**Universidad de las Ciencias Informáticas**

**Facultad 2**


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

# **Título:** Módulo de Gestión de Inventarios e Incidencias de Software en GNU/Linux.

**Autores:** David Delisle Flores.
Eilan Martin González.
**Tutor:** Ing. Ernesto Avilés Vázquez.
**Co-tutor:** Ing. Odaysa Rodríguez Huice.

La Habana, Junio 2012.

## DECLARACIÓN DE AUTORÍA

Declaramos que Eilan Martin Gonzalez y David Delisle Flores somos los únicos autores de este trabajo y autorizamos a la Facultad #2 de la Universidad de las Ciencias Informáticas (UCI) a hacer uso del mismo en su beneficio.

Para que así conste firmamos la presente a los 25 días del mes de junio del año 2012.

| | |
|---|---|
| _____ | _____ |
| Firma del Autor | Firma del Autor |
| Eilan Martin González | David Delisle Flores |
| _____ | _____ |
| Firma del Tutor | Firma del Tutor |
| Ing. Ernesto Avilés Vázquez | Ing. Odaysa Rodríguez Huice |

# RESUMEN

La mayoría de las entidades que poseen ordenadores para el desarrollo de sus actividades necesitan un sistema para el proceso de captura, análisis y consulta de la información de los activos informáticos de su red de computadoras, para un mejor control de los recursos y facilitar la toma de decisiones de los jefes de la Dirección de Tecnología y Sistemas respecto a los mismos. Actualmente existen diversas herramientas para controlar esta información, aunque algunas de estas presentan deficiencias como son: ser pasivas, puesto que no ejecutan acciones cuando ocurre un determinado cambio que no es permitido en la entidad, además de no realizar la monitorización de sus activos informáticos lo que es de gran ayuda pues puede detectar momentáneamente si son violadas las políticas de la entidad y no hay necesidad de esperar a realizar el inventario.

Después de haber estudiado las tendencias actuales de las herramientas de inventario a nivel mundial se necesita implementar un sistema que pueda cubrir las necesidades planteadas anteriormente y sobre todo que responda a las políticas establecidas en el centro donde se aplique, manteniendo cierto control en la red. Se necesita que el sistema sea capaz de inventariar y monitorizar a la vez toda una red de computadoras brindando la posibilidad de tomar acciones si ocurre alguna incidencia.

En la presente investigación se propone la fundamentación acerca de todas las herramientas utilizadas, además de todos los elementos que se necesitan para construir el Modulo de Gestión de Inventarios e Incidencias de Software en GNU\Linux ya que con la realización del mismo se pueden cubrir las necesidades expuestas anteriormente para las diferentes entidades que poseen una red de ordenadores.

# ÍNDICE

# ÍNDICE DE TABLAS

# ÍNDICE DE FIGURAS

# INTRODUCCIÓN

Desde tiempos inmemorables, los egipcios y demás pueblos de la antigüedad, acostumbraban almacenar grandes cantidades de alimentos para ser utilizados en los tiempos de sequía o calamidades. Debido a la necesidad de tener un control sobre estos y poder contabilizarlos surge el problema de los inventarios, para hacer frente a los periodos de escasez, que le asegurarán la subsistencia de la vida y el desarrollo de sus actividades normales.[1]

En un principio, el término inventario se utilizaba para referirse a una lista pormenorizada de mercaderías o artículos en la cual aparecía la descripción, el número, la cantidad y el valor de los bienes enumerados en dicha lista. En las empresas, actualmente, este término tiene un significado más genérico y se aplica también a la diversidad de materias primas, suministros, demás artículos y bienes que se encuentran en espera de su incorporación a la producción, que se hallan en proceso de elaboración o en forma de producto terminado. Este manejo contable permite a la empresa mantener el control oportunamente, así como también conocer al final del período contable un estado confiable de la situación económica de la empresa.[2]

Con el trascurso del tiempo, el avance de la sociedad y las tecnologías es de vital necesidad la mejora de las tareas que las personas realizan a diario en su beneficio, las tecnologías han permitido satisfacer necesidades esenciales como el mejoramiento de sistemas tanto manuales como físicos. Es de gran importancia confrontar el avance que se ha tenido en el campo de la informática, pues mediante los computadores el hombre ha logrado librarse de tareas manuales usando esta como recurso de efectividad.

Desde la llegada de la informática el control de la información es uno de los principales objetivos por el que fue creada. El hombre como ser racional en evolución constante, no solo le basta procesar la información con facilidad y alcanzar grandes deducciones o soluciones para sus problemáticas en el quehacer cotidiano, sino además almacenarlas, verificarlas y hasta compararlas con algún patrón que necesite seguir. De aquí la necesidad de aplicar inventario también a las computadoras.

---

[1] **Open E-learning.** http://www.mailxmail.com. *http://www.mailxmail.com.* [Online] Grupo Intercom. http://www.mailxmail.com/que-es-mailxmail_n. 1699-4914.

[2] **Imagina Network y Compas3 Comercio Electrónico.** http://aulafacil.com/. *http://aulafacil.com/.* [Online] Compás3 Comercio Electrónico S.L., 2000. http://aulafacil.com/gestion-stocks/curso/Lecc-33.htm. 28010.

Realizar una herramienta informática que obtenga un inventario en una red de ordenadores puede ser de gran utilidad puesto que este proceso puede ser muy sencillo o costar un poco de esfuerzo, depende del tamaño de la organización y del estado de sus registros, los cuales pueden almacenar gran cantidad de información acerca de los activos existentes. Una herramienta que permita automatizar el proceso de auditoría, posibilita un ahorro considerable de tiempo y recursos. Con esta el administrador pueda tener control de todo el software que tienen instalados sus usuarios, además de darle la posibilidad de saber si alguno de estos viola alguna norma establecida en su organización.

En Cuba se está trabajando para acelerar el desarrollo de la industria del software y potenciar su ubicación en el mercado internacional. Está encaminada hacia el software libre puesto a las ventajas que este presenta ante el software propietario.[3]

La Universidad de las Ciencias Informáticas (UCI) se ha ido convirtiendo en una potencial empresa desarrolladora de software, con el objetivo de ayudar en la economía del país. Con el fin de cumplir su objetivo se ha dividido en varias estructuras productivas especializadas en diferentes temas.[4]

Una de estas estructuras en la cual se ha dividido es el Centro de Telemática (TLM), el cual desarrolla sistemas y servicios informáticos en las ramas de las Telecomunicaciones y la Seguridad Informática. Entre los proyectos que se desarrollan en este centro se encuentra Gestión de Recursos de Hardware y Software el cual pretende realizar una herramienta que realice inventario en una red de computadoras.

En la mayoría de las entidades donde se utilizan ordenadores para el desarrollo de sus procesos, es necesario inventariar los activos informáticos, por lo que contar con un sistema automatizado para el proceso de captura, análisis y consulta de la información de su red de computadoras puede ser de gran utilidad para un mejor control de los recursos y facilitar la toma de decisiones a los administradores de la red. Por lo antes mencionado se plantea la siguiente **situación problemática**: Necesidad de gestionar y controlar de forma automatizada los recursos de software de una red de computadoras para los sistemas operativos Debian y Ubuntu. Aunque actualmente se utilizan varias aplicaciones destinadas a la gestión de inventarios de software, estas son pasivas, sólo notifican los cambios no siendo capaces de tomar

---

[3] **EUMEDNET, grupo de investigación.** http://www.eumed.net/. *http://www.eumed.net/.* [Online] Universidad de Málaga y de la Fundación Universitaria Andaluza Inca Garcilaso, 1994 . http://www.eumed.net/cursecon/ecolat/la/09/vsh.htm. SEJ-309.

[4] **Ramón Paumier Samón, Yoandy Pérez Villazón, Abel Meneses Abad.** es.scribd.com. *es.scribd.com.* [Online] [Cited: 10 25, 2011.] http://es.scribd.com/doc/17779155/Guia-Cubana-Migracion-a-Software-Libre.

decisiones, además de no presentar monitorización para mejorar la detección de cambios ocurridos en la red. Debido a tales dificultades enfrentadas se plantea como **problema a resolver:** ¿Cómo facilitar el proceso de inventariar el software en una red de computadoras para los sistemas operativos Debian y Ubuntu? Dando lugar a definir como **objeto de estudio**: El Proceso de Inventarios e Incidencias de Software en Debian y Ubuntu para la red de ordenadores de una entidad. Enmarcado en el **campo de acción**: el proceso de realizar inventario de software en los sistemas operativos Debian y Ubuntu

Para darle solución al problema a resolver se plantea como **objetivo general:** Desarrollar un Módulo de Gestión de Inventarios e Incidencias de Software en Debian y Ubuntu para la red de computadoras de una entidad. Desglosándose en los siguientes **objetivos específicos:**

- Seleccionar una metodología como guía para el proceso de desarrollo del sistema.
- Seleccionar las formas de obtener la información del software en los sistemas operativos Ubuntu y Debian.
- Seleccionar las formas de obtener las propiedades del sistema operativo Ubuntu y Debian.
- Detectar si ocurren cambios en el software y el sistema operativo de una computadora.
- Seleccionar las formas de almacenamiento para los inventarios e incidencias de software.
- Desarrollar un módulo que sea capaz de obtener la información, los cambios y las incidencias de todos los programas instalados en una computadora en los sistemas operativos Ubuntu y Debian.
- Desarrollar un módulo que sea capaz de realizar la monitorización de los identificadores de la PC, la instalación y desinstalación de los paquetes de software.

Por todo lo antes expuesto se plantea como **idea a defender**: El desarrollo de un Módulo de Gestión de Inventarios e Incidencias de Software para los sistemas operativos Debian y Ubuntu facilitará el proceso de inventario del software en las redes de computadoras de una entidad.

Para darle cumplimiento a los objetivos específicos se plantean las siguientes **tareas de la investigación**.

- Estudio del arte de las aplicaciones que realizan inventario de software.
- Diseño del plugin que realiza inventario y detecte cambios e incidencias en una red de computadores en Ubuntu y Debian.
- Implementación del plugin que realiza inventario, detecta cambios e incidencias en una red de computadores en Ubuntu y Debian.
- Prueba del plugin que realiza inventario, detecta cambios e incidencias en una red de computadores en Ubuntu y Debian.

- Diseño, del plugin que realiza la monitorización de los identificadores del ordenador, la instalación y desinstalación de paquetes de software.
- Implementación del plugin que realiza la monitorización de los identificadores del ordenador, la instalación y desinstalación de paquetes de software.
- Prueba del plugin que realiza la monitorización de los identificadores del ordenador, la instalación y desinstalación de paquetes de software.

En esta investigación fueron empleados métodos de investigación teóricos y empíricos.
Como métodos teóricos se utilizan:
**Método Analítico-Sintético:** Analizar características positivas y negativas de las herramientas de inventario de hardware y software desarrolladas hasta la actualidad, lograr reunir el conjunto de cualidades realmente útiles que ellas poseen
**Análisis Histórico-Lógico:** Se utiliza para realizar el estudio de la evolución de los sistemas de gestión de inventarios, así como las tendencias y tecnologías actuales en el desarrollo de este tipo de sistemas.
**Modelación:** Se utiliza para realizar la modelación del sistema propuesto, convertir los procesos en diagramas, modelos y artefactos que se puedan utilizar en el diseño.
Como métodos empíricos se utilizan:
**Entrevista:** Consultar con personas que están informadas sobre el tema de aplicaciones que realizan inventario de hardware y software en el país además de obtener conocimiento acerca de las necesidades de los usuarios en una red.
**Observación:** Permite conocer el funcionamiento de los procesos para determinar el problema a resolver y comprender como se realiza el proceso de inventario de software en una red de computadoras de determinada entidad.

El presente documento está estructurado en 4 capítulos.
**Capítulo 1:** Fundamentación teórica.
Estado del arte de las herramientas de inventario de hardware y software en una red de computadoras, a nivel internacional, de las tendencias, técnicas, tecnologías, metodologías y software usados en la actualidad o en las que se apoya para la creación de una herramienta similar que cumpla con los requisitos funcionales que realmente necesita la entidad.
**Capítulo 2:** Características del sistema.

En este capítulo queda conformado el modelo de negocio, se levantan los requisitos del sistema, estos incluyen funcionales y no funcionales para de esta forma definir lo que debe hacer el sistema.

**Capítulo 3:** Diseño del sistema.

Se desarrollan los diagramas para la realización del sistema, tales como diagrama de clases del diseño, además de definirse la arquitectura .Su enfoque es la construcción técnica del software.

**Capítulo 4:** Implementación y Prueba.

En este capítulo se muestra cómo será implementado el sistema. Se realizan los diagramas de componentes y el de despliegue además de los casos de prueba necesarios para determinar si el software cumple con los requisitos planteados.

# CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

## 1.1 Introducción

En el presente capítulo se exponen algunos conceptos de gran importancia para la realización de un inventario de software en una empresa, además de hacer un estudio exhaustivo de algunos de los sistemas ya existentes en el mercado a nivel mundial. También se abordan características de las herramientas y metodologías utilizadas.

## 1.2 Conceptos Fundamentales.

**Software:** Se puede definir como conjunto de programas, instrucciones y reglas informáticas que hacen posible la ejecución de diversas tareas en un ordenador. Igualmente es considerado como equipamiento lógico e intangible del ordenador, compendiando así a todas las aplicaciones informáticas.[5]

**Inventario de software**: Hacer un inventario de software implica analizar los productos de software instalados en ordenadores y confrontarlo con las licencias.[6]

**Incidencia**: Algo que se produce en el transcurso de un asunto o negocio y que repercute en estos alterándolos o interrumpiéndolos. [7]

---

[5] Microsoft.com. *Microsoft.com.* [Online] http://www.microsoft.com/spain/softlegal/msia/quees.aspx.

[6] **(RAE), Diccionario de la Lengua Española.** WordReference.com. *WordReference.com.* [Online] [Cited: Novienmbre 26, 2011.] http://www.wordreference.com/es/en/frames.asp?es=inventario.

[7] **SL, Larousse Editorial.** diccionarios.com. *diccionarios.com.* [Online] Larousse Editorial. http://www.diccionarios.com/detalle.php?palabra=incidencia&Buscar.x=51&Buscar.y=22&dicc_51=on&dicc_51=on.

## 1.3 Herramientas de inventario de software en el mundo.

Existen varias herramientas de inventario de software y hardware actualmente en el mercado, estas realizan principalmente el inventario de software instalados en los ordenadores (PC), permitiendo generalmente, descubrir y eliminar todo programa que ha sido instalado sin permiso, aunque es necesario destacar que ninguna de estas herramientas realiza monitorización en las PC clientes. Muchas de estas herramientas son multiplataforma, mientras que otras solo pueden realizar el inventario en un único sistema operativo. A continuación se exponen características principales de las herramientas de inventario de software identificadas como respaldo a la siguiente investigación.

### 1.3.1 Software Propietarios de Inventario de Hardware y Software.

#### 1.3.1.1 Login Inventory

Login Inventory es una aplicación permite realizar en pocos minutos un inventario de todo el software y hardware en una red informática sin instalar software adicional o agente en los clientes. Es instalado en un lugar central de la red (PC o servidor), apenas inicia su funcionamiento es capaz de realizar la recopilación de datos de todo ordenador con sistema operativo Windows.

Login Inventory solo se puede instalar en Windows NT, XP y 7[8], sin embargo puede realizar también la recopilación de datos de ordenadores con sistema operativo GNU/Linux y Mac OS. Este presenta características importantes, como son:

- ✓ Realiza los escáneres de la red en pocos minutos.
- ✓ Toda la información se muestra a través de Microsoft Management Console o a través de la interfaz web de Login Inventory.
- ✓ No existe límite de dispositivos u ordenadores en la red.

Aunque son varios los beneficios que trae el uso de esta aplicación es una herramienta privativa y no presenta agentes clientes, además, puede ser catalogada como "pasiva" debido a su poca interacción con el usuario por centrarse prácticamente solo en la obtención de informes. Su agente servidor no se encuentra implementado para distribuciones Linux, no ser una herramienta multiplataforma constituye

[8] **Iventory, Login.** Linux Party Group. *Linux Party Group.* [Online] Login Iventory. http://www.linux-party.com/modules.php?name=News&file=article&sid=4468.

una traba para el uso de la misma en otros sistemas operativos, debido a la variedad de sistema operativos en la actualidad, siendo esto un impedimento para su plena explotación.

#### 1.3.1.2 VEO

La aplicación VEO permite administrar los ordenadores de una red de computadoras tomando como vía el protocolo TCP/IP, por administrar se entiende tener el control y acceso a las PC(s) que se encuentran en la red. También permite realizar el inventario de hardware y software de estas computadoras. Su objetivo es evaluar la productividad de sus usuarios.

Otras de las funcionalidades reconocidas del VEO son:

- ✓ Permite cerrar aplicaciones en ejecución.
- ✓ Posibilita el despliegue de mensajes cuando se realiza algo indebido.
- ✓ Presenta la capacidad de controlar remotamente los equipos.

Está conformado por dos programas: la Consola (da las órdenes a los equipos remotos) y el agente remoto (recibe las órdenes y procede a realizarlas).

Aunque la aplicación VEO posee una gran cantidad de aspectos positivos vale señalar que es muy simple y de pobre interacción con el usuario, además no es multiplataforma ya que obtiene solamente el inventario de ordenadores con sistema operativo Windows debido a que solo presenta clientes para estaciones de trabajo Windows. No permite realizar reportes en pantalla debido a que exporta estos directamente a Excel. Es más una herramienta encaminada al control de la productividad en cada estación de trabajo que una herramienta de control de inventarios.

#### 1.3.1.3 NetSupport DNA

NetSupport DNA tiene la capacidad de realizar un informe acerca de los activos informáticos en una red de ordenadores, esta aplicación va más allá de una simple LAN o WAN, o sea, puede identificar activos que normalmente pueden “ocultarse” en una red remota o detrás de un cortafuego. Esta herramienta ofrece uno de los más completos y detallados módulos de inventario de hardware en la actualidad, además de hacer un resumen de todos los programas y aplicaciones instalados, detectados en cada ordenador ya sean Windows o Linux que tengan instalado el cliente DNA, posibilitando posteriormente visualizar esta información para un ordenador específico o bien para un departamento o grupo personalizado.

Presenta otras funcionalidades útiles, como son:

- ✓ Ofrece un soporte multiplataforma para los sistemas Windows y Linux.
- ✓ Su componente llamado Historial brinda un resumen cronológico completo de todas las actividades registradas por NetSupport DNA para todos los componentes principales basados en el inventario (hardware, software y usuarios), información que puede ser archivada y exportada componente por componente para el almacenamiento y el uso futuro.[9]

NetSupport DNA sin duda alguna es una aplicación de inventario de hardware y software efectiva de numerosas formas, pero tiene el inconveniente de ser una herramienta privativa, además, no es multiplataforma con respecto a su agente servidor pues no se encuentra implementado para Linux, aunque vale señalar que sus agentes clientes si están implementados para las distribuciones Windows y Linux.

#### 1.3.1.4 Total Network Inventory

Total Network Inventory es una aplicación de inventario de hardware y software para redes de escala. Es capaz de interrogar todos los ordenadores en la red y generar una información completa acerca del sistema operativo, hardware y software, posteriormente esta información es agregada a una base de datos centralizada permitiendo a los administradores de la red generar informes para cada uno o todos los ordenadores de la red. Es una de las aplicaciones que no instala software especial (agente de software) en las máquinas remotas.

Los informes que genera Total Network Inventory son excelentes ya que la información que dan sobre las máquinas en red es de fácil lectura y comprensión, como por ejemplo: la creación de informes con la información de un adaptador de red, direcciones IP, direcciones MAC, DHCP, DNS y WINS para cualquier número de ordenadores.

Ejemplos de la información que permite obtener este software son:

Software y sistema operativo:

- ✓ Programas instaladas en cada equipo.
- ✓ Nombre y versión del programa antivirus.[10]

---

[9] **NetSupport.** www.netsupportdna.com. *www.netsupportdna.com.* [Online] NetSupport. http://www.netsupportdna.com/es/inventory.asp.

[10] filecluster.es. *filecluster.es.* [Online] File Cluster. http://www.filecluster.es/programas/Total-Network-Inventory-30605.html.

Se puede resumir que Total Network Inventory es una aplicación con diversas funcionalidades apropiadas para el inventario de hardware y software en una red informática, sin embargo, es una herramienta privativa y no es multiplataforma. Además, al carecer de agentes clientes tiende a sobrecargar el tráfico de red debido al proceso de consulta que realiza sobre los ordenadores de la red.

#### 1.3.1.5 Lansweeper

Lansweeper es una aplicación de software que permite realizar una revisión completa de inventario de todos los elementos pertenecientes o instalados en una red local, algunos de estos elementos pueden ser el software, hardware e información sobre los usuarios. Esta herramienta no instala cliente alguno en las estaciones de trabajo debido a que todo el escaneo lo realiza mediante el uso de WMI (Windows Management Instrumentation), archivos compartidos y accesos al registro remoto.[11]

Principales funciones y características:

- ✓ No existen restricciones en cuanto al número de clientes.
- ✓ Interfaz web muy intuitiva.
- ✓ Ejecuta comandos y programas a través del interfaz web de Lansweeper, lo que resulta ideal para atender a los clientes o usuarios de la red.

Esta herramienta tiene requisitos previos de software, como por ejemplo, para instalar el "servidor" Lansweeper es necesario Windows 2003 o superior (Windows XP, Vista o Windows 7) y SQL Server o SQL Server Express.[12]

También presenta características negativas como el hecho de que es privativa, su agente servidor solo puede ser instalado en Windows y no usa agente cliente para realizar el escaneo de los ordenadores.

#### 1.3.1.6 Network Inventory Advisor

Network Inventory Advisor es una aplicación que recopila datos de inventario de red, creando informes personalizados posteriormente. Es una herramienta libre de agente que fue desarrollada para realizar el inventario de software, hardware y otros activos de tecnología informática en una red.

---

[11] programs-gratis.net. *programs-gratis.net.* [Online] http://lansweeper.programas-gratis.net/.

[12] programas.com. *programas.com.* [Online] http://www.programas.com/descargar_/lansweeper.

Brinda la posibilidad de agrupar software inventariados por título, efectuar un seguimiento de sus versiones, especificar el estado de uso de estos, fecha de instalación, ordenarlos y filtrarlos por varios parámetros y generar informes flexibles.

La instalación de este solo es posible en un ordenador Windows, sin embargo de forma remota es capaz de escanear también ordenadores de Linux. Hace posible programar escaneos de red y recibir informes de inventario por correo electrónico o cargados en el servidor, o simplemente exportados a la ubicación de red especificada.

Network Inventory Advisor puede realizar detección de software de seguridad fiable, incluso con Windows 2003 o 2008 detecta exactamente donde hay faltas de protección.[13]

Se puede decir que esta herramienta posee funcionalidades ventajosas para realizar un inventario de software y hardware en los ordenadores de la red, sin embargo presenta propiedades que inducen a su descarte, como el ser una herramienta privativa, de poco interacción con el usuario y que solo es posible su instalación en ordenadores con sistema Windows.

### 1.3.2 Software Libre de Inventario de Hardware y Software.

#### 1.3.2.1 OCS Inventory NG

OCS Inventory es una aplicación en software libre, la cual basada en un modelo cliente-servidor recopila información del software y hardware instalado en los equipos de la red en un sistema centralizado. Es capaz de detectar los dispositivos activos en la red (switches, routers, impresoras en red) y cualquier dispositivo desatendido, almacenando para cada uno de ellos la dirección MAC y la dirección IP, permitiendo clasificarlos posteriormente. Cuando el servidor central se ejecuta sobre un sistema Linux también puede ser posible escanear una dirección IP o una subred para obtener información detallada de equipos no inventariados.

Funcionamiento

El agente que se está ejecutando en el equipo envía un estado actualizado del inventario al servidor OCS; el servidor procesa dicho inventario y comprueba si hay paquetes de software asignados para el

[13] softpedia.es. *softpedia.es.* [Online] Softpedia. http://www.softpedia.es/programa-PC-Inventory-Advisor-96829.html.

despliegue en el equipo que ha enviado el reporte; si existen paquetes asignados para este equipo, el servidor de comunicación y/o despliegue lo envía al agente mediante protocolo SSL; el agente descarga el paquete y lo ejecuta. Una vez ejecutado, devuelve un código notificando el estado de la instalación.

Toda la información que extrae OCS se realiza mediante la instalación de un agente, en cada uno de los equipos a gestionar (ya sean Windows o Linux), que envía la información al servidor mediante HTTP e intercambiando archivos XML. El entorno requerido no es nada complejo: Apache + Perl + PHP + MySQL, por lo que es posible instalar el servicio sin mucha dificultad.[14]

OCS Inventory, a pesar de ser un software libre, multiplataforma y que utiliza agentes clientes para Windows y Linux, carece de sistemas de alertas e incidencias caracterizándola como una aplicación pasiva. Esta herramienta, aunque posibilita incluir nuevos paquetes e instalarlos desde el servidor en los ordenadores clientes, no permite definir nuevos componentes que complementen o modifiquen el paquete de instalación de sus aplicaciones clientes. Además, aunque permite la ejecución remota desde el servidor de determinados scripts fundamentalmente en Visual Basic, no posibilita la instalación de nuevos plugins que se integren al núcleo de la aplicación cliente, por lo que no es lo suficientemente extensible para incluir nuevas funcionalidades a través de componentes.

#### 1.3.2.2 GLPI

GLPI es un software libre basado en Web escrito en PHP, el cual permite la administración y registro de los inventarios del hardware y el software de una red informática. Su sección de Inventario puede listar y consultar el inventario de hardware y software del parque informático, haciendo posible realizar búsquedas parametrizadas y ordenadas, sobre alguno de los tipos de inventarios (computadores, material de red, impresoras, monitores, periféricos externos, software), además, permite desplegar la ficha que contiene los datos particulares de cada elemento inventariado, o agregar un elemento.[15]

- ✓ Realiza el inventario preciso de todos los recursos informáticos, y el software existente, cuyas características son almacenadas en bases de datos.
- ✓ Es software libre y se puede distribuir y/o modificar bajo los términos de la licencia GNU/GPL versión 2.

---

[14] bitelia.com. *bitelia.com.* [Online] http://bitelia.com/2010/09/automatiza-la-gestion-de-tus-equipos-con-ocs.

[15] pide.wordpress.com. *pide.wordpress.com.* [Online] http://pide.wordpress.com/2010/11/01/helpdesk-glpi/.

En resumen GLPI es una herramienta libre y multiplataforma, muestra las incidencias de los últimos eventos ocurridos y permite la gestión de notificaciones, sin embargo estas notificaciones no son parte del proceso automatizado de la aplicación, sino notificaciones realizadas por los usuarios de la red al servidor.

#### 1.3.2.3 CACIC

CACIC (Configurador Automático y Colector de Información Computacional) es una herramienta que proporciona un diagnóstico del parque informático, brindando diversidad de información como el número de equipos, propiedad y ubicación de estos, el tipo de software utilizado y configuraciones de hardware. Es un sistema distribuido donde la comunicación se rige por los protocolos HTTP y FTP, y sus módulos se distribuyen en las plataformas y el uso de diferentes tecnologías.

Esta herramienta sostiene su funcionamiento sobre la base de módulos, como son:

Agente: Se encarga de recoger los datos de hardware y software con una frecuencia establecida por el módulo el gerente.

Gerente: Recibe los datos recogidos de los agentes, los organiza y proporciona informes y consultas, a través de la interfaz web existente. Además, permite configurar algunas características de comportamiento de los módulos agentes.

Entre sus funcionalidades que interesan para este proyecto se encuentran:

- ✓ Recoge información sobre los componentes de hardware instalados en cada equipo.
- ✓ Envía alerta a los administradores de sistemas cuando se identifican cambios en la configuración de los componentes de hardware de cada uno de los ordenadores.
- ✓ Recoge información sobre los distintos programas informáticos instalados en cada ordenador.
- ✓ Envía alerta a los administradores de sistemas cuando se identifica cambios en la ubicación física de un equipo.

Esta herramienta de inventario de hardware y software en una red de ordenadores es libre y multiplataforma cuya interacción con los clientes es baja y sin gestión de incidencias, limitándose a la creación de reportes globales predefinidos por los desarrolladores durante su implementación. No posibilita la integración con nuevos componentes, salvo con la tecnología Intel vPro (INTEL AMT/VPRO – PLUGIN: tecnología desarrollada y disponible por Intel para la administración remota de

estaciones de trabajos corporativas que contemplan esta tecnología integrada), o sea no permite la integración con nuevos plugins o componentes.

Mediante el estudio de estas herramientas que realizan inventario de software actualmente en el mundo, es posible determinar que funcionalidades puede necesitar realmente la aplicación a crear. Por ejemplo, algunas de estas aplicaciones (NetSupport, Total Network Inventory, LOGINventory) presentan la limitación de no ser multiplataforma, otras como Total Network Inventory al no presentar aplicaciones clientes tienden a sobrecargar el tráfico de red debido al proceso de consulta que realiza sobre los ordenadores de la red, además de no tener forma de obtener el inventario de una máquina cliente en caso de que esta tuviera el firewall activado; como también puede suceder que los agentes clientes solo pueden ser instalados en un solo tipo de plataforma. Vale señalar que la necesidad de una herramienta que no sea costosa, notifique los eventos y envíe alertas se forma como resultado del análisis y evaluación de las características de las herramientas de inventario de hardware y software antes mencionadas, es esencial dejar atrás esas herramientas pasivas que no interactúan con el usuario limitadas solamente a la creación de reportes.

## 1.4 Metodologías de desarrollo.

El éxito al desarrollar un proyecto de software se mide evaluando tres variables fundamentales: costo, tiempo y calidad. Para que un proyecto sea exitoso se debe entregar en tiempo, bajo el presupuesto asignado y con la calidad requerida. Para lograr este éxito, los ingenieros de software emplean modelos, procesos y herramientas, creando así nuevas soluciones, empleando el término metodología.

Existen las metodologías que tienen mayor énfasis en la planificación y control del proyecto, en una especificación precisa de los requisitos y el modelado, estas reciben el apelativo de metodologías tradicionales o pesadas. Las metodologías tradicionales se centran principalmente en el proceso de desarrollo del software, con el fin de obtener un software más eficiente. Primeramente se hace énfasis en planificar bien todo el trabajo a realizar y una vez que está todo detallado, comienza entonces el ciclo de desarrollo del producto. Quedan bien definidos los roles, actividades, artefactos, herramientas, notaciones para el modelado y documentación detallada, para de esta forma tener un

buen control del proceso. Estas metodologías no se adaptan fácilmente a los cambios, por lo que no es adecuado usarlas cuando los requisitos no pueden predecirse o pueden sufrir variaciones.

También existen metodologías que tienen como filosofía centrarse en el factor humano o en el producto de software, son llamadas ágiles. Estas dan más importancia al individuo, a la colaboración con el cliente y al desarrollo incremental del software con iteraciones muy cortas. Son de gran efectividad en proyectos donde tienen requisitos muy cambiantes y además exigen reducir el tiempo de desarrollo, manteniendo una alta calidad.

Principios de un proceso ágil. [16]

- ✓ La prioridad es satisfacer al cliente mediante tempranas y continuas entregas de software que le aporte un valor.
- ✓ Dar la bienvenida a los cambios. Se capturan los cambios para que el cliente tenga una ventaja competitiva.
- ✓ Los implicados en el negocio y los desarrolladores deben trabajar juntos a lo largo del proyecto.
- ✓ El diálogo cara a cara es el método más eficiente y efectivo para comunicar información dentro de un equipo de desarrollo.

### 1.4.1 Metodología RUP (Proceso Unificado Racional)

En la ingeniería de software RUP es un proceso destinado a la asignación de tareas y responsabilidades en una organización de desarrollo, su objetivo principal es realizar un software de calidad que satisfaga las necesidades del cliente, se enfoca en los casos de uso, manejo de riesgos y un buen manejo de la arquitectura[17]. Además, vale señalar que se encuentra entre las metodologías tradicionales más utilizadas a nivel mundial, y constituye el estándar que más se utiliza para el

---

[16] **José H. Canós, Patricio Letelier, Carmen Penadés.** willydev.net. *willydev.net.* [Online] http://www.willydev.net/descargas/prev/TodoAgil.Pdf.

[17] google.com. *google.com.* [Online] http://www.google.com.cu/url?sa=t&rct=j&q=metodologia+RUP&source=web&cd=9&ved=0CGcQFjAI&url=http%3A%2F%2Fingenieriasoftwaredos.wikispaces.com%2Ffile%2Fview%2FMETODOLOGIA%2BRUP%2Btrabajo1.doc&ei=pq7ZTrnENKHc0QHQ2c3CDQ&usg=AFQjCNE3yTsnu2nXShzyI20PXebC1I3H4g.

análisis, implementación y documentación de sistemas orientados a objetos. Presenta características de gran importancia, como son:

Es un proceso dirigido por casos de uso: Con los casos de uso se puede mostrar que quieren y necesitan los usuarios, esto es recogido en la modelación del negocio y posteriormente es plasmado en los requisitos.

Es un proceso centrado en la arquitectura: La arquitectura permite obtener una visión completa del sistema entre los involucrados, para así llevar a cabo el control del desarrollo.

Es un proceso iterativo e incremental: Se propone dividir el trabajo en partes más pequeñas (mini proyectos), logrando un equilibrio entre casos de uso y arquitectura durante cada mini proyecto, en todo este proceso de desarrollo, logrando un crecimiento total del proyecto.

### 1.4.2 Metodología de Desarrollo seleccionada

Después de efectuar un análisis acerca de las metodologías de desarrollo más usadas a nivel mundial, se ha decidido optar por la metodología tradicional o pesada para la realización del software, a pesar del auge que tienen las metodologías ágiles en la actualidad, se considera que las características de las metodologías pesadas se ajustan más, por lo que se exponen algunos criterios que se tuvieron en cuenta para dicha elección[18][19]:

- ✓ Existe un contrato prefijado con la entidad para la cual se está realizando la aplicación.
- ✓ El cliente interactúa con el equipo de desarrollo mediante reuniones, en las cuales los desarrolladores pueden conocer lo que espera el cliente al terminar el producto, o sea, el cliente no forma parte del equipo de desarrollo como se precisa en las metodologías ágiles.
- ✓ RUP genera gran cantidad de artefactos y documentación los cuales servirán de referencias para continuar con el proyecto GRHS debido a que la mayoría de sus integrantes son estudiantes que se encuentran culminando la carrera.

---

[18] **Joskowicz, Jose.** iie.fing.edu.uy. *iie.fing.edu.uy.* [Online] Febrero 10, 2008. http://iie.fing.edu.uy/~josej/docs/XP%20-%20Jose%20Joskowicz.pdf.

[19] rational.com.ar. *rational.com.ar.* [Online] http://www.rational.com.ar/cmmi/cmmiesp.html.

## 1.5 Lenguaje de Modelado (UML)

En la rama de la ingeniería es de gran utilidad representar los diseños de forma gráfica. Desde que surgió la informática se han puesto en práctica distintas formas de hacer representaciones gráficas ya sea una forma más bien personal o con algún modelo gráfico. Al no existir una estandarización general, impedía que los diseños gráficos pudieran ser compartidos por los demás diseñadores. Era de gran necesidad un lenguaje para que existiera una uniformidad entre los desarrolladores y poder trasmitir sus ideas, además serviría a la hora de realizar el proceso de análisis de determinado problema. Este fue uno de los principales objetivos de la creación del Lenguaje Unificado de Modelado (UML: Unified Modeling Language).

UML facilita un vocabulario y reglas que permiten realizar una comunicación, cuyo objetivo principal es representar gráficamente un sistema. También puede aplicarse en el desarrollo de software, pues se entrega una diversidad de formas para dar soporte a una metodología de software, aunque no especifica la metodología a usar.

Se puede decir que UML es un modelado visual, pues este hace una abstracción de las partes esenciales del sistema y las plasma en una notación gráfica. Este tipo de modelado permite manejar la complejidad de los sistemas a analizar o diseñar[20].

Un Diagrama es la representación gráfica de un conjunto de elementos con sus relaciones. Con un diagrama se puede representar una vista del sistema a modelar, para la representación correcta del sistema desde varias perspectivas UML brinda una gran variedad de diagramas.

## 1.6 BPMN (Notación para el Modelado de Procesos de Negocio).

BPMN, es una notación gráfica estandarizada para el modelado de procesos de negocio. Esta notación gráfica inicialmente desarrollada por Business Process Management Initiative es actualmente mantenida por el Object Management Group después de la fusión de las dos organizaciones en el año 2005. Su versión actual, a abril de 2011, es la 2.0.

BPMN provee una notación estándar fácilmente legible y entendible para todos los involucrados e interesados del negocio, ejemplos de estos son: los analistas de negocio (quienes definen y redefinen

---

[20] **Orallo, Enrique Hernández.** disca.upv.es. *disca.upv.es.* [Online] http://www.disca.upv.es/enheror/pdf/ActaUML.PDF.

los procesos), los desarrolladores técnicos (responsables de implementar los procesos) y los gerentes y administradores del negocio (quienes monitorizan y gestionan los procesos). En resumen, BPMN sirve como lenguaje común para cerrar la brecha de comunicación que frecuentemente se presenta entre el diseño de los procesos de negocio y su implementación. [21]

## 1.7 Visual Paradigm

Visual Paradigm para UML es una herramienta profesional, la cual soporta el ciclo de vida completo del desarrollo de un software, con ciclo de vida completo se hace referencia a análisis y diseño orientados a objetos, construcción, pruebas y despliegue del software. Esta herramienta posibilita realizar una construcción más rápida de aplicaciones de calidad, mejores y a un menor coste. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. La herramienta UML CASE, además, proporciona abundantes tutoriales de UML, demostraciones interactivas de UML y proyectos UML.

Las características gráficas que presenta Visual Paradigm son realmente muy cómodas, facilitan la realización de los diagramas de modelado que sigue el estándar de UML que son: Diagramas de clase, Casos de Uso, Comunicación, Secuencia, Estado, Actividad y Componentes.

## 1.8 Lenguaje de programación Python

Python es un lenguaje de programación de alto nivel, su trabajo está dirigido al mantenimiento de una sintaxis muy limpia y un código legible, por lo que su simplicidad, versatilidad y rapidez de desarrollo le ha dado gran popularidad recientemente. Sin embargo, estas no fueron las únicas razones de su selección para este proceso de desarrollo, este lenguaje además presenta otras características:

- ✓ Su intérprete es multiparadigma, por lo que brinda así al programador la facilidad de no tener que usar un estilo en particular.

[21] www.bizagi.com. *www.bizagi.com.* [Online] Rickmansworth Hertfordshire WD3 IRE. http://www.bizagi.com/docs/BPMNbyExampleSPA.pdf.

- ✓ Es fuertemente tipado, usa tipado dinámico y además es un lenguaje interpretado por lo que no necesita compilar el código fuente para ser ejecutado.
- ✓ Es posible usarlo de muchas maneras en el desarrollo de software, es capaz de realizar desde aplicaciones de servidores de red hasta páginas web.
- ✓ Es multiplataforma.
- ✓ Es distribuido bajo la licencia Open Source OSI, por lo que es libre.
- ✓ Contiene diversas funciones en el propio lenguaje, para el tratamiento de cadenas, números, archivos, entre otras.
- ✓ También existen bibliotecas que pueden ser importadas para el tratamiento de ventanas, sistemas en red. [22]

## 1.9 Eclipse

La plataforma Eclipse constituye un Entorno de Desarrollo Integrado (IDE, Integrated Development Environment) abierto y extensible. Un IDE es un programa compuesto por un conjunto de herramientas útiles para un desarrollador de software. Como elementos básicos, un IDE cuenta con en un editor de código, un compilador/intérprete y un depurador. Eclipse sirve como IDE Java y cuenta con numerosas herramientas de desarrollo de software. También da soporte a otros lenguajes de programación, como son C/C++, PHP o Python. La arquitectura de plugins de Eclipse permite, además de integrar diversos lenguajes sobre el mismo IDE, introducir otras aplicaciones accesorias que pueden resultar útiles durante el proceso de desarrollo como: herramientas UML, editores visuales de interfaces, ayuda en línea para librerías, etc. [23][24]

Otras de las características importantes de este IDE para su selección, son:

- ✓ La instalación de sus plugins posibilita tener un entorno de desarrollo ampliado.

---

[22] **Alvarez, Miguel Angel.** desarrolloweb.com. *desarrolloweb.com.* [Online] http://www.desarrolloweb.com/articulos/1325.php.

[23] **Chabarría, Raúl Eduardo.** slideshare.net. *slideshare.net.* [Online] http://www.slideshare.net/Benedeti/ide-eclipse-breve-gua-201399.

[24] **Joaquín Seoane Pascual, Jesús M. González Barahona y Gregorio Robles.** http://www.atenas.cult.cu. *http://www.atenas.cult.cu.* [Online] Free Software Foundation, 2007. http://www.atenas.cult.cu/rl/informatica/manuales/sl/introduccion_al_SL/book1.html.

- ✓ Posee un asistente de etiquetas. Según se va escribiendo código aparecen las etiquetas que puedes incluir, facilitándote así la programación.

En conclusión, Eclipse es muy buen IDE pues su multitud de plugins lo adaptan a las necesidades del proyecto en desarrollo y con una instalación de los complementos realmente sencilla, así como de actualización de componentes ya instalados.

### 1.10 PyUtilib: Arquitectura basada en Componentes

El PCA (PyUtilib Component Arquitectura) se deriva de la arquitectura basada en componentes de TRAC y está incluido en el software PyUtilib. Su desarrollo fue motivado a raíz de la experiencia en una variedad de aplicaciones informáticas y la necesidad de un marco independiente. Esta arquitectura contiene en su núcleo determinadas clases e interfaces que definen nuevos conceptos presentes en su interacción, tales como: interfaces, componentes Singleton y no Singleton, ambientes y otras definiciones, pero fundamentalmente permite la integración entre los diferentes componentes.

Un plugin es una clase que implementa un conjunto de métodos relacionados en el contexto de una aplicación. Por lo tanto, un plugin puede ser descrito como un componente de definición.

Un servicio es una instancia de una clase de plugin, por lo que los servicios son instancias de componentes.

Una interfaz es una de clase que es implementada por un plugin para registrar sus capacidades.

Un punto de extensión en esencia hace referencia a la clase ExtensionPoint del núcleo de PyUtilib, utilizada por una clase determinada para conectarse a una interface específica y ejecutar funciones de los plugin subscritos a esta u obtener información.

De este modo se puede definir una arquitectura flexible y modular, que permite extender una aplicación de forma dinámica, pues todos los plugins definidos se registran automáticamente, y dinámicamente proveen puntos de extensión para definir los servicios correspondientes en tiempo de ejecución. Lo que garantiza que un desarrollador podrá definir puntos de extensión de una interfaz determinada sin conocer la implementación de los plugins que la implementan y desarrollar nuevos plugin sin necesidad de conocer cómo y dónde se emplean dentro de la aplicación. Por lo que la ventaja principal es la posibilidad de registrar componentes dinámicamente.

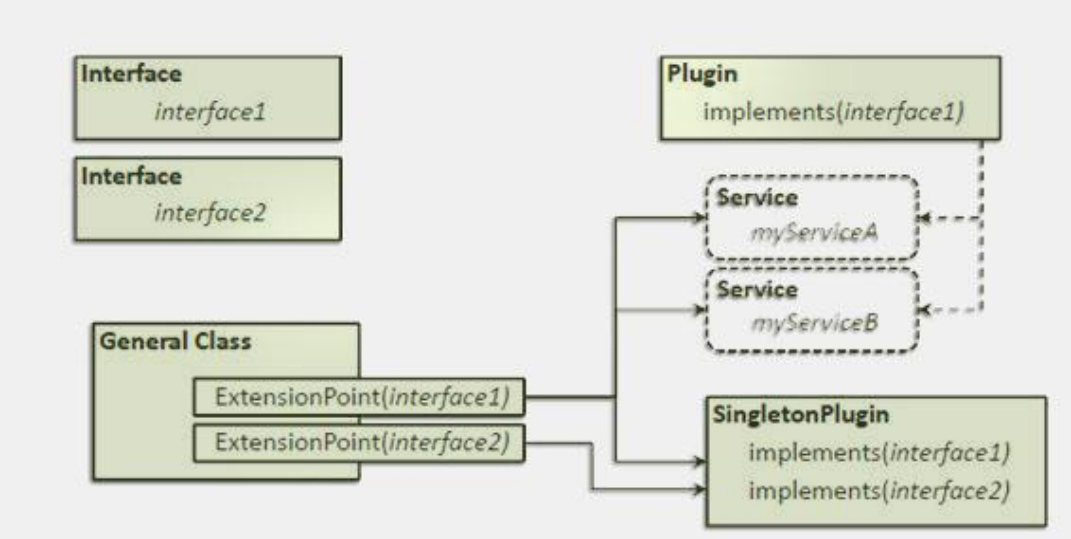


Figura 1: Arquitectura PCA.

## 1.11 Conclusiones

En este capítulo se realiza un análisis de un conjunto de conocimientos, conceptos y definiciones relacionados con el estudio del estado del arte sobre herramientas que realizan inventario de software y hardware en una red informática actualmente. Se explican herramientas, tecnologías y metodologías que fueron seleccionadas para la realización del Módulo de Gestión de Inventarios e Incidencias de Software en Debian y Ubuntu, y razones de la selección de estas.

# CAPÍTULO 2: CARACTERÍSTICAS DEL SISTEMA

## 2.1 Introducción

En este capítulo se realiza una descripción de los procesos de negocio correspondientes al proceso de inventario de software para una red computadoras con la intención de comprender su estructura. Queda conformado el modelo de negocio, se levantan los requisitos del sistema, estos incluyen funcionales y no funcionales, además de realizar una descripción de los casos de uso identificados para lograr un mejor entendimiento.

## 2.2 Problema y Situación Problemática

Necesidad de gestionar y controlar de forma automatizada los recursos de software de una red de computadoras para los sistemas operativos Debian y Ubuntu. Aunque actualmente se utilizan varias aplicaciones destinadas a la gestión de inventarios de software, estas son pasivas, sólo notifican los cambios no siendo capaces de tomar decisiones, además de no presentar monitorización para mejorar la detección de cambios ocurridos en la red. Por lo que se requiere de una aplicación capaz de inventariar todo el software y realice la monitorización.

Obtener la información del software de una computadora y la detección de incidencias, como tener instalado programas no autorizados así como los cambios ocurridos en las propiedades del sistema operativo son algunos de los procesos que el administrador tiene que realizar manualmente en los servidores donde se encuentra instalados los sistemas operativos Debían y Ubuntu, pues aun no cuentan con una aplicación automatizada que pueda realizar este proceso y poder detectar los cambios.

## 2.3 Objeto de Automatización

El proceso de inventario de software en la red de computadoras se realiza de forma manual en aquellas computadoras que su sistema operativo es Ubuntu y Debian. El administrador de la red es el encargado de realizar un informe con todos los programas instalados en la PC, recoger la información referente a los controladores de hardware, las características del sistema operativo al igual que toda

la información referente con la PC y el BIOS, de esta forma puede comprobar con su informe anterior si ha ocurrido algún cambio en las PC de sus usuarios y poder informar el mismo a sus superiores.

### 2.4 Propuesta del Sistema.

Desarrollar un módulo que sea capaz de obtener la información de todo el software instalado en una computadora en los sistemas operativos Debian y Ubuntu. Detectar las incidencias para una mejor gestión y control en una red de computadoras además de realizar la monitorización de los identificadores del ordenador (nombre, ip) y la instalación y desinstalación de paquetes.

### 2.5 Modelo de Negocio.

Para efectuar el modelado de los procesos de negocio se empleó BPMN (Notación para el Modelado de Procesos de Negocio), la cual es una notación gráfica estandarizada para el modelado de procesos de negocio. BPMN sirve como lenguaje común para cerrar la brecha de comunicación que frecuentemente se presenta entre el diseño de los procesos de negocio y su implementación. Los procesos descritos mediante esta notación se muestran en las siguientes imágenes (Figura 2, Figura 3):

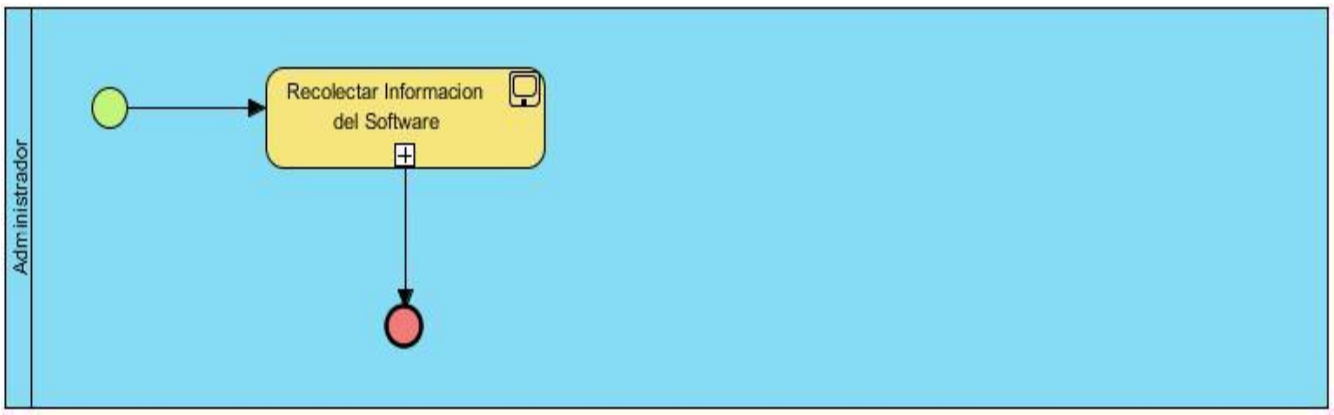


Figura 2: Proceso de Negocio Realizar Inventario de Software.

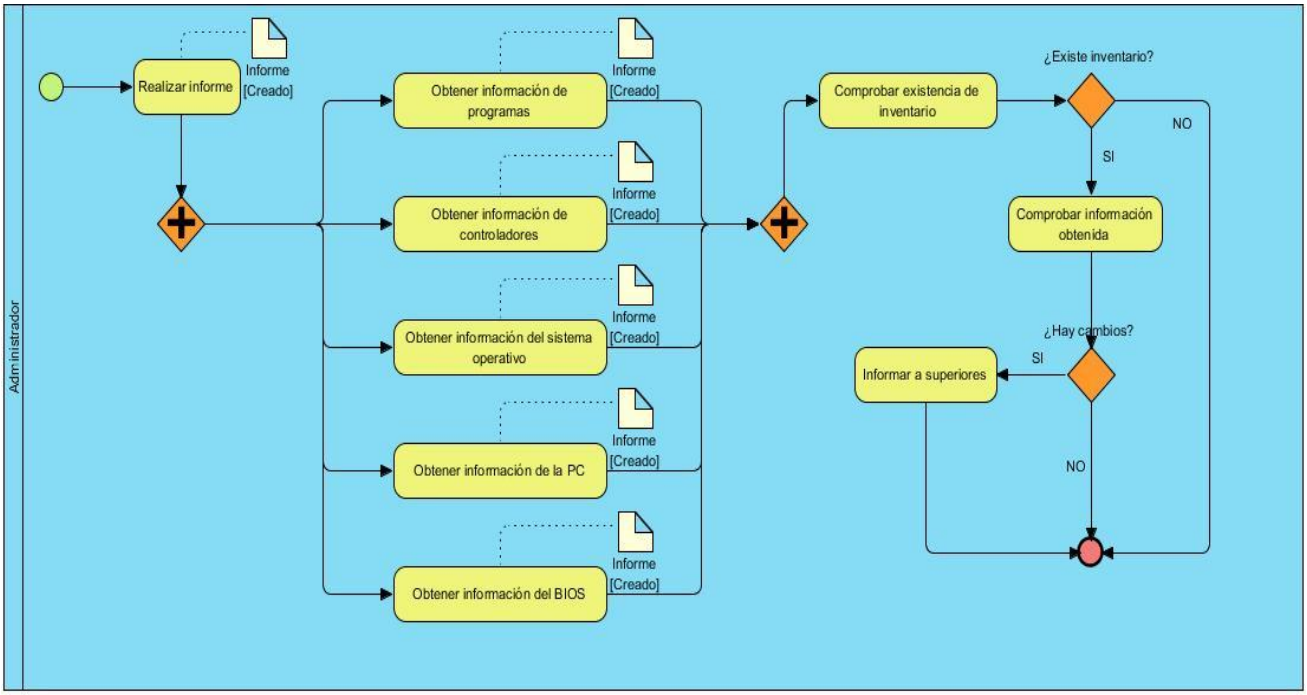


Figura 3: Proceso de Negocio Recolectar Información de Software.

### 2.5.1 Descripción de los procesos de negocio (Figura 2).

**Recolectar Información del Software**: El proceso comienza cuando el administrador de la red para realizar un inventario de software de la PC realiza el sub-proceso Recolectar Información del Software, obteniendo toda la información que necesita.

**Responsable:** Administrador.

**Entradas**: No Aplica.

**Salidas**: Informe.

### 2.5.2 Descripción de los procesos de negocio (Figura 3).

**Realizar Informe**: El administrador de la red realiza un informe el cual va llenando según obtiene la información de la PC.

**Responsable:** Administrador.

**Entradas:** No aplica.

**Salidas:** Informe.

**Obtener información de programas**: El administrador de la red recopila toda la información referente al software instalado en la PC.
**Responsable:** Administrador.
**Entradas:** No aplica.
**Salidas:** Informe.

**Obtener información de controladores**: El administrador de la red obtiene información de los drivers que tiene instalada la PC.
**Responsable:** Administrador.
**Entradas:** No aplica.
**Salidas:** Informe.

**Obtener información de Sistema Operativo**: El administrador de la red obtiene información del sistema operativo que tiene instalado la PC.
**Responsable:** Administrador.
**Entradas:** No aplica.
**Salidas:** Informe.

**Obtener información de la PC**: El administrador de la red obtiene información de la PC.
**Responsable:** Administrador.
**Entradas:** No aplica.
**Salidas:** Informe.

**Obtener información del BIOS**: El administrador de la red obtiene toda la información referente con el BIOS de la PC.
**Responsable:** Administrador.
**Entradas:** No aplica.
**Salidas:** Informe.

**Comprobar existencia de inventario:** El administrador comprueba la existencia de información registrada de algún inventario realizado anteriormente para compararla con la información obtenida de la PC.

**Responsable:** Administrador.

**Entradas:** No aplica.

**Salidas:** Informe.

**Comprobar Información Obtenida**: El administrador compara la información obtenida con la anterior obtenida y registrada del último inventario realizado, para detectar si existen cambios en los ordenadores de la red.

**Responsable:** Administrador.

**Entradas:** No aplica.

**Salidas:** Informe.

**Informar a Superiores**: Este proceso se inicia en caso de detectar cambios en la red, el administrador tiene la obligación de informar estos a sus superiores.

**Responsable:** Administrador.

**Entradas:** No aplica.

**Salidas:** Informe.

Personas que intervienen en los procesos de negocio.

| Nombre | Justificación |
|---|---|
| Administrador. | Es el encargado de obtener toda la información del software instalado en las PC de la red. |

Tabla 1: Persona que interviene en el proceso de negocio.

### 2.6 Requisitos Funcionales.

Los requisitos funcionales son declaraciones de los servicios que debe proporcionar el sistema, de la manera que este debe reaccionar a entradas particulares y de cómo se debe comportar en situaciones particulares. Estos requisitos dependen del tipo del software que se desarrolle, de los

posibles usuarios del software y del enfoque general tomado por la organización al redactar requerimientos. En algunos casos también pueden declarar explícitamente lo que el sistema debe hacer. En síntesis, los requerimientos funcionales del sistema describen con detalle la función de este, sus entradas y salidas, excepciones.

Para el diseño de este sistema se identificaron como requisitos funcionales:

RF1. Realizar inventario de software.

RF2. Realizar inventario de controladores.

RF3. Realizar inventario de sistema operativo.

RF4. Realizar inventario de computadora.

RF5. Realizar inventario de BIOS.

RF6. Monitorizar instalación y desinstalación de software.

RF7. Monitorizar cambio de IP o nombre de computadora.

RF8. Comprobar inventario.

RF9. Notificar incidencia.

RF10. Guardar inventario.

RF11. Leer inventario.

RF12. Guardar configuración de incidencia.

RF13. Leer configuración de incidencia.

RF14. Guardar períodos de cambio.

RF15. Leer períodos de cambio.

RF16. Enviar inventario.

RF17. Enviar incidencia.

RF18. Enviar traza.

RF19. Actualizar configuración de incidencia.

RF20. Actualizar Inventario

## 2.7 Requerimientos no Funcionales.

Los requerimientos no funcionales son aquellos que se refieren a las propiedades emergentes del sistema como la fiabilidad, el tiempo de respuesta y la capacidad de almacenamiento. Por lo tanto,

pueden especificar el rendimiento del sistema, la protección, la disponibilidad, usabilidad, y otras propiedades emergentes. En síntesis, los requerimientos no funcionales son propiedades o cualidades que el producto debe tener, por lo que en muchos casos son fundamentales en el éxito del producto.

Usabilidad.

Ambiente

Características del hardware del cliente:

- ✓ 256 MB de RAM
- ✓ 1 procesador de 2.0 GHz o superior
- ✓ 2 GB de espacio en el disco duro

Características de software en el cliente:

- ✓ Sistema operativo: Debian 6.0, Ubuntu 11.04
- ✓ Python 2.7

Restricciones de diseño.

- ✓ La codificación se regirá mediante el estilo de codificación TLM-GRHS-0120_55 EstandarPythonv1.0 definido por el proyecto para garantizar un mejor soporte.
- ✓ Se harán tratamientos de excepciones para todo el código incluyendo el importado de bibliotecas externas para garantizar conocer cuando no están instaladas algunas de sus dependencias.

Requisitos de Licencia.

- ✓ Se utilizan varios módulos GPL por lo que se debe cumplir las especificaciones de esta licencia.

## 2.8 Modelo de Casos de Uso del Sistema.

El Modelo de Casos de Uso del Sistema permite que los desarrolladores de software y los clientes lleguen a un acuerdo sobre los requisitos, es decir sobre las condiciones y posibilidades que debe

cumplir el sistema. Además proporciona la entrada fundamental para el análisis, diseño y las pruebas. [25]

### 2.8.1 Actores del Sistema a automatizar.

El Módulo Gestión de Inventario e Incidencias de Software en GNU/Linux presenta la intervención de un actor:

| Actores | Justificación |
|---|---|
| Sistema de Administración de Recursos y Acciones (SARA) | Es el sistema encargado de gestionar la monitorización e inventarios efectuados por la aplicación agente para la detección y control de cambios realizados en el ordenador como pueden ser modificación en sus identificadores, software, controladores, o vencimiento de las licencias del antivirus (Kaspersky). |

Tabla 2: Actores del Sistema.

[25] **Jacobson, Ivar, Booch, Grady y Rumbaugh, James. El proceso Unificado de Desarrollo de Software. 1era edición. Madrid : Pearson Educación, S.A. - Addison Wesley, 2000. 84-7829-036-2.**

### 2.8.2 Diagrama de Casos de Uso del Sistema.

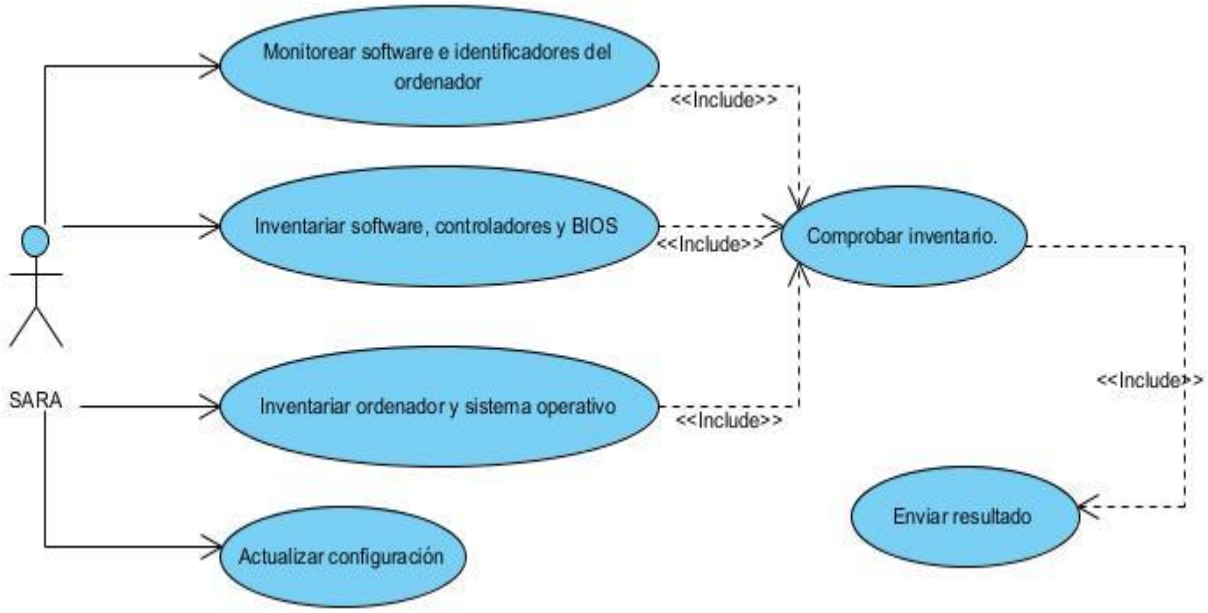


Figura 4: Diagrama de casos de uso Realizar inventario de software.

#### 2.8.2.1 Casos de Uso del Sistema.

Los casos de uso son artefactos narrativos que describen, bajo la forma de acciones y reacciones, el comportamiento del sistema desde el punto de vista del usuario. Los casos de uso son fragmentos de funcionalidad que el sistema ofrece para aportar un resultado de valor observable para sus actores [26]

#### 2.8.2.2 Casos de Uso del módulo Gestión de Inventario e Incidencias de Software en GNU/Linux.

CU 1. Monitorear software e identificadores del ordenador.

| | |
|---|---|
| Objetivo | Monitorizar la desinstalación e instalación de algún software, al igual que si el usuario cambia el nombre de la PC o el IP. |

[26] **Ibídem – pág. 129.**

| | | |
|---|---|---|
| Actores | Sara: (Inicia) Solicita la monitorización de instalación o desinstalación de algún software o cambio de los identificadores de la PC. | |
| Resumen | El caso de uso se inicia cuando SARA solicita realizar la monitorización de la PC, para controlar si existe alguna instalación o desinstalación de software o si es cambiado el nombre o IP de la PC. | |
| Complejidad | Alta. | |
| Prioridad | Crítico. | |
| Precondiciones | La PC debe estar encendida. | |
| Postcondiciones | No procede. | |
| Flujo de eventos | | |
| Flujo básico Monitorear software e identificadores. | | |
| | Actor | Sistema |
| 1. | Solicita realizar la monitorización de software, o de identificadores de la PC. | |
| 2. | | El sistema realiza la monitorización de la PC cliente para controlar si ocurren cambios de software. |
| 3. | | El sistema realiza la monitorización de la PC cliente para controlar si ocurren cambios en el IP o el nombre de la PC. |
| 4. | | El sistema detecta un cambio |

| | | |
|---|---|---|
| 5. | | El sistema comprueba el cambio, comparando con el inventario guardado en cache.<br><br>Ver caso de uso (Comprobar Inventario). |
| 6. | | Termina el caso de uso. |
| Relaciones | CU Incluidos. | • El sistema comprueba el cambio: Paso 5 del flujo Básico Monitorear software e identificadores.<br><br>El sistema comprueba el cambio en el CU Comprobar Inventario. |
| | CU Extendidos. | No procede. |
| Requisitos no funcionales | No procede. | |
| Asuntos pendientes | No procede. | |

Tabla 3: Descripción del CUS Monitorear software, identificadores del ordenador.

CU 2. Inventariar software, controladores y BIOS.

| | |
|---|---|
| Objetivo | Obtener información del software, controladores y BIOS del cliente. |
| Actores | Sara: (inicia): Solicita realizar el inventario de software, controladores y BIOS del ordenador. |

| | |
|---|---|
| Resumen | Este caso de uso consiste en realizar un inventario en la PC acerca del software (nombre, tamaño, versión, fabricante, autor y fecha de instalación), los controladores (nombre, fabricante, número de revisión y dispositivo que controla) y el BIOS (fabricante, tamaño, versión, fecha de lanzamiento, dispositivos de arranque, funciones, estándar soportados y expansión) para una posterior comparación con el último inventario realizado guardado y así encontrar cambios realizados en el ordenador. |
| Complejidad | Alta. |
| Prioridad | Crítico. |
| Precondiciones | La PC debe estar encendida. |
| Postcondiciones | No procede. |
| Flujo de eventos | |
| Flujo básico Inventario de software, controladores y BIOS. | |

| | Actor | Sistema |
|---|---|---|
| 1. | Solicita los datos del software, controlador y BIOS del ordenador. | |
| 2. | | Pide al servidor el último inventario de BIOS realizado. |
| 3. | | Actualiza el inventario de BIOS en la cache. |
| 4. | | Recopila los datos del software en el cliente. |

| | | |
|---|---|---|
| 5. | | Recopila los datos de los controladores del cliente. |
| 6. | | Recopila los datos del BIOS del cliente. |
| 7. | | Comprueba que haya cambios, comparando con el inventario guardado en cache.<br><br>Ver caso de uso (Comprobar Inventario). |
| 8. | | Termina caso de uso. |
| Relaciones | CU Incluidos. | Comprueba que haya cambios: Paso 5 del flujo Básico Inventario de software, controladores y BIOS.<br><br>Comprueba que haya cambios en el CU Comprobar Inventario. |
| | CU Extendidos. | No procede. |
| Requisitos no funcionales | No procede. | |
| Asuntos pendientes | No procede. | |

Tabla 4: Descripción del CUS Inventariar software, controladores y BIOS.

CU 3. Inventariar ordenador y sistema operativo.

| | | |
|---|---|---|
| Objetivo | Obtener información de la PC cliente y del sistema operativo que tiene instalado la misma. | |
| Actores | Sara: (inicia) Solicita realizar el inventario de PC y SO. | |
| Resumen | Este caso de uso consiste en realizar un inventario de los datos de la PC (nombre, dominio, dirección ip, puerta de enlace, máscara, usuarios locales y usuarios conectados) y el sistema operativo (uuid, versión y nombre) que tiene instalado la misma para una posterior comparación con el último inventario realizado guardado y así encontrar cambios realizados en el ordenador. | |
| Complejidad | Alta. | |
| Prioridad | Crítico. | |
| Precondiciones | La PC debe estar encendida. | |
| Postcondiciones | No procede. | |
| Flujo de eventos | | |
| Flujo básico Inventario de PC y Sistema Operativo. | | |
| | Actor | Sistema |
| | Solicita los datos de la PC y el sistema operativo. | |
| | | Recopila los datos de la PC en el cliente. |
| | | Recopila los datos del sistema operativo que tiene instalado el cliente |

| | | |
|---|---|---|
| | | (Debían-Ubuntu) en el cliente. |
| | | Comprueba que haya cambios, comparando con el inventario guardado en cache.<br><br>Ver caso de uso (Comprobar Inventario). |
| | | Termina caso de uso. |
| Relaciones | CU Incluidos. | Comprueba que haya cambios: Paso 4 del flujo Básico Inventario de PC y sistema operativo.<br><br>Comprueba que haya cambios en el Comprobar Inventario. |
| | CU Extendidos. | No procede. |
| Requisitos no funcionales | No procede. | |
| Asuntos pendientes | No procede. | |

Tabla 5: Descripción del CUS Inventariar ordenador y Sistema Operativo**.**

CU 4. Comprobar inventario.

| | |
|---|---|
| Objetivo | El actor interactúa con este caso de uso con el objetivo de comprobar la información obtenida acerca del software, controladores, SO o información de la computadora con la última |

| | |
|---|---|
| | información obtenida guardada. |
| Actores | SARA: (Inicia) Obtiene información del software, controladores, SO o información de la computadora. |
| Resumen | Este caso de uso consiste en la comparación del inventario tomado y el último inventario realizado y guardado en la cache, para así detectar cambios en software, controladores, SO, o información de la computadora, así como la verificación con las configuraciones de incidencias y períodos de cambio. |
| Complejidad | Alta. |
| Prioridad | Crítico. |
| Precondiciones | No procede. |
| Postcondiciones | No procede. |

| Flujo de eventos | | |
|---|---|---|
| Flujo básico Comprobar información obtenida. | | |
| | Actor | Sistema |
| 1. | | Lee el último inventario realizado guardado. |
| 2. | | Compara la información obtenida con el último inventario realizado obteniendo algunos cambios. |
| 3. | | Lee la última configuración de incidencias guardada. |

| | | |
|---|---|---|
| 4. | | Lee la última configuración de periodos de cambios guardados. |
| 5. | | Comprueba que el cambio detectado constituye en sí una de las incidencias definidas. |
| 6. | | Comprueba que dicha incidencia no se efectuó durante algún periodo de cambio. |
| 7. | | Notifica la incidencia al Módulo de Alarmas y Acciones ante Incidencias (MAAI). |
| 8. | | Envía al servidor el resultado obtenido.<br><br>Ver caso de uso (Enviar Resultado). |
| 9. | | Guarda el inventario realizado en el ordenador. |
| 10. | | Termina el caso de uso. |
| Flujos alternos | | |
| 1a .No existe inventario en cache. | | |
| | Actor | Sistema |
| 1a.1 | | Guarda el inventario por primera vez. |

| | | |
|---|---|---|
| 1a.2 | | Envía al servidor el mismo. Ver caso de Uso Enviar Resultado. |
| 1a.3 | | Termina el caso de uso. |
| Flujos alternos | | |
| 2a. No se encontró cambios al comprobar la información obtenida | | |
| | Actor | Sistema |
| 2a.1 | | Envía al servidor el resultado obtenido.<br><br>Ver Caso de Uso Enviar Resultado |
| 2a.2 | | Guarda el inventario realizado en el ordenador. |
| 2a.3 | | Termina el caso de uso. |
| Flujos alternos | | |
| 3a. El cambio detectado no constituye una de las incidencias definidas. | | |
| | Actor | Sistema |
| 3a.1 | | Envía al servidor el cambio detectado. Ver caso de uso Enviar Resultado. |
| 3a.2 | | Guarda el inventario realizado en el ordenador. |
| 3a.3 | | Termina el caso de uso. |

| Flujos alternos | | |
|---|---|---|
| 4a. La incidencia se efectuó durante un periodo de cambios | | |
| | Actor | Sistema |
| 4a.1 | | Envía al servidor los cambios detectados. Ver Caso de Uso Enviar Resultado. |
| 4a.2 | | Guarda el inventario realizado en el ordenador. |
| 4a.3 | | Termina el caso de uso. |

| Relaciones | CU Incluidos. | ✓ Envía al servidor el mismo: Paso 1a2 del Flujo alterno No existe inventario en cache. Envía al servidor el mismo en el CU Enviar Resultado.<br>✓ Envía al servidor el resultado obtenido: Paso 2 a1 del Flujo alterno No se encontró cambios al comprobar la información obtenida**.** Envía al servidor el resultado obtenido en el CU Enviar Resultado.<br>✓ Envía al servidor el inventario obtenido en Paso 3 a1 del Flujo alterno El cambio detectado no constituye una de las incidencias definidas. Envía al servidor el inventario obtenido en el CU Enviar Resultado. |
|---|---|---|

| | | |
|---|---|---|
| | | ✓ Envía al servidor el inventario obtenido: en el Paso 4 a1 Flujo alterno La incidencia se efectuó durante un periodo de cambio. Envía al servidor el inventario obtenido en el CU Enviar Resultado.<br>✓ Envía al servidor el resultado obtenido: en el Paso 8 del Flujo básico Comprobar información obtenida. Envía al servidor el resultado obtenido en el CU Enviar Resultado. |
| | CU Extendidos. | No procede. |
| Requisitos no funcionales | No procede. | |
| Asuntos pendientes | No procede. | |

Tabla 6: Descripción del CUS Comprobar inventario.

CU 5. Enviar resultado.

| | |
|---|---|
| Objetivo | El actor interactúa con este caso de uso con el objetivo de enviar al servidor el resultado final obtenido del inventario realizado. |
| Actores | SARA: (Inicia) Solicita enviar resultado final obtenido del inventario realizado. |
| Resumen | Este caso de uso consiste en enviar al servidor el resultado obtenido sobre la comprobación del inventario realizado y así |

| | |
|---|---|
| | informar acerca del estado actual del software del ordenador e incidencias ocurridas en dicha PC. Este resultado enviado podría estar conformado por incidencias detectadas, trazas o el mismo inventario realizado. |
| Complejidad | Alta. |
| Prioridad | Crítico. |
| Precondiciones | Se obtuvo información referente a software, controladores, SO o información de la computadora. |
| Postcondiciones | No procede. |

Flujo de eventos

Flujo básico Enviar Resultado.

| | Actor | Sistema |
|---|---|---|
| 1. | | Registra la fecha y hora de del resultado a enviar |
| 2. | | Permite enviar los diferentes resultados obtenidos del inventario.<br>✓ Enviar incidencias detectadas mediante el inventario. Ver sección 1: “Enviar Incidencias”.<br>✓ Enviar inventario realizado. Ver sección 2: “Enviar Inventario”. |

| | | |
|---|---|---|
| | | ✓ Enviar trazas obtenidas. Ver sección 3: “Enviar Traza”. |
| 3. | | Termina el caso de uso. |
| Sección 1: “Enviar Incidencias”. | | |
| Flujo básico Enviar Incidencias. | | |
| | Actor | Sistema |
| 1. | | Envía las incidencias detectadas. Donde cada incidencia está compuesta por:<br>• Nivel.<br>• Tipo de Incidencia.<br>• Componente.<br>• Descripción. |
| Sección 2: “Enviar Inventario”. | | |
| Flujo básico Enviar Inventario. | | |
| | Actor | Sistema |
| 1. | | Envía el inventario. |
| Sección 2: “Enviar Traza”. | | |
| Flujo básico Enviar Traza. | | |

| | Actor | Sistema |
|---|---|---|
| 1. | | Envía la traza. Donde la traza está compuesta por:<br>• Usuarios conectados.<br>• Fecha actual y hora.<br>• Tipo de sistema operativo “Linux“.<br>• Tipo “software“.<br>• Identificador del agente.<br>• Componente. |

| | | |
|---|---|---|
| Relaciones | CU Incluidos. | No procede. |
| | CU Extendidos. | No procede. |
| Requisitos no funcionales | No procede. | |
| Asuntos pendientes | No procede. | |

Tabla 7: Descripción del CUS Enviar resultado.

CU 6. Actualizar configuración.

| | |
|---|---|
| Objetivo | El actor interactúa con este caso de uso con el objetivo de mantener actualizado las configuraciones de incidencias, y períodos de cambio. |

| | |
|---|---|
| Actores | SARA: (Inicia) Solicita guardar nuevas configuraciones de incidencias o de periodos de cambio recibidas. |
| Resumen | Este caso de uso consiste en guardar en la caché nueva configuración de incidencias o nueva configuración de periodos de cambios, recibida del servidor. |
| Complejidad | Alta. |
| Prioridad | Crítico. |
| Precondiciones | No procede. |
| Postcondiciones | No procede. |

| Flujo de eventos | | |
|---|---|---|
| Flujo básico Guardar nuevas configuraciones recibidas. | | |
| | Actor | Sistema |
| 1 | Solicita guardar en la caché nueva configuración de incidencias y periodos de cambio definida por el servidor. | |
| 2 | | Obtiene la nueva configuración de incidencias del servidor. |
| 3 | | Obtiene la nueva configuración de periodos de cambio del servidor. |
| 4 | | Actualiza las configuraciones de incidencias en cache. |

| | | |
|---|---|---|
| 5 | | Actualiza las configuraciones de periodos de cambio en cache. |
| 6 | | Guarda las actualizaciones de incidencia en cache. |
| 7 | | Guarda las actualizaciones de períodos de cambio en cache. |
| 8 | | Termina el caso de uso. |
| | CU Incluidos. | No procede. |
| | CU Extendidos. | No procede. |
| | No procede. | |
| | No procede. | |

Tabla 8: Descripción del CUS Actualizar Configuración.

## 2.9 Conclusiones

En este capítulo se realizó el análisis y descripción acerca de los procesos de negocio correspondientes al proceso de inventario de software para una red de computadores, posibilitando así, una mejor comprensión de su estructura. Se efectuó el levantamiento de requisitos del sistema, los que incluyeron tanto los requisitos funcionales como los no funcionales. También se logró la identificación de los casos de uso y una posterior descripción de estos para un mejor entendimiento por parte de los usuarios.

# CAPÍTULO 3: DISEÑO DEL SISTEMA

## 3.1 Introducción

En el presente capitulo se realiza el diseño de clases del sistema, se define las relaciones entre las mismas agrupándolas en paquetes, queda definida la arquitectura del módulo y patrones del diseño que serán empleados en la construcción del mismo. Con el diseño se crea un punto de partida para la implementación.

## 3.2 Patrones Arquitectónicos

El diseño arquitectónico representa la estructura de datos y los componentes del programa necesarios para construir un sistema computacional. Asume el estilo arquitectónico que adoptará el sistema, la estructura y las propiedades de los componentes que constituyen el sistema y las relaciones entre todos los componentes arquitectónicos de un sistema. [27]

### 3.2.1 Arquitectura de MGIISL

El Módulo de Gestión de Inventarios e Incidencias de Software en GNU/Linux forma parte de un sistema el cual presenta una arquitectura basada en gradas, cada uno de sus módulos definirán su propia arquitectura. Para la realización de este módulo en específico se selecciona de los Estilos Arquitectónicos el Estilo de Llamada y Retorno y dentro de estos la Arquitectura en Capas y la Arquitectura basada en Componentes. Estos patrones arquitectónicos fueron seleccionados debido a que el objetivo principal de este módulo en específico es realizar un plugin que realice inventario en una PC cliente y plugin para controlar lo que se está realizando en la PC mediante la monitorización los cuales se integrarán en la aplicación general y a su vez cada uno de estos estará estructurado en diferentes capas. Se mezclan estas dos arquitecturas ya que se quiere lograr realizar dos componentes por separados, agrupándolos en módulos distintos y a su vez estos

[27] **Larman, Craig.** *UML y Patrones.* Vancouver, Canada : Prentice Hall.

tomaran una organización mediante capas, ya que cada uno de los componentes tendrá clases de la capa negocio y de la capa de acceso a datos, teniendo así una mejor organización en cada módulo a realizar.

**Diagrama de Arquitectura**

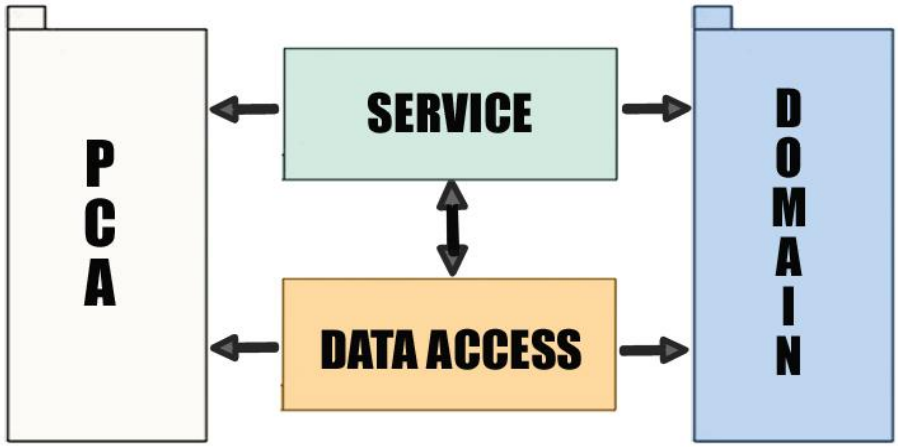


Figura 5: Diagrama de la arquitectura.

### 3.2.2 Arquitectura basada en componentes.

Un componente contiene un conjunto de clases que colaboran entre sí. Cada clase de un componente se ha elaborado completamente para incluir todos los atributos y las operaciones relevantes para su implementación.

Cuando se elige un método de ingeniería de software basado en componentes, el diseño a nivel de estos se concentra en la elaboración de las clases de análisis (clases especificas del dominio del problema), y la definición y la afiliación de las clases de infraestructura. La descripción detallada de los atributos, las operaciones y las interfaces empeladas por estas clases representan el detalle como precursor de la actividad de construcción.

### 3.2.3 Arquitectura n Capas

El estilo en capas como una organización jerárquica tal que cada capa proporciona servicios a la capa inmediatamente superior y se sirve de las prestaciones que le brinda la inmediatamente inferior. [28]

Las restricciones topológicas del estilo pueden incluir una limitación, más o menos rigurosa, que exige a cada capa operar sólo con capas adyacentes, y a los elementos de una capa entenderse sólo con otros elementos de la misma; se supone que si esta exigencia se relaja, el estilo deja de ser puro y pierde algo de su capacidad heurística; también se pierde, naturalmente, la posibilidad de reemplazar completamente una capa sin afectar a las restantes, disminuye la flexibilidad del conjunto y se complica su mantenimiento. [29]

**La capa de negocio** encapsula la lógica de implementación para lograr una automatización del negocio planteado, es la capa más específica de la aplicación, es decir en esta se encuentran los servicios generales que brinda el sistema.

**La capa de acceso a datos** agrupa la implementación necesaria para obtener los datos requeridos en determinados procesos de negocio y abstraer así la forma en que los datos persisten o son obtenidos.

**Dominio del módulo** en este paquete se tendrán todas las entidades que se manejan en el sistema, clases que su función principal es la información que contienen y no su comportamiento.

---

[28] **Shaw, David Garlan y Mary.** *An introduction to software architecture.* s.l. : CMU Software Engineering Institute Technical Report, 1994. GS94.

[29] **Microsoft Patterns & Practices.** msdn.microsoft.com. *msdn.microsoft.com.* [Online] Microsoft Patterns & Practices, 2004. http://msdn.microsoft.com/architecture/patterns/default.aspx?pull=/library/enus/.

**Comunicación entre capas** la comunicación entre capas se realiza mediante *los* extension points (puntos de extensión), pues mediante ellos se podrán obtener todos los plugin(servicios) que implementan una interfaz o uno en específico.

### 3.3 Diagrama de paquetes del Diseño.

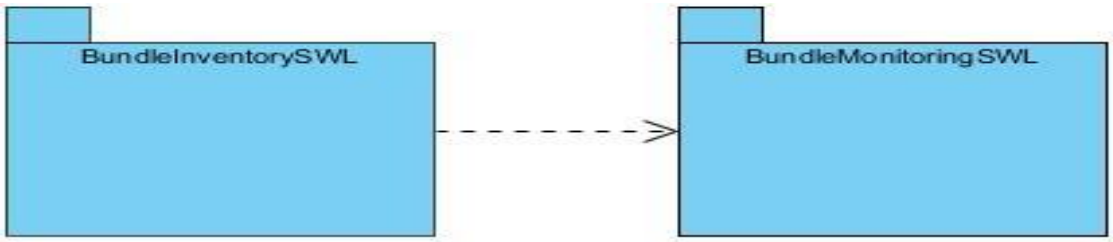


Figura 6: Diagrama de paquetes del diseño.

### 3.4 Diagramas de clases del Diseño.

Un diagrama de clases es “un diagrama que describe un conjunto de objetos que comparten las mismas responsabilidades, relaciones, operaciones, atributos y semánticas”.

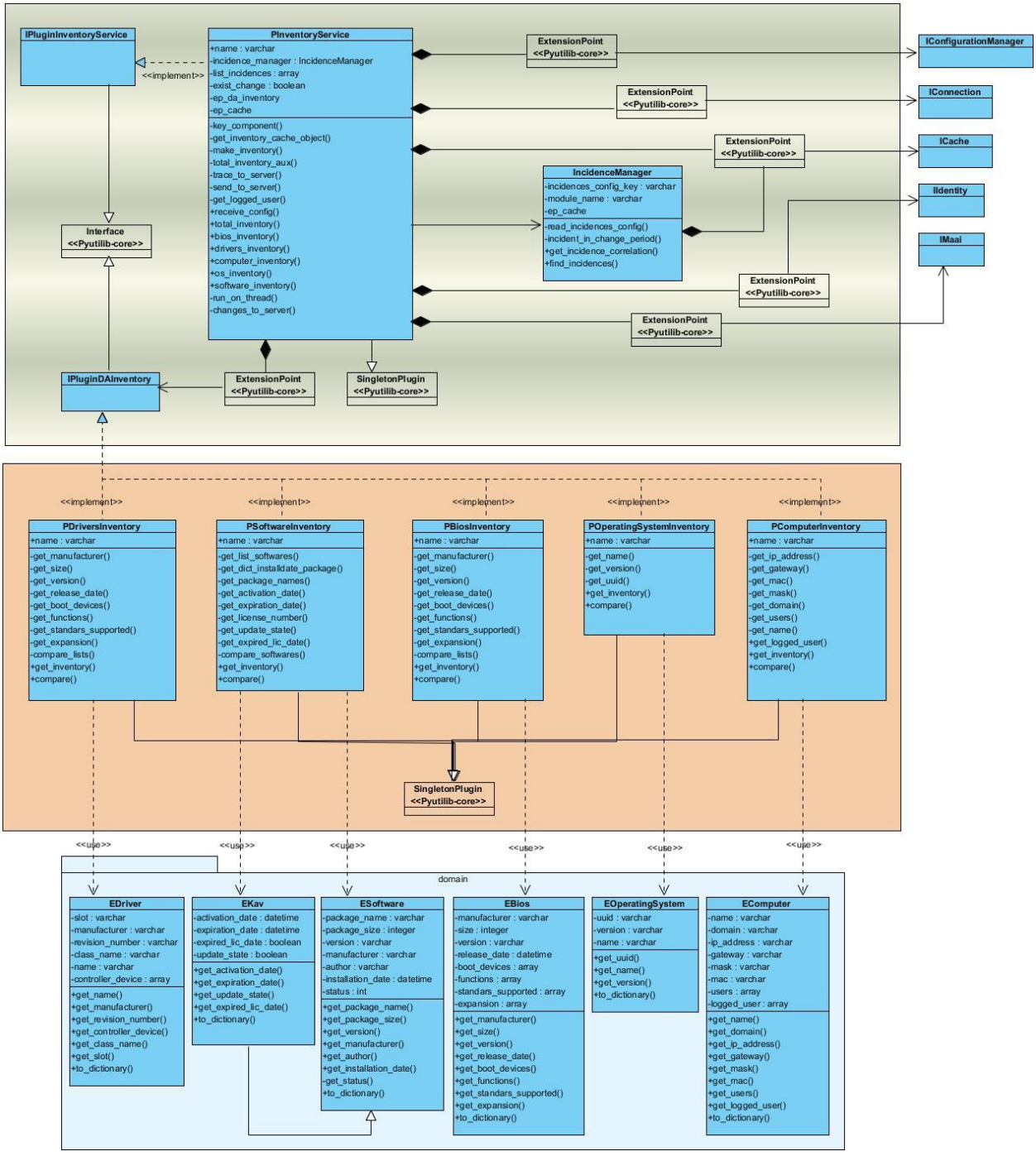


Figura 7: Diagrama de clases del diseño del Bundle Inventory Software.

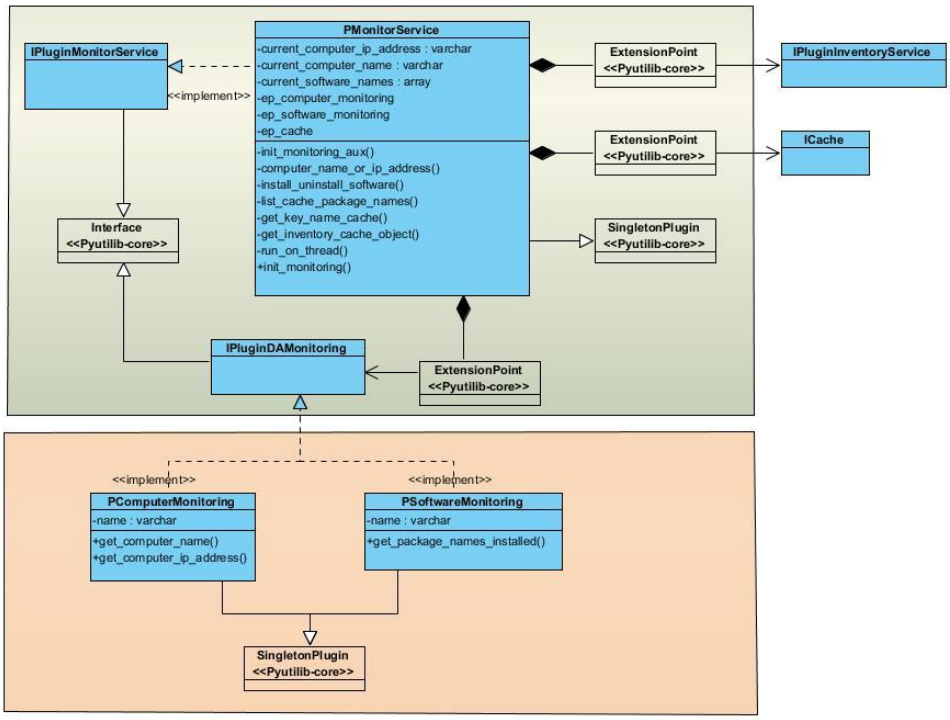


Figura 8: Diagrama de clases del diseño del Bundle Monitoring Software.

## 3.4 Patrones de Diseño

Los patrones de diseño son soluciones de diseño a los problemas recurrentes en la construcción de software. Son a menudo mal interpretados como aplicables sólo a la programación en los grandes sistemas, pero en realidad, se pueden aplicar a la solución de problemas en la programación pequeña como en la implementación de estructuras de datos o algoritmos simples. Los patrones de diseño se pueden combinar en los componentes que resuelven grandes problemas**."**

### 3.4.1 Patrones GOF (Gang of Four)

Los patrones GoF, se agrupan en tres categorías atendiendo las funciones que realizan: "Creacionales: Abarcan los procesos de creación de objetos. Estructurales: Tratan con la composición de las clases y objetos. De comportamiento: Caracterizan el modo en que las clases u objetos interactúan y distribuyen responsabilidades.

#### 3.4.1.1 Observer

**Problema:**

Cuando un objeto debe ser capaz de notificar a otros objetos sin suponer acerca de que son esos objetos. En otras palabras, no se quiere que los objetos estén perfectamente acoplados.

**Solución:**

Definir una dependencia de uno-a-muchos entre objetos, de manera que cuando un objeto cambia de estado todos los que dependan de él sean notificados y actualizados automáticamente.

**Consecuencias:**

El patrón Observer permite variar los sujetos y los observadores independientemente. Se puede rehusar sujetos sin el rehúso de observadores y viceversa. Permite agregar observadores sin modificar el sujeto o los observadores.

**Aplicación en el sistema**

En la aplicación se hace uso del patrón pues los servicios (instancias de los plugins) funcionan como observadores y el "notificador" seria donde se hace el extension point de la interfaz que implementen esos plugins. Ejemplo tendremos como notificador PInventorySErvice que es el plugins donde se hace el extension point y como observadores todos los plugins de acceso a datos.

**3.4.1.2 Singleton**

**Problemas**

Varios clientes precisan referenciar a un mismo elemento y se necesita asegurar de que no hay más de una instancia de ese elemento.

**Soluciones**

Garantizar una única instancia.

La solución clásica es utilizar exclusión mutua en el método de creación de la clase que implementa el patrón.

**Consecuencias**

Acceso controlado a la única instancia. Puede tener un control estricto sobre cómo y cuándo acceden los clientes a la instancia.

Permite el refinamiento de operaciones y la representación. Se puede crear una subclase de Singleton.

Permite un número variable de instancias. El patrón hace que sea fácil cambiar de opinión y permitir más de una instancia de la clase Singleton.

**Aplicación en el sistema**

En la aplicación este patrón se pone de manifiesto ya que se tendrán varios plugins los cuales heredaran de PyUtilib y serán Singleton, existirá una única instancia de estos en la aplicación, ejemplos de estos son los plugins del acceso a dato los cuales se encargan de recopilar la información para realizar cada uno de los inventarios en específico, al utilizarse estos posteriormente siempre se trabajará con su única instancia creada.

### 3.4.2 Patrones de Principios Generales para Asignar Responsabilidades (GRASP).

Los patrones GRASP describen los principios fundamentales del diseño de objetos y la asignación de responsabilidades, expresados como patrones.

GRASP es un acrónimo de *General Responsibility Assignment Software Patterns* (patrones generales de software para asignar responsabilidades). El nombre se eligió para sugerir la importancia de *aprehender (grasping* en inglés) estos principios para diseñar con éxito el software orientado a objetos. [30]

Patrones GRASP de mayor importancia:

- ✓ Experto en Información.
- ✓ Creador.
- ✓ Alta Cohesión.
- ✓ Bajo Acoplamiento.

#### 3.4.2.1 Experto en Información (o Experto)

**Aplicación en el sistema**

Este patrón se observa en el sistema a la hora de distribuir la responsabilidades, cada uno de los plugin de acceso a dato hacen su trabajo en específico, pues son ellos los que tienen la información necesaria para realizar los inventarios específicos.

[30] **Larman, Craig.** *UML y Patrones.* Vancouver, Canada : Prentice Hall.

#### 3.4.2.2 Creador

**Aplicación en el sistema**

En el sistema el patrón se pone de manifiesto en la clase PInventoryServicio que es necesario crear un objeto de IncidenceManager pues mediante esta clase se detecta si existen incidencias, ya que es en ella donde se implementan estos métodos.

#### 3.4.2.3 Bajo Acoplamiento

**Aplicación en el sistema**

Este patrón se pone de manifiesto pues el sistema tienen dos bundle el de inventory y el de monitoring de querer realizar algún cambio al bundle de monitoring este no afecta para nada al inventory y viceversa, aunque también se puede ver la presencia de este patrón dentro del mismo bundle de inventory ya que si se quiere agregar algún tipo nuevo de inventario no se verían afectados los demás, solo se agregaría a la interfaz y se implementaría en un plugins específico.

#### 3.4.2.4 Alta Cohesión

**Aplicación en el sistema**

Este patron es aplicable en el sistema puesto que los plugins de acceso a datos tienen pocos métodos y estos se encuentran relacionados, en estos solo se obtiene el nuvo objeto del inventario hecho y se compara con el anterior que se encuentra guardado en la cache, son utilizados con un proposito especifico, tambien colaborando con el bajo acoplamiento como ya se explicó anteriormente.

## 3.5 Conclusiones

En este capítulo se realizó la definición de la arquitectura del módulo y los patrones de diseño que son empleados en la construcción del mismo. Se alcanzó realizar el diseño de clases del sistema y definir las relaciones entre las mismas agrupándolas en paquetes, por lo que como resultado son conformados los diagramas de clase, que facilitan el entendimiento de las largas cantidades de datos del sistema y la relación entre diferentes partes de los mismos. Gracias al diseño de clases del sistema se logra ya tener un punto de partida para la implementación de este módulo.

# CAPÍTULO 4: IMPLEMENTACIÓN Y PRUEBA

## 4.1 Introducción

En este capítulo se explicará todo lo relacionado con la fase de implementación y prueba del sistema, realizando los diagramas de componentes de cada subsistema de implementación y el diagrama de despliegue el cual se aplicará con la culminación del sistema. También se mostrarán los casos de prueba para los principales casos de uso.

## 4.2 Diagrama de Despliegue.

Un diagrama de despliegue es utilizado para representar la distribución física de un sistema. El mismo muestra la configuración de los nodos de procesamiento en tiempo de ejecución, los links de comunicación entre ellos y las instancias de los componentes y objetos que residen en ellos. Para el módulo de GIISL el diagrama de despliegue es[31]:

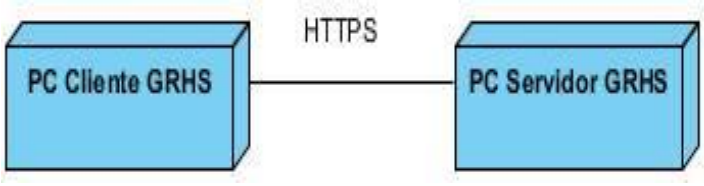


Figura 9: Diagrama de Despliegue.

**Descripción de los nodos del diagrama de despliegue.**

PC Cliente GRHS: PC donde se encuentra instalada la aplicación cliente, mediante la cual se podrán detectar las distintas incidencias que comenten los usuarios.

PC Servidor GRHS: PC donde se encuentra el servidor, contiene todo lo necesario para montar la aplicación y la parte web de la aplicación.

## 4.3 Subsistema de Implementación.

[31] **Ivar Jacobson, Grady Booch y James Rumbaugh.** *El Lenguaje Unificado de Modelado. Manual de Referencia.* 1era edición. Madrid : Addison Wesley, 2000. 84-7829-036-2.

Una colección de componentes y otros subsistemas de implementación usadas para estructurar el modelo de implementación y dividirlas en pequeñas partes que pueden ser integradas y probadas de forma separadas. Estos podrían incluir modelos claves del subsistema (modelo de componentes y diagrama de despliegue).

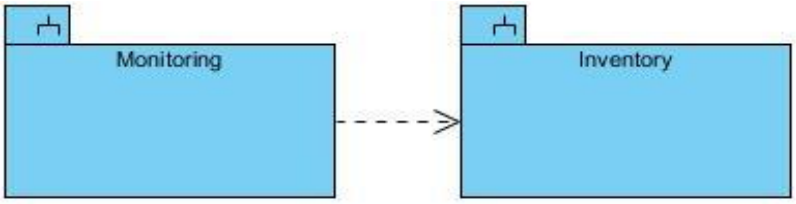


Figura 10: Diagrama de Subsistema de Implementación

### 4.4 Diagrama de Componentes.

"Un diagrama de componentes muestra las organizaciones y dependencias entre componentes."
Los diagramas de componentes se utilizan para modelar la vista de implementación estática de un sistema, estos muestran las organizaciones y dependencias lógicas entre componentes software. Los elementos de modelado dentro de un diagrama de componentes son componentes y paquetes. [32]

[32] **Ivar Jacobson, Grady Booch y James Rumbaugh.** *El Lenguaje Unificado de Modelado. Manual de Referencia.* 1era edición. Madrid : Addison Wesley, 2000. 84-7829-036-2.

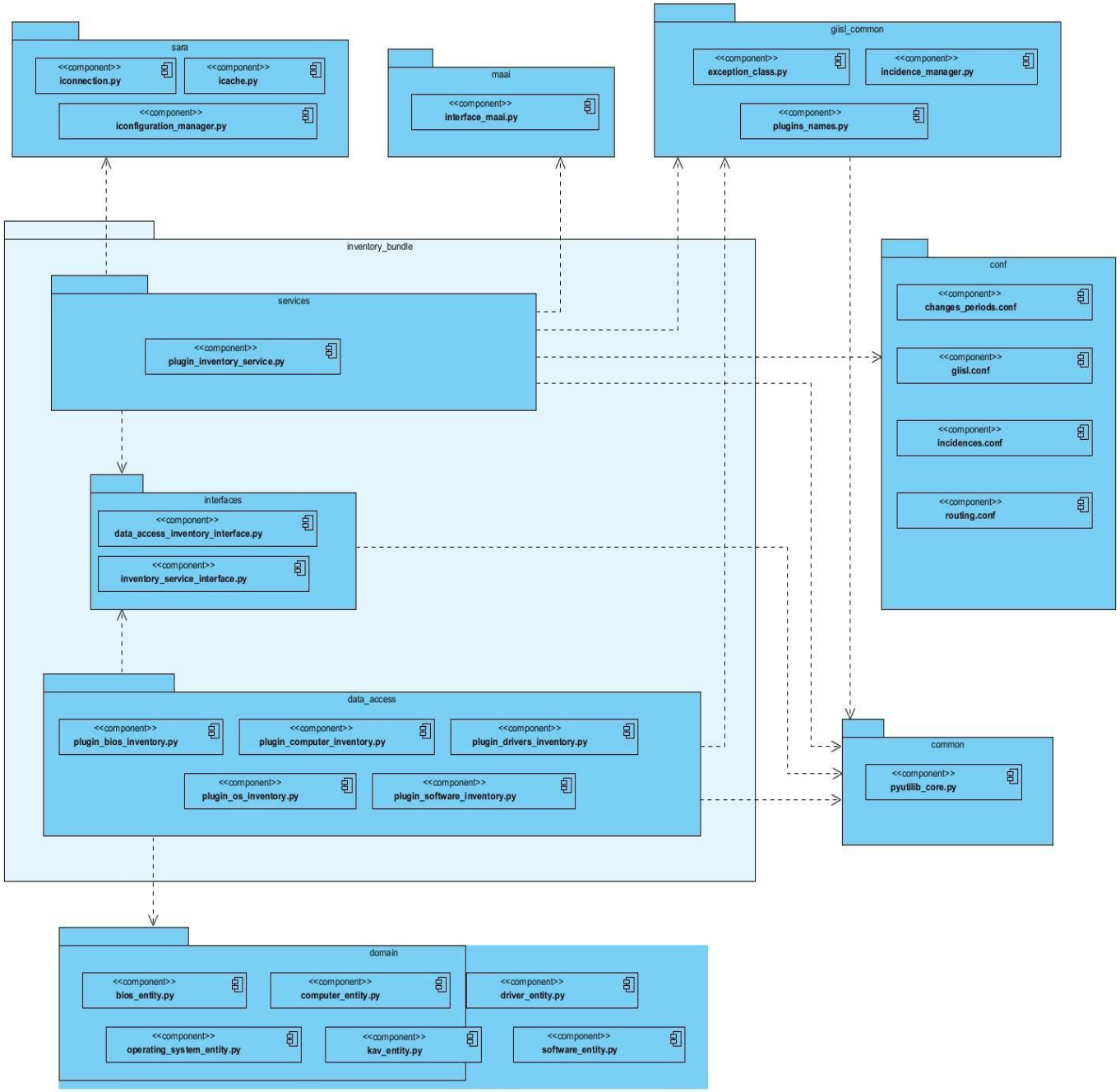


Figura 11: Diagrama de componentes para el Subsistema de Inventario.

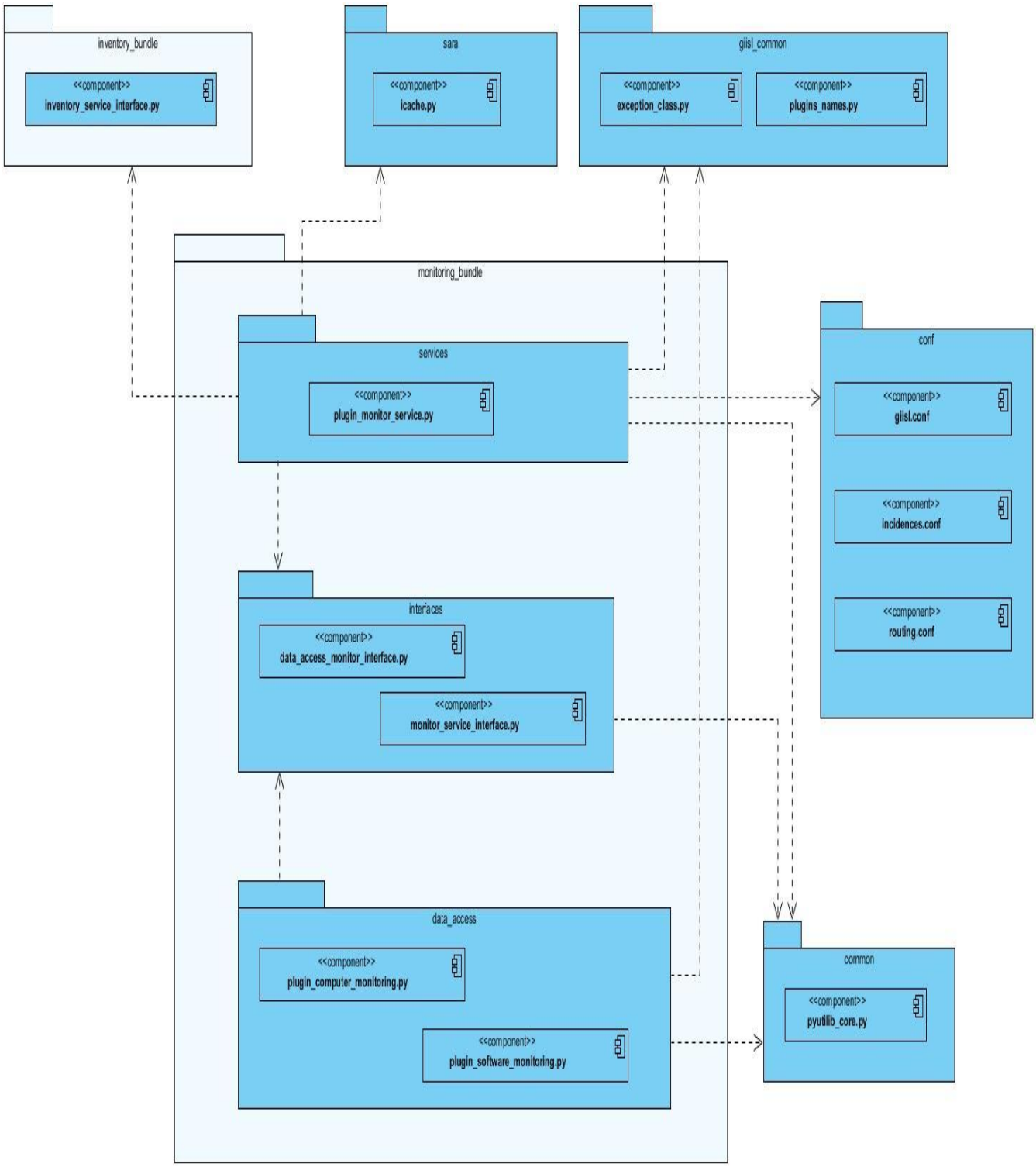


Figura 12: Diagrama de componentes para el Subsistema de Monitoreo.

### 4.5 Pruebas

El principal objetivo de realizar pruebas a un sistema es buscar y documentar errores, validando de esta forma que se cumplan con el cumplimiento de los requisitos y poder dar una indicación de calidad.

Se le denomina prueba aquellas acciones que se llevan a cabo sobre el software para verificar o revelar la calidad del producto, dando la posibilidad de identificar posibles fallos tanto de implementación como de calidad o usabilidad.

#### 4.5.1 Niveles de Prueba

Las pruebas se aplican en diferentes niveles de trabajo, para realizar las pruebas al sistema se escogieron los siguientes niveles.

Pruebas de Unidad: Son el proceso de comprobar los componentes individuales en el sistema. Este es un proceso de prueba de defecto, por lo que sus objetivos es encontrar defectos en estos componentes. Los desarrolladores de los componentes son los responsables de probarlos.

Existen diferentes objetos que pueden probarse en esta etapa:

- ✓ Funciones individuales o métodos dentro de un objeto.
- ✓ Clases de objetos que tiene varios atributos y métodos.

Pruebas de Integración: El proceso de integración del sistema implica construir este a partir de sus componentes y probar el sistema resultante para encontrar problemas que puedan surgir debido a la integración de los componentes. Los componentes que se integran pueden ser componentes comerciales, componentes reutilizables que han sido adaptados a un sistema en particular o componentes nuevos desarrollados.

Algunas veces, primero se desarrolla el esqueleto del sistema en su totalidad y se le añaden los componentes, esto de denomina integración descendente. [33]

[33] **Sommerville, Ian.** *Ingeniaria del Software.* 7ma. Madrid : PEARSON EDUCACION, S.A., 2005. 84-7829-074-5.

### 4.5.2 Tipos de Prueba

Los tipos de prueba tienen objetivos específicos:

**Función:** Pruebas fijando su atención en la validación de las funciones, métodos, servicios, caso de uso.

### 4.5.3 Métodos de Prueba

Son dos fundamentales: el método de la caja negra y de la caja blanca.

Prueba de caja blanca:

Comprueba los caminos lógicos del software proponiendo casos de prueba que se ejerciten conjuntos específicos de condiciones y/o bucles. Se puede examinar el estado del programa en varios puntos para determinar si el estado real coinciden con el esperado o mencionado.

Pruebas de caja negra:

Se refiere a las pruebas que se llevan a cabo sobre la interfaz del software. O sea los casos de prueba pretenden demostrar que las funciones del software son operativas, que la entrada se acepta de forma adecuada y que se produce una salida correcta, así como que la integridad de la información externa se mantiene. Esta prueba examina algunos aspectos del modelo fundamentalmente del sistema sin tener mucho en cuenta la estructura interna del software.

Algunas técnicas utilizadas en la prueba de caja negra son:

Partición de Equivalencia: Técnica que divide el campo de entrada de un programa en clases de datos de los que se pueden derivar casos de prueba. La partición equivalente se dirige a una definición de casos de prueba que descubran clases de errores, reduciendo así el número total de casos de prueba que hay que desarrollar. [34]

### 4.5.4 Estrategias de pruebas

Después de hacer una breve explicación de cada uno de los elementos que se utilizaran en la realización de las pruebas para el sistema solo falta presentar las estrategias.

---

[34] **Ivar Jacobson, Grady Booch y James Rumbaugh.** *El Lenguaje Unificado de Modelado. Manual de Referencia.* 1era edición. Madrid : Addison Wesley, 2000. 84-7829-036-2.

Nivel de prueba**:** Unidad

Tipo de Prueba: Funcionalidad

Método de prueba: Caja Blanca

Para cumplir con esta estrategia fue empleada la técnica automática pues se empleó la herramienta Pyunit la cual forma parte de la librería estándar de Python desde la versión 2.1.

```
import unittest
from giisl.inventory_bundle.data_access.plugin_bios_inventory \
    import PBiosInventory
from giisl.domain.bios_entity import EBios

class TestFunctionsBIOS(unittest.TestCase):

    def setUp(self):
        self._epb = PBiosInventory()
        self.bios = self._epb.get_inventory()

    def test_bios(self):
        self.assertTrue(isinstance(self.bios, EBios))
    def test_bios_manufacturer(self):
        self.assertTrue(isinstance(self.bios.get_manufacturer, str)
                        or self.get_manufacturer.get_name == None)
    def test_bios_size(self):
        self.assertTrue(isinstance(self.bios.get_size, str)
                        or self.computer.get_size == None)
    def test_bios_version(self):
        self.assertTrue(isinstance(self.bios.get_version, str)
                        or self.computer.get_version == None)
    def test_bios_release_date(self):
        self.assertTrue(isinstance(self.bios.get_release_date, str)
                        or self.computer.get_release_date == None)
    def test_bios_functions(self):
        self.assertTrue(isinstance(self.bios.get_functions, list))
    def test_bios_standars_supported(self):
        self.assertTrue(isinstance(self.bios.get_standars_supported, list))
    def test_bios_expansion(self):
        self.assertTrue(isinstance(self.bios.get_expansion, list))

if __name__ == '__main__':
    unittest.main()
```

Figura 13: Prueba realizada al plugins bios inventory.

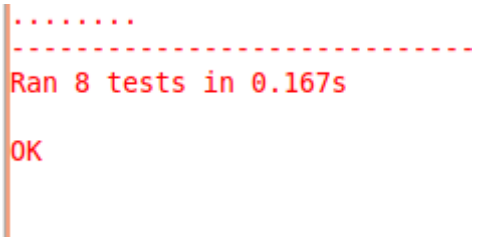


Figura 14: Resultado de la prueba aplicada al plugins bios inventory.

**Anexo 1: ¡Error! No se encuentra el origen de la referencia.**

**Anexo 2:** ¡Error! No se encuentra el origen de la referencia.

Nivel de prueba**:** Integración

Tipo de Prueba: Funcionalidad

Método de prueba: Caja Negra

La técnica utilizada para realizar esta estrategia fue la manual, realizando una tabla de integración para los módulos con que se interactúa.

| Módulo actual | Módulo integrado | Funcionalidad | Condiciones de ejecución | Escenarios de prueba | Resultado previsto | Resultado real |
|---|---|---|---|---|---|---|
| | | | | | | |

| | | | | | | |
|---|---|---|---|---|---|---|
| Módulo de Gestión de Inventario e Incidencias de Software en Linux | Sistema de Administración de Recursos y Acciones (SARA). | Enviar Resultado. | Se obtuvo información referente al software, de la computadora.<br>SARA y el módulo actual han estado ejecutándose correctamente. | Enviar Resultados correctamente. | Enviar correctamente el resultado, el cual puede ser un inventario, una incidencia o una traza, estos son enviados mediante el plugin PInventoryService el cual se conectara con la interfaz IConecction para acceder al método que nos permite enviar el resultado al servidor. | Se envió el resultado (traza, inventario, incidencia) satisfactoriamente. |

| | | | | | | |
|---|---|---|---|---|---|---|
| Módulo de Gestión de Inventario e Incidencias de Software en Windows | Sistema de Administración de Recursos y Acciones (SARA). | Realizar Inventario. | La PC debe estar encendida. SARA y el módulo actual han estado ejecutándose correctamente | Obtener correctamente la orden de realizar Inventario. | Recibir correctamente la solicitud para realizar el inventario mediante patrones definidos. | Se recibió correctamente la solicitud para realizar el inventario. |
| Módulo de Gestión de Inventario e Incidencias de Software en Windows. | Sistema de Administración de Recursos y Acciones (SARA). | Obtener configuraciones. | La PC debe estar encendida. SARA y el módulo actual han estado ejecutándose correctamente. | Obtener configuraciones correctamente. | Obtener correctamente las configuraciones necesarias tales como configuraciones de incidencia y configuraciones para realizar la monitorización. | Se obtuvieron correctamente las configuraciones necesarias para la aplicación. |
| Módulo de Gestión de Inventario e Incidencias de Software en | Sistema de Administración de Recursos y Acciones | Pedir Actualizaciones (inventario BIOS) | SARA y el módulo actual han estado ejecutándose | Solicitar correctamente la actualización del inventari | Se solicita correctamente la actualización de inventario mediante los | Se envió satisfactoriamente la solicitud de actualización del |

| | | | | | | |
|---|---|---|---|---|---|---|
| Windows. | (SARA). | | correctamente | o del BIOS. | patrones definidos. | inventario. |
| Módulo de Gestión de Inventario e Incidencias de Software en Windows. | Sistema de Administración de Recursos y Acciones (SARA). | Obtener Actualizaciones (inventario BIOS) | SARA y el módulo actual han estado ejecutándose correctamente. | Obtener correctamente la actualización del inventario del BIOS. | Obtener correctamente la actualización del inventario que envía el servidor a través de SARA | Se obtuvo satisfactoriamente la actualización del inventario. |
| Módulo de Gestión de Inventario e Incidencias de Software en Windows. | Módulo de Alarmas y Acciones ante Incidencias (MAAI). | Notificar incidencia. | SARA y el módulo actual han estado ejecutándose correctamente. | Notificar correctamente incidencias detectadas. | Notificar correctamente las incidencias detectadas a MAAI mediante el PInventoryService para que tomen acciones. | Se notificó correctamente las incidencias detectadas a MAAI. |

Tabla 9: Pruebas de Integración.

En la siguiente figura (Figura 15), se muestra el comportamiento de las pruebas de integración realizadas a este módulo con respecto a los módulos SARA y MAAI, donde se efectúan 5 pruebas en base a 100 puntos cada una de ellas. Debido a que en total se realizan 10 acciones conjuntas

con estos módulos, se calculó el porciento de las mismas que aplicaban correctamente en cada una de estas iteraciones.

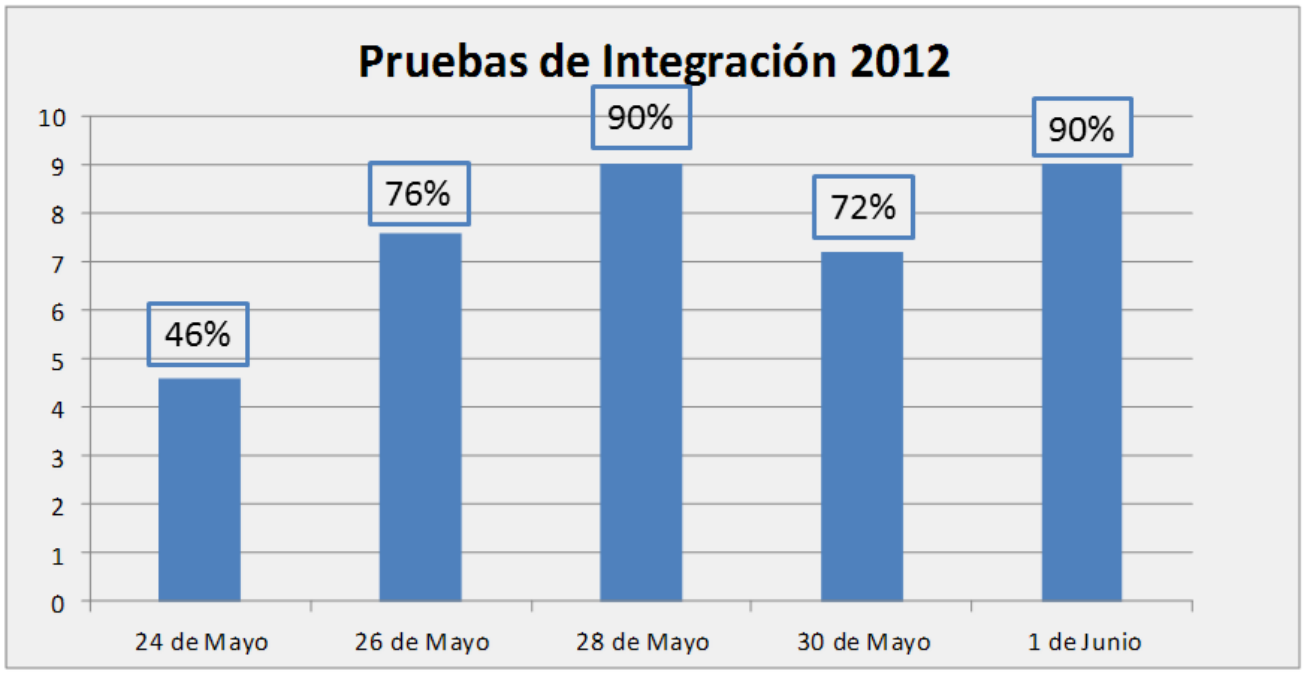


Figura 15: Pruebas de Integración con los módulos SARA y MAAI.

Como se observa en el gráfico anterior (Figura 15), la integración de este módulo con SARA y MAAI fue ascendiendo positivamente. Además, vale señalar que aunque en la prueba efectuada el 30 de Mayo se detectó un descenso debido a actualizaciones de SARA que provocaron ligeros ajustes en GIISL, fueron fácil y rápidamente implementadas las soluciones correspondientes a dichos inconvenientes.

**Resultados de las Pruebas**

Durante las pruebas realizadas fueron obtenidas 9 no conformidades, entre las principales se pueden mencionar:

- Intento innecesario de obtener datos del estado del Kaspersky y su licencia sin que el mismo estuviera instalado en el ordenador cuando SARA solicitaba la realización del inventario total.

- Repetición de envío al servidor de incidencia detectada por modificación en el BIOS cuando un ordenador presenta ambos sistemas operativos instalados (Linux y Windows).
- Ocurrencia de error en el sistema debido al intento de obtener algún inventario almacenado sin que este exista en el fichero de los datos.
- Incorrecta estructuración de las incidencias que son notificadas al Módulo de Alarmas y Acciones ante Incidencias.

Estas no conformidades encontradas fueron resueltas posteriormente.

## 4.5 Conclusiones

En este capítulo se abordó todo lo relacionado con la fase de implementación y prueba del sistema, realizando los diagramas de componentes de cada subsistema de implementación y el diagrama de despliegue, el cual se aplica con la culminación del sistema. También se mostraron los casos de prueba para los principales casos de uso, necesarios para determinar si el software cumple con los requisitos planteados.

# CONCLUSIONES GENERALES

Con la realización del trabajo de diploma Módulo de Gestión de Inventarios e Incidencias de Software en GNU Linux, se cumplió con los principales objetivos planteados realizando un sistema que sea capaz de realizar un inventario de software en una red de ordenadores y a su vez monitorizar los identificadores de la PC, instalación y desinstalación de los paquetes de software. Este sistema es parte de la aplicación Gestión de Recursos de Hardware y Software la cual se pretende mejorar el trabajo de los administradores de una red haciendo cumplir con las políticas establecidas en su entidad.

Se estructuraron cuatro capítulos en los cuales se evidencia todo el proceso de desarrollo para la construcción del módulo, realizando un estudio detallado del problema existente, estado del arte, se sentaron las bases para la implementación del sistema, además se hacer una selección de la metodología, herramientas y lenguaje ajustándose a las posibilidades de nuestro país y a la emigración hacia software libre.

La arquitectura utilizada permitió obtener un sistema escalable, ya que permite incorporar nuevos componentes y funcionalidades sin que sean necesarios cambios excesivos en la implementación.

A partir de las pruebas realizadas con PyUnit se constató que todas las funcionalidades del sistema son operativas, ya que producen el resultado esperado a partir de la entrada correspondiente.

# RECOMENDACIONES

Después de la realización de esta investigación se recomienda su análisis para posteriores trabajos relacionados con el tema del inventario de software en la redes de computadoras. La herramienta se puede poner en práctica en los laboratorios de la universidad así mejoraría el trabajo de los técnicos y podrían mantener un mayor control acerca de las actividades que realizan los estudiantes en los horarios docentes. Incluir nuevas funcionalidades que garanticen el control periódico en cuanto a fecha de actualización y expiración de llaves para todos los antivirus, no solamente para el Kaspersky. Mejorar el funcionamiento del sistema para que se pueda ejecutar satisfactoriamente en distribuciones Linux que no usen sistema de paquetería Debian.

# GLOSARIO DE TÉRMINOS

**PC (Computadora Personal)**: Computadora digital personal basada en un microprocesador y diseñada para ser utilizada por un solo individuo a la vez.

**Paquete:** Es un mecanismo de propósito general para organizar elementos en grupos.

**DNS (Domain Name Server):** Servidor de nombres de dominio.

**DHCP:** Protocolo de Configuración Dinámica de Cliente.

**HTTP (Hyper Text Transfer Protocol):** Protocolo que usa la World Wide Web (WWW) para comunicar los servidores con los navegadores.

**HTTPS (Hypertext Transfer Protocol Secure):** Es un protocolo de red basado en el protocolo HTTP, para la transferencia segura de datos de hipertexto (es la versión segura de HTTP).

**CASE (Computer Aided Software Engineering):** La Ingeniería de Software Asistida por Computación es la aplicación de métodos y técnicas a través de las cuales se hacen útiles a las personas comprender las capacidades de las computadoras, por medio de programas, de procedimientos y su respectiva documentación.

**Herramientas CASE:** Representan una forma que permite Modelar los Procesos de Negocios de las empresas y desarrollar los Sistemas de Información Gerenciales.

**PHP (Personal Home Page):** Lenguaje script avanzado para diseño de sitios web.

**FTP:** Protocolo de Transferencia de Archivos.

**TCP/IP (Transmission-Control-Protocol/Internet Protocol):** Es un sistema de protocolos que hacen posibles servicios Telnet, FTP, E-mail, y otros entre ordenadores que no pertenecen a la misma red.

**GOF:** Plantea que los patrones se clasifican según el propósito para el que han sido definidos, en Creacionales (solucionan problemas de creación de instancias. Ayudan a encapsular y abstraer dicha creación), Estructurales (solucionan problemas de composición (agregación) de clases y objetos) y de Comportamiento (soluciones respecto a la interacción y responsabilidades entre clases y objetos, así como los algoritmos que encapsulan).

**GPL (General Public License):** Es una licencia que protege la libre distribución, modificación y uso de software. Su propósito es declarar que el software cubierto por esta licencia es software libre y protegerlo de intentos de apropiación que restrinjan esas libertades a los usuarios.

**XML (Extensible Markup Language):** Es un lenguaje simple el cual posibilita la estructuración de documentos electrónicos de manera lógica tanto para humanos como para máquinas.

**WMI (Windows Management Instrumentation):** *Instrumental de Administración de Windows* es una iniciativa que pretende establecer normas estándar para tener acceso y compartir la información de administración a través de la red de una empresa.

**LAN (Local Area Network):** Red de área local, es una red que conecta los ordenadores en un área relativamente pequeña y predeterminada (como una habitación, un edificio, o un conjunto de edificios).

**WAN (Wide Area Network)**: Red de área amplia normalmente consiste en dos o más redes de área local. Los ordenadores conectados a una red de área ancha normalmente están conectados a través de redes públicas, como la red de teléfono. También pueden estar conectados a través de líneas alquiladas o de satélites.

**DNA (Digital Network Architecture):** Es una arquitectura de red, creada por DEC (Digital Equipment Corporation, la cual fue una compañía americana considerada pionera en la fabricación de minicomputadores).

**Direcciones MAC:** Control de Acceso al Medio en las redes de computadoras es un identificador de 48 bits (6 bloques hexadecimales) que corresponde de forma única a una tarjeta o dispositivo de red. Se conoce también como dirección física, y es única para cada dispositivo.

**WINS:** es una aplicación de Microsoft que resuelve los nombres que se utilizan generalmente para referirse a los ordenadores (por ejemplo, SERVER1, NOMINAS, etc.). El servicio WINS cambia estos nombres a direcciones IP.

**SQL Server (Structured Query Language):** Es un lenguaje declarativo de acceso a bases de datos relacionales que permite especificar diversos tipos de operaciones sobre las mismas. Aún a características del álgebra y el cálculo relacional permitiendo lanzar consultas con el fin de recuperar información de interés de una base de datos, de una forma sencilla.

**SSL (Secure Socket Layer):** El protocolo SSL es un sistema de seguridad utilizado actualmente por la mayoría de empresas que comercian a través de Internet. Es un sistema de seguridad ideado para acceder a un servidor garantizando la confidencialidad de los datos mediante técnicas de encriptación modernas.

**Apache:** Es el servidor de páginas web. Un servidor de páginas web es un programa que permite acceder a páginas web alojadas en un ordenador.

**MySQL:** Es un sistema de gestión de bases de datos relacional, licenciado bajo la GPL de la GNU.

**CMMI (Capability Maturity Model for Integration):** Modelo de Madurez de la Capacidad Integrado, es un modelo de procesos que contiene las mejores prácticas de la industria para el desarrollo, mantenimiento, adquisición y operación de productos y servicios.

**Kaspersky:** Es una empresa especializada en productos para la seguridad informática.