A mis compañeros de aula en especial a Darelys, Deilis, Aliuska, Naimí, Karla y Maiko, por aguantar mis malcriadeces y permitirme contar con ellos en lo que necesitara. Nunca los olvidaré.

A mis tutores por su apoyo, confianza y dedicación para la realización de la tesis. Gracias por las horas extras que nos dedicaron.

A todos los profesores que han contribuido con mi formación profesional y personal.

A todos los que de una forma u otra han contribuido a la realización del presente trabajo.

A todos aquellos que no he mencionado por falta de espacio pero que estuvieron ahí brindando su ayuda y compartiendo conmigo en estos cinco años.

A todos, muchas gracias.

Arianna

A mi mami Zove, a mi hermana Lipi, a mi hermanita Tamy y a mi padrastro Leo por todo su amor, cariño, apoyo y comprensión; por darme todo lo que pudieron y lo que no para que pudiera cumplir este gran sueño.

A mi beba por ser la que me impulsa a seguir para poder darle una vida cada vez mejor.

A toda mi familia por creer siempre en mí y ser un tesoro en mi vida.

A mi papá que aunque no me ha criado, siempre me ha apoyado en todas mis decisiones y lo quiero con la vida.

A mi compañera de tesis y gran amor por todos esos momentos que hemos vivido juntos, por todo el apoyo y cariño que siempre me ha dado, por cada noche de desvelo y no solo en la tesis sino en toda la carrera. Gracias, siempre vas a ser alguien muy importante en mi vida.

A Ana, María, Osvaldo, Balle y Ale por acogerme en su casa como un miembro más de la familia, gracias por brindarme su apoyo en todo momento.

A mis grandes amigos Yordi, Eblis, Tito, Pedro, Lisvet, Lester, Eduardo, Idalmis, Zailet, Carlos, Maiko, Mamao, Madelis, Jorge y Andy que han estado conmigo en las buenas y en las malas, me permitieron formar parte de su vida durante estos años y con los cuales he pasado excelentes e inolvidables momentos. Los quiero mucho.

A mis compañeros de aula, por estos años que hemos compartido y por permitirme contar con ellos en lo que necesitara. Nunca los olvidaré.

A mis tutores, porque más que tutores son mis compañeros de tesis y mis amigos.

A todos los que de una forma u otra han contribuido a mi formación como profesional y como persona.

A todos aquellos que no he mencionado por falta de espacio pero que estuvieron ahí brindando su ayuda y compartiendo conmigo en estos cinco años.

A todos, muchas gracias.

Yoel

DEDICATORIA

A mis padres por ser mi ejemplo a seguir, por todo su amor y dedicación. Porque a pesar de todo, son los mejores y lo que más quiero en el mundo.

A mi travieso hermano, por ser el protagonista de mis disgustos y alegrías. Te quiero mucho.

A mi tía Ana y a mi tío Ballester por quererme como si fuera la niña de sus ojos, por darme tanto amor y cariño y demostrarme que familia no es aquel que lleva tu sangre sino quien se preocupa por ti y da hasta lo imposible por tu bienestar. Los adoro.

A Yoel, por ser mi gran compañero, mi mejor amigo y el gran amor de mi vida.

A mis grandes amigos por estar ahí cada día incondicionalmente y soportar mis berrinches.

Arianna

A mi mamá por ser incondicional con sus hijos y dedicarnos toda su vida. Por darme tanto amor y por enseñarme con su ejemplo a enfrentar la vida.

A mi hermana Lipi que es esa rosa llena de espina pero con una fragancia y belleza incomparable.

A mi hermanita Tamy que es la dulzura de la casa y la perdición de sus padres.

A mi segundo papá que ha hecho tanto por mí y por las mujeres que tanto quiero, además de ser mi ejemplo a seguir como persona, profesional y como padre.

A mi pequeña que es lo que más quiero en la vida.

Yoel

RESUMEN

En la actualidad gran parte de las empresas e instituciones poseen y manejan considerables recursos de hardware y software, por lo que llevar un control de estos se hace necesario para los propietarios o administradores de las redes de computadoras. A raíz de esto la Universidad de las Ciencias Informáticas se dio a la tarea de realizar un sistema que permita realizar la gestión de los recursos de hardware y software de una red de computadoras. El sistema Gestor de Recursos de Hardware y Software está constituido por tres subsistemas fundamentales: Gserver, Gadmin y Gclient. El sistema Gclient es el encargado de obtener la información del hardware y software de las computadoras donde se encuentre instalado y detecta algún cambio en la configuración de dicho computador. Actualmente la instalación de Gclient no está automatizada, por lo que resulta complejo el despliegue en redes de computadoras extensas. Para dar solución a estos inconvenientes, después de haber realizado un análisis de las tendencias y paradigmas actuales de los sistemas dedicados a la instalación automática de aplicaciones en la red, se desarrolló un módulo que es capaz de automatizar el proceso de instalación de Gclient para el sistema Gestor de Recursos de Hardware y Software. Para el desarrollo de la investigación se empleó XP como metodología de desarrollo de software, Django 1.6 como marco de trabajo, PostgreSQL 9.3 como sistema gestor de bases de datos, Python 2.7 como lenguaje de programación y Eclipse Helios 3.6.1 como entorno de desarrollo integrado. Como resultado de la investigación se obtuvo un módulo que permite automatizar el proceso de instalación de Gclient para el sistema GRHS.

Palabras clave: Gclient, Gestor de Recursos de Hardware y Software, instalación automática, red de computadoras.

ÍNDICE DE CONTENIDOS

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

INTRODUCCIÓN

El avance de la ciencia y la técnica ha propiciado el desarrollo de una sociedad donde tienen un protagonismo cada vez mayor las Tecnologías de la Información y las Comunicaciones (TIC), trayendo consigo un auge en el ámbito de la informática y las redes de computadoras. Producto de las ventajas que brinda el empleo de dichas redes y los beneficios que aportan para las instituciones y entidades, su uso se ha extendido hacia todos los sectores de la sociedad.

Una red de computadoras está constituida por un conjunto de dispositivos que permiten el intercambio de información, servicios y recursos entre ellos (Alegsa, 2014). Las redes de computadoras actuales se caracterizan por un constante incremento del número, complejidad y heterogeneidad de los recursos que las componen. Los principales problemas relacionados con la expansión de estas, son la gestión del correcto funcionamiento diario y la planificación estratégica del crecimiento.

Tener el control de la información referente al hardware y software de los dispositivos que operan y componen la red, es una labor necesaria para los propietarios o administradores. Con el objetivo de satisfacer esta necesidad surgen sistemas con la capacidad de brindar información de hardware y software reflejada a modo de inventario. Cuba no está exenta del avance tecnológico del mundo actual, debido a la utilidad que representa el uso de sistemas para el control de la información concerniente a los dispositivos que integran la red, muchas instituciones han optado por su empleo.

Hace dieciséis años el país se ha visto inmerso en un gran proceso de transformaciones educacionales que están enmarcadas dentro de los programas de la Batalla de Ideas. De estos programas es que surge la Universidad de las Ciencias Informáticas (UCI), con la misión de “Formar profesionales comprometidos con su Patria y altamente calificados en la rama de la Informática. Producir aplicaciones y servicios informáticos, a partir de la vinculación estudio-trabajo como modelo de formación y servir de soporte a la industria cubana de la informática.” (UCI, 2012)

La Universidad cuenta con gran cantidad de dispositivos conectados a la red, haciendo que sea extensa y compleja. Está constituida por varios centros de desarrollo de software entre los que se encuentra el Centro de Telemática (TLM), que desarrolla sistemas y servicios informáticos en las ramas de las Telecomunicaciones y la Seguridad Informática.

El Centro de TLM dispone en la actualidad de un producto que permite realizar el inventario de hardware y software en una red de computadoras, el cual se denomina Gestor de Recursos de Hardware y Software (GRHS). Este sistema permite mantener un control sobre los recursos de hardware presentes en las computadoras de la red que han sido inventariadas, así como conocer el software que se utiliza en las estaciones de trabajo de la organización.

GRHS está constituido por tres subsistemas: servidor, administración y agente. El servidor procesa y almacena la información obtenida por los agentes, envía a las personas responsables notificaciones de las incidencias detectadas e informa al agente qué acción tomar ante incidentes que no estén contemplados en su configuración. El subsistema de administración muestra toda la información obtenida por el sistema agente-gestor y además funciona como una aplicación para visualizar información, configura las alarmas para que se activen ante las diferentes incidencias, permite configurar períodos para autorizar el cambio de algún componente en una o un grupo de estaciones de trabajo.

El agente (Gclient, Cliente de GRHS) obtiene la información del hardware y el software de las computadoras donde se encuentre instalado y detecta algún cambio en la configuración de dicho computador. Posee una instalación sencilla y altamente configurable según las necesidades de la organización que desee utilizarlo y se debe instalar en cada computadora. Esta instalación se realiza de forma directa por el personal responsable de administrar la red o por los usuarios de las computadoras.

A pesar de todas las facilidades que brinda GRHS existen actualmente un conjunto de deficiencias relacionadas con el proceso de instalación de Gclient. Instalar Gclient en una red pequeña de forma manual con personal del área de las tecnologías de la información puede ser una tarea sencilla, sin embargo a medida que aumenta el tamaño de la red es complejo el proceso de instalación de forma manual y tiene un gran costo en tiempo. Además cuando surge la necesidad de incorporar nuevas estaciones de trabajo a la red, restaurar las existentes o incorporar una actualización de Gclient, se vuelve a repetir el tedioso proceso de instalación. El método que se emplea actualmente para realizar el proceso de instalación de Gclient puede conllevar a errores ya que se realiza de manera manual.

Teniendo en cuenta la problemática antes descrita se define el siguiente **problema** a resolver: ¿Cómo automatizar el proceso de instalación de Gclient para el sistema GRHS en la red de computadoras de la UCI?

El **objeto de estudio** de la investigación se centra en el proceso de instalación automática de aplicaciones en la red.

Determinándose como **objetivo general**: Desarrollar un módulo para el sistema GRHS que permita automatizar el proceso de instalación de Gclient en la red de computadoras de la UCI.

Para ello se identifica como **campo de acción** el proceso de instalación automática de Gclient en la red de computadoras de la UCI.

Para dar solución al objetivo planteado se establecen las siguientes **tareas de la investigación**:

1. Estudio y caracterización de los sistemas existentes que realicen la instalación automática de aplicaciones en la red para conocer sus principales funcionalidades.
2. Estudio y revisión de la documentación referente al sistema GRHS para comprender sus procesos.
3. Estudio de las herramientas, metodologías y tecnologías necesarias para el desarrollo de la solución informática propuesta.
4. Análisis e identificación de requisitos para describir las características y restricciones que debe cumplir el módulo Instalación automática de Gclient para el sistema GRHS.
5. Estudio y análisis de patrones de diseño con el fin de elaborar un diseño robusto y flexible.
6. Estudio de los diferentes tipos de pruebas para verificar el correcto funcionamiento del sistema desarrollado.

La investigación se basa en la siguiente **hipótesis**: El desarrollo del módulo “Instalación automática de Gclient para el sistema GRHS”, permitirá automatizar el proceso de instalación de Gclient en la red de computadoras de la UCI.

Con el objetivo de resolver el problema, guiar la investigación y lograr el objetivo planteado se hizo necesaria la utilización de **métodos de investigación**. Los cuales están divididos en dos grandes grupos denominados teóricos y empíricos.

Métodos Teóricos:

- ✓ Analítico-Sintético: se utilizó para realizar el estudio teórico de la investigación, facilitando el análisis de documentos y la extracción de los elementos más importantes acerca del proceso de instalación automática de aplicaciones en la red.

- ✓ Inductivo - Deductivo: se empleó para inferir conocimientos lógicos como las conclusiones parciales y generales a partir del estudio de la información recopilada sobre el proceso de instalación automática de aplicaciones en la red y las características generales de los elementos de interés para la investigación.

Métodos Empíricos:

- ✓ Observación: se refleja durante todo el transcurso de la investigación fundamentalmente en la observación de la presentación de la información, así como en el funcionamiento de otros módulos del sistema GRHS y en los resultados de las pruebas.

El contenido de este trabajo consta de tres **capítulos**, definidos de la siguiente forma:

Capítulo 1: “Fundamentación Teórica”.

Incluye un estudio del estado del arte del proceso de instalación automática de aplicaciones en la red y se exponen las principales características de dicho proceso. Se hace referencia a los principales conceptos asociados al dominio del problema y se explica la metodología que guiará todo el proceso de desarrollo de software, los lenguajes de programación, el marco de trabajo (framework), el entorno de desarrollo integrado (IDE por las siglas en inglés) y el sistema gestor de bases de datos utilizado para el desarrollo del sistema.

Capítulo 2: “Planificación y diseño de la solución”.

El capítulo está constituido por la descripción de los procesos que serán objeto de automatización y el diseño de la aplicación, partiendo del estudio del estado del arte realizado en el capítulo anterior y teniendo en cuenta las fases de planificación y diseño definidas por la metodología XP. Se definen también los elementos importantes dentro del proceso de desarrollo tales como: las historias de usuario (HU), el plan de iteraciones, el plan de entregas, la arquitectura, los patrones de diseño y las tarjetas CRC (Clase, Responsabilidad y Colaboración).

Capítulo 3: “Implementación y pruebas”.

En este capítulo se muestra el desarrollo de las fases de implementación y pruebas que propone la metodología seleccionada. Se realiza una descripción de los principales artefactos a generar, como son: las tareas de ingeniería por cada HU identificada, los estándares de codificación y las pruebas empleadas para validar la solución propuesta.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

En este capítulo se presenta un estudio del estado del arte del proceso de instalación automática de aplicaciones en la red y se exponen las principales características de dicho proceso. Se hace referencia a los principales conceptos asociados al dominio del problema y se explica la metodología que guiará todo el proceso de desarrollo del software, los lenguajes de programación, el marco de trabajo, el entorno de desarrollo integrado y el sistema gestor de bases de datos utilizado para el desarrollo del sistema.

1.1 Conceptos fundamentales asociados al dominio del problema

Inventario: es el documento que contiene la relación ordenada de los bienes de una persona o comunidad. Estimación de las mercancías en almacén y de los diversos valores que componen la fortuna del comerciante.

Red de computadoras: es un conjunto de equipos informáticos y software conectados entre sí por medio de dispositivos físicos que envían y reciben impulsos eléctricos, ondas electromagnéticas o cualquier otro medio para el transporte de datos, con la finalidad de compartir información, recursos y ofrecer servicios.

Hardware: conjunto de los componentes que conforman la parte material de una computadora. Hace referencia a las partes tangibles de un sistema informático; sus componentes son: eléctricos, electrónicos, electromecánicos y mecánicos.

Software: se refiere a las instrucciones que se incorporan a un sistema informático para que este lleve a cabo una determinada función. Es el soporte lógico e intangible de los dispositivos electrónicos.

Aplicaciones web: software cliente/servidor que interactúa con usuarios y sistemas utilizando HTTP (Hypertext Transfer Protocol, Protocolo de Transferencia de Hipertexto). Desde el punto de vista del usuario, el cliente suele ser un navegador[1], en tanto que para las aplicaciones convencionales sería cualquier agente de usuario http, es decir, una aplicación que manejara ese protocolo.

1 Navegador: programa que permite navegar por Internet u otra red informática de comunicaciones.

1.2 Herramientas que realizan instalación automática

En la actualidad existen disímiles herramientas cuyo objetivo principal es automatizar el proceso de instalación de aplicaciones en una red de computadoras. Con el fin de adquirir experiencias en cuanto al funcionamiento de dichas herramientas, se realiza un estudio de los sistemas creados con esta finalidad. A continuación se exponen las características de cada uno de ellos.

Apple Remote Desktop 3: es el software que administra todos los ordenadores Mac[2] que pertenezcan a una misma red. Permite la distribución de programas y aplicaciones, automatiza determinadas tareas administrativas y crea informes detallados sobre software y hardware. Dentro de sus características se encuentran la auto-instalación (AutoInstall), que se encarga de actualizar programas de forma automática en toda una red de ordenadores Mac. Una de las funcionalidades principales y más útil es la instalación de paquetes (Install Package), la cual permite instalar paquetes de aplicaciones en toda la red en lugar de tener que hacerlo de forma individual.

Apple Remote Desktop 3 brinda más de una docena de comandos para controlar los sistemas remotos fácilmente. Permite apagar todos los sistemas a la vez y poner a dormir, despertar o reiniciar cualquiera o todos los equipos, todo ello sin moverse del escritorio. Los entornos de administración y de cliente de Apple Remote Desktop 3 han sido diseñados para trabajar sobre cualquier ordenador con Mac OS X[3] versión 10.3.9 o posterior. (Cook, 2011)

Manage Engine Desktop Central 9: es un software de gestión centralizada de ordenadores y dispositivos móviles, basado en la web. Ayuda a configurar, administrar y controlar de forma remota y automatizada miles de equipos desde una ubicación central. Ofrece una versión distribuida y multi-cliente.

Manage Engine Desktop Central 9 acepta las plataformas: Windows, Mac, iOS[4]. Es compatible con los sistemas operativos: Windows 2000 Professional, Windows XP Professional, Windows Vista, Windows 7, Windows 8, Mac 10.6 y Mac 10.7 para las computadoras de escritorio. Windows 2000, Windows 2003, Windows 2008, Windows 2008 R2, Windows 2012 para los servidores y iOS 4 o una

2 Macintosh abreviado como Mac, es la línea de ordenadores personales diseñada, desarrollada y comercializada por Apple Inc. (Incorporation).

3 Mac OS X, sistema operativo de Apple, basado en Unix, desarrollado, comercializado y vendido por Apple Inc.

4 iOS es un sistema operativo móvil de la multinacional Apple Inc.

versión posterior para dispositivos móviles. Microsoft[5] y Apple son los proveedores de dicho producto. (Correa, 2013)

EMCO Remote Installer 4.1.7: es la aplicación para instalar software remotamente, escanea la red local en busca de otros equipos conectados y permite que los usuarios instalen paquetes. Un punto interesante de EMCO Remote Installer es que trabaja silenciosamente y no interrumpe la actividad en las computadoras que reciben el software. La utilidad no se limita a la instalación de software, sino también a la desinstalación y la auditoría remota.

EMCO Remote Installer 4.1.7 fue diseñada para la gestión de paquetes con extensión MSI[6]; está en inglés y funciona en los sistemas operativos Windows XP, Windows Vista, Windows 7, Windows 8 y Windows 8.1. Cuenta con dos ediciones, la profesional (Professional Edition) y la libre (Free Edition). La edición profesional es una herramienta comercial que proporciona características de implementación y auditoría de software avanzada. Permite instalar y desinstalar software en un número ilimitado de ordenadores remotos en el ámbito de una sola operación. Mientras que la edición libre es una herramienta gratuita con características de implementación básicas, permite instalar y desinstalar software de forma remota en un máximo de 5 computadoras. (Emco, 2014)

El estudio del estado del arte permitió conocer un conjunto de herramientas que se dedican a la instalación automática de aplicaciones en la red, donde se apreció que la mayoría de ellas presentan características comunes, aunque cada una tiene particularidades propias a la hora de realizar dicho proceso. Se encontró como principal desventaja que son propietarias y el país está inmerso en una política de migración a Software Libre, a la cual se ha sumado la Universidad. Además no son compatibles con distribuciones GNU/Linux y el Gclient tiene versiones para los sistemas operativos Debian 6, Debian 7, Nova 2013, Ubuntu 12.04, Ubuntu 14.04, Centos 7, Windows 7 y Windows 8+. Por estas razones las herramientas estudiadas no son útiles para realizar el proceso de instalación automática de Gclient, por lo cual se decide implementar el módulo “Instalación automática de Gclient para el sistema GRHS”.

5 Microsoft: empresa multinacional, fundada el 4 de abril de 1975 por Bill Gates y Paul Allen; dedicada al sector del software y el hardware.

6 MSI conocido como Microsoft Installer, Windows Installer es un motor para la instalación, mantenimiento y eliminación de programas en plataformas Windows.

1.3 Metodología de Desarrollo de Software

Una metodología de desarrollo de software puede definirse como un conjunto de procedimientos, técnicas, herramientas y un soporte documental que ayuda en la construcción de un software. (Pressman, 2001)

En la actualidad existen diversas metodologías clasificadas en dos grandes grupos, las denominadas metodologías tradicionales o formales y las ágiles o ligeras. El primer grupo se caracteriza por llevar una documentación exhaustiva de todo el desarrollo del software y cumplir con la planificación elaborada en la fase inicial del proyecto. Muestra cierta resistencia a los cambios debido que, al consumarse uno, se ven afectados varios componentes del proceso de desarrollo de software y de igual modo se caracteriza por necesitar un proyecto de gran cantidad de participantes, pues requiere de un equipo de trabajo capaz de administrar un proceso complejo en varias etapas.

Como alternativa a estos inconvenientes surgen las metodologías ágiles, que se caracterizan por tener equipos de trabajo pequeños (menos de 10 integrantes), ser adaptables, configurables, de poca documentación y realizar pruebas constantemente para que se puedan obtener productos funcionales en poco tiempo en aras de satisfacer las necesidades del cliente. Permite además que estos productos sean fáciles y simples de modificar.

Dentro de las metodologías ágiles se encuentra la Programación Extrema (XP [7], Extreme Programming) y se utiliza para proyectos a corto plazo y equipos de trabajo pequeño. Consiste en una programación rápida o extrema, cuya particularidad es tener como parte del equipo, al usuario final, pues es uno de los requisitos para llegar al éxito del proyecto.

El ciclo de trabajo de esta metodología consta de 4 fases: (Wallace, y otros, 2002)

- ✓ **Planificación:** fase donde se crean las historias de usuario (HU), el plan de iteraciones y el plan de entrega.
- ✓ **Diseño:** en esta fase se crean las tarjetas clase, responsabilidad y colaboración (CRC) las cuales definen una clase expresando sus funcionalidades y las otras clases con las que colabora.

7 www.extremeprogramming.org, www.xprogramming.com, c2.com/cgi/wiki?ExtremeProgramming

- ✓ **Implementación:** es donde se definen las tareas de ingeniería para que los desarrolladores tengan una guía para implementar todas las HU.
- ✓ **Pruebas:** en esta fase se realizan las pruebas de aceptación a cada una de las HU para probar que cumplen con las funcionalidades que desea el cliente.

XP se centra en potenciar la comunicación entre los usuarios y los desarrolladores, permite la simplicidad al desarrollar y codificar los módulos del sistema. Tiene dos objetivos principales: lograr la satisfacción del cliente, tratando de darle un software que verdaderamente necesita y en el momento que lo requiera, además de potenciar al máximo el trabajo en grupo, identificando a cada miembro del equipo de desarrollo con el producto entregable al cliente. (Wallace, y otros, 2002)

Los autores del presente trabajo de diploma deciden utilizar XP como metodología de desarrollo de software para el desarrollo del módulo “Instalación automática de Gclient para el sistema GRHS” por los aspectos mencionados anteriormente. Además el equipo de trabajo está compuesto por una pareja de programadores y el módulo a desarrollar es pequeño y de corta duración. El cliente también forma parte del equipo de desarrollo y puede incorporar o cambiar las características funcionales del sistema sin afectar el ciclo de vida del proyecto.

1.4 Lenguajes y Herramientas de Desarrollo de Software

A continuación se describen los lenguajes y herramientas utilizados para desarrollar el módulo “Instalación automática de Gclient para el sistema GRHS”.

1.4.1 Lenguaje de programación Python v2.7

Un lenguaje de programación está formado por un conjunto de símbolos, palabras claves utilizables y reglas gramaticales para construir sentencias sintácticas y semánticamente correctas (Sala, 2003); pone dichas sentencias a disposición del programador o desarrollador para que pueda comunicarse con los dispositivos de hardware y software existentes.

Para el desarrollo del presente trabajo de diploma se selecciona Phyton en su versión 2.7 como lenguaje de programación, ya que es un lenguaje potente, flexible y con una sintaxis clara y concisa. No requiere dedicar tiempo a su compilación debido a que es interpretado y posee extensas bibliotecas estándar. Un programa en Python puede tener de 3 a 5 líneas de código menos que su equivalente en Java o C. Además el sistema al que se va a integrar el módulo está implementado en ese lenguaje.

1.4.2 HTML5

HTML es un lenguaje de marcado para la elaboración de páginas web (Hickson, y otros, 2014). Se decide utilizar HTML5 porque brinda un código sencillo y simplificado, permitiendo que se carguen más rápido las páginas en el navegador y es el lenguaje que utiliza GRHS. Además tiene una estructura compuesta por etiquetas que permiten insertar los diferentes elementos que componen la presentación de la información que brindará el módulo “Instalación automática de Gclient para el sistema GRHS”.

1.4.3 Hojas de Estilos en Cascada 3 (CSS3)

CSS es un lenguaje utilizado en la presentación de documentos HTML y XML, que permite dar formato y estilos de presentación. Es la simplificación del mantenimiento de las páginas web generadas en HTML que suponen una mejora del lenguaje en cuanto a presentación, mantenimiento y estandarización del contenido (Sierra, 2015). Se selecciona CSS3 por los aspectos mencionados y por ser uno de los lenguajes utilizados por el sistema al que se va a integrar el módulo.

1.4.4 Lenguaje Unificado de Modelado (UML) v2.0

UML es un lenguaje de modelado visual que se usa para construir, documentar, visualizar y especificar un sistema de software. Proporciona un conjunto de elementos de modelado, anotaciones, relaciones y normas que pueden aplicarse a una actividad de desarrollo de software. Permite entender, diseñar, hojear, configurar, mantener y controlar la información de tales sistemas. (Alegsa, 2015) Se utiliza para modelar las clases del sistema.

1.4.5 JavaScript

“JavaScript es uno de los lenguajes de script u orientado a documento” (Boticario, y otros, 2001) más usados en Internet, para añadir interactividad a las páginas web (Maza, 2012). Para el desarrollo del presente trabajo se utiliza este lenguaje para manejar los datos del lado del cliente, ya que permite dar respuestas a eventos iniciados por el usuario, como la entrada de datos en un formulario o la elección de algún enlace del documento HTML, sin realizar peticiones al servidor. Este lenguaje es utilizado por el sistema GRHS.

1.4.6 Marco de trabajo (framework) Django v1.6

Un marco de trabajo para aplicaciones web es un software o conjunto de librerías diseñado para dar soporte al desarrollo de sitios y aplicaciones web. (Vázquez, 2011)

Django es un framework web de código abierto escrito en Python que permite construir aplicaciones web más rápido y con menos código. Fomenta el bajo acoplamiento: filosofía de programación que dice que las distintas partes de la aplicación deben ser intercambiables y deben comunicarse unas con otras mediante Interfaces de Programación de Aplicaciones (API, Application Programming Interface) claras y concisas. (Keith-Magee, 2014)

Teniendo en cuenta los elementos antes mencionados, la definición del uso de Django por el centro TLM para el sistema GRHS y las facilidades que brinda para el desarrollo de actividades comunes como: el acceso a base de datos, el uso de plantillas, el manejo de sesiones y la reutilización de código, se selecciona dicho framework en su versión 1.6 para el desarrollo del módulo “Instalación automática de Gclient para el sistema GRHS”.

1.4.7 Entorno de Desarrollo Integrado (IDE) Eclipse Helios v3.6.1

Un Entorno de Desarrollo Integrado (IDE, Integrated Development Environment) es un programa informático compuesto por un conjunto de herramientas de programación que les permite a los programadores escribir, compilar, depurar y ejecutar programas. (Maldonado, 2013)

Eclipse Helios es un IDE de código abierto y multiplataforma que se ejecuta sobre la máquina virtual de Java. Eclipse Helios es mucho más que un simple IDE, es toda una comunidad de desarrolladores de código libre, dedicados a la implementación de mejoras del entorno (Arthorne, 2004). Permite añadir soporte a varios lenguajes (entre ellos se encuentra el lenguaje de programación seleccionado para desarrollar el sistema), ya que posee una arquitectura basada en módulos.

Debido a las razones antes expuestas y teniendo en cuenta que la herramienta Eclipse Helios es libre, ajustándose a las políticas de soberanía tecnológica trazadas por el país, se selecciona en la versión 3.6.1 para el desarrollo del módulo “Instalación automática de Gclient para el sistema GRHS”.

1.4.8 Sistema Gestor de Base de Datos (SGBD) PostgreSQL v9.3

Un Sistema Gestor de Base de Datos (SGBD) es un conjunto de programas que administran y gestionan la información contenida en una base de datos. Ayuda a realizar la definición, la manipulación, el control de la seguridad y la privacidad de los datos y el mantenimiento de la integridad de éstos dentro de la base de datos. (Álvarez, 2007)

PostgreSQL es un SGBD relacional, robusto, confiable y mantiene la integridad de los datos. Se caracteriza por ser un sistema de alto rendimiento y gran flexibilidad, lo que permite a los desarrolladores generar nuevas aplicaciones o mantener las existentes. (Global Development Group, 2015)

Teniendo en cuenta los aspectos mencionados anteriormente y por ser el SGBD que utiliza el sistema GRHS, se selecciona dicha herramienta en la versión 9.3 para gestionar la base de datos del módulo.

1.4.9 PgAdmin III

PgAdmin III es una herramienta de código abierto para la administración de bases de datos PostgreSQL. Incluye una interfaz administrativa gráfica, una herramienta de consultas SQL y un editor de código procedural. La interfaz gráfica soporta todas las características de PostgreSQL y hace simple la administración. Está disponible en varios sistemas operativos incluyendo GNU/Linux, FreeBSD[8], Mac, Windows y Solaris[9]. (Puro Software, 2007)

1.4.10 Herramienta CASE Visual Paradigm v8.0

Las herramientas CASE10 comprenden un conjunto de programas de diferentes tipos, empleados para ayudar a las actividades del proceso de desarrollo de software, tales como: análisis de requerimientos, modelado de sistemas, depuración y pruebas (Sommerville, 2005).

Visual Paradigm es una herramienta CASE que soporta el lenguaje UML, dispone de varios diagramas como los de clases, de objetos, de casos de uso del negocio y de paquetes; brinda facilidades de uso para la creación de estos y permite generar código a partir de ellos. Es un software

8 FreeBSD: sistema operativo libre para computadoras con CPU de arquitectura Intel, incluyendo procesadores Intel 80386, Intel 80486 (versiones SX y DX) y Pentium.

9 Solaris: sistema operativo de tipo Unix desarrollado desde 1992 inicialmente por Sun Microsystems y actualmente por Oracle Corporation como sucesor de SunOS. Es un sistema certificado oficialmente como versión de Unix.

10 CASE: (computer aided software engineering), ingeniería de software asistida por computadora.

gratuito, aunque se encuentra bajo una licencia que no permite su modificación o venta, ajustándose esta última a las políticas de soberanía tecnológica trazadas por el país.

Teniendo en cuenta la importancia del uso del software libre en Cuba y la existencia de una licencia UCI para el uso de Visual Paradigm, se selecciona en su versión 8.0 para realizar el diseño UML de la herramienta propuesta en el presente trabajo. Además se puede utilizar en cualquier fase del desarrollo del software, contribuyendo esto a garantizar una mayor flexibilidad y agilidad para la adaptación al cambio.

1.4.11 Backbone v1.1.0

Backbone es el framework ligero para JavaScript utilizado por el sistema GRHS. Facilita la construcción de interfaces de usuario para aplicaciones web, permitiendo que sean escalables, eficientes y robustas ante fallos, basado en el paradigma de diseño de aplicaciones Modelo Vista Controlador. Tiene como objetivo probar y definir un conjunto de estructuras de datos junto al manejo de la interfaz por medio de Vistas y URLs[11], que sean útiles para la constitución de aplicaciones JavaScript (Ashkenas, 2010). Se utilizará para la construcción de tablas paginadas encargadas de visualizar la información contenida en el módulo “Instalación automática de Gclient para el sistema GRHS”.

1.4.12 Paramiko v1.10.1

Paramiko es una biblioteca para Python que implementa el protocolo SSH[12] para conexiones seguras (cifradas y autenticadas) a máquinas remotas. Paramiko no viene por defecto con Python, se debe instalar. Se utilizará esta biblioteca en su versión 1.10.1 para establecer la conexión con las estaciones de trabajo y mediante ella automatizar el proceso de instalación de Gclient.

11 URLs: Uniform Resource Locator (localizador de recursos uniforme), es una secuencia de caracteres de acuerdo a un formato modélico y estándar, que se usa para nombrar recursos en Internet para localizarlos o identificarlos.

12 SSH (Secure SHell), protocolo que facilita las comunicaciones seguras entre dos sistemas usando una arquitectura cliente/servidor y que permite acceder a máquinas remotas a través de una red. SSH cifra la sesión de conexión haciendo imposible que alguien pueda obtener contraseñas no cifradas.

1.5 Conclusiones del capítulo

El análisis de los principales conceptos asociados al dominio del problema facilitó una mejor comprensión del presente trabajo. Además, el estudio del estado del arte realizado permitió analizar herramientas que realizan instalación automática de aplicaciones en una red de computadoras y definir elementos y características para el diseño y funcionamiento del módulo “Instalación automática de Gclient para el sistema GRHS”. Por otra parte, la selección de la metodología XP para guiar el proceso de desarrollo de software permitirá lograr mayor productividad, mejores resultados y garantizará la obtención de un producto de calidad que satisfaga plenamente las necesidades del cliente. Las herramientas PostgreSQL v9.3, Django v1.6 y Eclipse Helios v3.6.1 facilitarán la gestión de los datos y la construcción de una aplicación de forma rápida, que sea escalable, flexible y sencilla para el usuario.

CAPÍTULO 2: PLANIFICACIÓN Y DISEÑO DE LA SOLUCIÓN

En el presente capítulo se elabora una propuesta de solución, teniendo en cuenta las fases de planificación y diseño definidas por la metodología XP. Se definen también los elementos importantes dentro del proceso de desarrollo tales como: las historias de usuario (HU), el plan de iteraciones, el plan de entregas, la arquitectura, los patrones de diseño y las tarjetas CRC (Clase, Responsabilidad y Colaboración). Además, se abordarán temas relacionados con la estructura del framework Django para un mejor entendimiento, así como los principales componentes utilizados para el desarrollo del sistema.

2.1 Descripción del sistema propuesto

La solución informática propuesta se basa en el desarrollo de un módulo para el sistema GRHS que permita instalar Gclient de forma automática en una red de computadoras. El módulo mostrará la información (dirección IP, si tiene o no instalado Gclient, subred, fecha y localización) de todas las máquinas registradas en la base de datos, obtenida mediante el descubrimiento de la red, permitiendo al personal autorizado seleccionar las máquinas a las que desee instalar Gclient.

Luego de la selección, se solicitarán las credenciales para acceder a las estaciones de trabajo y se establecerá una conexión con ellas mediante el protocolo SSH. El sistema obtendrá de un servidor FTP[13] el instalador correspondiente a la arquitectura y sistema operativo del computador, lo copiará y lo ejecutará, logrando así la instalación. En la Figura 1 se muestra la propuesta de solución descrita anteriormente.

13 FTP (File Transfer Protocol) protocolo de red para la transferencia de archivos entre sistemas conectados a una red TCP (Transmission Control Protocol), basado en la arquitectura cliente-servidor.

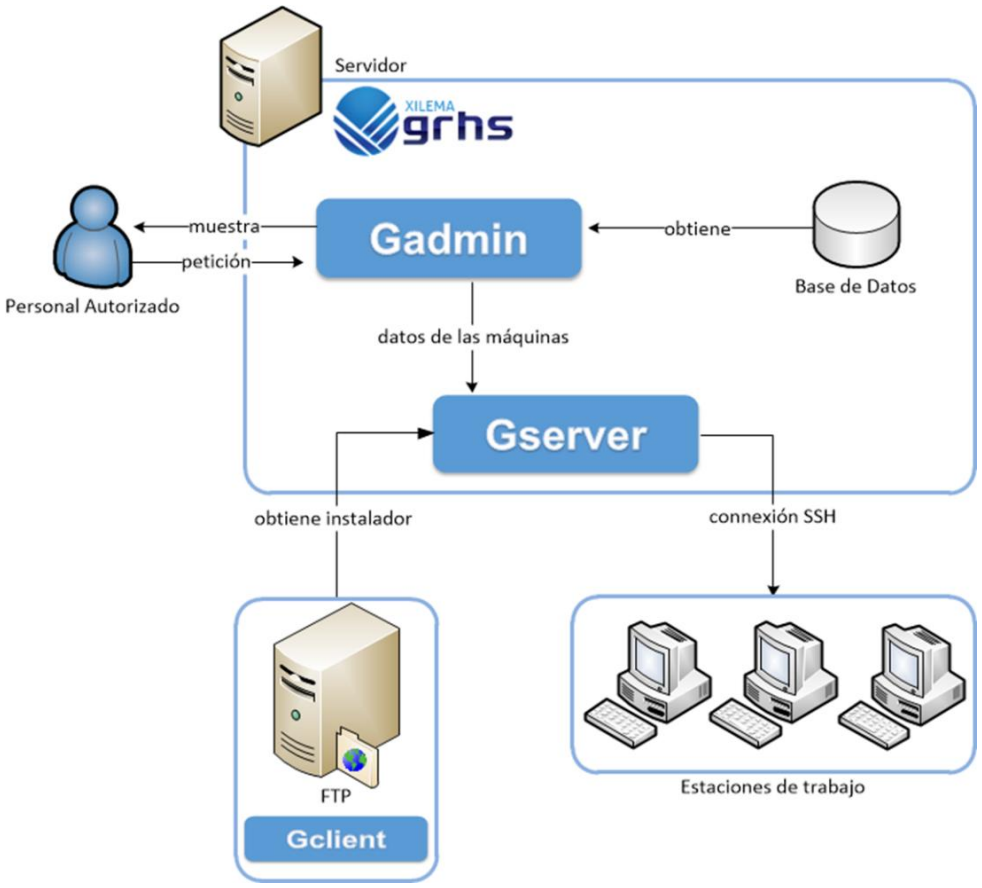


Figura 1. Propuesta de solución

2.2 Características no funcionales del sistema

Las características no funcionales (CNF) del sistema se refieren a las cualidades que debe tener el producto para que posea una interfaz legible, simple de usar, atractiva, rápida y confiable. Para el desarrollo del módulo se definieron un conjunto de CNF que se clasificaron en usabilidad, hardware, software y seguridad.

Usabilidad

El módulo “Instalación automática de Gclient para el sistema GRHS” deberá contar con una interfaz sencilla, descriptiva y fácil de usar, con el objetivo de propiciar un mejor entendimiento e interacción con el sistema a los usuarios finales.

Todas las interfaces de la aplicación deben permitir la traducción a los idiomas: español, inglés, francés y portugués.

Hardware

Para el funcionamiento del módulo se debe disponer de una computadora con un microprocesador cuya velocidad sea superior a los 2.2 GHz, 1 GB de RAM o superior y 40 GB de disco duro o superior.

Software

Para el funcionamiento del módulo las máquinas clientes deben tener habilitado el protocolo SSH. Se requiere la instalación de Python 2.7 y de la biblioteca paramiko. Django v1.6 como marco de trabajo y funciona en navegadores web como Mozilla Firefox 30.0+, Chrome 20.0+ e Internet Explorer 6+.

2.3 Planificación

La metodología de desarrollo XP comienza con la fase de planificación. En el transcurso de dicha fase los clientes plantean a grandes rasgos las historias de usuario (HU) y los programadores realizan una estimación del esfuerzo necesario de cada una de ellas. Se toman acuerdos sobre el contenido de la primera entrega y se establece un cronograma en conjunto con el cliente. Además se deben incluir varias iteraciones para lograr una entrega, siendo así el plan de entregas el resultado de esta fase. (Wallace, y otros, 2002)

2.3.1 Historias de Usuario (HU)

Uno de los artefactos más importantes que genera la metodología XP son las HU, éstas son una caracterización de una o más funcionalidades del sistema, escritas por el cliente o propietario del producto en conjunto con los programadores. Deben estar escritas en un lenguaje sin terminología técnica y entendible por los clientes, en caso de que el equipo de desarrollo sea quien las escriba. (Beck, 2000)

Las HU representan una forma rápida de administrar los requerimientos de los usuarios sin tener que elaborar gran cantidad de documentos formales y sin requerir de mucho tiempo para administrarlos. Además son la base para realizar las pruebas de aceptación, así como la estimación y planificación del proyecto.

El propietario del producto clasifica cada HU según la prioridad para el negocio en: Alta (fundamentales en el desarrollo del sistema), Media (necesarias pero no imprescindibles para el funcionamiento del sistema) y Baja (sirven de ayuda al control de elementos asociados al equipo de desarrollo), con el objetivo de determinar según las necesidades cuáles son más valiosas de resolver para poder realizar una correcta planificación de la implementación.

El equipo de desarrollo las clasifica según el riesgo en desarrollo, aquí se tiene en cuenta la dificultad y posible existencia de errores durante la implementación de cada HU. Las clasificaciones son: Alto (cuando se considera la posible existencia de errores que lleven a la inoperatividad[14] del código), Medio (cuando pueden aparecer errores que puedan retrasar la entrega de la versión) y Bajo (cuando pueden aparecer errores que serán tratados con relativa facilidad sin que traigan perjuicios para el desarrollo del proyecto).

Las HU se representan mediante tablas, las cuales se dividen en varias secciones denominadas:

- ✓ **Número:** representa el número, incremental en el tiempo, de la HU que se describe.
- ✓ **Nombre:** identifica la HU que se describe entre los desarrolladores y el cliente.
- ✓ **Usuario:** rol de la persona que ejecuta la funcionalidad.
- ✓ **Prioridad en negocio:** se otorga una prioridad (Alta, Media, Baja) a las HU de acuerdo a la necesidad de desarrollo.
- ✓ **Riesgo en Desarrollo:** se otorga una medida (Alto, Medio, Bajo) a la ocurrencia de errores en el proceso de desarrollo de la HU.
- ✓ **Iteración asignada:** número de la iteración donde va a desarrollarse la HU.
- ✓ **Programador responsable:** encargado de programar la HU.
- ✓ **Puntos Estimados:** tiempo estimado en semanas (de lunes a viernes) que se demorará el desarrollo de la HU.
- ✓ **Descripción:** breve descripción de la HU.
- ✓ **Observaciones:** señalamiento o advertencia del sistema.

14 Inoperatividad: falta de eficacia en la consecución de un propósito o fin.

- ✓ **Prototipo de interfaz:** imagen de cada interfaz relacionada con la HU.

Durante la fase de planificación se identificaron 6 HU definidas por el cliente en conjunto con el equipo de desarrollo, las cuales se exponen a continuación:

Tabla 1. HU: Mostrar información de las máquinas identificadas en la red

Historia de Usuario	
Número: 1	**Nombre:** Mostrar información de las máquinas identificadas en la red
Usuario: Personal autorizado	
Prioridad en Negocio: Alta	**Riesgos en Desarrollo:** Medio
Puntos Estimados: 0.4	**Iteración Asignada:** 1
Programadores responsables: Yoel Hernández Camejo y Arianna Martínez Sánchez	
Descripción: El sistema busca en la base de datos y muestra la dirección IP, si tiene o no instalado Gclient, la subred, la fecha y la localización del listado de máquinas que se encontraron mediante el descubrimiento de la red.	
Observaciones: El usuario tiene que estar previamente autenticado en el sistema.	
Prototipo de Interfaz:	

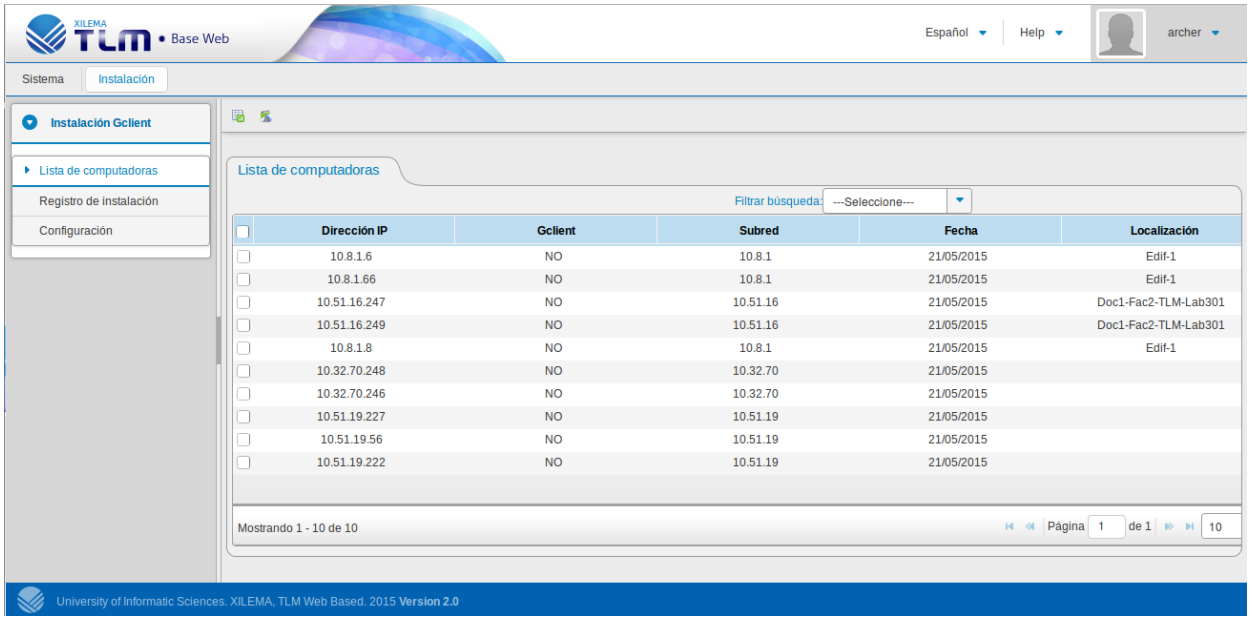


Tabla 2. HU: Instalación de Gclient para distribuciones GNU/Linux

Historia de Usuario	
Número: 2	**Nombre:** Instalación de Gclient para distribuciones GNU/Linux
Usuario: Personal autorizado	
Prioridad en Negocio: Alta	**Riesgos en Desarrollo:** Alto
Puntos Estimados: 2.6	**Iteración Asignada:** 1
Programadores responsables: Yoel Hernández Camejo y Arianna Martínez Sánchez	
Descripción: Para realizar la instalación el usuario selecciona las máquinas que desee instalar e introduce las credenciales para acceder a ellas.	
Observación: El usuario tiene que estar previamente autenticado en el sistema. Se debe conocer el usuario y la contraseña de las máquinas a instalar. Las estaciones de trabajo deben tener habilitado el protocolo SSH.	
Prototipo de Interfaz: No aplica	

Tabla 3. HU: Instalación de Gclient para el sistema operativo Windows

Historia de Usuario	
Número: 3	**Nombre:** Instalación de Gclient para el sistema operativo Windows
Usuario: Personal autorizado	
Prioridad en Negocio: Alta	**Riesgos en Desarrollo:** Alto
Puntos Estimados: 3	**Iteración Asignada:** 2
Programadores responsables: Yoel Hernández Camejo y Arianna Martínez Sánchez	
Descripción: Para realizar la instalación el usuario selecciona las máquinas que desee instalar e introduce las credenciales para acceder a ellas.	
Observación: El usuario tiene que estar previamente autenticado en el sistema. Se debe conocer el usuario y la contraseña de las máquinas a instalar. Las estaciones de trabajo deben tener habilitado el protocolo SSH.	
Prototipo de Interfaz: No aplica	

Tabla 4. HU: Registro de instalación

Historia de Usuario	
Número: 4	**Nombre:** Registro de instalación
Usuario: Personal autorizado	
Prioridad en Negocio: Media	**Riesgos en Desarrollo:** Medio
Puntos Estimados: 1	**Iteración Asignada:** 3
Programadores responsables: Yoel Hernández Camejo y Arianna Martínez Sánchez	
Descripción: Muestra la dirección IP, el sistema operativo, la fecha, un mensaje que indica si se instaló o no Gclient y responsable de realizar la instalación de las máquinas.	
Observación: El usuario tiene que estar previamente autenticado en el sistema.	
Prototipo de Interfaz:	

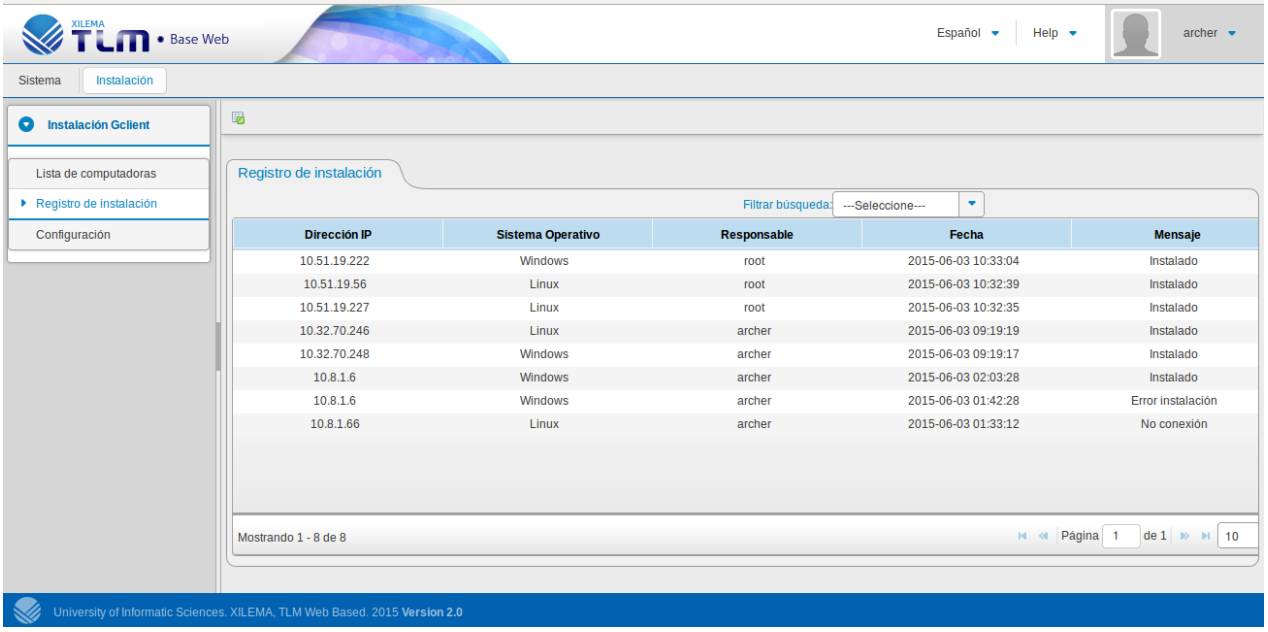


Tabla 5. HU: Configuración de instalación

Historia de Usuario	
Número: 5	**Nombre:** Configuración de instalación
Usuario: Personal autorizado	
Prioridad en Negocio: Media	**Riesgos en Desarrollo:** Media
Puntos Estimados: 1	**Iteración Asignada:** 3

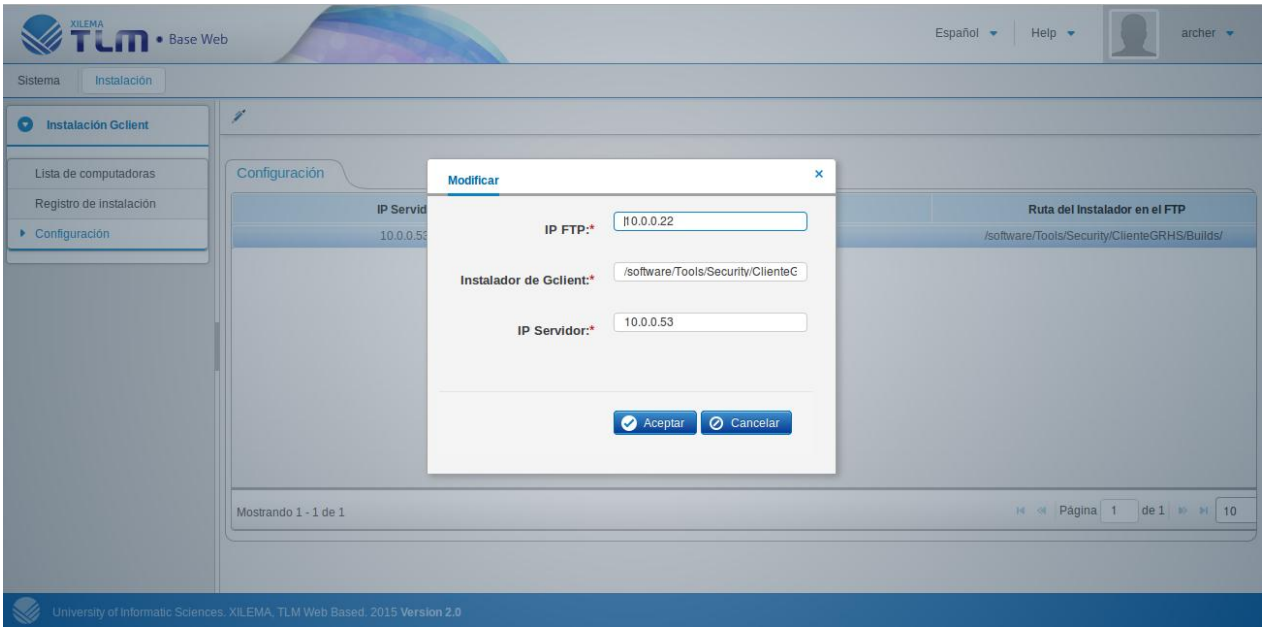
Programadores responsables: Yoel Hernández Camejo y Arianna Martínez Sánchez
Descripción: El usuario debe poder visualizar y modificar el IP y la ruta del FTP donde se encuentren los instaladores de Gclient y el IP del servidor de GRHS.
Observación: El usuario tiene que estar previamente autenticado en el sistema.
Prototipo de Interfaz: 

Tabla 6. HU: Generar informe de instalación de Gclient

Historia de Usuario	
Número: 6	**Nombre:** Generar informe de instalación de Gclient
Usuario: Personal autorizado	
Prioridad en Negocio: Baja	**Riesgos en Desarrollo:** Bajo
Puntos Estimados: 1	**Iteración Asignada:** 3
Programadores responsables: Yoel Hernández Camejo y Arianna Martínez Sánchez	
Descripción: El usuario podrá exportar en formato de hoja de cálculo el listado de máquinas identificadas con sus respectivos datos (dirección IP, fecha, sistema operativo, mensaje que indica si se instaló o no Gclient y responsable de realizar la instalación).	

Observación: El usuario tiene que estar previamente autenticado en el sistema.
Prototipo de Interfaz: No aplica

2.3.2 Estimación de esfuerzo por Historias de Usuario

Para la realización de la herramienta informática propuesta se efectuó una estimación de esfuerzo por cada HU identificada, arrojando los siguientes resultados:

Tabla 7. Estimación de esfuerzo por HU

Iteración	Historia de Usuario		Puntos estimados (semanas)
1	1	Mostrar información de las máquinas identificadas en la red	0.4
	2	Instalación de Gclient para distribuciones GNU/Linux	2.6
2	3	Instalación de Gclient para el sistema operativo Windows	3
3	4	Registro de instalación	1
	5	Configuración de instalación	1
	6	Generar informe de instalación de Gclient	1
Total			**9**

2.3.3 Plan de Iteraciones

Una vez identificadas y descritas las historias de usuario y estimado el esfuerzo dedicado a la realización de cada una de ellas, se procede a la planificación de la etapa de implementación, esto se refleja en el plan de iteraciones. El plan incluye iteraciones sobre la herramienta antes de ser entregada y muestra que HU serán implementadas en cada iteración. Las funcionalidades básicas e indispensables del sistema estarán listas al final de la última iteración, permitiendo que esté listo para entrar en producción. (Beck, y otros, 2000)

A continuación se describen cada una de las iteraciones propuestas, donde la duración total de iteraciones se obtiene a partir del esfuerzo estimado por el desarrollador para implementar cada HU:

Iteración 1: en esta iteración se implementan las HU con prioridad alta, relacionadas con la instalación de Gclient para estaciones de trabajo con sistema operativo correspondiente a distribuciones GNU/Linux y la visualización de la información de las máquinas detectadas por el escaneo de la red. Al finalizar esta iteración se contará con una primera versión del sistema.

Iteración 2: en esta iteración se implementa la HU número 3 con prioridad alta en el negocio, la cual tiene como finalidad realizar la instalación de Gclient en estaciones de trabajo con sistema operativo Windows. Como resultado se obtendrá una versión más completa de las funcionalidades del sistema.

Iteración 3: en esta iteración se implementan las HU relacionadas con la configuración y exportación de los datos, completando así el desarrollo de la aplicación. Se obtendrá la versión 1.0 del módulo de forma tal que los usuarios puedan probar todas las funcionalidades y explotar el sistema a plenitud.

A modo de resumen se presenta una tabla que muestra la duración de cada iteración, así como el orden en que serán implementadas las HU en cada una de ellas según la prioridad asignada por el cliente:

Tabla 8. Plan de duración de las iteraciones

Iteración	Historia de Usuario	Duración
1	Mostrar información de las máquinas identificadas en la red	3 semanas
	Instalación de Gclient para distribuciones GNU/Linux	
2	Instalación de Gclient para el sistema operativo Windows	3 semanas
3	Registro de instalación	3 semanas
	Configuración de instalación	
	Generar informe de instalación de Gclient	

2.3.4 Plan de Entregas

A continuación se muestra el plan de entrega elaborado por el equipo de desarrollo para la fase de implementación, la cual comienza el 23 de febrero de 2015. En el plan se especifica la fecha en la que se realiza el encuentro entre el grupo de desarrollo y los propietarios del software para la liberación de las versiones del producto.

Tabla 9. Plan de entregas

Iteración	Fecha de entrega
1	13 de marzo de 2015
2	3 de abril de 2015
3	24 de abril de 2015

2.4 Diseño

La metodología de desarrollo de software XP sugiere la elaboración de diseños sencillos para lograr un fácil entendimiento en la fase de desarrollo, permitiendo la reducción del tiempo de elaboración de la tarea asignada al desarrollador. Se deben evitar a toda costa la complejidad innecesaria y el código extra. El software que posee un diseño adecuado es aquel que refleja claramente la intención de implementación de los programadores, supera con éxito todas las pruebas, no tiene lógica duplicada y tiene el menor número posible de clases y métodos.

XP construye un proceso de diseño evolutivo, que se basa en describir la modificación del código fuente sin cambiar el comportamiento del sistema y lograr así que sea simple en cada iteración. En el diseño no se realiza nada de forma anticipada para necesidades futuras, solo se centra en la iteración actual. Se obtiene como resultado un proceso de diseño disciplinado, que combina la disciplina con la adaptabilidad y logra que sea una de las más desarrolladas de entre todas las metodologías ágiles. (Canós, y otros, 2011)

2.4.1 Arquitectura de Software

La Arquitectura de Software es la organización fundamental de un sistema encarnada en sus componentes, las relaciones entre ellos y el ambiente y los principios que orientan su diseño y evolución. (Etcheverry, 2010)

Para el desarrollo del presente trabajo se utilizó la arquitectura cliente/servidor, en la cual el cliente hace peticiones al servidor, el cual procesa dichos requerimientos y retorna los resultados al cliente apropiado (Pillou, 2015). Esta arquitectura permite la centralización del control de los recursos, datos y accesos a datos. En la Figura 2 se muestra la arquitectura empleada para la solución propuesta.

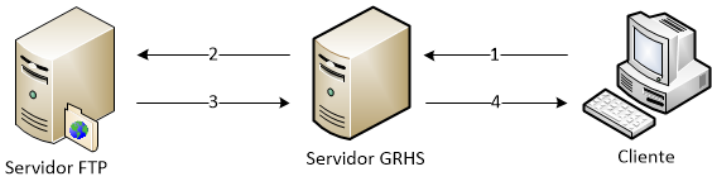


Figura 2. Arquitectura del sistema

1) El cliente le pide al servidor de GRHS que instale las máquinas seleccionadas.
2) El servidor GRHS le pide al servidor FTP el instalador de Gclient.
3) El servidor FTP le brinda el instalador de Gclient al servidor GRHS.
4) El servidor GRHS informa al cliente si se instalaron o no las máquinas seleccionadas.

2.4.2 Patrón arquitectónico

Un patrón arquitectónico define la estructura básica de una aplicación. Proporciona un conjunto de subsistemas predefinidos para especificar responsabilidades, reglas y directrices que determinan la organización, comunicación, interacción y relaciones entre ellos. Además, heredan gran parte de la terminología y conceptos de patrones de diseño, pero se centran en proporcionar modelos y métodos reutilizables, específicamente para la arquitectura general de los sistemas de información. (Gómez, 2013)

La arquitectura empleada está basada en el patrón arquitectónico Modelo-Vista-Controlador (MVC), el cual puntualiza una serie de subsistemas con las respectivas responsabilidades, e incluye las reglas y criterios para organizar las relaciones existentes entre ellos (Martín, y otros, 2010). Este patrón está formado por tres niveles: modelo, vista y controlador. El *modelo* representa la información con la que trabaja la aplicación, es decir, son los datos puros que puestos en contexto del sistema proveen de información al usuario o a la aplicación. La *vista* convierte el modelo en una página web con contenido dinámico sobre la cual el usuario puede realizar operaciones. El *controlador* es la capa encargada de manejar y responder las solicitudes del usuario; para lograrlo, procesa la información necesaria y modifica el modelo en caso de ser necesario. (Bernabe, y otros, 2014)

El framework que se utilizará para el desarrollo de la aplicación realiza algunas modificaciones a la forma de llamar los elementos y las respectivas funciones del patrón MVC y establece su propia filosofía para interpretarlo, a pesar de hacer un uso intensivo de este patrón (Alchin, 2013). Django no cambia el significado del modelo, pero la forma de ver la vista y el controlador varía en cierta medida. Los desarrolladores de Django denominaron otro nombre para dicho patrón (Modelo-Plantilla-Vista o Model-Template-View, MTV por sus siglas en inglés), a pesar de identificar en su arquitectura los mismos beneficios que en el MVC.

En MTV la vista comparte el nombre y algunos elementos de la definición de vista del MVC, sin embargo se asocia más al controlador. Se manejan de forma que permitan modificaciones de los datos a enviar en torno a una petición; tienen acceso a los modelos, consultan y modifican la información necesaria para cumplir la tarea requerida por el usuario. Las plantillas (templates) son una parte esencial del framework y una capa independiente que sirve de intermediaria entre la vista y el modelo; se encargan de presentar los datos al usuario. (Alchin, 2013)

Este patrón permite la reutilización de los elementos que lo componen y el desarrollo de sistemas escalables, de fácil implementación y mantenimiento. Además, al separar la plantilla del modelo facilita mostrar múltiples plantillas para un mismo negocio, favoreciendo la personalización de los sistemas. En la Figura 3 se muestra como queda evidenciado el patrón arquitectónico (MTV) para la solución propuesta.

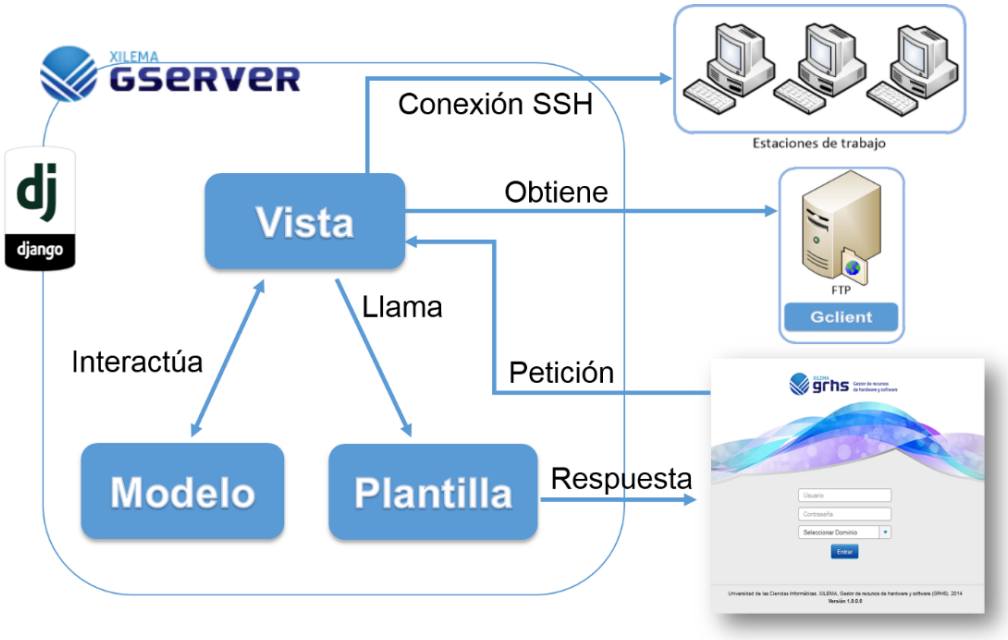


Figura 3. Patrón arquitectónico (MTV)

A continuación se presentan diagramas de clases que describen la constitución del módulo “Instalación automática de Gclient para el sistema GRHS” en las capas de la arquitectura MTV, para un mejor entendimiento del marco de trabajo Django.

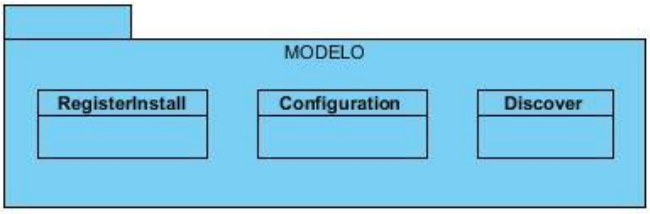


Figura 4. Clases de la capa: Modelo

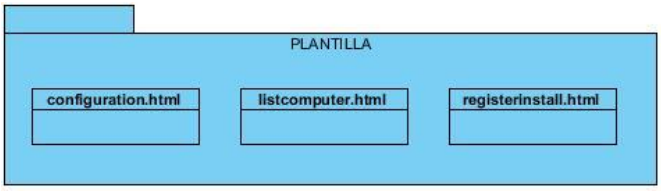


Figura 5. Clases de la capa: Plantilla

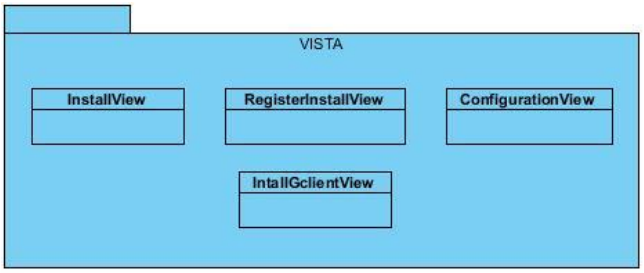


Figura 6. Clases de la capa: Vista

2.4.3 Patrones de diseño

Los patones de diseño son descripciones de clases y objetos relacionados, que tienen como particularidad la resolución de problemas de diseño comunes en un determinado contexto. Ayudan a capturar conocimiento y a crear un vocabulario técnico de los aspectos claves de una estructura de diseño común, lo cual los hace útiles para crear un diseño orientado a objetos flexible, elegante y reutilizable. (Varela, 2011)

Identifican además las clases e instancias participantes, los roles, las colaboraciones y la distribución de responsabilidades. Cada patrón de diseño se centra en un problema concreto, describe cómo aplicarlo y si tiene sentido hacerlo, teniendo en cuenta otras restricciones de diseño, así como las consecuencias, las ventajas e inconvenientes del uso. (Gamma, y otros, 2002)

Para diseñar la herramienta se emplearon un conjunto de patrones de diseño que constituyen soluciones simples y elegantes a problemas específicos y comunes del diseño orientado a objetos. A continuación se describen los patrones empleados.

Patrones Generales de Software para Asignar Responsabilidades (GRASP[15])

Los patrones GRASP describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en forma de patrones (Visconti, y otros, 2004). Son a la vez principios o técnicas de trabajo utilizadas para mejorar los diseños orientados a objetos, como el polimorfismo y la alta cohesión. Para el desarrollo del diseño del sistema se definen los siguientes patrones GRASP:

- ✓ **Alta cohesión:** asigna una responsabilidad de modo que la cohesión siga siendo alta. La cohesión es una medida de cuán relacionadas y enfocadas están las responsabilidades de una clase. Un elemento con responsabilidades altamente relacionadas y que no hace una

15 GRASP: object-oriented design General Responsibility Assignment Software Patterns, por las siglas en inglés.

gran cantidad de trabajo tiene alta cohesión. El patrón se refleja en las clases *RegisterInstall* y *Configuration*.

- ✓ **Bajo acoplamiento:** mide la fuerza con que una clase está conectada a otra, de esta forma una clase con bajo acoplamiento debe tener un número mínimo de dependencia con otras clases. Este patrón se tuvo presente debido a la importancia que se le atribuye a realizar un diseño de clases independientes que puedan soportar los cambios de una manera fácil y a la vez permitan la reutilización. Se evidencia en la clase *InstallGclient*.

- ✓ **Controlador:** está representado por una clase a la cual se le asigna la responsabilidad de las operaciones del sistema, ofrece también una guía para tomar decisiones apropiadas. Este patrón se refleja en las clases *InstallView, RegisterInstallView, InstallGclientView* y *ConfigurationView*, estas clases se encargan de controlar las acciones que realiza el usuario con las interfaces de la aplicación y dar respuesta a las peticiones realizadas.

- ✓ **Creador:** guía la asignación de responsabilidades relacionadas con la creación de objetos. El propósito fundamental de este patrón es encontrar un creador que se debe conectar con el objeto producido en cualquier evento. Brinda soporte a un bajo acoplamiento lo que supone menos dependencias respecto al mantenimiento y mejores oportunidades de reutilización. Este patrón se pone de manifiesto en la clase *Instalar*.

- ✓ **Experto:** se pone en práctica con el uso de clases que poseen responsabilidades específicas a cumplir, de acuerdo con la información que manejan. El uso de este patrón se evidencia en las clases *InstallView, RegisterInstallView, InstallGclientView, ConfigurationView, RegisterInstall* y *Configuration*, ya que poseen funciones concretas acorde con la información que gestionan.

2.4.4 Tarjetas CRC

La técnica CRC propone una forma de trabajo, preferentemente grupal, para encontrar los objetos del dominio de la aplicación, sus responsabilidades y cómo colaboran con otros para realizar tareas. Esta técnica utiliza las llamadas tarjetas CRC, las cuáles registran el nombre de las clases, sus responsabilidades y las otras clases con las que colaboran. (Beck, y otros, 1989)

Las tarjetas CRC ayudan al equipo a identificar las clases que participan en el diseño de la herramienta (Vallespir, 2002). Las responsabilidades de una clase son cualquier información que ella conoce o actividad que realiza y los colaboradores son clases que se requieren para recibir información necesaria y completar una responsabilidad.

Tabla 10. Tarjeta CRC: InstallGclient

Clase: InstallGclient	
Responsabilidades: ✓ Iniciar conexión ✓ Copiar instalador de Gclient en las estaciones de trabajo ✓ Actualizar el archivo *config.yml* ✓ Instalar Gclient para Windows y distribuciones GNU/Linux ✓ Guardar el registro de instalación	**Colaboradores:** ClienteSSHParamiko Utils Config RegisterInstall

Tabla 11. Tarjeta CRC: Views

Clase: Views	
Responsabilidades: ✓ Obtener el listado de las máquinas obtenidas mediante el descubrimiento de la red ✓ Iniciar proceso de instalación ✓ Obtener la información referente al registro de instalación ✓ Obtener la información referente a la configuración de la instalación	**Colaboradores:** Discover RegisterInstall Configuration InstallGclient

Tabla 12. Tarjeta CRC: ClienteSSHParamiko

Clase: ClienteSSHParamiko	
Responsabilidades: ✓ Establecer conexión mediante el protocolo SSH ✓ Ejecutar comandos en las estaciones de trabajo ✓ Cerrar la conexión	**Colaboradores:** Paramiko

Tabla 13. Tarjeta CRC: Utils

Clase: Utils	
Responsabilidades: ✓ Actualizar identificador e IP del servidor al que se conecta Gclient ✓ Obtener la distribución y la arquitectura de las estaciones de trabajo	**Colaboradores:** Config

2.5 Conclusiones del capítulo

La construcción del plan de iteraciones permitió conocer las historias de usuario a implementar en cada iteración y el orden de prioridad de cada una durante su desarrollo. Además, la confección del plan de entregas permitió conocer la fecha aproximada en que serán entregadas las versiones definidas como partes funcionales del sistema.

La utilización de la arquitectura cliente/servidor sustentada en el patrón arquitectónico MTV, permitió puntualizar las respectivas responsabilidades, reglas y criterios para organizar las relaciones existentes en el sistema. Por otra parte, el empleo de patrones de diseño garantizó una solución que tiene como premisa la reutilización de código y la solución a problemas que tienen contextos similares en el desarrollo de software. El diseño de las tarjetas CRC permitió identificar las principales clases, las relaciones entre ellas y las responsabilidades, posibilitando una reducción del acoplamiento y un aumento de la reutilización en la herramienta.

CAPÍTULO 3: IMPLEMENTACIÓN Y PRUEBAS

En este capítulo se muestra el desarrollo de las fases de implementación y pruebas que propone la metodología seleccionada. Se realiza una descripción de los principales artefactos a generar, como son: las tareas de ingeniería por cada HU identificada y las pruebas empleadas para validar la solución propuesta.

3.1 Implementación

La implementación del sistema es la parte más importante dentro del desarrollo de software en XP. Existe una idea equivocada de lo que significa programar, la implementación es mucho más que sentarse frente a la computadora y escribir líneas de código, es la resolución de problemas y requiere gran cantidad de atención y creatividad. La metodología propone una serie de prácticas que sirven para el desarrollo exitoso del software a realizar y plantea que esta fase debe realizarse de forma iterativa, para que al culminar cada iteración se obtenga un producto funcional que debe ser probado y mostrado al cliente, para retroalimentar a los desarrolladores con la opinión de este. (Wallace, y otros, 2002)

En esta fase las HU se descomponen en tareas de programación o ingeniería, que a la vez son convertidas en código fuente. Se tiene en cuenta el desarrollo de las iteraciones según el plan de iteraciones definido, la entrega de la herramienta en iteraciones pequeñas y un diseño simple y poco redundante del código con las funcionalidades necesarias.

3.1.1 Tareas de Ingeniería

Una vez identificadas las HU los programadores proceden a descomponer cada una en tareas específicas, las denominadas tareas de programación, que son escritas técnicamente y darán solución a la historia correspondiente. Se utilizan para describir las tareas que se realizan sobre el proyecto, las cuales pueden ser de tipo: desarrollo, corrección y mejora. Además permiten a los desarrolladores obtener un nivel de detalle más avanzado que el propiciado por las HU. Como se menciona en el capítulo anterior, se realiza una planificación de 3 iteraciones para el desarrollo del software y a continuación se especifican las tareas de ingeniería correspondientes a cada iteración.

Tabla 14. Tareas de ingeniería

Iteración	Historia de Usuario	Tareas de Ingeniería
1	Mostrar información de las máquinas identificadas en la red	Mostrar información
		Filtrar información
	Instalación de Gclient para distribuciones GNU/Linux	Establecer conexión para máquinas con distribuciones GNU/Linux
		Obtener el instalador de Gclient
		Instalar Gclient
2	Instalación de Gclient para el sistema operativo Windows	Establecer conexión para máquinas con sistema operativo Windows
		Obtener el instalador de Gclient
		Instalar Gclient
3	Registro de instalación	Mostrar registro de instalación
	Configuración de instalación	Mostrar configuración de instalación
		Modificar configuración de instalación
	Generar informe de instalación del Gclient	Generar informe de instalación

A continuación se detalla cada una de las tareas de ingeniería correspondientes a las tres iteraciones planificadas por el equipo de desarrollo.

Tabla 15. Tarea de ingeniería: Mostrar información

Tarea de ingeniería	
Número de tarea: 1	**Número de HU:** 1
Nombre de la tarea: Mostrar información	
Tipo de tarea: Desarrollo	**Puntos Estimados:** 0.2
Fecha de inicio: 23 de febrero de 2015	**Fecha de fin:** 23 de febrero de 2015
Programadores responsables: Yoel Hernández Camejo y Arianna Martínez Sánchez	
Descripción: El usuario selecciona el menú “Instalar Gclient” y el sistema busca en la base de datos y muestra un listado con el identificador, la dirección IP, si tienen o no instalado Gclient, la localización, la subred y la fecha de las máquinas que se encontraron mediante el descubrimiento de la red.	

Tabla 16. Tarea de ingeniería: Filtrar información

Tarea de ingeniería	
Número de tarea: 2	**Número de HU:** 1
Nombre de la tarea: Filtrar información	
Tipo de tarea: Desarrollo	**Puntos Estimados:** 0.2
Fecha de inicio: 24 de febrero de 2015	**Fecha de fin:** 24 de febrero de 2015
Programadores responsables: Yoel Hernández Camejo y Arianna Martínez Sánchez	
Descripción: El usuario selecciona en la lista desplegable “Filtrar” uno o varios de los criterios de búsqueda (dirección IP, localización, subred y si tiene o no instalado Gclient) y el sistema muestra un listado con la información de las máquinas asociadas al criterio seleccionado.	

Tabla 17. Tarea de ingeniería: Establecer conexión para máquinas con distribuciones GNU/Linux

Tarea de ingeniería	
Número de tarea: 3	**Número de HU:** 2
Nombre de la tarea: Establecer conexión para máquinas con distribuciones GNU/Linux	
Tipo de tarea: Desarrollo	**Puntos Estimados:** 0.6
Fecha de inicio: 25 de febrero de 2015	**Fecha de fin:** 27 de febrero de 2015
Programadores responsables: Yoel Hernández Camejo y Arianna Martínez Sánchez	
Descripción: Se elabora un método que permite conectarse del servidor a las estaciones de trabajo mediante el protocolo SSH, para ello se utiliza el biblioteca paramiko.	

Tabla 18. Tarea de ingeniería: Obtener el instalador de Gclient

Tarea de ingeniería	
Número de tarea: 4	**Número de HU:** 2
Nombre de la tarea: Obtener el instalador de Gclient	
Tipo de tarea: Desarrollo	**Puntos Estimados:** 1
Fecha de inicio: 2 de marzo de 2015	**Fecha de fin:** 6 de marzo de 2015
Programadores responsables: Yoel Hernández Camejo y Arianna Martínez Sánchez	

Descripción:
Se elabora un método que permite obtener de un FTP el instalador de Gclient correspondiente a la arquitectura y sistema operativo de las máquinas seleccionadas, copiarlo en cada una y ejecutarlo.

Tabla 19. Tarea de ingeniería: Instalar Gclient

Tarea de ingeniería	
Número de tarea: 5	**Número de HU:** 2
Nombre de la tarea: Instalar Gclient	
Tipo de tarea: Desarrollo	**Puntos Estimados:** 1
Fecha de inicio: 9 de marzo de 2015	**Fecha de fin:** 13 de marzo de 2015
Programadores responsables: Yoel Hernández Camejo y Arianna Martínez Sánchez	
Descripción: El usuario selecciona las máquinas que desee instalar, oprime el icono “Instalar” y el sistema le solicita las credenciales para acceder a las máquinas seleccionadas. Luego el sistema debe guardar en la base de datos la dirección IP, mensaje del resultado de la instalación, fecha y responsable de realizar la instalación automática de las máquinas.	

Tabla 20. Tarea de ingeniería: Establecer conexión para máquinas con sistema operativo Windows

Tarea de ingeniería	
Número de tarea: 6	**Número de HU:** 3
Nombre de la tarea: Establecer conexión para máquinas con sistema operativo Windows	
Tipo de tarea: Desarrollo	**Puntos Estimados:** 1
Fecha de inicio: 16 de marzo de 2015	**Fecha de fin:** 20 de marzo de 2015
Programadores responsables: Yoel Hernández Camejo y Arianna Martínez Sánchez	
Descripción: Se elabora un método que permite conectarse del servidor a las estaciones de trabajo mediante el protocolo SSH, para ello se utiliza el biblioteca paramiko.	

Tabla 21. Tarea de ingeniería: Obtener el instalador de Gclient para Windows

Tarea de ingeniería	
Número de tarea: 7	**Número de HU:** 3
Nombre de la tarea: Obtener el instalador de Gclient para Windows	
Tipo de tarea: Desarrollo	**Puntos Estimados:** 1
Fecha de inicio: 23 de marzo de 2015	**Fecha de fin:** 27 de marzo de 2015
Programadores responsables: Yoel Hernández Camejo y Arianna Martínez Sánchez	
Descripción: Se elabora un método que permite obtener de un FTP el instalador de Gclient correspondiente a la arquitectura y sistema operativo de las máquinas seleccionadas, copiarlo en cada una y ejecutarlo.	

Tabla 22. Tarea de ingeniería: Instalar Gclient para Windows

Tarea de ingeniería	
Número de tarea: 8	**Número de HU:** 3
Nombre de la tarea: Instalar Gclient para Windows	
Tipo de tarea: Desarrollo	**Puntos Estimados:** 1
Fecha de inicio: 30 de marzo de 2015	**Fecha de fin:** 3 de abril de 2015
Programadores responsables: Yoel Hernández Camejo y Arianna Martínez Sánchez	
Descripción: El usuario selecciona las máquinas que desee instalar, oprime el icono “Instalar” y el sistema le solicita las credenciales para acceder a las máquinas seleccionadas. Luego el sistema debe guardar en la base de datos la dirección IP, mensaje del resultado de la instalación, fecha y responsable de realizar la instalación automática de las máquinas.	

Tabla 23. Tarea de ingeniería: Mostrar registro de instalación

Tarea de ingeniería	
Número de tarea: 9	**Número de HU:** 4
Nombre de la tarea: Mostrar registro de instalación	
Tipo de tarea: Desarrollo	**Puntos Estimados:** 1
Fecha de inicio: 6 de abril de 2015	**Fecha de fin:** 10 de abril de 2015
Programadores responsables: Yoel Hernández Camejo y Arianna Martínez Sánchez	

Descripción: El usuario selecciona el menú “Registro de Instalación” y el sistema busca en la base de datos y muestra un listado con la dirección IP, la fecha, un mensaje que indica si se instaló o no Gclient y responsable de realizar la instalación automática de las máquinas.

Tabla 24. Tarea de ingeniería: Mostrar configuración de instalación

Tarea de ingeniería	
Número de tarea: 10	**Número de HU:** 5
Nombre de la tarea: Mostrar configuración de instalación	
Tipo de tarea: Desarrollo	**Puntos Estimados:** 0.4
Fecha de inicio: 13 de abril de 2015	**Fecha de fin:** 14 de abril de 2015
Programadores responsables: Yoel Hernández Camejo y Arianna Martínez Sánchez	
Descripción: El usuario selecciona el menú “Configuración” y el sistema busca en la base de datos y muestra un listado con el IP y la ruta del FTP donde se encuentren los instaladores de Gclient y el IP del servidor de GRHS.	

Tabla 25. Tarea de ingeniería: Modificar configuración de instalación

Tarea de ingeniería	
Número de tarea: 11	**Número de HU:** 5
Nombre de la tarea: Modificar configuración de instalación	
Tipo de tarea: Desarrollo	**Puntos Estimados:** 0.6
Fecha de inicio: 15 de abril de 2015	**Fecha de fin:** 17 de abril de 2015
Programadores responsables: Yoel Hernández Camejo y Arianna Martínez Sánchez	
Descripción: El usuario después de acceder al menú “Configuración” selecciona la máquina a la que desea cambiar la configuración y oprime el icono “Modificar”. Luego el sistema le muestra un formulario que le permite cambiar el IP y la ruta del FTP donde se encuentren los instaladores de Gclient y el IP del servidor de GRHS. El usuario modifica los datos y el sistema los guarda en la base de datos y muestra el mensaje “Se realizó el cambio satisfactoriamente”.	

Tabla 26. Tarea de ingeniería: Generar informe de instalación

Tarea de ingeniería	
Número de tarea: 12	**Número de HU:** 6
Nombre de la tarea: Generar informe de instalación	
Tipo de tarea: Desarrollo	**Puntos Estimados:** 1
Fecha de inicio: 20 de abril de 2015	**Fecha de fin:** 24 de abril de 2015
Programadores responsables: Yoel Hernández Camejo y Arianna Martínez Sánchez	
Descripción: El usuario oprime el icono “Exportar” y el sistema muestra una ventana que le permite seleccionar la ubicación donde desea guardar el archivo *.exe* que contendrá el listado de las máquinas identificadas mediante el descubrimiento de la red con sus respectivos datos (dirección IP, fecha, mensaje que indica si se instaló o no Gclient y responsable de realizar la instalación).	

3.2 Estándares de Codificación

Los estándares de codificación son aquellos que permiten entender de manera rápida, fácil y sencilla el código empleado en el desarrollo de un software. Además, garantizan un mantenimiento óptimo de dicho código por parte del programador. (Avilez, y otros, 2010)

Comprenden todos los aspectos de generación de código dentro un proyecto, deben reflejar un estilo armonioso como si un único programador hubiera escrito todo el código de una sola vez. La confección de estos estándares debe ser definida al comienzo de la implementación para garantizar que todos los programadores trabajen de manera coordinada. (Microsoft, 2010)

Estos estándares ayudan a asegurar que el código tenga una alta calidad, menos errores y pueda ser mantenido fácilmente. Para lograr este objetivo se utilizó la Guía de estilo para el código Python (PEP 8). Esta guía posee una gran cantidad de convenciones para escribir código legible, dentro de las cuales se destacan: (Rossum, y otros, 2001)

- ✓ Usar cuatro espacios por indentación.
- ✓ No se deben mezclar tabulaciones y espacios.
- ✓ Limitar todas las líneas a un máximo de caracteres (120 en este proyecto).
- ✓ Las sentencias de *import* deben de estar generalmente separadas una en cada línea.

- ✓ Las sentencias *import* deben estar siempre en la parte superior del archivo, agrupadas de la siguiente manera:
  - Librería estándar
  - Librerías de terceros
  - *import's* de la aplicación local.
- ✓ Usar espacios alrededor de los operadores aritméticos.
- ✓ No usar espacios alrededor del signo igual cuando se encuentre en un listado de argumentos de una función.
- ✓ Evitar usar espacios en blanco innecesarios.
- ✓ Utilizar el estilo CamelCase para nombrar clases y el lower_case_with_underscores para funciones y métodos.

3.3 Pruebas

Las pruebas son procesos que permiten verificar y revelar la calidad de un producto de software, en las cuales un sistema o componente es ejecutado bajo condiciones o requerimientos especificados. Una de las principales fortalezas de la metodología de desarrollo XP es el proceso de pruebas, el cual permite asegurar el éxito del producto al realizarse de manera continua. La continuidad proporciona la obtención de un producto de mayor calidad, pues disminuye el tiempo transcurrido entre la aparición de un error y su detección, permitiendo así que se corrijan de forma sencilla. XP divide las pruebas del sistema en dos grupos: las pruebas unitarias, que son diseñadas por los programadores y tienen como finalidad la verificación del código y las pruebas de aceptación o pruebas funcionales que están destinadas a evaluar si al final de cada iteración se obtuvo la funcionalidad requerida y diseñada por el cliente final. (Corbea, y otros, 2007)

Para la validación del módulo “Instalación automática de Gclient para el sistema GRHS” se realizaron un conjunto de pruebas unitarias y de aceptación, las cuales se detallan a continuación.

3.3.1 Pruebas unitarias

Las pruebas unitarias son una forma de probar el correcto funcionamiento de un módulo, sirven para asegurar que cada uno de los módulos funcione correctamente por separado y son diseñadas por los programadores. Las pruebas unitarias tienen como base realizar pruebas en pequeños fragmentos del código de la aplicación. Los fragmentos deben ser unidades estructurales del programa encargados de una tarea específica, en programación orientada a objetos se puede

afirmar que estas unidades son los métodos o las funciones que se tienen definidas. Estas pruebas no revelan errores en la integración de las partes unitarias ni tampoco otros problemas como el bajo rendimiento de las aplicaciones o problemas derivados del sistema sobre el que está ejecutándose el programa. El objetivo de dichas pruebas es el aislamiento de partes del código y la demostración de que estas partes no contienen errores. (Covelo, 2013)

Al culminar cada funcionalidad se aplicó una prueba unitaria, para lograr este fin se utilizó el módulo *unittest* que permite realizar pruebas automatizadas al código. Se realizaron un total de 7 pruebas unitarias, a continuación se muestra el código de la prueba realizada a la clase *RegisterInstall*.

```
class RegisterInstallTestCase(TestCase):
    """
    Clase para represntar las pruebas del modelo RegisterInstallerTestCase
    """
    def test_create_registro(self):
        """
        Esta es una prueba para ver el comportamiento de la vista crear
        para el modelo RegisterInstall
        """
        register = RegisterInstall.objects.create(address="10.8.1.8", responsable="Yoel",
                                                  fecha="2015-05-27 00:20:54.989549-04", message="Instalado")
        self.assertEqual(RegisterInstall.objects.count(), 1)
        this_register = RegisterInstall.objects.get(id=register.id)
        self.assertEqual(register.address, this_register.address)
        RegisterInstall.objects.filter(id=register.id).delete()
        self.assertEqual(RegisterInstall.objects.count(), 0)
    def test_update_registro(self):
        """
        Esta es una prueba para realizar para ver el comportamiento de la vista
        actualizar para el modelo RegisterInstall
        """
        register = RegisterInstall.objects.create(address="10.8.1.8", responsable="Yoel",
                                                  fecha="2015-05-27 00:20:54.989549-04", message="Instalado")
        self.assertTrue(register.address is not None and register.address is '10.8.1.8')
        address = '10.0.0.1'
        self.assertFalse(register.address is not register.address)
        register.address = address
        register.save()
        self.assertTrue(RegisterInstall.objects.get(address=address), '10.0.0.1')
    def test_delete_registro(self):
        """
        Esta es una prueba para ver el comportamiento de la vista eliminar para
        el modelo RegisterInstall
        """
        self.assertTrue(RegisterInstall.objects.create(address="10.8.1.8", responsable="Yoel",
                                                  fecha="2015-05-27 00:20:54.989549-04", message="Instalado"))
        self.assertEqual(RegisterInstall.objects.count(), 1)
        RegisterInstall.objects.get(message="Instalado").delete()
        self.assertFalse(RegisterInstall.objects.count(), 1)
```

Figura 7. Pruebas unitarias de la clase RegisterInstallTestCase

Al aplicar las pruebas unitarias a la aplicación en la última iteración se obtuvo un resultado satisfactorio, permitiendo comprobar que el código de la aplicación no presenta fallos o errores y aumenta la calidad del desarrollo. En la Figura 10 se muestran los resultados obtenidos al aplicar las pruebas.

```
archer@archer-Satellite-Pro-L650:~/Documentos/tlm_web$ python manage.py test src.apps.install
Creating test database for alias 'default'...
[<Permission_Menu: User list | Items List >]
[<Permission_Menu: User list | Can delete user >]
[<Permission_Menu: User list | Can change user >]
[<Permission_Menu: User list | User Details >]
[<Permission_Menu: User list | Can add user >]
[<Permission_Menu: User list | Password Change >]
[<Permission_Menu: Group Permissions list | Save Permission >]
[<Permission_Menu: Group Permissions list | Items List >]
[<Permission_Menu: Groups list | Can change group >]
[<Permission_Menu: Groups list | Can delete group >]
[<Permission_Menu: Groups list | Can add group >]
[<Permission_Menu: Groups list | Items List >]
[<Permission_Menu: Installation log | Excel Export >]
[<Permission_Menu: Installation log | You can install >]
[<Permission_Menu: Configuration | You can install >]
[<Permission_Menu: Configuration | Can change configuration >]
[<Permission_Menu: List Computers | Excel Export >]
[<Permission_Menu: List Computers | You can install >]
.......
----------------------------------------------------------------------
Ran 7 tests in 4.869s

OK
Destroying test database for alias 'default'...
archer@archer-Satellite-Pro-L650:~/Documentos/tlm_web$
```

Figura 8. Resultados obtenidos de las pruebas unitarias

3.3.2 Pruebas de aceptación

Las pruebas de aceptación o pruebas funcionales son creadas a partir de las historias de usuario. Durante una iteración, las historias de usuario seleccionadas durante la planificación de cada iteración se convertirán en pruebas de aceptación. El cliente o usuario especifica los escenarios a probar cuando una historia de usuario ha sido correctamente implementada. Una historia de usuario puede tener más de una prueba de aceptación para asegurar un funcionamiento correcto y no se considera completa hasta que no supera las pruebas correspondientes. (Del Valle, 2010)

El objetivo de las pruebas de aceptación es validar que un sistema cumple con el funcionamiento esperado y permitir al usuario de dicho sistema determinar el criterio de aceptación, desde el punto de vista de la funcionalidad y el rendimiento. (Palacios, 2008)

Las pruebas de aceptación son pruebas de caja negra[16], cada una de ellas representa una salida esperada del sistema. Los clientes son responsables de verificar el correcto funcionamiento de las pruebas de aceptación y revisar las puntuaciones de cada una para indicar cuáles de las pruebas fallidas tienen mayor prioridad. El aseguramiento de calidad es una parte esencial del proceso de XP. (Del Valle, 2010)

Se elaboraron un conjunto de pruebas de aceptación para comprobar el correcto funcionamiento del módulo, estas pruebas quedan registradas en tablas que comprenden las siguientes secciones:

- ✓ **Código:** identificador de la prueba.
- ✓ **Historia de Usuario (HU):** nombre de la HU a la que hace referencia.
- ✓ **Nombre:** identifica la prueba que se describe.
- ✓ **Descripción:** breve descripción de la funcionalidad que se desea probar.
- ✓ **Condiciones de Ejecución:** condiciones que deben cumplirse para poder realizar la prueba.
- ✓ **Entradas / Pasos de Ejecución:** descripción de cada uno de los pasos realizados durante el desarrollo de la prueba, se tendrá en cuenta cada una de las entradas que hace el usuario con el objetivo de verificar si se obtiene el resultado esperado.
- ✓ **Resultado esperado:** breve descripción del resultado que se espera obtener con la prueba.
- ✓ **Evaluación de la prueba:** acorde al resultado de la prueba realizada se emitirá una evaluación.
- ✓ **Resultado obtenido:** breve descripción del resultado obtenido con la prueba.

Se elaboraron un conjunto de 7 pruebas de aceptación, las cuales se exponen a continuación:

Tabla 27. Prueba de aceptación: Mostrar información

Prueba de aceptación	
Código: HU1_P1	**HU:** Mostrar información de las máquinas identificadas en la red
Nombre: Mostrar información	

16 Pruebas de caja negra: se llevan a cabo sobre la interfaz del software, obviando el comportamiento interno y la estructura del programa.

Descripción: Probar que el sistema muestra el identificador, la dirección IP, si tienen o no instalado Gclient, la localización, la subred y la fecha de las máquinas que fueron descubiertas.
Condiciones de ejecución: El usuario tiene que estar previamente autenticado en el sistema.
Entradas / Pasos de Ejecución: Autenticarse en el sistema GRHS. Oprimir el botón “Instalación”.
Resultado esperado: El sistema muestra el identificador, la dirección IP, si tienen o no instalado Gclient, la localización, la subred y la fecha de las máquinas que fueron descubiertas.
Resultado obtenido: El sistema muestra el identificador, la dirección IP, si tienen o no instalado Gclient, la localización, la subred y la fecha de las máquinas que fueron descubiertas.
Evaluación de la prueba: Satisfactoria

Tabla 28. Prueba de aceptación: Filtrar información

Prueba de aceptación	
Código: HU1_P2	**HU:** Mostrar información de las máquinas identificadas en la red
Nombre: Filtrar información	
Descripción: Probar que el sistema filtre la información de acuerdo a los criterios (dirección IP, localización, subred y si tiene o no instalado Gclient).	
Condiciones de ejecución: El usuario tiene que estar previamente autenticado en el sistema.	
Entradas / Pasos de Ejecución: Autenticarse en el sistema GRHS. Oprimir el botón “Instalación”. Seleccionar el o los criterio(s) de búsqueda en el campo “Filtrar”.	
Resultado esperado: El sistema filtra la información de acuerdo a los criterios (dirección IP, localización, subred y si tiene o no instalado Gclient).	

Resultado obtenido: El sistema filtra la información de acuerdo a los criterios (dirección IP, localización, subred y si tiene o no instalado Gclient).
Evaluación de la prueba: Satisfactoria

Tabla 29. Prueba de aceptación: Instalación de Gclient para distribuciones GNU/Linux

Prueba de aceptación	
Código: HU2_P1	**HU:** Instalación de Gclient para distribuciones GNU/Linux
Nombre: Instalación de Gclient para distribuciones GNU/Linux	
Descripción: Probar que el sistema permite instalar Gclient en las máquinas con distribuciones GNU/Linux.	
Condiciones de ejecución: El usuario tiene que estar previamente autenticado en el sistema. Se debe conocer el usuario y la contraseña de las máquinas a instalar. Las estaciones de trabajo deben tener habilitado el protocolo SSH.	
Entradas / Pasos de Ejecución: Autenticarse en el sistema GRHS. Oprimir el botón “Instalación”. Seleccionar las máquinas a instalar. Oprimir el icono “Instalar”. Insertar las credenciales. Oprimir el botón “Aceptar”.	
Resultado esperado: El sistema instala Gclient en las máquinas con distribuciones GNU/Linux.	
Resultado obtenido: El sistema instala Gclient en las máquinas con distribuciones GNU/Linux.	
Evaluación de la prueba: Satisfactoria	

Tabla 30. Prueba de aceptación: Instalación de Gclient para el sistema operativo Windows

Prueba de aceptación	
Código: HU3_P1	**HU:** Instalación de Gclient para el sistema operativo Windows
Nombre: Instalación de Gclient para el sistema operativo Windows	

Descripción: Probar que el sistema permite instalar Gclient en las máquinas con sistema operativo Windows.
Condiciones de ejecución: El usuario tiene que estar previamente autenticado en el sistema. Se debe conocer el usuario y la contraseña de las máquinas a instalar. Las estaciones de trabajo deben tener habilitado el protocolo SSH.
Entradas / Pasos de Ejecución: Autenticarse en el sistema GRHS. Oprimir el botón “Instalación”. Seleccionar las máquinas a instalar. Oprimir el icono “Instalar”. Insertar las credenciales. Oprimir el botón “Aceptar”.
Resultado esperado: El sistema instala Gclient en las máquinas con sistema operativo Windows.
Resultado obtenido: El sistema instala Gclient en las máquinas con sistema operativo Windows.
Evaluación de la prueba: Satisfactoria

Tabla 31. Prueba de aceptación: Mostrar registro de instalación

Prueba de aceptación	
Código: HU4_P1	**HU:** Mostrar registro de instalación
Nombre: Mostrar registro de instalación	
Descripción: Probar que el sistema muestra el identificador, la dirección IP, mensaje del resultado de la instalación, fecha y responsable de realizar la instalación automática de las máquinas.	
Condiciones de ejecución: El usuario tiene que estar previamente autenticado en el sistema.	
Entradas / Pasos de Ejecución: Autenticarse en el sistema GRHS. Oprimir el botón “Instalación”. Seleccionar “Registro de Instalación” en el menú lateral.	

Resultado esperado: El sistema muestra el identificador, la dirección IP, mensaje del resultado de la instalación, fecha y responsable de realizar la instalación automática de las máquinas.
Resultado obtenido: El sistema muestra el identificador, la dirección IP, mensaje del resultado de la instalación, fecha y responsable de realizar la instalación automática de las máquinas.
Evaluación de la prueba: Satisfactoria

Tabla 32. Prueba de aceptación: Mostrar configuración de instalación

Prueba de aceptación	
Código: HU5_P1	**HU:** Configuración de instalación
Nombre: Mostrar configuración de instalación	
Descripción: Probar que el sistema muestra la dirección IP del servidor y la dirección del FTP donde está el instalador de Gclient en la ventana correspondiente a la configuración del sistema.	
Condiciones de ejecución: El usuario tiene que estar previamente autenticado en el sistema.	
Entradas / Pasos de Ejecución: Autenticarse en el sistema GRHS. Oprimir el botón “Instalación”. Seleccionar “Configuración” en el menú lateral.	
Resultado esperado: El sistema muestra la dirección IP del servidor y la dirección del FTP donde está el instalador de Gclient en la ventana correspondiente a la configuración del sistema.	
Resultado obtenido: El sistema muestra la dirección IP del servidor y la dirección del FTP donde está el instalador de Gclient en la ventana correspondiente a la configuración del sistema.	
Evaluación de la prueba: Satisfactoria	

Tabla 33. Prueba de aceptación: Modificar configuración de instalación

Prueba de aceptación	
Código: HU5_P2	**HU:** Configuración de instalación
Nombre: Modificar configuración de instalación	

Descripción: Probar que el sistema modifica la dirección IP del servidor y la dirección del FTP donde está el instalador de Gclient.
Condiciones de ejecución: El usuario tiene que estar previamente autenticado en el sistema.
Entradas / Pasos de Ejecución: Autenticarse en el sistema GRHS. Oprimir el botón “Instalación”. Seleccionar “Configuración” en el menú lateral. Seleccionar el icono “Modificar”.
Resultado esperado: El sistema modifica la dirección IP del servidor y la dirección del FTP donde está el instalador de Gclient.
Resultado obtenido: El sistema modifica la dirección IP del servidor y la dirección del FTP donde está el instalador de Gclient.
Evaluación de la prueba: Satisfactoria

Tabla 34. Prueba de aceptación: Generar informe de instalación

Prueba de aceptación	
Código: HU6_P1	**HU:** Generar informe de instalación
Nombre: Generar informe de instalación	
Descripción: Probar que el sistema exporta a hojas de cálculo la información correspondiente al registro de instalación y a las máquinas descubiertas por la red.	
Condiciones de ejecución: El usuario tiene que estar previamente autenticado en el sistema.	
Entradas / Pasos de Ejecución: Autenticarse en el sistema GRHS. Oprimir el botón “Instalación”. Seleccionar el icono “Exportar”.	

Resultado esperado: El sistema exporta a hojas de cálculo la información correspondiente al registro de instalación y a las máquinas descubiertas por la red.
Resultado obtenido: El sistema exporta a hojas de cálculo la información correspondiente al registro de instalación y a las máquinas descubiertas por la red.
Evaluación de la prueba: Satisfactoria

Resultados de las pruebas de aceptación:

Las pruebas aplicadas contribuyeron a mejorar la calidad y las funcionalidades del sistema, detectándose en la primera iteración de pruebas de la aplicación un total de cuatro no conformidades, las cuales fueron resueltas. En una segunda iteración de pruebas para esta versión se detectaron tres no conformidades, las cuales fueron resueltas satisfactoriamente. En la tercera iteración no se detectaron no conformidades, logrando así la satisfacción del cliente y el cumplimiento del objetivo planteado. A continuación se muestra el comportamiento de las no conformidades por iteraciones.

Figura 9.Resultados obtenidos al realizar las pruebas de aceptación

3.4 Conclusiones del capítulo

El empleo de los estándares de codificación facilitó la lectura, comprensión y mantenimiento del código para los desarrolladores. Por otra parte, las pruebas de aceptación permitieron comprobar que las funciones son operativas a través de la interfaz del software, que la entrada se acepta de forma adecuada y se produce un resultado correcto, manteniendo así la integridad de la información externa. Las pruebas unitarias sirvieron para comprobar internamente las funciones del módulo, facilitando la detección de no conformidades para su corrección.

CONCLUSIONES GENERALES

Con la realización del presente trabajo de diploma se desarrolló un módulo que permitió automatizar el proceso de instalación de Gclient para el sistema GRHS y arribar a las siguientes conclusiones:

- ✓ El estudio de sistemas encargados de realizar instalación automática de aplicaciones en la red, permitió determinar que era necesario realizar un módulo para automatizar el proceso de instalación de Gclient, ya que gran parte de ellos son sistemas propietarios y no son compatibles con distribuciones GNU/Linux y Gclient cuenta con versiones para los sistemas operativos Debian, Nova y Ubuntu.

- ✓ La selección de la metodología XP permitió obtener un modelo guía para la implementación del sistema mediante la generación de los artefactos correspondientes a los flujos de trabajo que propone, así como la culminación en tiempo a partir del cronograma planificado para cumplir los objetivos de la investigación.

- ✓ La validación del módulo haciendo uso de las pruebas unitarias y de aceptación, permitió obtener una solución estable, correcta, que garantiza robustez y flexibilidad a cambios, lo cual posibilita que se cumpla satisfactoriamente el objetivo trazado para el presente trabajo de diploma, dando solución a la problemática planteada inicialmente por el cliente.

RECOMENDACIONES

Debido a los resultados de la investigación efectuada y a la experiencia adquirida durante la realización de este trabajo, se expone la siguiente recomendación:

- ✓ Incorporar al módulo la posibilidad de instalar Gclient utilizando el dominio LDAP[17].

17 LDAP: (Lightweight Directory Access Protocol) el protocolo ligero de acceso a directorios es un conjunto de protocolos abiertos usados para acceder a la información guardada centralmente a través de la red.

REFERENCIAS BIBLIOGRÁFICAS

Alchin, Marty. 2013. *"Pro Django".* Second Edition. s.l. : Apress, 2013. pág. 300. ISBN 978-1-4302-5809-4.

Alegsa, Leandro. 2014. "Definición de Red de Computadoras". *Diccionario de Informática y Tecnología. ALEGSA.com.arg.* [En línea] Santa Fe, Argentina, 2014. [Citado el: 3 de Noviembre de 2014.] http://www.alegsa.com.ar/Dic/red%20de%20computadoras.php.

—. 2015. "Definición de UML". *Diccionario de Informática y Tecnología. ALEGSA.com.arg.* [En línea] 2015. [Citado el: 2015 de Abril de 22.] http://www.alegsa.com.ar/Dic/uml.php.

Álvarez, Sara. 2007. "Sistemas gestores de bases de datos". *DesarrolloWeb.com.* [En línea] 31 de Julio de 2007. [Citado el: 10 de Noviembre de 2014.] http://www.desarrolloweb.com/articulos/sistemas-gestores-bases-datos.html.

Arthorne, John. 2004. eclipse. [En línea] 2004. [Citado el: 16 de Diciembre de 2014.] https://eclipse.org/org.

Ashkenas, Jeremy. 2010. BACKBONE.js. [En línea] 10 de Octubre de 2010. [Citado el: 10 de Noviembre de 2014.] http://backbonejs.org/.

Avilez, Anny Almanza, y otros. 2010. "Estándares de Codificación". *CEDES (Contenidos Educativos Digitales para Educación Superior).* [En línea] 2010. [Citado el: 8 de Mayo de 2015.] http://www.aves.edu.co/ovaunicor/recursos/view/265.

Beck, Kent. 2000. *"Extreme Programming Explained".* Boston : Addison-Wesley, 2000.

Beck, Kent y Cunningham, Ward. 1989. *"A Laboratory For Teaching Object-Oriented Thinking ".* New Orleans, Louisiana : SIGPLAN Notices, 1989.

Beck, Kent y Fowler, M. 2000. *"Planning Extreme Programming".* s.l. : Addison Wesley, 2000.

Bernabe, Roberto Aguilar y Espil, Martín Ruís. 2014. *"Model View Controller".* Trujillo : Escuela Académico Profesional de Ingeniería Informática, Facultad de Ciencias Físicas y Matemáticas, Universidad Nacional, 2014.

Boticario, Jesús González y Vázquez, Elena Gaudioso. 2001. *"Capítulo 6. JavaScript. Aprender y formar en Internet".* Madrid, España : Internacional Thomson Editors Spain Paraninfo, S.A., 2001. ISBN 84-283-2743-2.

Canós, José H., Letelier, Patricio y Penadés, Mª Carmen. 2011. *"Métodologías Ágiles en el Desarrollo de Software".* s.l. : DSIC -Universidad Politécnica de Valencia, 2011.

Cook, Tim. 2011. "Apple Remote Desktop 3". *Apple.* [En línea] 2011. [Citado el: 20 de Enero de 2015.] https://www.apple.com/es/remotedesktop/.

Corbea, Maite Rodríguez y Pérez, Meylin Ordóñez. 2007. *"LA METODOLOGÍA XP APLICABLE AL DESARROLLO DEL SOFTWARE EDUCATIVO EN CUBA".* La Habana, Cuba : Universidad de las Ciencias Informáticas (UCI), 2007.

Correa, Luis Manzano. 2013. "Desktop Central - software de Gestión e Inventario de Estaciones de Trabajo y Servidores". *REUSCH.* [En línea] 2013. [Citado el: 15 de Enero de 2015.] http://www.manageengine.cl/desktop-central/index.html.

Covelo, Abraham . 2013. Unit testing (pruebas unitarias). [En línea] 2013. [Citado el: 26 de Marzo de 2015.] http://www.novanebula.net/blog/archives/99-Unit-testing-pruebas-unitarias.html.

Del Valle. 2010. "Pruebas de aceptación". *WikiCC.* [En línea] 30 de Agosto de 2010. [Citado el: 10 de Mayo de 2015.] http://streaming.uvg.edu.gt/mediawiki/index.php?title=Pruebas_de_aceptaci%C3%B3n.

Emco. 2014. "EMCO Remote Installer". *EMCO Software.* [En línea] 3 de Febrero de 2014. [Citado el: 16 de Enero de 2015.] http://emcosoftware.com/remote-installer.

Etcheverry, Ing. Lorena. 2010. "Arquitectura de un Sistema de Información. Definiciones de arquitectura (II)". *PEDECIBA - Programa de Desarrollo de las Ciencias Básicas.* [En línea] Marzo de 2010. [Citado el: 2 de Abril de 2015.] http://www.pedeciba.edu.uy/bioinformatica/sibdyw/Clase_3.pdf.

Gamma, Erich, y otros. 2002. *"Patrones de diseño : elementos de software orientado a objetos reutilizable".* [trad.] César Fernández Acebal Universidad de Oviedo. Madrid : Addison Wesley, 2002. pág. 364. ISBN 8478290591 9788478290598.

Global Development Group, The PostgreSQL. 2015. "About". *PostgreSQL.* [En línea] 2015. [Citado el: 8 de Enero de 2015.] http://www.postgresql.org/about/.

Gómez, Mauro. 2013. "Patrones Arquitectónicos". *Ingenio DS.* [En línea] 16 de Septiembre de 2013. [Citado el: 25 de Marzo de 2015.] https://ingeniods.wordpress.com/2013/09/16/patrones-arquitectonicos/.

Hickson, I. y Hyatt, D. 2014. "HTML 5.1 Nightly". *World Wide Web Consortium (W3C).* [En línea] 13 de Noviembre de 2014. [Citado el: 25 de Noviembre de 2014.] http://www.w3.org/html/wg/drafts/html/master/introduction.html#html-vs-xhtml.

Keith-Magee, Russell. 2014. "The web framework for perfectionists with deadlines". *django.* [En línea] 2014. [Citado el: 16 de Diciembre de 2014.] https://www.djangoproject.com/.

Maldonado, Daniel. 2013. "Que son los IDE de Programación". *El CoDiGo K.* [En línea] 2013. [Citado el: 15 de Enero de 2015.] http://www.elcodigok.com.ar/2007/09/que-son-los-ide-de-programacin.html.

Martín, E. LLeonart, García, A. y Rovira, M. 2010. *"Patrones".* Valencia : Facultad de Informática-Universidad Politécnica, 2010.

Maza, Miguel Ángel Sánchez. 2012. *"JavaScript".* s.l. : IC Editorial, 2012. ISBN 978-8495733184.

Microsoft. 2010. "Revisiones de código y estándares de codificación". *Microsoft Developer Network.* [En línea] 2010. [Citado el: 8 de Mayo de 2015.] https://msdn.microsoft.com/es-es/library/aa291591%20%28v=vs.71%29.aspx.

Palacios, Ricardo Colomo. 2008. "Pruebas. Ingeniería del Software III". *OpenCourseWare.* [En línea] Marzo de 2008. [Citado el: 10 de Mayo de 2015.] http://ocw.uc3m.es/ingenieria-informatica/ingeniera-del-software-iii/materialclase/ISIII_09_PRUE.pdf.

Pillou, Jean-François. 2015. "Entorno cliente/servidor". *Kioskea.net.* [En línea] Mayo de 2015. [Citado el: 6 de Mayo de 2015.] http://es.kioskea.net/contents/148-entorno-cliente-servidor.

Pressman, Roger S. 2001. *"Ingeniería del Software: una tecnología estratificada. Ingeniería del Software. Un enfoque práctico".* Quinta Edición. España : McGraw Hill, 2001.

Puro Software. 2007. "PgAdmin - Excelente gestor PostGreSQL". *Puro Software ...Tu Portal de Software Libre y Gratuito.* [En línea] 2007. [Citado el: 12 de Diciembre de 2014.] http://www.purosoftware.com/programacion-bases-de-datos/11-pg-admin-3.html.

Rossum, Guido Van, Warsaw, Barry y Coghlan, Nick. 2001 . "Style Guide for Python Code". *python.* [En línea] 5 de Julio de 2001 . [Citado el: 12 de Mayo de 2015.] https://www.python.org/dev/peps/pep-0008/.

Sala, Jesús Javier Rodríguez. 2003. *"Introducción a la programación: teoría y práctica".* s.l. : Editorial Club Universitario, 2003. ISBN 9788484542742.

Sierra, Manuel. 2015. "Qué es y para qué sirve el lenguaje CSS (Cascading Style Sheets - Hojas de Estilo)". *aprenderaprogramar.com.* [En línea] 2015. [Citado el: 9 de Enero de 2015.] http://www.aprenderaprogramar.com/index.php?option=com_content&view=article&id=546:que-es-y-para-que-sirve-el-lenguaje-css-cascading-style-sheets-hojas-de-estilo&catid=46:lenguajes-yentornos&Itemid=163%29. DV00203A.

Sommerville, Ian. 2005. *"Ingeniería del software".* Séptima Edición. Madrid. España : Pearson Educaión. S.A., 2005. ISBN 84-7829-074-5.

UCI, Universidad de las Ciencias Informáticas. 2012. "Misión". *Portal de la Universidad de las Ciencias Informáticas.* [En línea] La Habana, Cuba, 2012. [Citado el: 14 de Octubre de 2014.] http://www.uci.cu/?q=mision.

Vallespir, Diego. 2002. CRC y un Taller. [En línea] 2002.

Varela, Luis González. 2011. "Tema 6. Patrones de diseño". [En línea] 2011. [Citado el: 15 de Abril de 2015.] https://eseida.wikispaces.com/file/view/Tema+6+-+Patrones+de+Dise%C3%B1o.pdf/407459462/Tema%206%20-%20Patrones%20de%20Dise%C3%B1o.pdf.

Vázquez, Ing. Iván Mendoza. 2011. "Definición de un Framework para aplicaciones Web con navegación sensible a concerns". *Trabajo de tesis para la obtención del título de Máster en Ingeniería de Software.* [En línea] 2011. [Citado el: 2 de Diciembre de 2014.] http://sedici.unlp.edu.ar/bitstream/handle/10915/4192/Documento_completo.pdf?sequence=1.

Visconti, Marcello y Astudillo, Hernán. 2004. *"Fundamentos de Ingeniería de Software".* 2004.

Wallace, Doug, Ragget, Isobel y Joel Aufgang. 2002. *"Extreme Programming for Web Projects".* s.l. : Addison Wesley, 2002. pág. 192. ISBN 0-201-79427-6.

Wallace, Doug, Raggett, Isobel y Aufgang, Joel. 2002. *"Extreme Programming for Web Projects".* s.l. : Addison Wesley, 2002. pág. 192. ISBN 0-201-79427-6.

BIBLIOGRAFÍA CONSULTADA

Alegsa, Leandro. 2015. "Definición de cliente/servidor (computación)". *Diccionario de Informática y Tecnología. ALEGSA.com.arg.* [En línea] 2015. [Citado el: 22 de Abril de 2015.] http://www.alegsa.com.ar/Dic/cliente%20servidor.php.

—. 2010. "Qué significa inventario - Información y significado de inventario". *Diccionario General de Español. Definiciones-de.com.* [En línea] Santa Fe, Argentina, 28 de Agosto de 2010. [Citado el: 3 de Noviembre de 2014.] http://www.definiciones-de.com/Definicion/de/inventario.php.

Álvarez, Miguel Angel. 2008. "Introducción a CSS3". *DesarrolloWeb.com.* [En línea] 9 de Junio de 2008. [Citado el: 16 de Diciembre de 2014.] http://www.desarrolloweb.com/articulos/introduccion-css3.html.

—. 2003. "Qué es Python". *DesarrolloWeb.com.* [En línea] 19 de Noviembre de 2003. [Citado el: 14 de Diciembre de 2014.] http://www.desarrolloweb.com/articulos/1325.php.

Beck, Kent. 1999. *"Extreme Programming Explained. Embrace Change." Una explicación de la programación extrema. Aceptar el cambio.* [trad.] Addison Wesley. s.l. : Pearson Education, 1999.

Benchimol, Daniel. 2011. *"HACKING DESDE CERO: Manuales Users".* Argentina : Creative Andina Corp., 2011. ISBN 987177303X, 9789871773039.

Etcheverry, Ing. Lorena. 2010. "Definiciones de arquitectura (II)". *Arquitectura de un Sistema de Información.* s.l. : IEEE 1471-2000, 2010.

Huice, Ing. Odaysa Rodríguez. 2012. *"Programa de Mejoara".* Carretera a San Antonio Km 2 ½. Torrens. Boyeros. Ciudad de La Habana. Cuba : Universidad de las Ciencias Informáticas (UCI), 2012. 0208_Proyecto Técnico.

IEEE. 2000. *Arquitectura Std 1471-2000.* 2000.

Letelier, Patricio y Penadés, Mª Carmen. 2004. "Métodologías ágiles para el desarrollo de software: eXtreme Programming (XP)". [En línea] 2004. [Citado el: 12 de Diciembre de 2014.] http://www.willydev.net/descargas/masyxp.pdf.

Melo, Julie Esperanza Daza. 2015. "Conceptos Básicos de Software". *Buenas Tareas.* [En línea] 25 de Febrero de 2015. [Citado el: 27 de Febrero de 2015.] http://www.buenastareas.com/ensayos/Conceptos-De-Software/68471920.html.

Moreno, Nohelia Martínez. 2009. "Sistemas de la Información de la Mercadotecnia. Unidad III. Organizaciones Virtuales y Sistemas Inteligentes". *Buenas Tareas.* [En línea] 2009. [Citado el: 4 de Diciembre de 2014.] http://www.buenastareas.com/ensayos/Concepto-Hardware/906786.html.

Newkirk, James y Martin, Robert C. 2002. *"La Programación Extrema en la práctica".* s.l. : Addison-Wesley Iberoamericana Espanya, S.A., 2002. ISBN 8478290575.

Pelaez, Juan Carlos. 2009. Arquitectura basada en capas. [En línea] 2009. http://geeks.ms/blogs/jkpelaez/archive/2009/05/29/arquitectura-basada-en-capas.aspx.

Pressman, Roger S. 2003. *Ingeniería del Software. Un enfoque práctico. Sexta Edición.* s.l. : Mc Graw Hill, 2003. 970-10-5473-3.

Rodríguez Corbea, Maite, Ordóñez Pérez, Meylin. 2007. *La Metodología XP Aplicable al Desarrollo del Software Educativo en Cuba.* Universidad de las Ciencias Informáticas, Cuba : s.n., 2007.

Rojas, Prof. Edgar. 2013. "Computación 1. Redes, Datos Numéricos, Lógicos y Alfa Numéricos". *Buenas Tareas.* [En línea] 25 de Octubre de 2013. [Citado el: 6 de Noviembre de 2014.] http://www.buenastareas.com/ensayos/La-Redes-De-Computadoras/40477951.html?_t=1&_p=2.

Valdarrama, Santiago Luis Del Pino. 2005. *"Programación extrema en pocos minutos: planificando la transición".* Cuba : Tono. Revista Técnica de la Empresa de Telecomunicaciones de Cuba, S.A., 2005. págs. 41-44. ISBN 18135056.

Vértice, Equipo. 2009. *"Diseño Básico de Páginas Web en HTML".* España : Publicaciones Vértice S.L, 2009. ISBN 978-84-9931-034-3.