Universidad de las Ciencias Informáticas

Facultad 2

Módulo de Alarmas y Acciones ante Incidencias

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

Surenis Ramos Betancourt

Ivett León Rodríguez

Tutor

Ing. Julio Antonio Hernández Pérez

Co-tutor

Ing. Yoanni Ordoñes Leyva

Ciudad de La Habana, Junio del 2012.

"Año 54 de la Revolución"

PENSAMIENTO

"Los estudiantes son en su mayoría revolucionarios. Revolucionarios por naturaleza, porque pertenecen a ese estrato de jóvenes que se abren en la vida y que adquieren todos los días conocimientos nuevos."

Ché.

DECLARACIÓN DE AUTORÍA

Declaramos que somos los únicos autores de este trabajo y autorizamos a la Facultad 2 de la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

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

_____ _____

Surenis Ramos Betancourt
Firma del autor

Ivett León Rodríguez
Firma del autor

_____

Ing. Julio Antonio Hernández Pérez
Firma del tutor

DATOS DE CONTACTO

Ing. Julio Antonio Hernández. Graduado en el año 2011 de Ingeniería en Ciencias Informáticas en la Universidad de las Ciencias Informáticas. Experiencia como desarrollador en proyecto Servicios Telemáticos. Posee una publicación científica en la Serie Científica de la Universidad de la Ciencias Informáticas. Coautor de HERMINWEB, herramienta de minería de datos para la obtención de patrones, aplicados a registros de servidores proxy, en términos de tareas descriptivas: Agrupamiento y Reglas de Asociación. Investiga sobre el área de la minería de uso.

DEDICATORIA

Le dedico esta tesis a mi mamá Belkis Betancourt Pérez y a mi papá Victor Osvaldo Ramos, por su apoyo, comprensión y amor.

Surenis Ramos Betancourt.

Les dedico la tesis a mis padres Bárbara Rodríguez y Jorge Luis León por apoyarme incondicionalmente en cada una de mis travesías, por el amor que me han brindado y el sacrificio realizado.

Ivett León Rodríguez

AGRADECIMIENTOS

Agradezco a mi mamá Belkis Betancourt y a mi papá Victor Osvaldo Ramos por ser mi apoyo en los momentos difíciles y estar en los momentos alegres compartiendo mi felicidad. Por aconsejarme y ser mi guía en el transcurso de mi vida, por enseñarme lo bueno y lo malo. Por seguir y apoyar cada paso que he dado, por complacerme, por malcriarme, por cuidarme y por convertirme en la persona que soy.

Agradezco a mis abuelos Flora, Yeyo, Carmelina y Rolando, y a mi familia en general, por confiar siempre en mí y darme su cariño.

Agradezco a mis vecinos Nidia, Pipo y Tata, que estuvieron pendientes de mí, cuidándome como si fuera parte de su familia. Agradezco a Rosa y a mi madrina por su ayuda y apoyo, por considerarme una persona especial.

Agradezco a mis compañeros de aula, con los cuales he compartido mi vida en la universidad. A mis compañeras de apartamento Yisel Pavón, Luisa, Ivett, Yari, Elizabeth y Yanet; y en especial a Yuni, Yuya y Orquidea que más que mis compañeras han sido mi familia en el transcurso de mi vida universitaria.

Agradezco a mi novio por darme su cariño y nunca perder la confianza en mí, por compartir conmigo los buenos y malos momentos desde que nos conocimos, por comprenderme, por ayudarme y complacerme.

De manera general agradezco a todas las personas que me han ayudado no solo a terminar mi carrera universitaria, sino también en todos los momentos de mi vida.

Surenis Ramos Betancourt

AGRADECIMIENTOS

Agradezco a mi mamá Bárbara Rodríguez, haberme dado la vida, apoyarme siempre, cuidarme, guiarme, por confiar en mí, por ser la mejor madre del mundo.

Agradezco a mi papá Jorge Luis León por defenderme, protegerme, apoyarme, por enseñarme que las grandes cosas requieren de grandes sacrificios, por ser mi piedro.

Agradezco a mis tíos y primos, Anita, Normita, Tati, Yadian y Alexander por ser parte de mi familia y estar ahí cada vez que necesité de tu ayuda, sin darme nunca la espalda.

Agradezco a mi novio Ariel Martínez, por darme ánimo y por estar ahí esos días que ni yo me soportaba, por ser paciente y darme todo el amor del mundo.

Ivett León Rodríguez

Queremos agradecer a todas las personas que nos ayudaron y que formaron parte de nuestras vidas a lo largo de estos cinco años, en especial a nuestras compañeras, por enseñarnos que se puede tener una familia.

Agradecer a nuestro tutor Julio Antonio Hernández, por exigirnos en cada tarea y ser nuestro mejor guía en el desarrollo de este trabajo, por convertirse en nuestro amigo y en esa persona que uno siempre tendrá muchas cosas que agradecerle en la vida. Por dedicarnos su tiempo y ayudarnos en el desarrollo de la tesis.

Agradecer a Yoanni Ordoñes por ayudarnos siempre, por aconsejarnos y ser la otra persona a la cual le debemos mucho, por adoptarnos como sus niñas grandes.

Agradecer a todos los estudiantes y profesores del proyecto.

RESUMEN

El presente trabajo propone un módulo para la ejecución de acciones y envío de alarmas ante incidencias de forma automática para el Gestor de Recursos de Hardware y Software (GRHS). Posee dos módulos, uno para las estaciones clientes y otro para la aplicación servidora, ambos diseñados para Windows y GNU/Linux. El módulo de los clientes está basado en las arquitecturas N-Capas y en componentes, esta última usando PyUtilib Component Architecture (PCA). El módulo del cliente permite la ejecución de acciones de control ordenadas desde el servidor como el reinicio, apagado e hibernado de la computadora; además de ejecutar acciones ante una incidencia como: tomar el control del ratón, el teclado y las sesiones de los usuarios mediante la suspensión, así como la inhabilitación de los puertos USB y las interfaces de red Ethernet, también captura imágenes de la estación de trabajo y del usuario que se encuentre usándola. El módulo en el servidor está basado en la arquitectura Model-Template-View (MTV) del marco de trabajo Django y envía alarmas mediante correo electrónico, SMS y mensajería instantánea. Se realizaron pruebas de integración y de unidad obteniendo resultados satisfactorios.

PALABRAS CLAVE

Acciones, Alarmas, PCA, Incidencia.

ÍNDICE DE CONTENIDO

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

INTRODUCCIÓN

La administración de inventarios se centra fundamentalmente en el control de la cantidad existente de unidades, previendo la cantidad a ordenarse o a producirse en un momento dado. Se identifican además, cuáles artículos de la empresa merecen una atención especial en el inventario. Inicialmente, los inventarios se desarrollaban de forma manual, trayendo consigo algunas imprecisiones e inconsistencias. En la actualidad, la creciente dependencia tecnológica ha propiciado la migración de los tradicionales sistemas de inventarios a potentes aplicaciones informáticas. Dichas aplicaciones son significativas en cualquier tipo de empresa o compañía, mostrando mayor atractivo en su capacidad de almacenar gran cantidad de información. Estas aplicaciones informáticas también gestionan y controlan los recursos tangibles y no tangibles de las organizaciones (1).

Particularmente, el inventario de activos informáticos relaciona el conjunto de los bienes propios de hardware y software de una entidad, su estado y disponibilidad. Con el fin de registrar e inspeccionar los inventarios en las redes de computadoras, las organizaciones adoptan los sistemas de gestión de inventarios. Un sistema de gestión de inventario de hardware y software es una aplicación con soporte de datos, que acumula información sobre los activos informáticos en una red de ordenadores (2). El incremento en los procesos comerciales acarrea la necesidad de mantener un control de inventarios informáticos con el objetivo de prevenir posteriores incidencias. Además de contar con documentación detallada y personalizada de acuerdo a las necesidades de la organización, así como localizar de manera más simple equipos y aplicaciones.

Hoy en día, existen numerosas herramientas dedicadas a la gestión de inventarios de hardware y software en redes de computadoras. Así queda resuelto el problema de realizar los inventarios de forma manual. Sin embargo, los usuarios de este tipo de soluciones, requieren herramientas capaces de realizar un conjunto de operaciones que puedan ser chequeadas constantemente, sin la presencia de un administrador del sistema. Dichas operaciones ayudan a mitigar los procesos anómalos o incidencias detectados en los inventarios de hardware y software. Estas herramientas también deben estar capacitadas para enviar mensajes de alarmas ante la ocurrencia de incidencias. Estos mensajes pueden ser a través de correo electrónico, mensajería instantánea y Servicio de Mensajes Cortos (SMS, en inglés Short Messages Service).

INTRODUCCIÓN

La Universidad de las Ciencias Informáticas (UCI), cuenta con diversos centros de desarrollo, entre los que se encuentra el Centro de Telemática. Dicho centro tiene como misión fundamental el desarrollo de sistemas y servicios informáticos en las ramas de las telecomunicaciones y la seguridad informática (3). En este centro se está desarrollando el producto Gestor de Recursos de Hardware y Software conocido como GRHS. Esta plataforma es una solución tecnológica integral para los procesos de captura, análisis y consulta de la información de los activos informáticos de una red de computadoras. Dicha plataforma carece de un módulo que permita la ejecución de acciones y envío de alarmas ante incidencias al igual que todas las herramientas existentes de este tipo.

Tomando en cuenta lo expresado anteriormente, se plantea como problema a resolver ¿Cómo contribuir al mejoramiento del envío de alarmas y la realización de acciones ante incidencias en el proceso de inventario de hardware y software? El objetivo general de la investigación se dirige a desarrollar un módulo capaz de enviar alarmas y realizar acciones ante incidencias en la plataforma Gestor de Recursos de Hardware y Software. Como objeto de estudio se tienen los procesos de inventario de hardware y software. El campo de acción se enmarca en el proceso de envío de alarmas y realización de acciones ante incidencias en la plataforma Gestor de Recursos de Hardware y Software.

Los objetivos específicos a cumplir con esta investigación son:

1. Identificar limitaciones de las herramientas de gestión de inventario de activos informáticos conforme a la realización de acciones y el envío de alarmas ante la ocurrencia de una incidencia.
2. Desarrollar los componentes para enviar SMS, correos electrónicos y mensajes instantáneos.
3. Desarrollar los componentes para tomar el control del ratón y del teclado.
4. Desarrollar los componentes para capturar imágenes de las estaciones de trabajo y al usuario.
5. Desarrollar los componentes para la inhabilitación de los puertos USB.
6. Desarrollar los componentes para la inhabilitación de las interfaces de red Ethernet.
7. Desarrollar los componentes para la inhabilitación de las sesiones de la computadora donde ocurrió la incidencia.

Como idea a defender se plantea que: con la implementación del Módulo de Alarmas y Acciones ante Incidencias, se logra un subsistema capaz de automatizar los procesos de envío de alarmas y realización de acciones ante las incidencias detectadas en los inventarios de activos informáticos.

INTRODUCCIÓN

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

1. Elaborar un informe sobre las tendencias actuales de los sistemas de inventario de hardware y software, con el objetivo de comprobar sus funcionalidades y limitaciones.
2. Elaborar un informe sobre las bibliotecas y aplicaciones para el control de los puertos USB.
3. Elaborar un informe sobre las bibliotecas y aplicaciones para el control de las interfaces de red Ethernet.
4. Elaborar un informe sobre las bibliotecas y aplicaciones para el control del ratón y el teclado.
5. Identificar los requerimientos funcionales y no funcionales del módulo para especificar las funcionalidades y características que el sistema debe cumplir.
6. Desarrollar los componentes para enviar SMS, correos electrónicos y mensajes instantáneos.
7. Desarrollar los componentes para tomar el control del ratón y teclado.
8. Desarrollar los componentes para capturar imágenes de las estaciones de trabajo y al usuario.
9. Desarrollar los componentes para la inhabilitación de los puertos USB.
10. Desarrollar los componentes para la inhabilitación de las interfaces de red Ethernet.
11. Desarrollar los componentes para la inhabilitación de las sesiones de la computadora donde ocurrió la incidencia.
12. Elaborar un informe con los resultados de las pruebas internas del módulo, con el objetivo de comprobar el cumplimiento de las funcionalidades requeridas.

A continuación se muestran los métodos científicos utilizados en la investigación.

Métodos teóricos:

- Análisis-síntesis: Es utilizado durante todo el proceso investigativo, para el análisis de la bibliografía consultada y la extracción de los elementos esenciales relacionados con el objeto de estudio.
- Modelación: Es empleada en la representación de las propiedades y funcionalidades del sistema a desarrollar.
- Histórico-lógico: Posee gran importancia para la elaboración de la fundamentación teórica de la investigación, pues permite estudiar lo más notable en el plano teórico acerca de las herramientas

informáticas que realizan los inventarios de hardware y software en una red de computadoras, así como la evolución de los inventarios hasta la actualidad.

Método empírico:

- Observación**:** Se refleja durante todo el transcurso de la investigación, fundamentalmente en la observación de los procesos de realización de acciones y envío de alarmas, así como en el funcionamiento de otros módulos de la plataforma Gestor de Recursos de Hardware y Software y en los resultados de las pruebas del sistema.

El documento investigativo está estructurado en cuatro capítulos: **Capítulo 1** "Fundamentación Teórica", se hace un estudio de algunas herramientas que realizan inventarios de hardware y software y se describen las herramientas, metodología y tecnologías a utilizar durante el desarrollo de la investigación. **Capítulo 2** "Características del Sistema", se representa y describe el modelo de dominio del Módulo de Alarmas y Acciones ante Incidencias. Se especifican los actores, los casos de uso y los requerimientos funcionales y no funcionales del sistema. **Capítulo 3** "Diseño del Sistema", se describe y refleja la arquitectura del módulo y se caracteriza el diseño, describiendo los patrones de diseño empleados. **Capítulo 4** "Implementación y Pruebas", es representado y descrito el diagrama de despliegue del módulo y se caracteriza el desarrollo del sistema en término de componentes. Se especifica la estrategia de pruebas a seguir y los resultados obtenidos con dichas pruebas.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

1.1 Introducción

En el presente capítulo se presenta el marco teórico de la investigación, partiendo de los conceptos y definiciones adquiridos en el estudio de las tendencias actuales en el marco de las herramientas de inventario de hardware y software. Se definen los conceptos de inventario, incidencia, acción y alarma que sustentan la presente investigación. Contiene además, un estudio de las tecnologías, metodología y herramientas empleadas en el desarrollo del Módulo de Alarmas y Acciones ante Incidencias (MAAI).

1.1.1 Inventario

El inventario representa uno de los procesos más importantes de las empresas, donde se identifican y categorizan sistemáticamente los recursos. Se define como inventario al documento donde son registradas las comprobaciones periódicas de la existencia de cualquier artículo o recurso utilizado en una organización. El inventario tiene como objetivo fiscalizar los sistemas de control de administración y manejar los materiales y el método de almacenamiento (4). Las aplicaciones encargadas de realizar los inventarios de hardware y software llevan a cabo un análisis exhaustivo de cada activo informático. El término inventario, en el ámbito de la plataforma GRHS, se ve relacionado con otras definiciones como incidencia, acción y alarma.

Una incidencia es definida como: *acontecimiento que sobreviene en el curso de un asunto o negocio y tiene con él alguna conexión. Influencia o repercusión* (5). Una acción es el *ejercicio de la posibilidad de hacer, es el resultado de hacer* (6). Una alarma es el *mecanismo que, por diversos procedimientos, tiene por función avisar de algo* (7). En el contexto del MAAI estos conceptos tienen sus propias definiciones:

- Una incidencia es un acontecimiento no permitido en los reglamentos de una organización, ya sea la instalación de un programa en específico, la falta de actualización de licencia de un software, el extravío de algún dispositivo informático, entre otros sucesos detectados por los inventarios de hardware y software.
- Una acción, especifica la operación a realizar dada una incidencia, que bien puede considerarse como la inhabilitación del teclado y del ratón, así como de los puertos USB en la computadora donde ocurrió la incidencia.

- Una alarma se refiere al envío de mensajes para notificar la incidencia.

Existen diferentes herramientas encargadas de realizar inventarios de hardware y software en redes informáticas. A continuación se describirán las principales características de algunas de estas herramientas.

1.2 Herramientas de gestión de inventarios de hardware y software

Herramientas privativas

Total Network Inventory 2.0.6

Total Network Inventory es un programa de control de activos e inventario de computadoras, diseñado para compañías medianas y grandes. Soporta las plataformas GNU/Linux, Mac OS y Windows. Examina todos los ordenadores y portátiles en la red y genera la información sobre el sistema operativo, dispositivos y software. Es una herramienta privativa que envía notificaciones a los usuarios referentes a las incidencias detectadas en el inventario de su computadora. Sin embargo no es capaz de realizar acciones ante incidencias. Además, posibilita la creación de breves informes que pueden ser impresos, visualizados en la pantalla y exportados a diferentes formatos (8).

Login Inventory 5

Permite obtener un inventario automático de todo el software y el hardware de las computadoras, sin instalar agentes en los clientes, ya que realiza el inventario a través de acceso remoto. Permite conocer qué componentes de hardware y software se tienen instalados en cada estación de trabajo, facilitando la tarea de diagnóstico de problemas en los activos informáticos. Soporta las plataformas GNU/Linux, Mac OS y Windows. Se considera una aplicación pasiva, pues no envía alarmas y ni realiza acciones ante las incidencias detectadas en el inventario (9).

NetSupport DNA 2.7

Mediante el uso de esta herramienta privativa, se detectan cambios o situaciones tanto a nivel de sistema (adición de máquinas, cambios de hardware, uso de Internet, uso de licencias de software) como a nivel de máquina (uso de la Unidad Central de Procesamiento (CPU, en inglés Central Processing Unit),

espacio en disco, instalación de dispositivos USB). Soporta las plataformas GNU/Linux y Windows. Envía alarmas en tiempo real, con la información referente a la cantidad de espacio libre en disco o la ejecución de determinadas aplicaciones. Incluye la opción de imprimir y exportar a los formatos *.pdf (formato del documento portátil, en inglés Portable Document Format), *.doc (extensión de Microsoft Word) y *.xls (extensión de Microsoft Excel). Sin embargo, esta herramienta también tiene sus limitantes, no ejecuta acciones ante la ocurrencia de una incidencia en una red de ordenadores de forma automática, y realiza un resumen de los avisos activados en un período determinado con los datos del operador DNA asignado (10).

Herramientas libres

Gestión Libre de Parque Informático (GLPI) 0.65

GLPI es una aplicación de código abierto, muy útil en empresas con varias sedes, para gestionar el inventario. Es solamente la interfaz de administración de un sistema de inventario, ya que toma sus datos de otras herramientas como OCS Inventory NG y Fusion Inventory. Incluye datos administrativos como pueden ser: períodos de validez de las licencias de software, garantías y tiempos de caducidad de las mismas, o datos de ubicación de los equipos. Posibilita efectuar búsquedas parametrizadas y ordenadas sobre inventarios a computadoras, material de red, impresoras, monitores, periféricos externos y software. Soporta las plataformas GNU/Linux y Windows. Permite el envío de alarmas a través de correo electrónico cuando se detecta alguna incidencia y posibilita exportar informes a *.pdf, *.csv (archivo de valores separados por coma, en inglés comma separated values) y *.slk (en inglés Sylk Symbolic Link). Sin embargo, esta aplicación no es capaz de realizar acciones ante las incidencias detectadas en los inventarios de los activos informáticos (11).

OCS Inventory NG 2.0.3

Es una herramienta libre que brinda la posibilidad de realizar reportes de los inventarios realizados. Soporta una amplia gama de plataformas, entre las que se destacan Windows y GNU/Linux. Envía alarmas a través de correo electrónico, pero no es capaz de ejecutar acciones de forma automática. Para la comunicación entre el cliente y el servidor utiliza el protocolo HTTP. La aplicación cliente debe ser configurada por el administrador para que en determinados intervalos de tiempo, se realice un envío de la información obtenida al servidor, y se muestren datos actualizados con los últimos cambios efectuados en

cada ordenador de la red donde esté instalada la herramienta. Utiliza la Licencia Pública General (GPL, en inglés General Public License), versión GNU GPLv2 (12).

CACIC (Configurador Automático y Colector de Informaciones Computacionales) 2.2.2

Es capaz de proporcionar un diagnóstico preciso del ambiente informático como el número de equipos y su distribución en diversos órganos, el tipo de software utilizado y la licencia, las configuraciones de hardware, entre otros. También puede proporcionar información a la propiedad y la ubicación física del equipo, ampliando el control del ambiente computacional y la seguridad de la red. Soporta las plataformas GNU/Linux y Windows. Es capaz de enviar alarmas a los administradores de sistemas cuando se identifiquen cambios en la ubicación física del equipo, y en la configuración de los componentes de hardware de cada uno de los ordenadores (13).

Seguidamente se muestra una tabla comparativa con las características fundamentales de las herramientas descritas anteriormente.

Criterios/ Herramientas	Total Network Inventory 2	Login Inventory	NetSupport DNA	GLPI	OCS Inventory NG	CACIC
Multiplataforma	x	x	x	x	x	x
Privativas	x	x	x	-	-	-
Clientes Windows	-	-	x	-	x	x
Clientes GNU/Linux	-	-	x	-	x	x
Envío de Alarmas	x	-	x	x	x	x
Control Remoto	-	x	x	-	-	-
Exporta a otros formatos	x	x	x	x	x	-

Tabla 1. Comparación de herramientas de gestión de inventarios de hardware y software

Todas las herramientas descritas tienen como propósito fundamental realizar inventarios de hardware y software en una red de ordenadores, cada una con sus características específicas. Se puede destacar que las herramientas estudiadas son multiplataforma, sin embargo, no todas pueden exportar la información obtenida después de la realización de un inventario a diferentes formatos. Otras de las características que presentan estas herramientas, es que solo algunas cuentan con clientes en los sistemas operativos GNU/Linux y Windows; señalando además la posibilidad de envío de alarmas en casos específicos. Solo algunas de estas herramientas realizan el inventario a las computadoras a través de acceso remoto y ejecutan acciones pero no de forma automática.

En la UCI fue desarrollada la aplicación Sistema de Inventario de Hardware y Software (SIHS), una aplicación destinada a la realización de inventarios de los activos informáticos, para las plataformas Windows y GNU/Linux. La misma, cuenta con un módulo web para la administración y configuración del proceso de inventario y un segundo módulo para la recolección de la información. Presenta como ventaja el envío de correo electrónico al detectar una incidencia. Sin embargo, no es capaz de realizar acciones ante incidencias generadas durante los procesos de control de inventarios de hardware y software.

De manera general, todas las herramientas estudiadas son semejantes en cuanto a algunos criterios. Ninguna de las herramientas caracterizadas anteriormente, incluye a la vez, las funcionalidades de enviar alarmas a través de SMS, correo electrónico y mensajes instantáneos, así como realizar acciones ante incidencias de forma automática en los sistemas operativos Windows y GNU/Linux. Por lo tanto, se refleja la necesidad de integrar el MAAI a la plataforma GRHS, con el fin de lograr un sistema capaz de realizar acciones ante las incidencias detectadas por los módulos de inventarios de hardware y software, que a la vez pueda enviar alarmas para notificar las incidencias.

Seguidamente se hará una descripción de las herramientas, metodología y tecnologías a utilizar en el desarrollo de la investigación. Señalando que la metodología de desarrollo, el lenguaje de modelado y de programación, el entorno de desarrollo integrado y el sistema gestor de base de datos, están basados en la selección realizada por el equipo de arquitectura de la plataforma GRHS.

1.3 Metodología de desarrollo. El Proceso Unificado de Desarrollo de Software (RUP)

RUP es una metodología pesada y adaptable al contexto y necesidades de cualquier organización. Es utilizada para el análisis, implementación y documentación de los sistemas orientados a objetos. Además

define varios flujos de trabajo, que se distinguen en dos grupos: los de proceso (Modelamiento del negocio, Requerimientos, Análisis y Diseño, Implementación, Prueba y Despliegue), y los de apoyo (Gestión de la configuración y cambios, Gestión de proyecto y Ambiente).

RUP está dirigido por casos de uso, posibilitando que el proceso de desarrollo del MAAI siga una guía, avanzando a través de una serie de flujos de trabajo que parten de los casos de uso. La inestabilidad del personal en el grupo de desarrollo de la plataforma GRHS requiere de una correcta y detallada documentación del proceso de desarrollo, pues guían las actividades a realizar incluyendo el diseño, la implementación y las pruebas del sistema. Permite contar con el más mínimo detalle de las versiones preliminares, que serán usadas posteriormente al crear nuevas versiones del producto. Que sea una metodología centrada en la arquitectura, constituye las bases del sistema a desarrollar, permitiendo tener una vista del diseño completo del MAAI, resaltando las características fundamentales de los elementos más significativos del módulo. Al ser iterativo e incremental, destaca la división del trabajo en partes más pequeñas o mini-proyectos, pudiendo determinar si han surgido nuevos requisitos o han cambiado los existentes cuando termina cada iteración en el desarrollo del módulo. Esta característica permite que con el tiempo, al MAAI se le puedan ir adicionando nuevas funcionalidades en cada una de las iteraciones, así como refinando las funcionalidades ya existentes (14).

1.4 Lenguaje de Modelado Unificado (UML) 2.0

UML es un lenguaje gráfico para visualizar, especificar, construir y documentar cada una de las partes que comprende el desarrollo de software. Permite documentar los componentes de un sistema de software (arquitectura, requisitos, diseño, pruebas, versiones, planificación, entre otros). Modela con tecnología orientada a objetos todos aquellos elementos que configuran la arquitectura de un sistema de información, y por ende, los procesos de negocio y funciones de sistema, además de brindar formas concretas como son escribir clases en un lenguaje determinado, esquemas de base de datos y componentes de software reusables.

UML es el lenguaje de modelado propuesto por la metodología RUP. La investigación en curso requiere de un proceso de desarrollo con documentación correctamente fundamentada, y que en iteraciones posteriores sea bien interpretado todo el trabajo realizado. Este lenguaje de modelado permite expresar gráficamente el funcionamiento del módulo, de manera que cualquier persona capacitada pueda

entenderlo. Especifica las características del MAAI antes de su construcción. Detalla los artefactos y a partir de los modelos especificados permite construir el módulo. Los propios elementos gráficos sirven como documentación del MAAI para futuras versiones (15).

1.5 Visual Paradigm para UML 8.0

Es una herramienta que permite construir diagramas UML, como son los flujos de eventos del sistema, las clases, todo lo que es documentación tanto de desarrollo como procesos de negocio. Visual Paradigm soporta el ciclo de vida completo de RUP: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. El software de modelado UML ayuda a una construcción más ágil de aplicaciones de calidad a un menor coste. Visual Paradigm es útil en el desarrollo del sistema, pues utiliza UML como lenguaje de modelado y puede ser empleada en sistemas operativos GNU/Linux y Windows, integrándose con Python, y PostgreSQL. Además produce la documentación del sistema en diferentes formatos, por ejemplo: *.pdf, *.html (lenguaje de marcas de hipertexto, en inglés Hyper Text Markup Language). Sus soluciones posibilitan documentar el MAAI para facilitar la comunicación entre el equipo de desarrollo (16).

1.6 Lenguaje de Programación. Python 2.7

Python es un lenguaje interpretado que incluye programación orientada a objetos y ofrece una manera sencilla de desarrollar programas con componentes reutilizables. Elimina preocupaciones referentes a detalles de bajo nivel, como manejar la memoria empleada por el programa, permitiendo a los desarrolladores centrarse más en el código que en la sintaxis del programa. Al ser un lenguaje multiplataforma permite ser utilizado en la implementación de las funcionalidades del MAAI tanto para GNU/Linux como para Windows. Dado que presenta una biblioteca estándar con gran cantidad de funcionalidades, disminuyendo el uso de bibliotecas externas, de acuerdo a las necesidades del módulo. En cualquier plataforma que exista el intérprete de Python es posible extender la aplicación, por si en un futuro se desea utilizar el MAAI como parte de una aplicación informática. El código legible de Python permitirá una clara comprensión del sistema, facilitando el mantenimiento y la corrección del módulo. Además es un lenguaje dinámico, pues en tiempo de ejecución se le puede modificar el comportamiento de objetos y clases, siendo útil en el registro de plugins en la arquitectura PCA que se abordará posteriormente (17).

1.7 Entorno de Desarrollo Integrado Eclipse Helios 3.6

La plataforma Eclipse es un IDE (en inglés, Integrated Development Environment) multiplataforma y extensible. Brinda soporte al lenguaje de programación Python. Estructura los proyectos en directorios y recursos, lo que permite establecer una organización jerárquica. Al ser multiplataforma posibilita su uso en el desarrollo del MAAI en los sistemas operativos GNU/Linux y Windows. Eclipse permite la integración con el framework Django y con el servidor web Apache utilizados en el desarrollo del módulo. Además de integrarse con el framework PyUnit utilizado en la fase de pruebas del software. Además existe una extensión para este IDE llamada site que permite el trabajo con el control de versiones subversion utilizado en el desarrollo del módulo (18).

1.8 Framework Django 1.3.1

Django es un marco de desarrollo web para Python, utilizado en configuraciones, archivos y modelo de datos. Permite desarrollar aplicaciones web, reutilizar el código existente y hacer un uso correcto de los componentes de ingeniería. Presenta como característica su escalabilidad, lo que permite manejar el crecimiento continuo de funcionalidades. El framework Django es utilizado para el desarrollo de la aplicación encargada del envío de alarmas, correspondiente al MAAI. Es aprovechada la ventaja del uso de un mapeador de objeto relacional (ORM) para el almacenamiento y acceso de los datos de la información persistente referente a las incidencias y diferentes tipos de mensajes desde el gestor de base de datos PostgreSQL. Además permite el uso de las funcionalidades que brinda para el envío de mensaje a través de correo electrónico (19).

1.9 Servidor web Apache 2.2

Apache es un servidor web que publica aplicaciones web mediante los protocolos HTTP/ HTTPS. Permite que un navegador web pueda enviar solicitudes en paralelo, las cuales ahorran ancho de banda dejando de trasmitir las cabeceras HTTP en cada solicitud. Apache funciona en los sistemas operativos Windows y GNU/Linux (20). Apache puede publicar aplicaciones web escritas en Python mediante el módulo mod_python pudiendo integrar aplicaciones hechas con Django en este servidor web (21). Apache fue seleccionado como servidor web para el desarrollo de la aplicación Django encargada del envío de alarmas a través de mensajes, por las características mencionadas anteriormente. Además que es una

aplicación estable y permite el acceso de muchas aplicaciones clientes accediendo concurrentemente, ahorrando al equipo de trabajo de la gestión de numerables peticiones para el envío de alarmas.

1.10 Sistema Gestor de Base de Datos (SGBD). PostgreSQL 9.1

PostgreSQL es un SGBD objeto-relacional, cuyo código fuente está disponible libremente. Utiliza un modelo cliente-servidor y presenta buen rendimiento al manejar grandes cantidades de datos y una alta concurrencia de usuarios accediendo a la vez al sistema. Posee numerosos tipos de datos, además de los tipos estándares en cualquier base de datos. El SGBD es utilizado por el MAAI para almacenar la información referente a las alarmas enviadas, con el objetivo de tener constancia del envío de mensajes. Su velocidad en cuanto al procesamiento de consultas posibilita una mejor obtención y almacenamiento de los datos de las incidencias y alarmas. No suele perder información ni corromper los datos. Restringe el acceso a los datos, así los usuarios no autorizados no podrán interceptar información confiable, dando mayor seguridad en cuanto a la integridad de la información de la plataforma GRHS (22).

El trabajo con bases de datos requiere de una herramienta para su gestión. PgAdminIII es una interfaz de administración para gestionar bases de datos PostgreSQL. Permite la edición rápida de consultas y soporte para todos los tipos de objetos de PostgreSQL. Además, posee un editor de código para procedimientos y funciones. El uso de PgAdminIII es necesario para la administración de PostgreSQL, permitiendo crear la base de datos a utilizar en el MAAI, así como la gestión de las tablas y datos relacionados con las incidencias y las alarmas en la plataforma GRHS (23).

1.11 Framework PyUnit 2.7

PyUnit es el framework oficial para hacer pruebas unitarias en Python. Se incluye en la biblioteca estándar que facilita la creación y gestión de pruebas en módulos Python. Este framework es soportado por el IDE Eclipse utilizado en el desarrollo del MAAI. Permite separar el código de pruebas del código del propio módulo, realizando pruebas de fallo y devolviendo el resultado correspondiente. Este framework es utilizado por sus características para el desarrollo de las pruebas de unidad del MAAI (24).

1.13 RabbitMQ 2.8.1

RabbitMQ es un software de código abierto para la negociación de mensajes. Cuenta con una distribución independiente de plataformas y ofrece alta confiabilidad, disponibilidad y escalabilidad. Dicha escalabilidad permite mantener la comunicación con los módulos del sistema en el cliente, a través de la cola de mensajes en caso de algún cambio en el lenguaje. Se hace necesario su uso, teniendo en cuenta que consigue que las peticiones enviadas por los módulos de inventario de hardware y software al MAAI, puedan ser atendidas por orden de llegada. Este servidor posibilita el mismo mecanismo de cola de mensajes para diferentes lenguajes de programación, tales como C++, Java, PHP y Perl (25).

1.14 Notación de Objetos JavaScript (JSON en inglés JavaScript Object Notation)

JSON es un formato para la serialización de estructuras de datos y deserialización en un nuevo objeto Python. Está diseñado explícitamente para permitir su uso en diferentes lenguajes de programación. Python incluye un módulo json en su biblioteca estándar, el cual dispone de opciones para codificar la salida de datos con formato apropiado para la lectura. Este formato es utilizado para la serialización de los datos referentes a las incidencias. Cuando los módulos de inventario detectan una incidencia, serializan los datos correspondientes a dicha incidencia y los envían al servidor y al MAAI. En cuanto al MAAI, utiliza este formato para deserializar la información recibida de los módulos de inventario, así como la nueva configuración recibida del servidor. Además, el MAAI serializa la información de los niveles de incidencia que envía al servidor (26).

1.15 Devcon

Es una aplicación que brinda una serie de utilidades basadas en líneas de comandos, que actúa como alternativa al administrador de dispositivos de Windows. Mediante su utilización se puede habilitar, inhabilitar, reiniciar, actualizar, quitar y consultar dispositivos individuales o grupos de dispositivos. Es compatible con el sistema operativo Windows a partir de su versión Microsoft Windows 2000. Esta herramienta es utilizada en el flujo de implementación. Facilita el desarrollo de algunas de las funcionalidades en el cliente que darán cumplimiento a determinados objetivos específicos, como son el control de los puertos USB y de las interfaces de red en Windows (27).

1.16 Ethtool 1.3.2

Es una herramienta para gestionar los parámetros de la tarjeta de red en un sistema GNU/Linux. Se puede utilizar para consultar y cambiar la configuración, tales como la velocidad, auto-negociación y la descarga de suma de comprobación de los dispositivos de red y dispositivos de Ethernet en particular. Esta herramienta es usada para inhabilitar y habilitar las interfaces Ethernet en la ejecución de acciones ante incidencias que sean definidas tanto de software como de hardware (28).

1.17 Xinput 1.5

Es una herramienta para listar los dispositivos de entrada disponibles en el sistema operativo GNU/Linux. Brinda información de consulta acerca de un dispositivo, además de ofrecer la posibilidad de cambiar la configuración del dispositivo de entrada. Xinput en el desarrollo de la solución es necesario, pues al ser capaz de listar los dispositivos de entrada conectados en la computadora y proporcionar información sobre ellos, ofrece la posibilidad de manipular esos datos. Logra mantener el control del ratón y el teclado en una computadora, permitiendo inhabilitarlos y habilitarlos (29).

1.18 Fswebcam 20120717-1

Es un programa pequeño y rápido para el sistema operativo GNU/Linux. Su función está dada por la captura de imágenes a través de una cámara web, reduciendo el ruido de la imagen tomada. Se encarga de la compresión de la imagen en los formatos *.png (Gráficos de Red Portátiles, en inglés Portable Network Graphics) o *.jpg (en inglés Joint Photographic Experts Group) y guarda la imagen capturada en un archivo especificado. Se hace necesaria la utilización de esta aplicación, en la toma de imágenes con cámara web en el sistema operativo GNU/Linux. Esta funcionalidad tiene como objetivo conocer el usuario que está interactuando con la computadora cuando ocurre una incidencia (30).

1.19 Gammu-smsd

Es un programa que utiliza el módem de Sistema Global para las comunicaciones Móviles (GSM, en inglés Global System for Mobile communications) para verificar los mensajes recibidos y enviar mensajes ubicados en la cola de almacenamiento. La utilidad de línea de órdenes Gammu provee acceso a una amplia variedad de características en el sistema operativo GNU/Linux, entre las que soporta el listado,

inicio y manejo de llamadas. También permite la recuperación, copia de respaldo y envío de SMS. Gammu es utilizado en el desarrollo del módulo para el envío de alarmas a través de SMS, debido a las diferentes características especificadas anteriormente (31).

1.20 Conclusiones parciales

En el presente capítulo, se realizó un análisis de conceptos y definiciones relacionadas con el estudio de las tendencias actuales de herramientas que realizan inventario de hardware y software. Dichas herramientas permiten la obtención de información de hardware y software en una red de ordenadores, pero reflejan como deficiencia que no todas son capaces de enviar alarmas y tomar acciones ante incidencias de manera automática. Fueron definidas las herramientas, metodología y tecnologías a utilizar en el transcurso de la investigación. Este estudio previo brinda la posibilidad de lograr un módulo con la calidad requerida.

CAPÍTULO 2: CARACTERÍSTICAS DEL SISTEMA

2.1 Introducción

En el presente capítulo se muestra y describe el modelo de dominio, teniendo como objetivo fundamental comprender su estructura. Se identificarán y definirán las funcionalidades que deberá desarrollar el módulo, quedando agrupadas en los requerimientos funcionales, así como las propiedades o cualidades descritas en los requerimientos no funcionales. Los casos de uso (CU) agruparán las funcionalidades del módulo y serán representados en los diagramas correspondientes, con una explicación detallada de su funcionamiento en las descripciones de casos de uso. Estas especificaciones permitirán obtener un enfoque lo más detallado posible y una mejor comprensión del módulo a desarrollar.

2.2 Objeto de Automatización

Las funcionalidades a realizar en el MAAI tienen como propósito automatizar la realización de acciones ante incidencias detectadas por los Módulos de Inventario (MI) y el envío de alarmas. Como procesos a automatizar se tienen:

- Capturar imagen del usuario y de la pantalla de la computadora que utiliza el usuario.
- Inhabilitar, monitorizar y habilitar el ratón, el teclado, los puertos USB y las interfaces de red Ethernet de la computadora donde ocurrió la incidencia.
- Suspender, reiniciar, apagar e hibernar la computadora.
- El envío de SMS, correo electrónico y mensajes instantáneos cuando ocurra una incidencia.

2.3 Propuesta de Solución

Existe la necesidad de incluir funcionalidades para realizar acciones y enviar alarmas ante las incidencias detectadas en los inventarios de activos informáticos. Por tanto, se propone como solución, integrar a la plataforma GRHS un módulo capaz de cumplir con estas funcionalidades de forma automática. Dicha plataforma se enmarca en los procesos de análisis, captura y consulta de la información del hardware y software correspondiente a una red de computadoras. La plataforma GRHS está concebida en una arquitectura cliente-servidor.

CAPÍTULO 2: CARACTERÍSTICAS DEL SISTEMA

La aplicación cliente se ocupa de la realización de inventarios de hardware y software, así como de la ejecución de acciones ante las incidencias detectadas en los inventarios. En este documento se hará referencia como cliente a dicha aplicación que se encuentra en las estaciones de trabajo a controlar en una red de computadoras. La plataforma GRHS en el cliente está compuesta por los MI, el MAAI y el Servicio de Administración de Recursos y Acciones (SARA). Los MI son los responsables de realizar los inventarios de los activos informáticos y detectar las incidencias. El MAAI es el encargado de la realización de acciones ante las incidencias detectadas por los MI. El SARA es el núcleo del sistema en el cliente, gestionando y comunicando la información del MAAI y de los módulos que realizan el inventario de hardware y software. Además, el SARA es el puente para establecer la comunicación entre el cliente y el servidor de la plataforma GRHS.

Tras detectar una incidencia en los inventarios realizados, los MI envían directamente dicha incidencia al MAAI. Esta incidencia se clasifica por un nivel, el cual es considerado como una categoría que se le proporciona a una incidencia. Dicho nivel de incidencia tiene asociado una o varias acciones a realizar y un estado activo o no activo que define si las acciones asociadas al nivel se están ejecutando. Las acciones a realizar por el MAAI en el cliente son: inhabilitar ratón, teclado, puertos USB e interfaces de red Ethernet, hibernar, reiniciar, apagar y suspender la computadora, capturar una imagen del usuario y una imagen de la pantalla de la computadora que está utilizando dicho usuario. Estas acciones se dividen en tres grupos:

- Acciones que se monitorizan: Son las acciones que se ejecutan a partir de la incidencia recibida por los MI y solo pueden ser revertidas a través de SARA. El término monitorizan, se refiere al chequeo constante de la acción para garantizar el estado activo del nivel de la incidencia.
- Acciones especiales: Son aquellas que no necesitan ser monitorizadas. Algunas son realizadas al recibir una incidencia.
- Acciones de control: No están asociadas a ningún nivel de incidencia.

A continuación se muestra una tabla, estableciendo las relaciones entre las definiciones esclarecidas anteriormente:

Acciones/ Relación	Hibernar	Reiniciar	Apagar	Suspender	Red	Teclado	Ratón	USB	Imagen usuario	Imagen pantalla
Especiales	x	x	x	-	-	-	-	-	x	x
Control	x	x	x	-	-	-	-	-	-	-
Monitorizan	-	-	-	x	x	x	x	x	-	-
Revertir Usuario	x	x	x	-	-	-	-	-	-	-
Revertir Automático	-	-	-	-	-	x	x	x	-	-
Revertir administra-dor	-	-	-	x	x	-	-	-	-	-
Ejecuta MI	-	-	-	x	x	x	x	x	x	X
Ejecuta SARA	x	x	x	-	-	-	-	-	-	-

Tabla 2. Acciones del MAAI

El servidor, por su parte, es el encargado de gestionar la información referente a los inventarios y las incidencias, además de enviar las alarmas. La plataforma GRHS en el servidor está compuesta por el Módulo de Alarmas y Acciones ante Incidencias en el servidor (SMAAI), el Módulo de Administración de Recursos de Hardware, Software, Incidencias y Alarmas (MAHSIA), y el Módulo de Administración de Operaciones de Inventarios (MAOI). El SMAAI es el responsable del envío de alarmas ante las incidencias recibidas por el MAOI. El MAHSIA es el encargado de la administración del sistema. El MAOI gestiona la comunicación entre MAHSIA y los MI, almacenando la información referente a los inventarios. El MAOI, al recibir una incidencia de los MI a través del SARA, envía dicha incidencia al SMAAI. Esta incidencia se clasifica por un nivel, el cual tiene asociado una o varias alarmas a enviar. Los mensajes de alarmas a enviar por el SMAAI son: correo electrónico, SMS y mensajes instantáneos.

De manera general, como se ha explicado anteriormente, el MAAI está presente en el cliente y en el servidor de la plataforma GRHS. El MAAI se ocupa de la gestión y control de las acciones en el cliente, mientras que garantiza el envío de alarmas en el servidor. Sin embargo, estos módulos pertenecientes al MAAI, no se encuentran relacionados entre sí, ya que utilizan otros módulos para la ejecución de sus responsabilidades.

En la siguiente figura se muestran las relaciones entre los diferentes componentes que integran la plataforma GRHS, destacando que los componentes de color rojo son los correspondientes al MAAI.

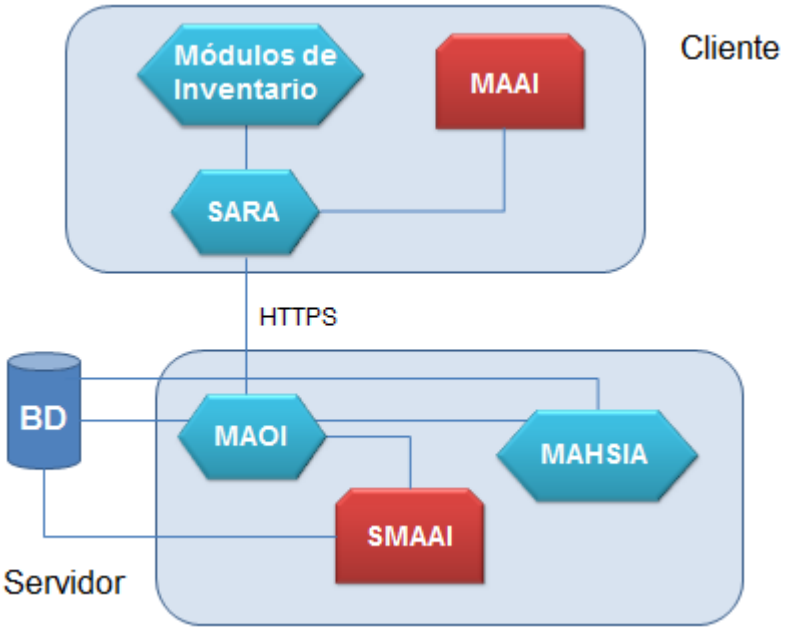

Figura 1. Propuesta de solución del MAAI, a partir de la plataforma GRHS

Después de una descripción de la propuesta de solución para el desarrollo del MAAI, se procede a realizar el modelo de dominio.

2.4 Modelo de Dominio

El modelo del dominio toma los tipos más importantes de objetos en el ámbito del sistema. *Los objetos del dominio representan las cosas que existen o los eventos que suceden en el entorno en el que trabaja el sistema.* El modelo del dominio se describe a través de diagramas UML, especialmente diagramas de clases (32). Seguidamente se muestra el modelo de dominio del MAAI:

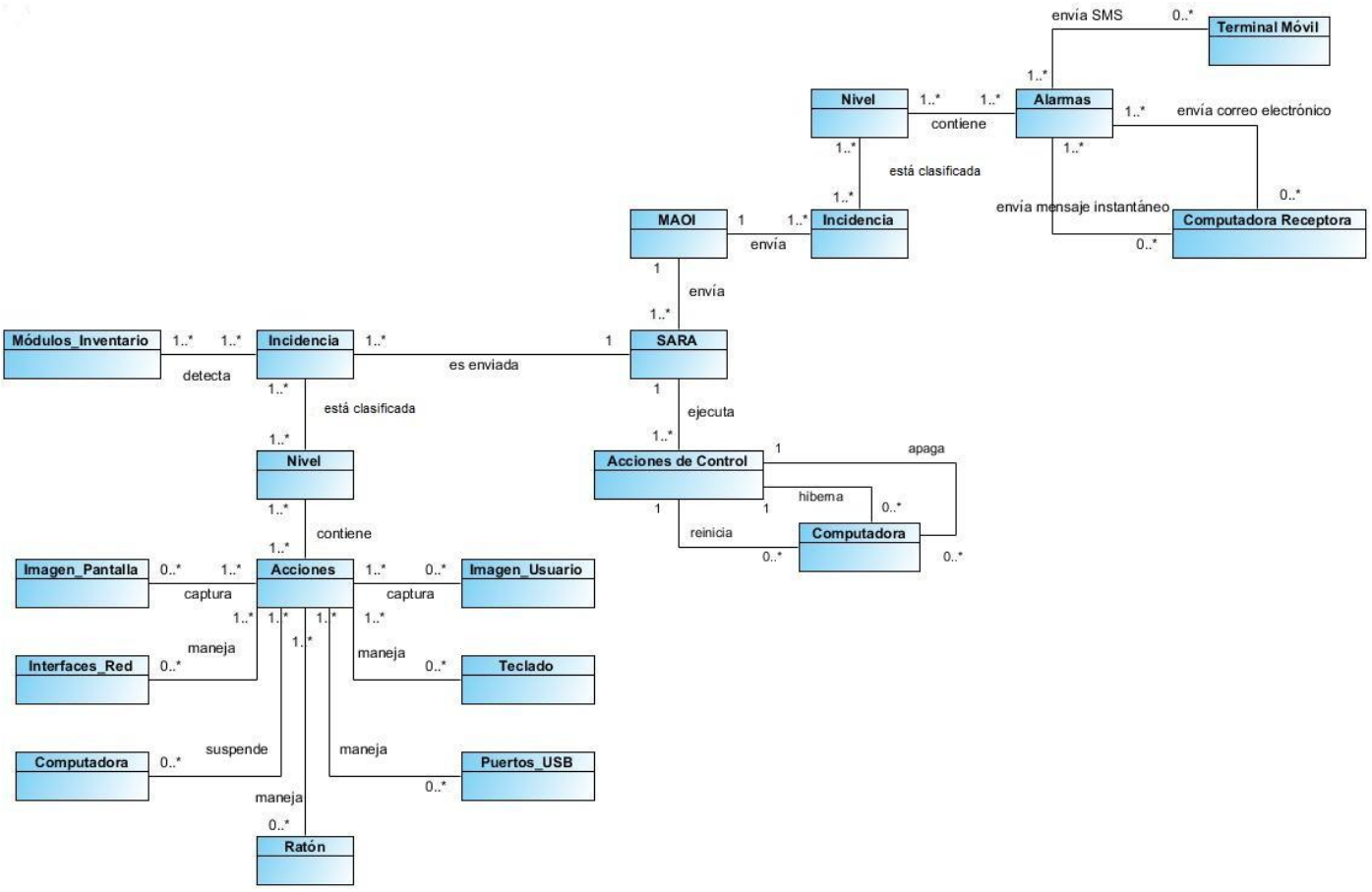

Figura 2. Modelo de Dominio

Módulos_Inventario: representa los módulos que realizan el inventario del hardware y software en los sistemas operativos Windows y GNU/Linux.

Incidencia: constituye la ocurrencia de un acontecimiento no permitido detectado por el inventario de hardware y software en una red de ordenadores.

Nivel: categoría que se le asigna a una incidencia y que contiene acciones o alarmas.

Acciones: son las acciones a realizar de acuerdo a un nivel de incidencia.

Imagen_Pantalla: es una imagen que se captura de la pantalla de la computadora donde ocurrió la incidencia.

Imagen_Usuario: es una imagen que se captura del usuario que está utilizando la computadora donde ocurrió la incidencia.

Puertos_USB: se controlan los puertos USB de la computadora donde ocurrió la incidencia.

Interfaces_Red: se controlan las interfaces de red Ethernet de la computadora donde ocurrió la incidencia**.**

Teclado: se controla el teclado de la computadora donde ocurrió la incidencia.

Ratón: se controla el ratón de la computadora donde ocurrió la incidencia.

Computadora: es el ordenador utilizado por el usuario que se suspende, reinicia, hiberna o apaga.

SARA: es el encargado de la comunicación en el cliente, sirve de pasarela entre el MAOI y los MI.

Acciones de Control: son las acciones ejecutadas por el SARA sin tener en cuenta un nivel de incidencia.

MAOI: gestiona la información de las incidencias y los inventarios en el servidor.

Alarmas: son las alarmas a enviar de acuerdo a un nivel de incidencia.

Terminal Móvil: es el teléfono celular que recibe el mensaje SMS.

Computadora Receptora: es la computadora que recibe el mensaje instantáneo o el correo electrónico.

Los Módulos de Inventario detectan la presencia de una incidencia. Esta incidencia se clasifica en un nivel, que a su vez en el cliente, tiene asociado un conjunto de acciones a realizar. Las acciones a realizar están relacionadas con el control del mouse, el teclado, los puertos USB y las interfaces de red Ethernet, además de capturar imágenes y suspender la computadora donde fue detectada la incidencia. Dicha incidencia detectada por los MI es enviada al MAOI a través del SARA, siendo clasificada por un nivel. En el caso del servidor el nivel de la incidencia tiene asociados diferentes alarmas a enviar: mensajes instantáneos, mensajes a través de SMS y de correo electrónico. Los mensajes instantáneos y los mensajes de correo electrónico son enviados a una computadora, mientras que los mensajes a través de SMS son enviados a un terminal móvil. El MAOI envía al SARA la petición de ejecutar las acciones de control: hibernar, reiniciar o apagar la computadora del usuario. A partir del modelo de dominio, son especificados los requerimientos del sistema que serán descritos seguidamente.

2.5 Especificación de los requerimientos del sistema

Los requerimientos son la descripción de los servicios proporcionados por un sistema, así como sus restricciones operativas. Reflejan las necesidades de los clientes de un sistema que ayude a resolver algún problema como el control de un dispositivo, hacer un pedido o encontrar información (33). La especificación de requisitos o requerimientos de software son descripciones completas del comportamiento del sistema que se va a desarrollar, incluyendo un conjunto de casos de uso que describen todas las interacciones que tendrán los actores con el software (34).

2.5.1 Requerimientos funcionales

Los requerimientos funcionales son una descripción de las necesidades de un producto, establecen los comportamientos del sistema y además, se estructuran de forma natural mediante casos de uso (35). A continuación se muestran los requerimientos funcionales con los que debe cumplir el sistema:

RF1. Recibir incidencia: Consiste en recibir una incidencia enviada por los MI, la cual se caracteriza por un nivel, un tipo de incidencias, un componente y una descripción.

RF2. Inhabilitar las interfaces de red Ethernet: Inhabilita y monitoriza el uso de las interfaces de red Ethernet a partir de la incidencia recibida.

RF3. Inhabilitar dispositivos: Impide el uso de los dispositivos a partir de la incidencia recibida

RF3.1. Inhabilitar ratón: Inhabilita y monitoriza el funcionamiento del ratón, impidiendo su uso.

RF3.2. Inhabilitar teclado: Inhabilita y monitoriza el funcionamiento del teclado, impidiendo su uso.

RF4. Inhabilitar los puertos USB: Inhabilita y monitoriza el uso de los puertos USB a partir de la incidencia recibida.

RF5. Controlar la computadora del usuario: Mantiene un control sobre la computadora del usuario.

RF5.1. Reiniciar la computadora: Reinicia la computadora del usuario.

RF5.2. Apagar la computadora: Apaga la computadora del usuario.

RF5.3. Hibernar la computadora: Hiberna la computadora del usuario.

RF5.4. Suspender la computadora: Suspende la computadora del usuario donde ocurrió la incidencia e inhabilita y monitoriza las sesiones de trabajo de los usuarios.

RF6. Obtener imagen: Obtiene una imagen como evidencia a partir de la incidencia recibida.

RF6.1. Obtener la imagen de la pantalla: Captura una imagen de la pantalla de la computadora del usuario donde ocurrió la incidencia y la envía al servidor.

RF6.2. Obtener la imagen del usuario: Captura una imagen del usuario que esté utilizando la computadora donde ocurrió la incidencia y la envía al servidor.

RF7. Actualizar configuración: Actualiza la configuración del MAAI a partir de la configuración recibida por el servidor.

RF8. Leer niveles de incidencia: Obtiene los niveles de incidencia almacenados en el fichero de configuración de niveles de incidencia del MAAI, y lee las acciones a realizar por cada nivel.

RF9. Habilitar los dispositivos: Cambia los dispositivos a su estado normal.

RF9.1. Habilitar el ratón: Habilita el ratón, permitiendo su uso.

RF9.2. Habilitar el teclado: Habilita el teclado, permitiendo su uso.

RF10. Habilitar los puertos USB: Habilita el uso de los puertos USB de la computadora.

RF11. Habilitar usuarios de la computadora: Permite el acceso de los usuarios locales, administradores y del dominio a la computadora.

RF12. Enviar alarmas: Envía alarmas de incidencias a partir del identificador de incidencia recibido.

RF12.1. Enviar mensaje SMS: Envía mensaje de alarma mediante SMS.

RF12.2. Enviar mensaje de correo electrónico: Envía mensaje de alarma mediante correo electrónico.

RF12.3. Enviar mensaje instantáneo: Envía mensaje de alarma mediante chat.

RF13. Guardar alarmas: Almacena las alarmas enviadas en la base de datos.

2.5.2 Requerimientos no funcionales

Los requerimientos no funcionales son propiedades o cualidades que el producto debe cumplir. Estas propiedades constituyen las características que hacen al producto atractivo, usable, rápido o confiable. En muchos casos dichos requisitos son fundamentales en el éxito del producto (36). Seguidamente se muestran los requerimientos no funcionales con los que debe cumplir el sistema:

Requerimientos de software

RnF1. Características de software en el cliente:

- Sistema operativo Windows XP
- Sistema operativo GNU/Linux Ubuntu 11.10
- Intérprete para Python2.7

RnF2. Características de software en el servidor:

- Sistema operativo Ubuntu 11.10
- Intérprete para Python2.7

Requerimientos de hardware

RnF3. Características del hardware del servidor:

- Procesador: Dual Core CPU 2.2GHz
- Memoria: 1GB
- Tarjeta Madre: Intel-965
- Ancho de Banda: 100 Mbps
- Módem GSM/GPRS (en inglés General Packet Radio Services) Huawei modelo E220.

RnF4. Características del hardware del cliente:

- Procesador: Dual Core CPU 2.2GHz
- Memoria: 1GB
- Tarjeta Madre: Intel-965
- Ancho de Banda: 100 Mbps
- Cámara Web

Restricciones en el diseño

RnF5. La codificación se rige mediante el estilo de codificación TLM-GRHS-0120_55 EstandarPythonv1.0 definido por el arquitecto del proyecto para garantizar un mejor soporte a la plataforma GRHS (37).

Al concluir la captura y especificación de requisitos, se da paso al modelamiento del sistema.

2.6 Modelo de casos de uso del sistema

El modelo de CU ayuda al cliente, a los usuarios y a los desarrolladores a llegar a un acuerdo sobre el funcionamiento del sistema. Cada actor utiliza el sistema al interactuar con los CU (definido más adelante), representando gráficamente el modelo de CU (38). En la siguiente tabla se reflejan los diferentes actores del sistema.

2.6.1 Actores del sistema

Actor	Objetivo
SARA	Módulo que gestiona y comunica la información de los módulos que realizan el inventario de hardware y software en una red de computadoras.
MAOI	Aplicación Django que gestiona la comunicación entre Módulo de Administración y Módulos en los Agentes (clientes), y almacena la información referente a los inventarios.
Módulos de Inventario	Módulo que ejecuta los inventarios de hardware y software.

Tabla 3. Descripción de los actores del sistema

Después de una descripción de los actores del sistema se representan los casos del uso del sistema y su relación con estos actores.

2.6.2 Casos de Uso del Sistema

Los CU son artefactos narrativos que describen el comportamiento del sistema desde el punto de vista del usuario. Los CU proporcionan, además, uno o más escenarios que indican cómo debería interactuar el sistema con el usuario o con otro sistema para conseguir un objetivo específico. El modelo de CU captura todos los requisitos funcionales del sistema (39). A continuación se muestra el diagrama de casos de uso del sistema:

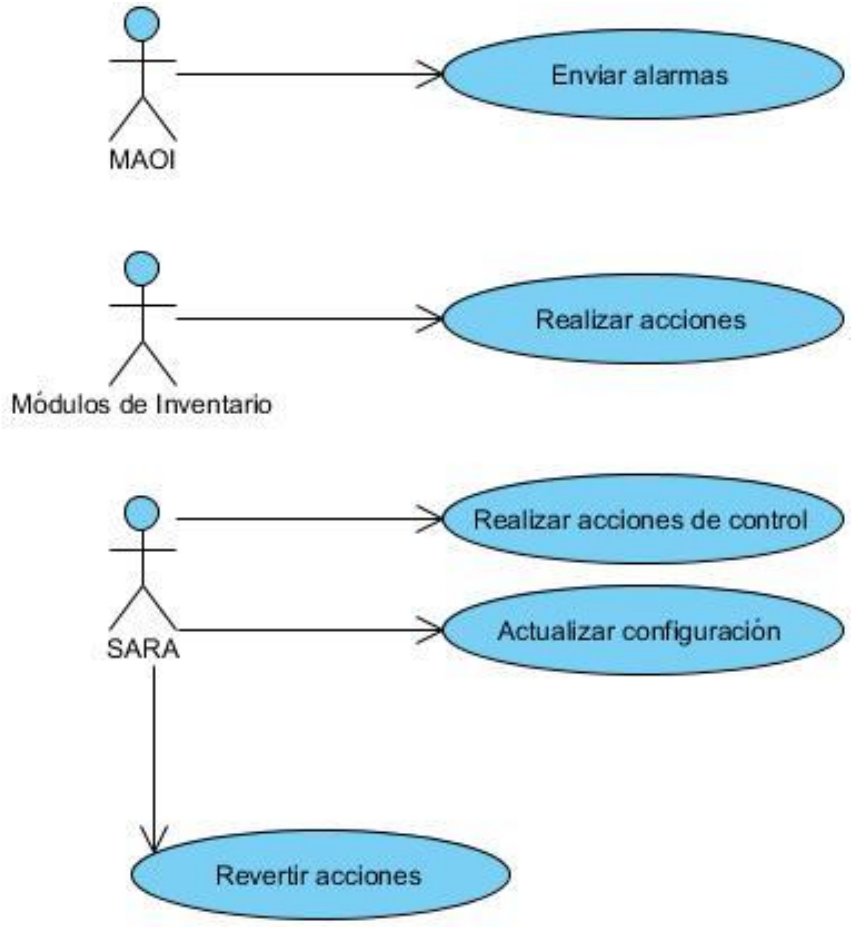

Figura 3. Diagrama de casos de uso del sistema

Los casos de uso del sistema se dividen en:

CU- 1 Enviar alarmas.
CU- 2 Realizar acciones.
CU- 3 Actualizar configuración.
CU- 4 Realizar acciones de control.
CU- 5 Revertir acciones.

Tomando en cuenta que el CU Realizar acciones es considerado como crítico se hará una descripción detallada del mismo. Los CU restantes se pueden encontrar en el Anexo 1. Este CU será utilizado como muestra en todo el documento.

2.6.3 Descripción detallada del caso de uso Realizar acciones.

Objetivo	El objetivo del caso de uso es realizar acciones ante las incidencias recibidas por los módulos de inventario.
Actores	Módulos de Inventario: (Inicia) Después de realizado el inventario, envía la incidencia detectada al MAAI.
Resumen	A partir de la incidencia recibida se realizan acciones como la captura de imagen del escritorio del usuario, la captura de imagen del usuario, suspender la computadora del usuario y bloquear el ratón, el teclado, los puertos USB y las interfaces de red Ethernet. Posteriormente se envían los niveles de incidencia activos al servidor, como prueba de su estado.
Complejidad	Alta
Prioridad	Crítico
Precondiciones	Se ha almacenado la configuración del MAAI en la caché. Se han ejecutado las acciones correspondientes a los niveles en estado activo. Se ha detectado una incidencia.
Postcondiciones	Se realizaron las acciones correspondientes al nivel de la incidencia recibida.

Sección “Realizar acciones”		
Flujo Normal de Eventos		
	Actor	**Sistema**
1.	Envía una incidencia.	
2.		Recibe una incidencia de los módulos de inventario.
3.		Obtiene la configuración almacenada en la caché.
4.		Verifica el nivel de la incidencia recibida en la configuración del MAAI almacenada en la caché.
5.		Comprueba cada una de las acciones a realizar dentro del nivel de la incidencia recibida.
6.		Verifica si la acción a realizar está incluida dentro de

		las acciones especiales.
7.		Ejecuta la acción correspondiente: • Ver sección Capturar imagen del usuario. • Ver sección Capturar imagen de la pantalla del usuario. Nota: Se ejecuta una u otra sección de acuerdo a la acción que se está verificando.
8.		Verifica si la acción no está en ejecución y no es una acción especial.
9.		Ejecuta la acción correspondiente: • Inhabilitar ratón. • Inhabilitar teclado. • Inhabilitar interfaces de red Ethernet. • Inhabilitar puertos USB. • Ver sección Suspender la computadora del usuario. Nota: Se ejecuta una u otra acción de acuerdo a la acción que se está verificando.
10.		Actualiza en la configuración del MAAI almacenada en la caché, el estado del nivel de la incidencia recibida.
11.		Envía al servidor una lista con los niveles activos registrados en la configuración del MAAI almacenada en la caché.
12.		Termina el CU.
Flujos Alternos		
1. No se puede establecer conexión con la computadora para el envío de la incidencia.		
	Actor	**Sistema**
1a1.		Se guarda una traza notificando que no se puede establecer conexión con la computadora para el

		envío de la incidencia.
1a2.		Ir al paso 11.
Flujos Alternos		
4. El nivel de la incidencia recibida no se encuentra registrado dentro de la configuración del MAAI almacenada en la caché.		
	Actor	**Sistema**
4a1.		Se guarda una traza notificando que el nivel de la incidencia recibida no se encuentra registrado en la configuración del MAAI almacenada en la caché.
4a2.		Ir al paso 11.
Flujos Alternos		
6. La acción a realizar no es una acción especial.		
	Actor	**Sistema**
6a1.		Ir al paso 8.
Flujos Alternos		
8. La acción está en ejecución o la acción es una acción especial.		
	Actor	**Sistema**
8a1.		Ir al paso 10.
Sección “Capturar imagen del usuario”		
Flujo Normal de Eventos		
	Actor	**Sistema**
1.		Captura la imagen del usuario.
2.		Envía al servidor la imagen tomada con la fecha y hora actualizada.
Flujos Alternos		

1. El MAAI no encuentra la cámara web para capturar la imagen.		
	Actor	**Sistema**
1a1.		Se guarda una traza notificando que no se encuentra la cámara web.
Flujos Alternos		
2. El MAAI no encuentra una conexión con el servidor.		
	Actor	**Sistema**
2a1.		No se puede enviar al servidor la imagen tomada al usuario.
2a2.		Se guarda una traza notificando que no se encuentra una conexión con el servidor.
Sección "Capturar imagen de la pantalla del usuario"		
Flujo Normal de Eventos		
	Actor	**Sistema**
1.		Captura la imagen de la pantalla del usuario.
2.		Envía al servidor la imagen tomada con la fecha y hora actualizada.
Flujos Alternos		
2. No puede establecer conexión con el servidor.		
	Actor	**Sistema**
2a1.		No se puede enviar al servidor la imagen tomada a la pantalla del usuario.
2a2.		Se guarda una traza notificando que no se encuentra una conexión con el servidor.
Sección "Suspender la sesión del usuario"		
Flujo Normal de Eventos		
	Actor	**Sistema**

1.		Suspende la sesión del usuario.
2.		Inhabilita el acceso a la computadora de todos los usuarios, excluyendo a los usuarios registrados como permitidos en la configuración del MAAI.

Tabla 4. Descripción detallada del CU Realizar Acciones

2.7 Conclusiones parciales

En el presente capítulo se realizaron los modelos de dominio con UML, sentando las bases para la captura de requisitos y posibilitando una mejor comprensión de los mismos. Se presentaron los requisitos funcionales y no funcionales; así como los casos de uso del sistema. Fue estructurado el modelo de casos de uso del sistema y su descripción, constituyendo una entrada importante para desarrollar el siguiente flujo de trabajo.

CAPÍTULO 3: DISEÑO E IMPLEMENTACIÓN DEL SISTEMA

3.1 Introducción

En el presente capítulo se muestra una vista general de la arquitectura del MAAI. Se especifican los patrones de diseño a utilizar de acuerdo a las necesidades del módulo, sentando las bases para su posterior implementación. Se representa además la estructura del sistema a partir de las funcionalidades previstas en el capítulo anterior. Se plasma el modelo físico de datos, así como el diagrama de paquetes tanto para el servidor como para el cliente. Son mostradas y descritas las clases del diseño correspondiente a cada diagrama de paquete con sus respectivas relaciones. A continuación se representa y caracteriza la arquitectura del MAAI.

3.2 Arquitectura del sistema

La arquitectura brinda una perspectiva clara del sistema completo, necesaria para controlar el desarrollo. Es necesario una arquitectura que describa los elementos del modelo que son más importantes para el módulo en cuestión, así guían el trabajo del equipo de desarrollo (40). A continuación se describe la arquitectura del MAAI.

3.2.1 Arquitectura basada en capas

La arquitectura basada en capas se enfoca en la distribución de roles y responsabilidades de forma jerárquica, proporcionando una forma muy efectiva de separación de responsabilidades. El rol indica el modo y tipo de interacción con otras capas, y la responsabilidad indica la funcionalidad que está siendo desarrollada. El estilo en capas se define 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 capa inmediatamente inferior (41).

Este estilo arquitectónico soporta un diseño basado en niveles de abstracción crecientes. Permite a los desarrolladores la partición de un problema complejo en una secuencia de pasos incrementales. El estilo admite muy naturalmente optimizaciones y refinamientos, además de proporcionar una amplia reutilización. El módulo a desarrollar utiliza este estilo arquitectónico en el cliente, contando con dos capas: la capa de servicios Services y la capa de acceso a datos DataAccess. La capa Services está

compuesta por las acciones a realizar ante una incidencia y agrupa toda la lógica de las funcionalidades del módulo. La capa DataAccess contiene todos los componentes (en inglés plugins) especializados en la implementación de diferentes funcionalidades para la realización de acciones.

3.2.2 Arquitectura basada en componentes

Una arquitectura basada en componentes describe una aproximación de ingeniería de software al diseño y desarrollo de un sistema. Esta arquitectura se enfoca principalmente en la descomposición del diseño en componentes funcionales o lógicos que expongan interfaces de comunicación bien definidas. Un componente es una parte importante, casi independiente y reemplazable de un sistema que satisface una función clara en el contexto de una arquitectura bien definida. En el MAAI los componentes son los elementos que ofrecen un conjunto de servicios o funcionalidades, a través de interfaces definidas (42).

La Arquitectura en Componentes de PyUtilib (PCA, en inglés PyUtilib Component Architecture) permite la implementación de una arquitectura que garantiza la extensibilidad del software en cuanto a la adición de nuevos componentes. Definiendo a un componente (en inglés plugin) como una clase que implementa un conjunto de métodos relacionados en el contexto de una aplicación. Mientras que un servicio (en inglés service) no es más que una instancia de una clase componente; la cual puede ser de tipo componente singleton o no singleton. La clase Interface es un modelo que describe las funcionalidades de los plugins, y la clase plugin incluye las implementaciones correspondientes a la estructura que define una o más interfaces. Mientras que un punto de extensión (en inglés extension point) se define con respecto a una clase interfaz específica, proporcionando de esta forma un mecanismo genérico a aplicaciones para utilizar las funcionalidades implementadas en los servicios.

Este mecanismo soporta un flexible paradigma de programación modular en el cliente del MAAI, permitiendo que la aplicación pueda ser extendida de manera dinámica. Se pueden definir puntos de extensión sin saber cómo serán implementadas las interfaces, y registrar extensiones para los componentes sin necesidad de conocer cómo o dónde son empleados. Esta capacidad facilita la aplicación e inscripción dinámica de componentes dentro del módulo (43). A continuación se muestra el diagrama de arquitectura del MAAI en el cliente.

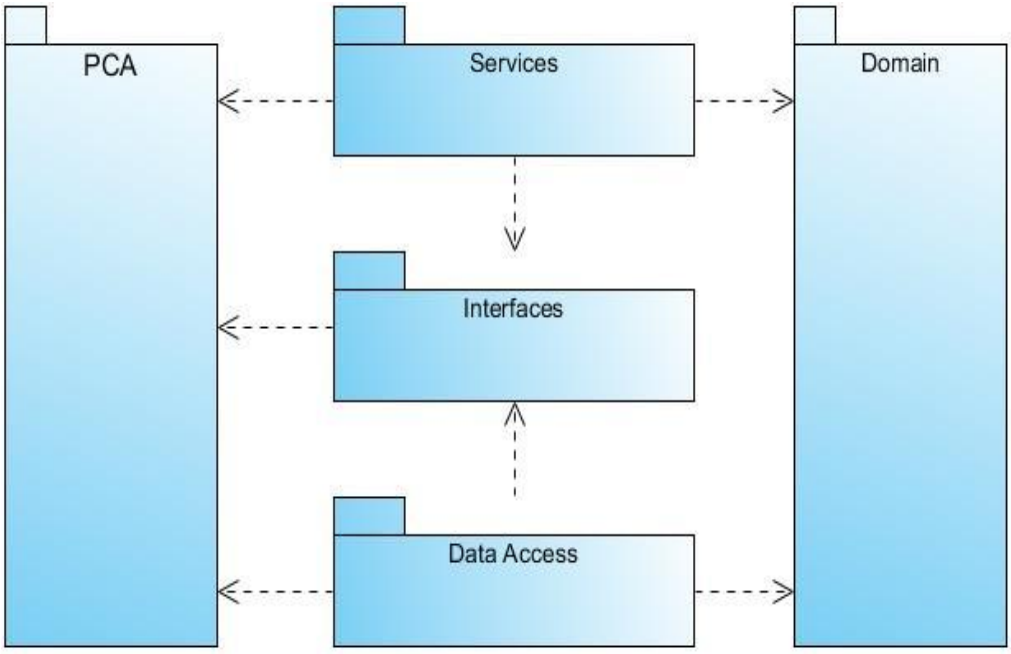

Figura 4. Arquitectura del sistema en el cliente

La figura 4 refleja la arquitectura a utilizarse en el cliente, donde se presenta una vista general de la distribución de las capas y los paquetes que conforman el sistema. La capa de servicios contiene toda la lógica del funcionamiento del módulo. Mientras que la capa de acceso a datos DataAccess contiene todos los componentes especializados en las funcionalidades para la realización de acciones. El paquete de interfaces Interfaces está compuesto por las interfaces del MAAI, que definen los métodos a implementar por los diferentes componentes que integran el módulo. En el paquete de dominio Domain, están incluidas todas las entidades, que no son más que objetos concretos o abstractos que presentan interés para el sistema. Sobre estas entidades se recoge información que será utilizada durante el proceso de ejecución. El paquete PCA incluye las clases de pyutilib.core, que se encuentran distribuidas por las diferentes capas y paquetes dentro de la arquitectura del MAAI en el cliente.

3.2.3 Arquitectura Modelo Vista Plantilla (MTV, en inglés Model-Template-View)

El framework Django se denomina como un framework MTV. La capa de modelos Models administra el comportamiento y contiene todo lo referido a los datos: cómo acceder a ellos, cómo validarlos, qué comportamiento tienen y las relaciones entre ellos. Los modelos responden a requerimientos de información sobre su estado (usualmente formulados desde la vista) y a instrucciones de cambiar el

estado. La capa de plantillas Templates contiene las decisiones relacionadas con la presentación, cómo debería mostrarse algo en una página Web u otro tipo de documento. Mientras que la capa de vistas Views maneja la visualización de la información, describiendo “qué” datos serán presentados y no “cómo” se verán los mismos. La capa Vistas se refiere a la capa de la lógica del funcionamiento del módulo, y contiene el acceso a los modelos (44).

En el caso del MAAI, esta arquitectura se establece únicamente en Modelos y Vistas, debido a que el SMAAI no necesita mostrar ni obtener los datos en un formulario. La comunicación entre SMAAI, MAOI y MAHSIA se realiza mediante la invocación de una URL, utilizando el protocolo HTTP pues son módulos de una aplicación web. El MAOI es el encargado de enviar el identificador de las incidencias al SMAAI. Las vistas interpretan la URL y se sirven de los modelos para la manipulación de la información. El servidor del MAAI se integra con las capas de almacenamiento como la base de datos. A continuación se muestra el diagrama de arquitectura en el servidor.

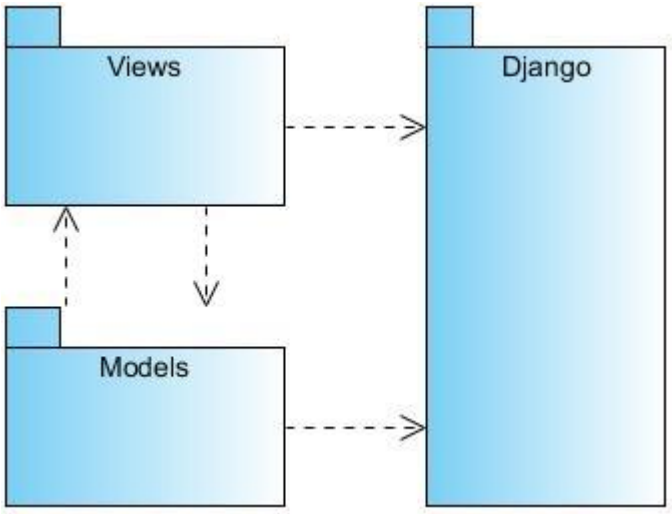

Figura 5. Arquitectura del sistema en el servidor

Después de estar definida la arquitectura, se procede a caracterizar el diseño, describiendo los patrones de diseño utilizados.

3.3 Patrones de diseño

Un patrón de diseño establece una solución estándar para un problema común de programación, siendo una técnica para flexibilizar el código, que a su vez, satisface ciertos criterios. Establece una manera más

práctica de describir algunos aspectos de la organización de un programa, así como conexiones entre componentes de programas. Los patrones de diseño clasifican y describen formas comunes de solucionar problemas frecuentes en el proceso de desarrollo. Su utilización permite ahorrar gran cantidad de tiempo en la construcción de software, siendo más fácil de comprender los diagramas de clases del diseño desarrollados. Los patrones de diseño pueden ser utilizados en diversas situaciones, favoreciendo la reutilización del código, a la hora de realizar determinados cambios en el sistema (45).

Los patrones se clasifican en: patrones de asignación de responsabilidades (GRASP, en inglés General Responsibility Asignment Software Patterns) y los patrones pandilla de los cuatro (GOF, en inglés Gang of Four). A continuación se muestra un análisis de su utilización en el desarrollo de la aplicación.

3.3.1 Patrones GRASP

Los patrones GRASP describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en forma de patrones. Estando estas responsabilidades relacionadas con las obligaciones de un objeto en cuanto a su comportamiento. Los patrones GRASP utilizados en el desarrollo del sistema son los siguientes:

- **Experto**

  Experto es un principio básico que suele utilizarse en el diseño orientado a objetos. El cumplimiento de una responsabilidad requiere a menudo información distribuida en varias clases de objetos, lo que significa que hay muchos expertos "parciales" que colaboran en la tarea.

  Se utiliza en todo el desarrollo de la aplicación, delegando las responsabilidades a los componentes que poseen la información necesaria para realizar una determinada acción, conservando el encapsulamiento. Por ejemplo: PluginKeyboardWindows es el encargado de manipular las acciones a realizar referentes al teclado en el sistema operativo Windows, mientras que PluginMouseLinux se ocupará de las acciones del ratón en GNU/Linux.

- **Creador**

El patrón Creador, como lo indica su nombre, se encarga de la creación de objetos, tarea muy frecuente en los sistemas orientados a objetos. El propósito fundamental de este patrón es encontrar un creador que debemos conectar con el objeto producido en cualquier evento.

Se utilizará en el desarrollo de la aplicación, evidenciándose en la clase ManagerIncidence donde se crea una instancia de la clase ManagerActions, con el objetivo de ejecutar las acciones ante las incidencias. Este patrón brinda soporte a un bajo acoplamiento, que se describirá a continuación, lo cual supone menos dependencias respecto al mantenimiento y mejores oportunidades de reutilización.

- **Bajo acoplamiento**

El acoplamiento es una medida de la fuerza con que una clase está conectada a otras clases, en otras palabras significa que una clase no depende de muchas clases a la hora de realizar determinada función, sino que puede ejecutarse de manera independiente.

Este patrón se evidencia en las relaciones entre las clases del negocio, como ManagerActions, y las clases de acceso a datos, como PluginUsbLinux y PluginMouseWindows, permitiendo un diseño de clases más independientes. Reduce el impacto que pueda tener algún cambio realizado en cualquier clase, posibilitando que las clases sean más reutilizables y fáciles de entender por separado.

- **Alta cohesión**

Una alta cohesión caracteriza a las clases con responsabilidades estrechamente relacionadas que no realicen un trabajo enorme, es decir, define a las clases que agrupan funciones relacionadas o similares, evitando la sobrecarga de trabajo.

El patrón alta cohesión está presente en la clase ManagerActions, la cual tiene como responsabilidad las acciones ante las incidencias detectadas; y en la clase ManagerIncidence, encargada del manejo y control de las incidencias. La división del trabajo entre estas clases incrementa la claridad y la facilidad con que se entiende el diseño; además de simplificar el mantenimiento y las mejoras en funcionalidad.

- **Controlador**

  Un Controlador es un objeto de interfaz no destinada al usuario que se encarga de manejar un evento del sistema. Definiendo además el método de su operación. El uso de este patrón responde al problema de: ¿Cómo lograr atender un evento del sistema?

  La clase ManagerIncidence se emplea como controladora. Esta clase es la encargada del control y manejo de las incidencias. Es válido aclarar que no se le debe asignar mucha responsabilidad a la clase controladora. Normalmente, un controlador debería delegar el trabajo que necesita ser realizado a otros objetos, coordinando las actividades (46).

3.3.2 Patrones GOF

Los patrones GOF presentan tres tipos de clasificaciones, las cuales son utilizadas en dependencia del propósito que se quiere alcanzar, estas son:

Patrones de Creación: Tratan la creación de instancias, y se abstraen a la forma en que los objetos son creados, de manera que permite tratar las clases a implementarse de forma genérica, y sin considerar las clases que serán desarrolladas ni la forma en que se hará (47).

- **Patrones Estructurales:** Tratan la relación entre clases, la combinación clases y la formación de estructuras más complejas (48).
- **Patrones de Comportamiento:** Tratan la interacción y la cooperación entre clases u objetos (49).

En el diseño del MAAI fueron utilizados los patrones GOF explicados a continuación:

Patrón de Creación

- **Singleton**

  Es un modelo que garantiza que solo exista una instancia y que todos puedan acceder a ella. Para esto en lugar de tener una variable global, la instancia se almacena en un atributo estático de la clase (50).

Las clases componentes del MAAI implementan el patrón Singleton, esta estructura se define como un Plugin Singleton, el cual construye un único servicio. A su vez permite a las restantes clases definir solo una instancia de este objeto. Brinda un mayor control, especialmente sobre el número de instancias, en la aplicación se evidencia este patrón en cada componente que cumple con un requisito funcional determinado, por ejemplo: en el plugin_usb, el plugin_mouse y el plugin_keyboard.

Patrón Estructural

- **Fachada (en inglés Facade)**

El patrón Fachada tiene como propósito brindar una interfaz unificada para un conjunto de interfaces de un subsistema, definiendo una interfaz de alto nivel que hace que el subsistema sea más fácil de usar. Este estructura un sistema en subsistemas, lo cual ayuda a reducir la complejidad (51).

La utilización de este patrón en el diseño permite al módulo minimizar las dependencias entre subsistemas, eliminando el acceso directo a los diferentes componentes encargados de las funcionalidades que debe cumplir la aplicación. Dicho acceso se realiza a través de la fachada, en este caso la clase IAction, evitando el contacto directo con las subclases del MAAI.

Patrón de Comportamiento

- **Observador (en inglés Observer)**

Define una dependencia de uno-a-muchos entre objetos, de forma que cuando un objeto cambie de estado se notifique y se actualicen automáticamente todos los objetos que dependen de él. Describe cómo establecer una consistencia entre objetos relacionados, sin hacer a las clases fuertemente acopladas ya que eso reduciría su reutilización (52).

El patrón Observador se emplea en los componentes del sistema encargados de realizar las acciones ante la ocurrencia de una incidencia. A través del extension point que se realiza de la interface IAction, que permite acceder a los componentes que implementan esa interface, se refleja la observación por dichos componentes a cualquier cambio de estado de la clase ManagerActions.

Después de caracterizar los patrones de diseño utilizados se caracteriza el mecanismo de seguridad empleado en el MAAI.

3.4 Seguridad del sistema

En la actualidad, el avance de las tecnologías permite la comunicación a través de redes informáticas. La información que se intercambia entre ordenadores circula por una serie de sistemas intermedios desconocidos, de los cuales no se tiene control alguno. Además no se tiene la seguridad que el sistema al que uno está conectado sea quien dice ser. En fin, no existe absoluta certeza de que los sistemas a los que se envía información sean en realidad los auténticos. En el caso que sea auténtico, no se conoce si recibe la información que realmente fue enviada, o si fue modificada por terceros. Por tanto, la seguridad es fundamental a la hora de afrontar tareas que se realizan en sistemas informáticos (53).

Existen diferentes algoritmos de cifrado para la protección de la información. La seguridad de la plataforma GRHS en el cliente se garantiza en el SARA, donde la información a enviar al MAOI se cifra con el algoritmo de cifrado asimétrico de Ron Rivest, Adi Shamir y Leonard Adleman (RSA) (54). Este algoritmo consiste en que el emisor solicita la clave pública del receptor para generar la información cifrada. Una vez realizado el cifrado, únicamente el receptor con su clave privada, puede recuperar la información inicial enviada por el emisor. Al cifrado de la información se une el uso de una Infraestructura de Clave Pública (PKI, en inglés Public Key Infraestructure) (55), la cual se basa en la posibilidad de firmar ficheros para conocer quién envió la información y garantizar el no repudio. Otro de los mecanismos de seguridad adoptados por el SARA es el monitoreo de ficheros, además de la verificación de la integridad de los componentes a través de su suma de verificación. Dicho mecanismo comprueba la existencia de algún cambio, notificándolo como una incidencia. En cuanto al servidor, utiliza también el algoritmo de cifrado RSA y el PKI para el envío de la información al cliente. Ambos hacen el proceso inverso en el caso de recibir la información. Después de una descripción de la seguridad en el sistema del MAAI se muestran los diagramas de paquetes del diseño del módulo.

3.5 Diagrama de paquetes del diseño

El diagrama de paquetes del diseño se elaboró en correspondencia con las clases pertenecientes al cliente y al servidor.

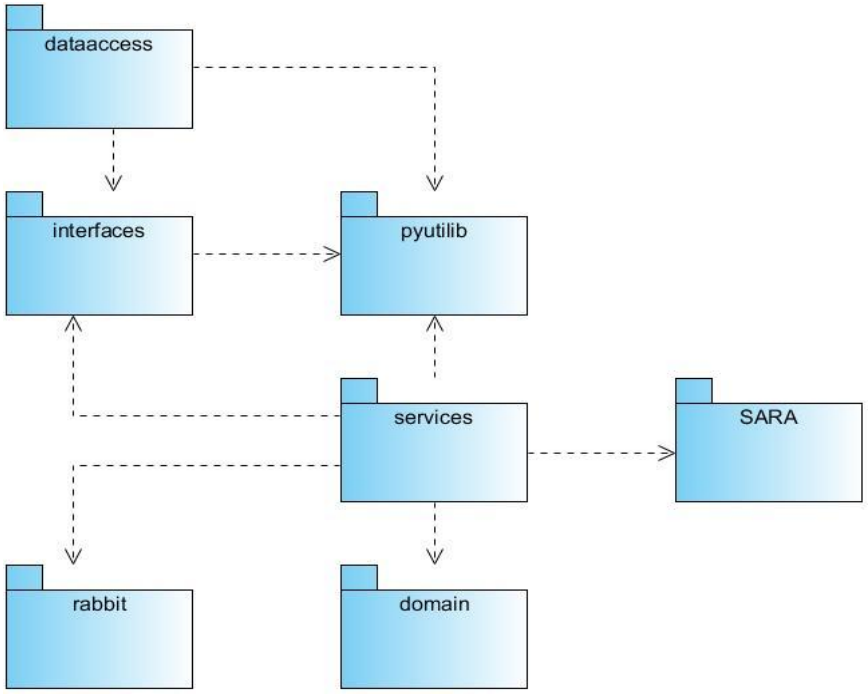

Figura 6. Diagrama de Paquetes para las clases del diseño en el cliente

El diagrama correspondiente a la figura 6 brinda una vista general de cómo está confeccionado el diseño en el cliente. Las clases del diseño están agrupadas en los diferentes paquetes y aplicaciones como se muestran. El paquete de acceso a datos dataaccess contiene los componentes encargados de la manipulación y control de las acciones a ejecutar ante una incidencia, tanto para el sistema operativo Windows, como para GNU/Linux. El paquete pyutilib contiene a pyutilib.core que encierra un conjunto de clases relacionadas entre sí, como son SingletonPlugin, Interface y ExtensionPoint.

El paquete interfaces contiene todas las interfaces del MAAI que son implementadas por los componentes. El paquete de servicios services está compuesto por las clases encargadas de la lógica del funcionamiento del módulo, utilizando la aplicación rabbitmq, que no es más que un servidor de cola de mensajes, empleado para adicionar las incidencias recibidas de los módulos de inventario de hardware y software. En el paquete de dominio domain están incluidas todas las clases entidades de interés para el sistema, acumulando información que será utilizada durante el proceso de ejecución del módulo. EL SARA es el encargado de la comunicación e integración de los módulos en la plataforma GRHS.

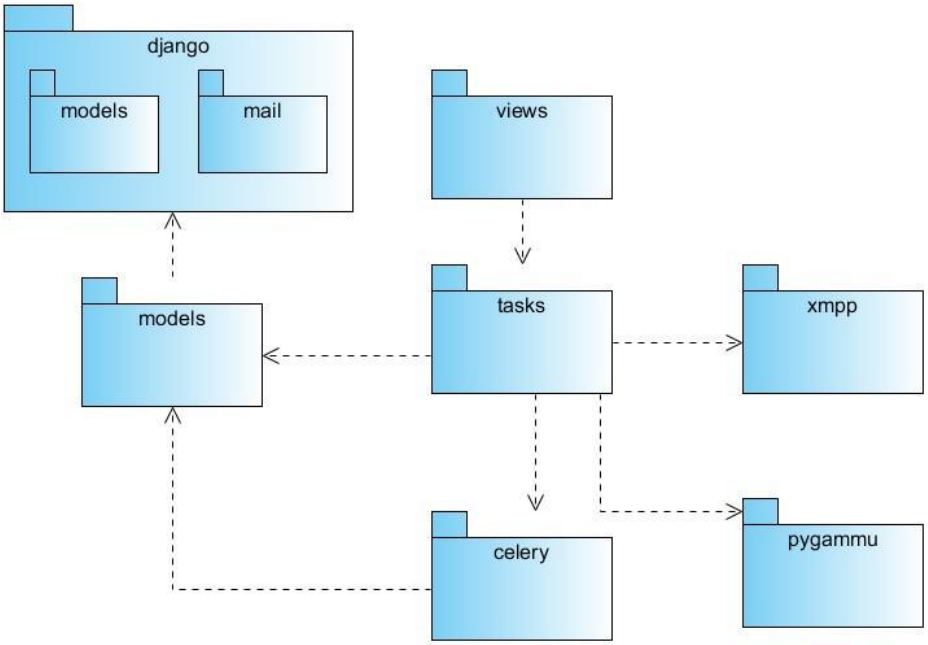

Figura 7. Diagrama de Paquetes para las clases del diseño en el servidor

El diagrama de paquetes establecido para las clases del diseño en el servidor que se muestra en la figura 7, está compuesto por paquetes. El paquete django está conformado por el framework de desarrollo web Django que facilita el desarrollo del sistema en el servidor y que a su vez, contiene el paquete de modelos (en inglés models) encargado de administrar la manipulación de los datos y el paquete mail utilizado para el envío de alarmas a través de correo electrónico. El paquete de modelos models establece dentro de sí, todos los modelos correspondientes a cada una de las funcionalidades a implementar por parte del servidor. El paquete de vistas views se encarga de almacenar las vistas, obteniendo el identificador de la incidencia a través de la URL y enviándolo a las tareas. El paquete de tareas tasks está relacionado con celery para la cola de tareas distribuidas y manipula el envío de alarmas utilizando el módulo xmpp para el enviar mensajes instantáneos y la aplicación gammu para enviar SMS. A partir del diagrama de paquetes para las clases del diseño, se modelan los diagramas de clases del diseño correspondiente al MAAI.

3.6 Diagrama de clases del diseño

El diagrama de clases del diseño es realizado durante el proceso de diseño de los sistemas, es aquí donde se crea el diseño conceptual de la información manipulada. El MAAI en su diagrama de clases del diseño representa todas las clases que serán utilizadas dentro del módulo y las relaciones que

existen entre ellas (56). A continuación se muestra el diagrama de clases del diseño correspondiente al CU Realizar acciones. Los otros diagramas de clases del diseño se pueden encontrar en el Anexo 2.

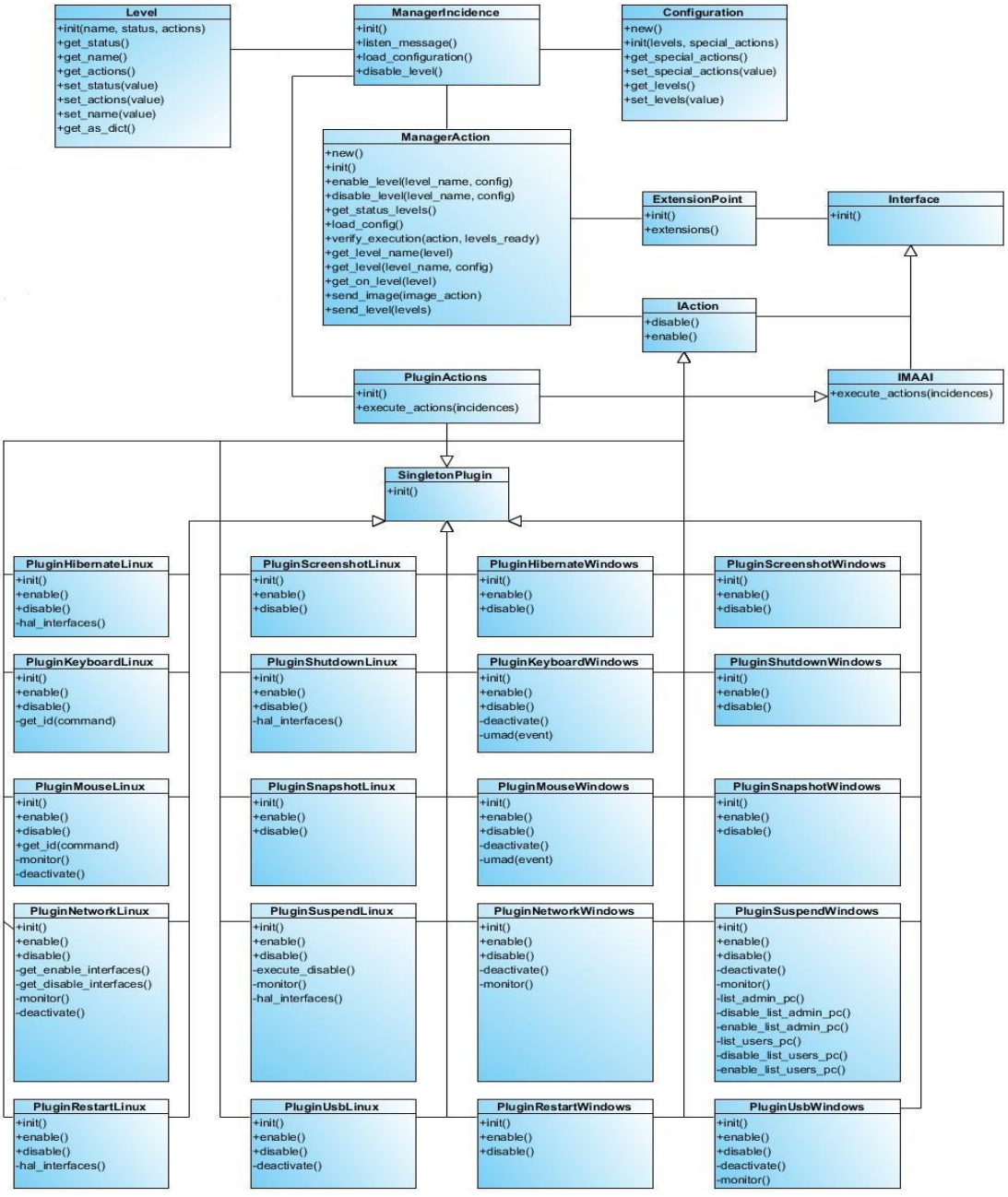

Figura 8. Diagrama de clases del diseño del CU Realizar acciones

El diagrama de clases del diseño correspondiente al CU Realizar acciones, ilustrado en la figura 8. El diseño en el cliente del MAAI se rige por la estructura que establece PyUtilib mediante componentes, puntos de extensión e interfaces. Las clases interfaces son IAction e IMAAI, las cuales heredan de la clase Interface de pyutilib.core. Los MI envían las incidencias detectadas a través de la interfaz IMAAI, que es implementada por la clase plugin PluginActions. La interfaz IAction es la encargada de modelar las funcionalidades relacionadas con las acciones a realizar, siendo implementada por las clases plugins que realizan las acciones.

Las clases de servicio son ManagerAction y ManagerIncidence. La clase ManagerIncidence es la encargada de gestionar las incidencias recibidas de los MI y tiene una relación estrecha con la clase ManagerAction. La clase ManagerAction gestiona y controla las acciones a realizar de acuerdo al nivel de la incidencia recibida. La clase ExtensionPoint de pyutilib.core es utilizada para crear puntos de extensión de las interfaces, permitiendo el acceso a los diferentes componentes. Las clases de dominio Configuration y Level representan información de interés referente a la configuración y los niveles respectivamente. La clase SingletonPlugin de pyutilib.core es heredada por todas las clases componentes que integran el MAAI, estableciendo que solo se puede crear una instancia de dichos componentes. Seguidamente se muestra el modelo físico de datos del MAAI en el servidor.

3.7 Modelo Físico de Datos

Describe las representaciones físicas de los datos persistentes utilizados en el MAAI en el servidor y que serán almacenados en la base de datos.

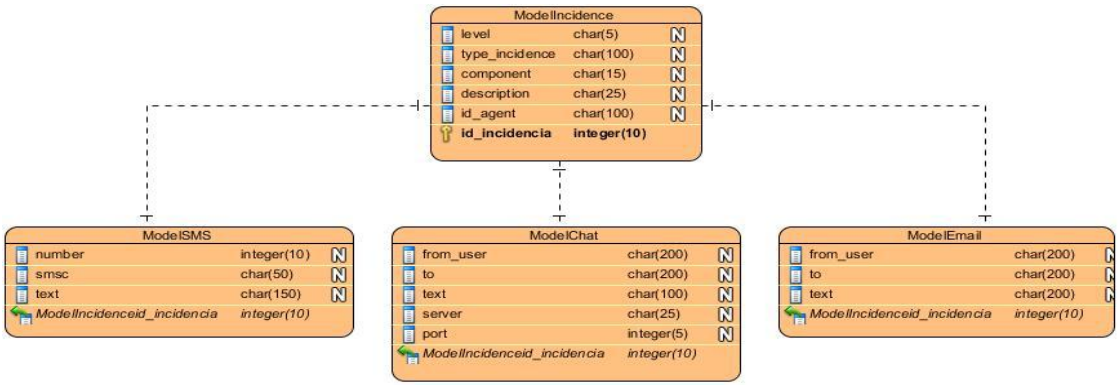

Figura 9. Modelo Físico de Datos del MAAI en el servidor

En el modelo físico de datos presentando en la figura 9, se muestra un modelo ModelIncidence, que está relacionado a su vez, con uno o varios tipos de mensajes, representados en los modelos: ModelSMS, ModelChat y ModelEmail. Cada modelo representa una serie de atributos en particular, de acuerdo a las necesidades de su uso.

3.8 Conclusiones parciales

En el presente capítulo fue mostrada una arquitectura base flexible y adaptada a las funcionalidades del sistema, caracterizándose por ser una arquitectura basada en capas y en componentes para el cliente, y basada en Model-Template-View para el servidor. Los patrones de diseño son una técnica para flexibilizar el código, por lo que se detallan y especifican los patrones de diseño utilizados en la aplicación para la asignación de responsabilidades, así como los patrones GOF que incluyen patrones de creación, de comportamiento y estructurales. Además se describió la seguridad del sistema, y la importancia que representa para el MAAI y la plataforma GRHS. Durante el desarrollo del capítulo se modelaron los diagramas de paquetes del diseño del MAAI y el diagrama de clases del diseño relacionado con el CU Realizar acciones. Se presentó el modelo físico de datos en el servidor, mostrando la relación entre los modelos y los datos de cada uno.

CAPÍTULO 4: IMPLEMENTACIÓN Y PRUEBAS

4.1 Introducción

El presente capítulo se muestra el modelo de implementación desarrollado a partir del modelo del diseño elaborado en el capítulo anterior. En el flujo de implementación, se describe el desarrollo del sistema en término de componentes como script, ejecutables y ficheros de código fuente, así como el modelo de despliegue del sistema que se empleará una vez terminado este flujo. Al concluir el flujo de implementación se dará inicio al flujo de trabajo pruebas, donde cada parte desarrollada del módulo, será evaluada mediante la estrategia de pruebas definida, para determinar si se logró el objetivo general de la investigación de manera satisfactoria. A continuación se muestra el diagrama de despliegue correspondiente al MAAI.

4.2 Diagrama de despliegue

Un diagrama de despliegue muestra la configuración de los nodos que participan en la ejecución y de los componentes que residen en ellos (57). Un nodo es un elemento físico que existe en tiempo de ejecución. Representa un recurso computacional que por lo general tiene memoria y capacidad de almacenamiento. Además, representa el despliegue físico de un componente (58). A continuación se muestra el diagrama de despliegue correspondiente al módulo:

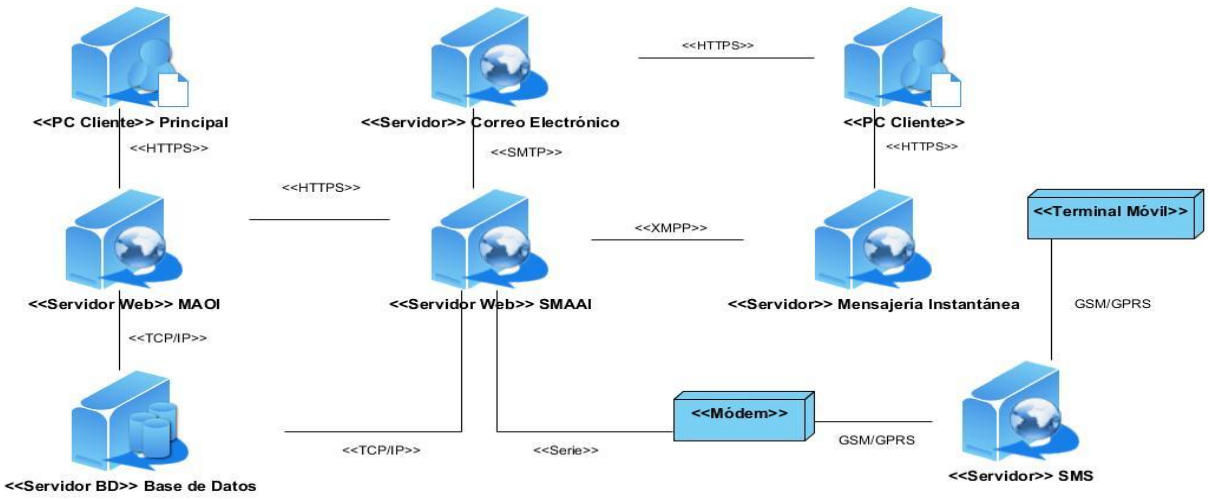

Figura 10. Diagrama de Despliegue

Nombre del Nodo	Descripción
PC Cliente Principal	Computadora desde la cual es ejecutado el cliente del MAAI.
Servidor de Base de Datos	Servidor de Base de Datos donde se guarda toda la información generada por el módulo.
Servidor Web MAOI	Representa el ordenador donde reside el servidor, encargado de deshabilitar los niveles de incidencias, y de ejecutar las acciones de reiniciar, hibernar y apagar la computadora.
Servidor Web SMAAI	Representa el ordenador donde se encuentra el servidor del MAAI, responsable del envío de las alarmas.
Servidor de Correo Electrónico	Representa el ordenador donde se encuentra el servidor de correo electrónico para el envío de este tipo de mensaje.
Servidor de Mensajería Instantánea	Permiten intercambiar información a una gran cantidad de usuarios ofreciendo la posibilidad de llevar a cabo conversaciones en tiempo real.
PC Cliente	Computadora cliente que recibe las alarmas enviadas a través de correo electrónico y mensajería instantánea.

Módem	Representa un dispositivo de hardware que se conecta a un ordenador y a una línea telefónica, permitiendo al ordenador transmitir información por la línea telefónica.
Servidor SMS	Representa el servidor para enviar SMS a los teléfonos celulares.
Terminal Móvil	Representa el teléfono celular al que se le envía las alarmas mediante SMS.

Tabla 5. Descripción de los nodos del diagrama de despliegue

4.3 Diagramas de componentes

Los diagramas de componentes describen los elementos físicos del sistema y sus relaciones, representan a su vez, todos los tipos de elementos software que intervienen en la fabricación de aplicaciones informáticas, como ejecutables, librerías y dependencias lógicas que existen entre ellos (59). A continuación se realiza una breve descripción del diagrama de componentes referente al CU Realizar acciones. El resto de los diagramas de componentes se pueden ver en el Anexo 3.

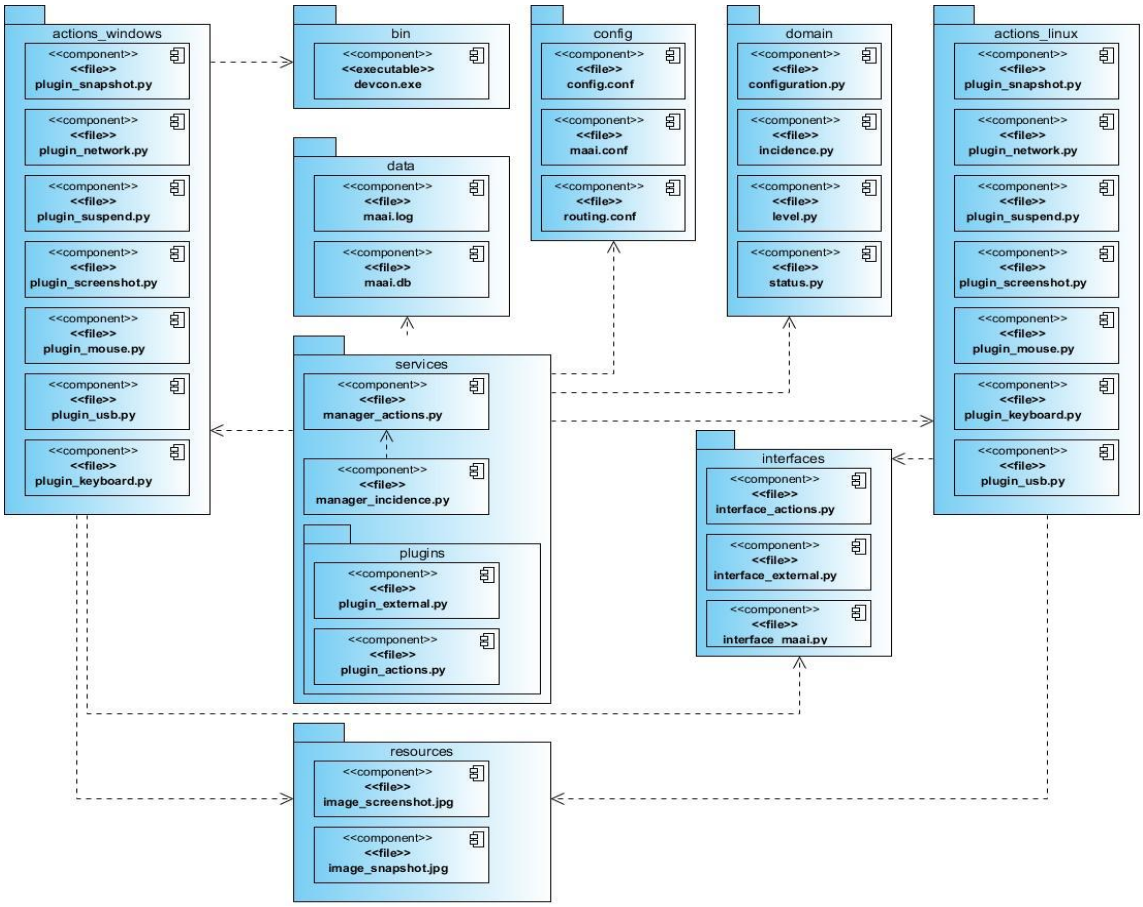

Figura 11. Diagrama de componentes del paquete cliente para el CU Realizar acciones

El diagrama de componentes para el cliente mostrado anteriormente en la figura 11, refleja el paquete de acciones de GNU/Linux (actions_linux), el cual representa los componentes utilizados para realizar las acciones en el sistema operativo GNU/Linux; y el paquete de acciones de Windows (actions_windows), compuesto por los componentes utilizados para realizar las acciones en el sistema operativo Windows. Este último, se relaciona a su vez, con el paquete bin, el cual contiene un componente devcon necesario para realizar determinadas acciones de este sistema operativo, como la habilitación e inhabilitación de la red y de los puertos USB.

El paquete de interfaces (interfaces) incluye los componentes interfaces que son implementados por los componentes contenidos en los paquetes actions_windows y actions_linux. El paquete de servicios services está integrado por los componentes encargados de la lógica del funcionamiento del módulo y se

relaciona con el paquete de dominio domain que le brinda información necesaria al paquete anterior. El paquete de recursos resources almacena las imágenes capturadas. El paquete de configuración config contiene las configuraciones utilizadas en el MAAI, mientras que el paquete data registra los componentes para la caché y los logs del sistema respectivamente.

Todas las dependencias usadas para dar solución a los requisitos funcionales, definidos en el Capítulo 2, correspondientes a los dos diagramas de componentes antes diseñados se podrán consultar en el Anexo 4.

4.4 Estrategia de Pruebas

La estrategia de pruebas vela porque el producto de software que se está construyendo, reúna los requerimientos de lógica del funcionamiento del módulo que el cliente ha pedido realizar mediante el debido contrato de desarrollo de software. Son los procesos que permiten verificar y revelar por la calidad de un producto de software (60). El MAAI definió una estrategia con niveles, tipos de pruebas, métodos y técnicas. Teniendo en cuenta en este análisis todos aquellos elementos que influyen en el desarrollo del sistema. La estrategia seleccionada permitió solucionar errores que presentaba el módulo y perfeccionar de forma satisfactoria la solución implementada.

Dentro de los niveles generales de pruebas se encuentran:

- Prueba de Desarrollador
- Prueba Independiente
- Prueba de Unidad
- Prueba de Integración
- Prueba de Sistema
- Prueba de Aceptación

Para elegir el nivel de prueba a utilizar se tuvo en cuenta las características de los mismos. Dentro de estos niveles de pruebas la estrategia elaborada hace énfasis en el nivel de Prueba de Desarrollador, debido a que estas pruebas son llevadas a cabo por el equipo de desarrollo y en el nivel de Prueba de

Integración, llevada a cabo en la combinación de los diferentes paquetes con que cuenta el MAAI para su correcto funcionamiento.

Niveles de Pruebas	Tipos de Pruebas	Métodos
Integración	Funcionalidad Función	Caja Negra
Desarrollador	Funcionalidad Función	Caja Blanca

Tabla 6. Estrategia de prueba

4.4.1 Niveles de prueba

Pruebas de Integración: es la fase de la prueba de software en la cual se combinan los diferentes módulos y son probados como un grupo. Con estas pruebas se asegura que los componentes del modelo de implementación funcionen de manera correcta, cuando se combinan para ejecutar todos los CU que conforman las funcionalidades con las cuales debe contar el sistema.

Pruebas del Desarrollador: son las pruebas realizadas durante el desarrollo de la aplicación para verificar el funcionamiento correcto de lo implementado, y son llevadas a cabo por los desarrolladores (61).

4.4.2 Tipos de Pruebas

- Funcionalidad

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

Pruebas de Caja Blanca

Se denomina cajas blancas al tipo de pruebas de software que es realizada sobre las funciones internas de un módulo determinado, es decir son las que tratan el código directamente no verifican si este trabaja o

no correctamente, solo se encarga de revisar estructuras determinadas. Estas se llevan a cabo en primer lugar, sobre un módulo concreto (62).

Pruebas PyUnit

PyUnit es el framework estándar para desarrollar los casos de prueba de unidad, para programas desarrollados en el lenguaje Python. Está diseñado para trabajar con cualquier versión de este lenguaje superior a la 1.5.2, además de ser multiplataforma (63). Se hizo necesaria la utilización de este framework en aras de comprobar la lógica de la implementación usada en la solución, en busca de posibles errores. Con la realización de este tipo de prueba de caja blanca se logró trabajar sobre los errores encontrados, como es el caso de los valores que se esperaban y el que realmente devolvía. Estos errores se considerados no relevantes debido a que fueron solucionados, dando la posibilidad de brindar una solución final que no presenta dificultades.

Pruebas de Caja Negra

Las pruebas de caja negra son aquellos elementos que se estudian desde el punto de vista de las entradas que reciben, y las respuestas que provee sin tener en cuenta el funcionamiento interno. Esta prueba examina algunos aspectos del modelo, fundamentalmente del sistema, sin tener mucho en cuenta la estructura interna del software. Se centran principalmente en los requisitos funcionales del software. Permiten obtener un conjunto de condiciones de entrada que ejerciten completamente todos los requisitos funcionales de un programa. En ellas se ignora la estructura de control, concentrándose en los requisitos funcionales del sistema y ejercitándolos (64)**.** A continuación se muestra la iteración 1 de las pruebas realizadas, el resto de las iteraciones se pueden consultar en el Anexo 5.

Iteración #1 Integración de los Módulos de Inventario y el MAAI.

Módulo actual	Módulo integrado	Funcionalidad	Condiciones de ejecución	Escenario de prueba	Resultado previsto	Resultado real
MAAI	Módulos de Inventario.	Realizar acciones.	- Los MI están inicializados. - La configuración del MAAI está almacenada en la caché.	Capturar imagen del usuario.	Capturar la imagen del usuario y enviarla al servidor.	Captura la imagen del usuario, pero al enviarla al servidor, esta llega distorsionada.
				Capturar imagen de la pantalla del usuario.	Capturar la imagen de la pantalla de la computadora y enviarla al servidor.	Captura la imagen de la pantalla de la computadora, pero al enviarla al servidor, esta llega distorsionada.
				Inhabilitar puertos USB.	Inhabilitar y monitorizar los puertos USB.	Inhabilita los puertos USB, pero la monitorización no se cumple correctamente.
				Inhabilitar ratón.	Inhabilitar y monitorizar el ratón.	Inhabilita y monitoriza el ratón satisfactoriamente.

				Inhabilitar teclado.	Inhabilitar y monitorizar el teclado.	Inhabilita y monitoriza el teclado satisfactoriamente.
				Inhabilitar interfaces de red Ethernet.	Inhabilitar y monitorizar las interfaces de red.	Inhabilita las interfaces de red, presentando problemas en la monitorización.
				Suspender la computadora.	Suspender la computadora e inhabilitar las sesiones de los usuarios.	Suspende la computadora, aunque no todas las sesiones de los usuarios son inhabilitadas.

Tabla 7. Integración de los Módulos de Inventario y el MAAI

4.4.3 Entorno de pruebas

Partiendo de los requisitos no funcionales que se definieron para el desarrollo de esta solución el entorno de pruebas utilizado cuenta con las particulares siguientes:

Computadora Servidor

Hardware	Datos
Procesador	Dual Core CPU 2.2GHz
Memoria	1GB
Tarjeta Madre	INTEL-965

Ancho de Banda	100 Mbps
Sistema Operativo	Ubuntu 11.10

Tabla 8. Características de la computadora servidor donde se realizan las pruebas

Computadora Cliente para los sistemas operativos GNU/Linux y Windows

Hardware	Datos
Procesador	Dual Core CPU 2.2GHz
Memoria	1GB
Tarjeta Madre	INTEL-965
Ancho de Banda	100 Mbps
Sistema Operativo	Ubuntu 11.10 y Windows XP

Tabla 9. Características de la computadora cliente donde se realizan las pruebas

4.4.4 Resultados de las pruebas

Las pruebas realizadas a la solución desarrollada se dividieron en tres bloques. El primer bloque fue probado detalladamente, detectando una serie de no conformidades entre las que se destacan problemas al monitorizar las funcionalidades referentes a los puertos USB y a las interfaces de red Ethernet, así como la inhabilitación de las sesiones de los usuarios, contando con un total de 12 no conformidades. Para las pruebas del segundo bloque, las no conformidades detectadas anteriormente habían disminuido, aunque persistían los problemas con los puertos USB y la red. Para el último bloque solo quedó registrada una única no conformidad, declarada como que no procede. Así resulta un módulo funcional con las características descritas en los requerimientos del sistema, cumpliendo así, con los objetivos especificados.

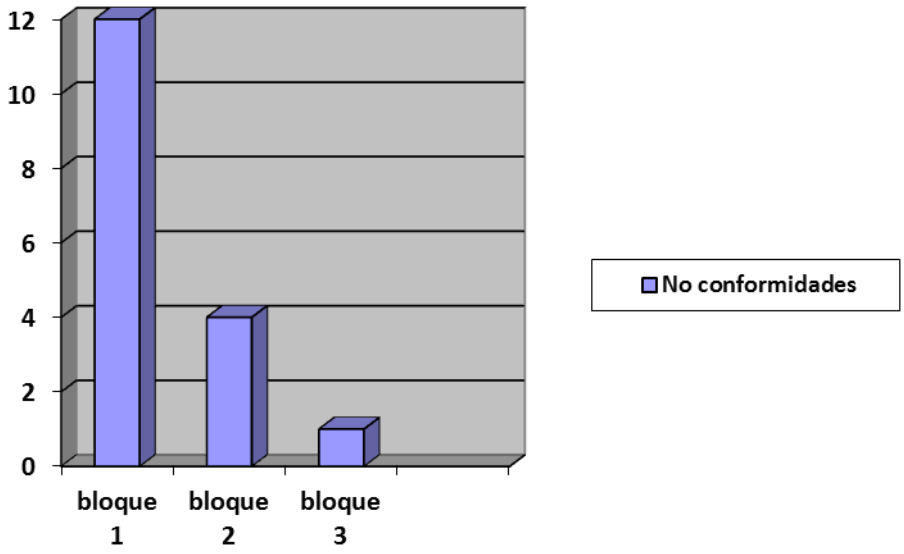

Figura 12. Resultado de las iteraciones de las pruebas realizadas en el MAAI

4.5 Conclusiones parciales

Se logró en el presente capítulo un mejor entendimiento del sistema desde el punto de vista de implementación, presentado una descripción general y bien detallada de los artefactos generados en la fase de implementación y de pruebas. Mediante la realización de los diagramas de componentes en el cliente y el servidor y el diagrama de despliegue, se describió la distribución del sistema en nodos. Los resultados obtenidos a través de las pruebas realizadas, ejemplifican la calidad del trabajo efectuado en la implementación del MAAI, mostrando que las funcionalidades desarrolladas no presentan anomalías en su funcionamiento ya que realizan un correcto tratamiento de los datos que manejan. Por lo que se concluye que el resultado de las pruebas realizadas ha sido satisfactorio.

CONCLUSIONES GENERALES

El resultado principal de este trabajo de investigación es la obtención de un módulo capaz de enviar alarmas y realizar acciones ante incidencias, funcional integrado a la plataforma Gestor de Recursos de Hardware y Software. Se cuenta con una documentación completa y bien explicada que se fue fundamentando durante la fase de diseño y de implementación, siendo guiada por la metodología RUP. Con el propósito de darle cumplimiento al objetivo general y basado en la problemática expuesta, se llevaron a cabo satisfactoriamente cada una de las tareas que fueron perfiladas al comienzo de la investigación.

- Se realizó un estudio de las tendencias actuales en el marco de los inventarios de hardware y software, identificando entre las herramientas estudiadas que algunas son capaces de enviar alarmas, sin embargo, solo envían un tipo de mensaje: correo electrónico. No ejecutan acciones de manera automática ante incidencias detectadas, ni son capaces de enviar alarmas con la realización de acciones.
- La utilización de RUP como metodología de desarrollo permitió guiar el proceso de desarrollo de software, ya que genera abundante documentación y artefactos necesarios para verificar la solución.
- Se definieron los requisitos funcionales y no funcionales de acuerdo a las necesidades del sistema.
- Fueron desarrollados lo componentes para el envío de alarmas a través de SMS, correos electrónicos y mensajes instantáneos, con el objetivo de notificar las incidencias.
- Se implementaron los componentes para la toma del control del ratón, el teclado, los puertos USB y las interfaces de red Ethernet.
- Se desarrollaron los componentes para tomar imágenes de las estaciones de trabajo y al usuario, que seguidamente son enviadas al administrador del sistema.
- Las sesiones de la computadora del usuario son controladas por el servidor, garantizando un correcto funcionamiento en su ejecución.
- Fueron realizadas las pruebas internas del módulo, obteniendo resultados satisfactorios.

Actualmente la plataforma Gestor de Recursos de Hardware y Software, cuenta con un módulo capaz de agrupar funcionalidades necesarias para un correcto control de los activos informáticos. Este módulo

presenta la ventaja atender las peticiones de una forma ordenada y realizar una serie de acciones de forma automática en redes distantes de computadoras, así como el envío de alarmas a través de SMS, correo electrónico y mensajes instantáneos.

RECOMENDACIONES

- Implementar funcionalidades para el envío de otras alarmas como mensaje a beeper y la activación de una sirena.
- Añadir funcionalidades para habilitar las sesiones de los usuarios de forma automática.
- Capturar imágenes y videos de los locales de las estaciones de trabajo, y enviarlos al servidor.
- Tomar el control de otros dispositivos como el monitor, impresoras y discos duros.

BIBLIOGRFÍA

1. **Parada, Prof. Jorge E.** Sistemas de Inventario. (Pag 2-13). 2006.

2. **Holguín, Carlos Julio Vidal.** Fundamentos de gestión y control de inventarios. Capítulo 2: Elementos para la toma de decisiones en sistemas de inventarios. 2012.

3. **UCI, Aprobado por la Rectora de la.** Manual de Funcionamiento Interno UCI. 2012.

4. **Española, Real Academia.** Real Academia Española. [En línea] http://buscon.rae.es/draeI/SrvltGUIBusUsual?TIPO_HTML=2&LEMA=inventario.

5. —. Real Academia Española. [En línea] http://buscon.rae.es/draeI/SrvltConsulta?TIPO_BUS=3&LEMA=incidencia.

6. —. Real Academia Española. [En línea] http://buscon.rae.es/draeI/SrvltObtenerHtml?LEMA=acción&SUPIND=0&CAREXT=10000&NEDIC=No.

7. —. Real Academia Española. [En línea] http://buscon.rae.es/draeI/SrvltConsulta?TIPO_BUS=3&LEMA=alarma.

8. **Inc, Softinventive Lab.** Softinventive Lab Inc. [En línea] [En línea] http://www.softinventive.com/es/products/total-network-inventory.

9. **LOGINventory.** LOGINventory. [En línea] http://www.loginventory.com/loginventory.

10. **DNA, Net Support.** Net Support DNA. [En línea] http://www.netsupportdna.com/ES/inventory.asp.

11. **INFORMATIQUE, GPLI GESTION LIBRE DE PARC.** GPLI GESTION LIBRE DE PARC INFORMATIQUE. [En línea] http://www.glpi-project.org/spip.php?article13.

12. **generation, OCS inventory nexi.** OCS inventory nexi generation. [En línea] http://www.ocsinventory-ng.org/en/about/features.

13. **documentation, Cacic v1.** Cacic v1 documentation. [En línea] http://www.latinuxpress.com/books/drafts/cacic/caps/09.html.

14. **Jacobson, Ivar, Booch, Grad y Rumbaugh, James.** El Proceso Unificado de Desarrollo de Software. (Pag 5-7).

15. **Vitalta, Josep.** UML Guía Visual.

16. **Paradigm, Visual.** Visual Paradigm. [En línea] http:// http://www.visual-paradigm.com/product/vpuml/.

17. **Duque, Raúl González.** Python para Todos. (Pag 7-9).

BIBLIOGRAFÍA

18. **Eclipse.** Eclipse. [En línea] http://help.eclipse.org/helios/index.jsp?topic=%2Forg.eclipse.platform.doc.isv%2Fporting%2Feclipse_3_6_porting_guide.html.

19. **Kaplan-Moss, Jacob y Holovaty, Adrian.** El libro de Django.(Pag 1-5). 2008.

20. **J.Kabir, Mohammed.** La biblia Servidor Apache 2. Capítulo 1. (Pag 39-43) .

21. **Kaplan-Moss, Jacob y Holovaty, Adrian.** El libro de Django. (Pag 229-232). 2008.

22. **Riggs, Simon y Krosing, Hannu .** PostgreSQL 9. Administration Cookbook. (Pag 8-12). 2012.

23. **PgAdmin.** PgAdmin. [En línea] http://www.pgadmin.org/docs/1.14/index.html.

24. **Duque, Raúl González.** Python para Todos.(Pag 139-142).

25. **Rhodes, Brandon y Goerzen, John.** Foundations of Python Network Programming. Second Edition. (Pag 131).

26. **Pilgrim, Mark.** Inmersión en Python 3. Capítulo 13. (Pag 288-299).

27. **Microsoft, Soporte de.** Soporte de Microsoft. [En línea] http://support.microsoft.com/kb/311272..

28. Ubuntu Server Guide. 2010.(Pag 31-32).

29. **Langdale, Philip.** Die.net. [En línea] http://linux.die.net/man/1/xinput.

30. **FireStorm.** FireStorm. [En línea] http://www.firestorm.cx/fswebcam/.

31. **Cihar, Michal.** Gammu SMSD Daemon Manual.(Pag 5-9). 2011.

32. **Jacobson, Ivar , Booch, Grad y Rumbaugh, James.** El Proceso Unificado de Desarrollo de Software. (Pag 112-113).

33. **Sommerville, Ian.** Ingeniería de Software. Capítulo 6. (Pag 108).

34. **Pressman, Roger S.** Ingeniería del Software. Un enfoque práctico.Capítulo 7. Pag(160-161).

35. **Sommerville, Ian.** Ingeniería de Software. Capítulo 6. Pag (110-111).

36. **Jacobson, Ivar, Booch, Grad y Rumbaugh, James.** El Proceso Unificado de Desarrollo de Software. Capítulo 6. Pag (121-122).

37. **Leyva, Ing. Yoanni Ordoñes.** Estilo de Codificación en Python de GRHS. 2012.

38. **Jacobson, Ivar, Booch, Grad y Rumbaugh, James.** El Proceso Unificado de Desarrollo de Software. Capítulo 3. (Pag 38).

39. —. El Proceso Unificado de Desarrollo de Software. Capítulo 3. (Pag 39-40).

40. —. El Proceso Unificado de Desarrollo de Software. Capítulo 4. Pag (80).

BIBLIOGRAFÍA

41. **Reynoso, Carlos y Kiccillof, Nicolás.** Estilos y Patrones en la Estrategia de Arquitectura de Microsoft. (Pag 19-22).

42. **Pressman, Roger S.** Ingeniería del software. Un enfoque práctico. Sexta edición. Capitulo 30. (Pag 879-883).

43. **E. Hart, William y Siirola, John.** The PyUtilib Component Architecture. 2010.

44. **Holovaty, Adrian y Kaplan-Moss, Jacob.** El libro de Django. (Pag 48). 2008.

45. **Erich , Gamma, y otros, y otros.** Design Patterns. Elements of Reusable Object- Oriented Software. (Pag 11-14).

46. **Larman, Craig.** UML y Patrones. Introducción al análisis y diseño orientado a objetos. (Pag 185-210).

47. **Gamma, Erich, y otros, y otros.** Design Patterns. Elements of Reusable Object- Oriented Software. (Pag 94-98).

48. **Erich, Gamma, y otros, y otros.** Design Patterns. Elements of Reusable Object- Oriented Software. (Pag 155-156).

49. —. Design Patterns. Elements of Reusable Object- Oriented Software. (Pag 249).

50. —. Design Patterns. Elements of Reusable Object- Oriented Software. (Pag 144-152).

51. —. Design Patterns. Elements of Reusable Object- Oriented Software. (Pag 208-217).

52. —. Design Patterns. Elements of Reusable Object- Oriented Software. (Pag 326-337).

53. **Ferrer, Jorge y Fernández-Sanguino, Javier .** Seguridad informática y software libre. (Pag 3).

54. **López, Manuel J. Lucena.** Criptografía y Seguridad en Computadores. 4ª Edición. Versión 0.7.0. Capítulo 12. (Pag 171-182).

55. EBOX Platform. eBox 1.4 para Administradores de Redes. Chapter 6. (Pag 181-183).

56. **Larman, Craig.** UML y Patrones. Introducción al análisis y diseño orientado a objetos. Capitulo 21. Pag (254-270).

57. —. UML y Patrones. Introducción al análisis y diseño orientado a objetos. Capítulo 36. (Pag 430).

58. **Jacobson, Ivar, Booch, Grad y Rumbaugh, James.** El Proceso Unificado de Desarrollo de Software. Capítulo 9. (Pag 217).

59. —. El Proceso Unificado de Desarrollo de Software. Capítulo 10. (Pag 257).

60. **Pressman, Roger S.** Ingeniería del Software. Un enfoque práctico. Capitulo 18. (Pag 305).

61. **Pressman, Roger S.** Ingeniería del software. Un enfoque práctico. Sexta edición. Capítulo 13. (Pag 391-409).

BIBLIOGRAFÍA

62. —. Ingeniería del software. Un enfoque práctico. Sexta edición.Capítulo 14. Pag (423-430).

63. **PyUnit.** PyUnit. [En línea] http://pyunit.sourceforge.net/pyunit.html.

64. **Pressman, Roger S.** Ingeniería del software. Un enfoque práctico. Sexta edición. Capítulo 14. (Pag 433-440).

GLOSARIO DE TÉRMINOS

Término	Significado
activos informáticos	Hace referencia a servidores, dispositivos, discos lógicos y aplicaciones.
aplicación	En informática, tipo de programa informático diseñado para facilitar al usuario la realización de un determinado tipo de trabajo. Esto lo diferencia principalmente de otros tipos de programas como los sistemas operativos, las utilidades y los lenguajes de programación, que realizan tareas más avanzadas y no pertinentes al usuario común.
artefactos	En UML, un artefacto es la especificación de un componente físico de información que es usado o producido por un proceso de desarrollo de software o por el desarrollo y operación de un sistema.
automatizar	
bytecode	Código intermedio más abstracto que el código de máquina.
caso de uso	En ingeniería del software, técnica para la captura de requisitos potenciales de un nuevo sistema o una actualización de software. Cada caso de uso proporciona uno o más escenarios que indican cómo debería interactuar el sistema con el usuario o con otro sistema para conseguir un objetivo específico.
ciclo de vida	Tiempo estimado que se espera esté un soporte, medio o aplicación en uso operativo. También se utiliza para designar las principales etapas que tienen lugar durante el desarrollo de un sistema.
código abierto	Práctica de desarrollo de software que promueve el acceso al código fuente de los sistemas computacionales. Algunos consideran al código abierto como una filosofía y otros como una metodología pragmática.

CPU	Unidad Central de Procesamiento, en inglés Central Processing Unit.
.csv	Guarda únicamente el texto y los valores como aparecen en las celdas de la hoja de cálculo activa para Excel.
.doc	Extensión de Microsoft Word.
emisor	En informática, es uno de los conceptos de comunicación, de la teoría de la comunicación y del proceso de información. El emisor es aquel objeto que codifica el mensaje y lo transmite por medio de un canal de comunicación hasta un receptor.
Ethernet	Es la red de área local (LAN), más ampliamente instalada tecnológicamente.
framework	Estructura de artefactos o módulos concretos con base en la que otro proyecto de software puede ser desarrollado.
hardware	Conjunto de dispositivos físicos que componen el ordenador: la pantalla, el teclado, el ratón, etc.
html	Lenguaje de programación que se utiliza para el desarrollo de páginas de Internet. Se trata de la sigla de Hyper Text Markup Language, es decir, Lenguaje de Marcas de Hipertexto. Permite describir la estructura y el contenido en forma de texto, además de complementar el texto con objetos tales como imágenes. Este lenguaje se escribe mediante etiquetas, que aparecen especificadas por corchetes angulares (< ejemplo >).
HTTP	Es el protocolo usado en las transacciones de la World Wide Web (WWW).
HTTPS	En inglés Hypertext Transfer Protocol Secure. Acrónimo de protocolo seguro de transferencia de hipertexto, es un protocolo de red basado en el protocolo HTTP, destinado a la transferencia segura de datos de hipertexto, es decir, es la versión segura de

	HTTP. El sistema HTTPS utiliza un cifrado basado en las Secure Socket Layers (SSL) para crear un canal cifrado (cuyo nivel de cifrado depende del servidor remoto y del navegador utilizado por el cliente) más apropiado para el tráfico de información sensible que el protocolo HTTP.
IP o TCP/IP	En inglés Internet Protocol, protocolo de comunicación mediante datagramas.
.jpg	Son las siglas de Joint Photographic Experts Group, el nombre del grupo que creó este formato. Es un formato de compresión de imágenes, tanto en color como en escala de grises, con alta calidad (a todo color).
JSON	Acrónimo de JavaScript Object Notation. Es un subconjunto de la notación literal de objetos de JavaScript que no requiere el uso de XML.
lenguaje alto nivel	Lenguaje donde es posible expresar los algoritmos mediante una sintaxis del modo más cercano a la capacidad cognitiva del hombre.
lenguaje interpretado	Se ejecuta utilizando un programa intermedio llamado intérprete, en lugar de compilar el código a lenguaje máquina que pueda comprender y ejecutar directamente una computadora (lenguajes compilados).
lenguaje orientado a objetos	Lenguaje diseñado para cumplir con los conceptos de la programación orientada a objetos.
licencia BSD	Licencia de software libre menos restrictiva que la GPL, asignada principalmente para los sistemas BSD.
Linux	Sistema operativo creado con la combinación de las herramientas de sistema GNU y el kernel o núcleo similar a Unix, llamado Linux.
Mac OS X	Sistema operativo desarrollado y comercializado por la compañía

	Apple.
ORM	El acceso a una base de datos por parte del programador convirtiendo toda sentencia sql a operaciones con objetos.
PC	Computadora personal, en inglés Personal Computer. Computadora digital personal basada en un microprocesador y diseñada para ser utilizada por una sola persona a la vez.
.pdf	En inglés portable document format (formato de documento portátil). Es un formato de almacenamiento de documentos desarrollado por la empresa Adobe Systems. Es de tipo compuesto (imagen vectorial, mapa de bits y texto).
plataforma	Es un sistema para el funcionamiento de determinados módulos de hardware o software con los que es compatible.
receptor	En informática, es el que recibe una información. El receptor es aquel objeto que decodifica el mensaje recibido.
.slk	Guarda únicamente los valores y fórmulas de la hoja de cálculo activa y un formato de celda limitado.
SMS	Servicio de Mensajes Cortos, en inglés Small Messages Service. Se refiere al envío de mensajes a teléfonos celulares.
software	Término en inglés para describir a los programas de computación.
TCP	En inglés Transmission Control Protocol. Es un protocolo de comunicación del nivel de transporte orientado a conexión.
USB	Bus Universal en Serie, en inglés Universal Serial Bus. Se trata de un concepto de la informática para nombrar el puerto que permite conectar periféricos a una computadora.
Windows	Sistema operativo desarrollado y comercializado por la compañía Microsoft.