Facultad 2

Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas

Título: “Módulo de inventario de actualizaciones y parches de seguridad del sistema operativo Windows en XILEMA GRHS”.

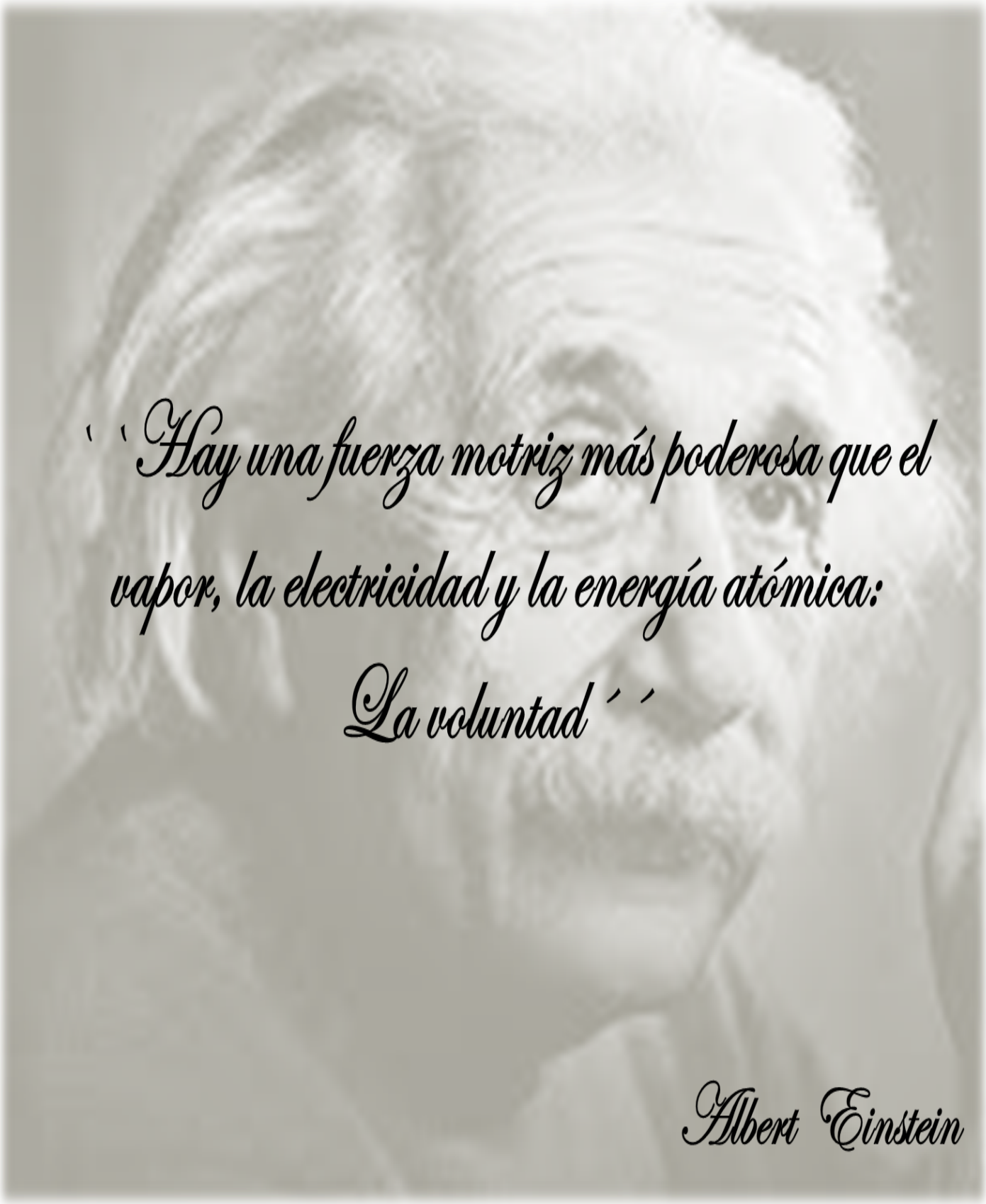
Autores: Erlin Torres Lage

Tutores: MSc. Mónica Peña Casanova

Ing. Leonel Fuentes García.

Ing. Ramón Guzmán Alemañy

La Habana, Junio de 2019


' 'Hay una fuerza motriz más poderosa que el vapor, la electricidad y la energía atómica: La voluntad ' '
Albert Einstein

Declaración de autoría

Declaramos ser autores de la presente tesis que tiene por título: Módulo de inventario de actualizaciones y parches de seguridad de sistemas operativos y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo. Para que así conste firmo la presente a los ____ días del mes de _____ del año _____.

Erlin Torres Lage

_____
Firma del Autor

MSc. Mónica Peña Casanova	Ing Leonel Fuentes García.	Ing. Ramón Guzmán Alemañy
_____	_____	_____
Firma del Tutor	Firma del Tutor	Firma del Tutor

Datos de contacto

MSc.Mónica Peña Casanova

Decana de: La facultad 2 de la Univercidad de las Ciencias Informáticas

Correo electrónico: monica@uci.cu

Ing. Leonel Fuentes García

Graduado de: Ingeniería en Ciencias Informáticas (2016) en la UCI.

Es: Administrador de la configuración y desarrollador del proyecto SavUnix perteneciente al departamento de Aplicaciones del Centro de TLM de la Facultad 2.

Correo electrónico: lfuentesg@uci.cu.

Ing. Ramón Guzmán Alemañy

Graduado de: Ingeniería en Ciencias Informáticas en la UCI.

Es: Especialista "A" en Ciencias Informáticas

Correo electrónico: ralemany@uci.cu

Agradecimientos

Dedicatoria

Resumen

En la medida que las organizaciones alcanzan un mayor nivel de informatización en sus procesos y se hacen más dependientes de la tecnología, carecen de la necesidad de responder rápidamente a nuevas amenazas a través de la reconfiguración de las aplicaciones o instalación de parches, que permita disminuir el número de vulnerabilidades que son descubiertas constantemente y utilizadas por los atacantes. Cada minuto perdido en la implantación de un parche de seguridad significa la exposición a amenazas concretas y un gran riesgo para las instituciones.

En la presente investigación se desarrolla un análisis de los principales elementos asociados a la automatización del inventario de actualizaciones y parches de seguridad del sistema operativo Windows para el gestor de recursos de hardware y software, desarrollada por el centro de telemática de la facultad 2 de la universidad de las ciencias informáticas.

Se presenta la descripción de las funcionalidades y los artefactos asociados a la metodología de desarrollo adoptada para dar solución a la problemática planteada por el cliente. Para la propuesta de solución se realizará un módulo que permita capturar y mostrar los datos de las actualizaciones y parches de seguridad instalados y pendientes del sistema operativo Windows en el gestor de recursos de hardware y software. Lo que facilitará la automatización de la captura, análisis y consulta de la información de las actualizaciones y parches de seguridad contribuyendo al control en el cumplimiento de las políticas de y a la reducción de riesgos.

Palabras clave: actualizaciones, parches de seguridad, inventario.

Abstract

As organizations reach a higher level of computerization in their processes and become more dependent on technology, there is a growing need to respond quickly to new threats through the reconfiguration of applications or the installation of patches to reduce the number of vulnerabilities that are constantly discovered and used by the attackers. Every minute lost in the implementation of a security patch means exposure to specific threats and a great risk for the institutions.

In the present investigation an analysis of the main elements associated to the automation of the Inventory of updates and security patches of the Windows operating system for the hardware and software resource manager, developed by the telematics center of the faculty 2 of the university of informatics sciences.

The description of the functionalities and artifacts associated with the development methodology adopted to solve the problem posed by the client is presented. For the solution proposal, a module will be made to capture and show the data of the updates and security patches installed and pending of the Windows operating system in the hardware and software resource manager. This will facilitate the automation of the capture, analysis and consultation of the information of security updates and patches, contributing to the control of compliance with the policies of and to the reduction of risks.

Keywords: *Updates, security patches, inventory.*

Índice

Índice de tablas

Índice de figuras

Índice de gráficos

Introducción

Las tecnologías de la información y las comunicaciones aportan grandes beneficios a la humanidad, su papel principal es crear herramientas útiles para el ahorro de tiempo y esfuerzo de trabajo, así como la creación de nuevas formas de servicios. La informatización ha penetrado en varios ámbitos de la sociedad, alcanzando cada día un papel más significativo en esta, la mayor parte de la información crítica se procesa y almacena en formato digital. Por una parte, las organizaciones son cada vez más dependientes de las soluciones informáticas que soportan sus procesos como resultado de lo cual, aumenta el impacto que la degradación de los niveles de funcionamiento de los servicios informáticos o las fallas en estos pueden provocar en las mismas. Este escenario se agudiza por estar los sistemas informáticos expuestos a un número creciente de amenazas (Alassafi, Hussain, Ghashgari, Walters, & Wills, 2017)

En la medida que las organizaciones tienen mayor cantidad de equipos interconectados, que alcanzan un mayor nivel de informatización, que avanza del desarrollo tecnológico de las tecnologías de la información y las comunicaciones; es mayor la “superficie de ataque[1]” y por tanto es necesario mantener el control sobre los medios tecnológicos que se emplean y dar seguimiento a los parches y actualizaciones presentes en los sistemas operativos(SO) y aplicaciones, convirtiendo a la informatización de este proceso en imprescindible (Montesino Perurena, 2012)

Las organizaciones deben ser conscientes sobre la necesidad de mantener permanentemente actualizado y parcheado todo software. Para esto existen un conjunto de aplicaciones que facilitan la realización de esta tarea de manera automática y que es recomendable aplicar (Ali, 2014).

En la Universidad de las Ciencias informáticas (UCI), se encuentran interconectadas más de 6000 computadoras, entre los SO usados se encuentra Windows en diferentes versiones, por lo que se hace necesario dar seguimiento de manera automatizada a los parches y actualizaciones del SO. Una de las políticas de seguridad definidas en la UCI es el uso del sistema Gestor de Recursos de Hardware y Software (XILEMA GRHS) siendo obligatoria su instalación en todas las computadoras como principal medida de

1 La superficie del ataque es un término empleado en el contexto de la seguridad informática para denotar las posibilidades de ataque de un atacante informático.

seguridad. Este sistema permite la verificación manual de un grupo de elementos de seguridad, no contando con un módulo que automatice la gestión de inventario de actualizaciones y parches de seguridad del SO, lo que permitiría tener un control más eficiente de las políticas de seguridad en las estaciones. Actualmente, el sistema realiza el inventario de hardware y software a todas las Computadoras conectadas a la red que tengan instalados el cliente de XILEMA GRHS (gclient), el cual envía la información del inventario realizado a un servidor de XILEMA GRHS (gserver) donde se procesa y almacena. A esta información se puede acceder mediante una aplicación web (gadmin) que permite la visualización de los datos, realización de reportes e informes, entre otras funcionalidades. XILEMA GRHS permite mejorar la gestión de los recursos, ofrece mayor facilidad en la obtención de informes tecnológicos, evita la duplicación y pérdida de la información, entre otros beneficios (TLM, 2015).

Además del control de los componentes de hardware de las máquinas y la acumulación de datos de software, es necesario llevar un registro de los elementos software básicos de los equipos, principalmente los relacionados con el SO y sus principales características, todo ello con el fin de conformar un inventario en cada equipo para el control eficiente de los medios con que cuenta una institución (Soomro, Shah, & Ahmed, 2016).

En el contexto actual, al realizar un análisis respecto a la relación de cambios entre los elementos de hardware y software, se observa que el software es el que está sujeto a modificaciones más frecuentes y significativas (Hsu, 2018). Un ejemplo fehaciente del planteamiento anterior, lo constituyen las actualizaciones del SO y la implementación de los parches de seguridad, que establecen de forma periódica cambios al software y que de forma gradual modifican sus componentes respecto a su versión inicial. (Montesino Perurena, 2012; Wolden, Valverde, & Talla, 2015).

El sistema XILEMA GRHS no tiene implementados métodos para la obtención y procesamiento de los datos referentes a las actualizaciones del SO, ni los parches de seguridad instalados.

A partir de lo anteriormente planteado cualquier modificación significativa que sufra el SO no queda registrada, ni se garantiza la información necesaria para el sellado de las vulnerabilidades reportadas en estos, por lo que se hace necesario la realización de modificaciones que permitan la captura de los datos referentes a las actualizaciones y

parches de seguridad instalados, identificados por el SO y que se aplican al XILEMA GRHS.

A partir de la situación problemática planteada se tiene el siguiente **problema a resolver**: El sistema XILEMA GRHS no es capaz de realizar de manera automatizada el inventario de las actualizaciones y parches de seguridad instalados y pendientes en el SO Windows lo que afecta el registro y control sobre el sellado de vulnerabilidades existentes en los mismos.

Basado en lo anterior, se define como **objeto de estudio**: Proceso de inventario automatizado de actualizaciones y parches de seguridad del SO Windows, enmarcado en el **campo de acción:** Proceso para la obtención de datos de las actualizaciones y parches de seguridad instalados y pendientes en el SO Windows para XILEMA GRHS.

Para darle solución al problema planteado se define como **objetivo general**: Desarrollar un módulo para XILEMA GRHS que permita llevar un inventario de los datos de las actualizaciones y parches de seguridad instalados y pendientes en el SO Windows.

Con el fin de dar cumplimiento al objetivo general se plantearon las siguientes **tareas de investigación**:

- Revisión de la bibliografía para elaborar el marco teórico conceptual referente a la captura, análisis y muestra de los datos de las actualizaciones y parches de seguridad.
- Análisis de sistemas homólogos para determinar características en común y conocer aspectos regulares en el diseño de las actualizaciones y parches de seguridad de SO aplicados al XILEMA GRHS.
- Elaboración de las principales funcionalidades del módulo para la posterior implementación del mismo, teniendo en cuenta los requisitos definidos.
- Diseño e implementación de un módulo para XILEMA GRHS que permita llevar un inventario de los datos de las actualizaciones y parches de seguridad instalados y pendientes en el SO.
- Ejecución de las pruebas a cada una de las funcionalidades para su posterior aplicación sobre el módulo desarrollado, y evaluar y validar la calidad del producto y para garantizar que se cumplan con las necesidades del cliente.

La investigación está sustentada en los siguientes **métodos científicos**:

- **Analítico-Sintético:** Se utilizó en la revisión de documentos y artículos, de donde se extrajeron ideas y elementos importantes vinculados con al Inventario de actualizaciones y parches de seguridad. Permitió ampliar más sobre el tema, estudiando sus particularidades, obteniendo ideas centrales y relacionándolas como un todo.
- **Histórico-Lógico:** mediante este método se logra el análisis de la evolución de sistemas similares en cuanto a Inventario de actualizaciones y parches de seguridad, de esta manera se profundizó sobre los rasgos que caracterizan a estos sistemas y en los aspectos principales para fundamentar la propuesta de solución a la problemática planteada.
- **Modelación:** Se utilizó para representar de forma simple las propiedades, características y funcionalidades detectadas en la investigación del módulo a desarrollar, mediante el uso de diagramas, relaciones entre objetos; y las actividades que intervinieron en el proceso de actualizaciones y parches de seguridad para el SO Windows.

Estructura de capítulos:

El contenido de esta investigación consta de Introducción, tres capítulos, conclusiones, referencias bibliográficas, bibliografía y anexos:

Capítulo 1: Fundamentación teórica para el módulo de inventario de actualizaciones y parches de seguridad del sistema operativo Windows en XILEMA GRHS.

En el capítulo se presentan los conceptos fundamentales relacionados con el problema a resolver, se efectúa un detallado estudio de los principales conceptos y los aspectos más importantes relacionados con los registros y control de los datos de actualizaciones y parches de seguridad instalados y pendientes en el sistema operativo Windows. Se describen la metodología de desarrollo de software y herramientas que serán utilizadas para la correcta realización de la implementación del módulo de inventario de actualizaciones y parches de seguridad instalados y pendientes para el SO en XILEMA GRHS.

Capítulo 2: Análisis y diseño de la propuesta de solución para el módulo de inventario de actualizaciones y parches del sistema operativo Windows en XILEMA GRHS.

En este capítulo se describe una presentación de la propuesta de solución a la problemática. Se determinan requerimientos funcionales y no funcionales identificados, y características del sistema. Se realiza la descripción y modelado del proceso de negocio. Se confecciona el diagrama de casos de uso a partir de los requisitos obtenidos y se definen los casos de uso y los actores que se relacionan con cada uno de ellos. Se modela el diagrama de modelo de dominio, se fundamenta el uso de patrones de diseño y se modela el diagrama de clases del diseño.

Capítulo 3: Implementación y pruebas para el módulo

En este capítulo se modelan los diagramas de despliegue y de componentes. Se presentan los resultados de las pruebas unitarias, de integración y de aceptación que se realizan a la solución y el resultado obtenido de las mismas se muestra gráficamente, así como las no conformidades detectadas durante el proceso de prueba.

Capítulo I: Fundamentación Teórica para el Módulo de Inventario de actualizaciones y parches del sistema operativo Windows en XILEMA GRHS.

Introducción:

En el presente capítulo se fundamentan los diferentes elementos teóricos que engloban la investigación para la comprensión del problema a resolver, así como el desarrollo del tema, haciendo énfasis en la captura y muestra de los datos de las actualizaciones y parches de seguridad y determinar las alternativas de solución posibles. Se profundiza en los aspectos y conceptos más importantes asociados al objeto de estudio de la investigación y se realiza un análisis detallado sobre la metodología de desarrollo de software, las herramientas y las tecnologías que complementan el desarrollo e implementación del módulo de inventario de actualizaciones y parches de seguridad instalados y pendientes en el SO.

1.1. Conceptos asociados al dominio del problema:

En el presente epígrafe se estudian los conceptos fundamentales, con el objetivo de que exista una mejor comprensión de la investigación realizada, entre ellos se destacan los siguientes:

1.1.1. Inventario

Según la Real Academia Española:

La palabra inventario significa obtener una lista de bienes pertenecientes a una persona o comunidad, hecho con orden y precisión. También se denomina inventario al documento escrito donde consta la anotación de dichos bienes.

Los inventarios sirven para saber los bienes existentes, y son muy útiles a la hora de evaluar los progresos o pérdidas patrimoniales que ocurren en un periodo de tiempo (RAE, 2010).

Inventario en el ámbito informático

Un inventario informático, consiste en hacer un listado de todo el hardware y software de una organización (Montesino Perurena, 2012).

- **Algunas actividades importantes en el proceso de realizar inventario de medios informáticos**

1. Identificar ámbito al cual realizar el inventario.

2. Guardar la información necesaria.

3. Mantener la información actualizada.

1.1.2. Seguridad Informática

La seguridad informática es una disciplina que se encarga de proteger la integridad y la privacidad de la información almacenada en un sistema informático[CITATION Pau161 \l 1033].

Es el área de la informática que se enfoca en la protección de la infraestructura computacional y todo lo relacionado con esta y, especialmente, la información contenida o circulante (Alassafi et al., 2017).

La seguridad informática trata de mantener segura la: integridad, privacidad, disponibilidad, control y autenticidad de la información, que se encuentra almacenada en una computadora (ISO/IEC, 2005; Montesino Perurena, 2012).

Considerando las descripciones planteadas anteriormente, en el contexto de este proyecto se opina que la seguridad informática se encarga de preservar la integridad, confidencialidad y la privacidad de la información almacenada en un sistema informático, impidiendo que intrusos internos o externos realicen operaciones no permitidas. Este concepto es de vital importancia porque uno de los aspectos principales de la necesidad de las actualizaciones es la seguridad.

1.1.3. Actualización de software

Una actualización es o consiste en cambiar o alterar una aplicación por una versión más actual de la misma. Esta actualización puede ser por un parche, un service-pack o una actualización completa del programa(ISO/IEC, 2005) .

Las actualizaciones permiten solucionar los problemas que se encuentran en el software que se utiliza diariamente. Cuando un error es encontrado en un programa o sistema determinado, se informa al proveedor, quien evalúa su importancia, desarrolla la solución normalmente llamada “parche”- y luego lanza públicamente la actualización lo cual permite la corrección de errores y el sellado de vulnerabilidades (Amin & Valverde, 2017).

Es de gran importancia mantener todos los sistemas actualizados ya que es el único modo de evitar problemas de vulnerabilidad en el funcionamiento del SO, aplicaciones y programas de nuestro ordenador. El registro y control de esta información sirve como método de prevención de muchos incidentes de seguridad.

1.1.4. Parche

En informática, un **parche** consta de cambios que se aplican a un programa, para corregir errores, agregarle funcionalidad, actualizarlo.

Si bien los parches suelen ser desarrollados por programadores ajenos a los autores iniciales del proyecto, esto no siempre es así. Un parche puede ser aplicado tanto a un binario ejecutable como al código fuente de cualquier tipo de programa, incluso, un sistema operativo o una aplicación para dispositivos móviles (ISO/IEC, 2005).

Tipos según su propósito (Wolden et al., 2015)

- **Parches de seguridad**

Los parches de seguridad solucionan agujeros de seguridad y, siempre que es posible, no modifican la funcionalidad del programa. Los parches de seguridad son especialmente frecuentes en aplicaciones que interactúan con Internet.

- **Parches de actualización**

Consiste en modificar un programa con el objetivo de incorporar metodologías más nuevas. Por ejemplo, optimizar en tiempo cierto programa, utilizar algoritmos mejorados, añadir funcionalidades, eliminar secciones obsoletas de software, etc.

1.2. Estudio de soluciones existentes que sirven de guía para el desarrollo del módulo*.*

En la actualidad existe gran diversidad de sistemas actualizadores y sistemas de inventario relacionados con las actualizaciones y parches de seguridad, con los que se facilita la seguridad de los SO.

Dado a que el objetivo de la investigación es implementar un módulo para el XILEMA GRHS, el estudio de estos sistemas se centra en como ellos verifican y procesan los datos de las actualizaciones en el SO Windows.

A continuación, se mencionan y describen algunos de los sistemas de actualización desarrollados en las últimas décadas. Los cuales se analizaron para determinar cómo es que ellos realizan el proceso de actualización, qué información es la que se obtiene del SO y donde se almacena dicha información. Todos ellos aplican su propio mecanismo de actualización y no todos pertenecen al mismo grupo de actualizadores:

1.2.1. Windows Update

A partir de la versión Windows XP, Microsoft desarrolló Microsoft Update, herramienta que facilita la gestión automatizada de las actualizaciones no únicamente para Windows, sino para el paquete Microsoft Office. Es exclusivamente para el SO Windows (Microsoft, 2015).

Esta herramienta chequea constantemente en busca de nuevas actualizaciones disponibles y las instala, busca nuevas versiones de programas que pueden mejorar el entorno de trabajo y notifica su existencia, se centra en determinar las vulnerabilidades del SO e instalar las actualizaciones que las corrijan, sin tener en cuenta las actualizaciones a otros programas que no pertenezcan a la gama de productos Microsoft Office.

1.2.2. LiveUpdate

Con el clic de un botón, LiveUpdate lanza un asistente que guía al usuario a través del proceso de actualización de software. En primer lugar, detecta la disponibilidad de una conexión a Internet o módem y automáticamente se conecta a un servidor de Symantec**,** donde *Symantec Corporation* es una corporación internacional que desarrolla y comercializa software para computadoras, particularmente en el dominio de la seguridad informática. Con la sede central en *Mountain View, California, Symantec* opera en más de cuarenta países. Una vez que LiveUpdate se conecta al sitio de Symantec, el sistema descargará la información de archivos correspondientes a ese producto, eliminando la necesidad de los usuarios de desplazarse a través de la información que no se aplica a su producto. Si un usuario utiliza LiveUpdate para Norton Antivirus, LiveUpdate tendrá que localizar y descargar toda nueva información relacionada con las definiciones de virus o cualquier otra información de Norton Antivirus. Una vez que el usuario confirma que quieren ejecutar la actualización, entonces se completa la operación. Para aumentar aún

más la productividad, funciona en segundo plano para que los usuarios puedan seguir trabajando mientras el programa se está actualizando (Symantec Corporation, 2014).

1.2.3. Appupdater

Es otra de las herramientas de instalación y actualización de programas que se maneja en línea de comandos. No hay diferencia con los comandos apt-get o yum de Linux, que son la vía tradicional de instalación de aplicaciones en este sistema operativo. Appupdater actualiza periódicamente la base de datos con información sobre cientos de aplicaciones para que el usuario la consulte e instale. Además de ser multiplataforma, incluye soporte para dispositivos USB y se pueden sugerir nuevas aplicaciones que se incluirán en la lista de aplicaciones soportadas. Crear un repositorio personalizado si se desea es otra de las posibilidades que brinda Appupdater(Nabber, 2014)

1.2.4. Gupdater 1.0

Es un software desarrollado con Python 2.7, es un actualizador automático desarrollado para actualizar el cliente GRHS (gclient). Se instala en cada cliente donde está ejecutándose Gclient y es el encargado de actualizarlo automáticamente.

Almacena en registro de logs las trazas generadas en el proceso de ejecución, detecta automáticamente la última versión disponible del software instalado, realiza actualizaciones automáticas puesto que no interviene un usuario en el proceso de actualización, es multiplataforma con soporte para Windows y Linux.

También existen varias herramientas de inventario de software y hardware actualmente en el mercado, estas realizan principalmente el inventario de software instalados en los ordenadores (PC), permitiendo generalmente, descubrir y eliminar todo programa que ha sido instalado sin permiso, aunque es necesario destacar que ninguna de estas herramientas realiza monitorización en las PC clientes. Muchas de estas herramientas son multiplataforma, mientras que otras solo pueden realizar el inventario en un único sistema operativo.

Algunos de estos sistemas gestores de hardware y software tienen implementado un módulo para el control de las actualizaciones del SO. Por esta razón son analizados, para ver cómo es que se realiza esa gestión, o sea, qué secuencia es la que se sigue para revisar los datos de las actualizaciones, qué datos son los que se solicitan, y de donde es

que se toman. A continuación, se exponen características principales de las herramientas de inventario de software identificadas como respaldo a la siguiente investigación:

1.2.5. Software Libre de Inventario de Hardware y Software.

- **GLPI *(*Free Computer Equipment Manager)**

GLPI, es una herramienta web en software libre que ofrece una gestión integral del inventario informático de una empresa además de incluir un sistema de gestión de incidencias.

La herramienta está desarrollada para entornos Apache-PHP-MySQL, por lo que puede ser instalada tanto en servidores Windows como Linux y su fácil instalación y manejo permite gestionar todo el soporte y mantenimiento de una empresa de una manera rápida y sencilla, por lo que el despliegue y la puesta en marcha son bastante reducidos («Gartner», 2019)

Principales características

- Gestión de información comercial y financiera (compra, la garantía y la extensión, la amortiguación)
- Equipo de gestión de estado
- Gestión de las solicitudes de asistencia de todo tipo de inventario de equipos
- Interfaz para permitir al usuario a presentar un ticket de soporte
- Inventario de equipos, periféricos, impresoras de red y los componentes asociados a través de una interfaz con OCS Inventory o FusionInventory.
- Gestión de Preguntas Frecuentes (FAQ).
- Informe del generador: hardware, red o intervenciones (de apoyo).

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

Por otra parte, GLPI tiene muchos plugins [2]para agregar características adicionales.

En resumen, GLPI es una herramienta libre y multiplataforma, muestra las incidencias de los últimos eventos ocurridos y permite la gestión de notificaciones, sin embargo, estas

2 Un plugin es aquella aplicación que en un programa informático añade una funcionalidad o una nueva característica al software.

notificaciones no son parte del proceso automatizado de la aplicación, sino notificaciones realizadas por los usuarios de la red al servidor.

- **OCS Inventory**

Open Computer and Software Inventory Next Generation (OCS Inventory) es un software libre que permite a los administradores de TI (Tecnología de Información) gestionar el inventario de sus activos de TI. OCS-NG recopila información sobre el hardware y software de equipos que hay en la red que ejecutan el programa de cliente OCS ("agente OCS de inventario"). OCS Inventory puede utilizarse para visualizar el inventario a través de una interfaz web (OCS, 2010).

Es un sistema para mantener el inventario de máquinas de forma fácil y automatizada. Permite disponer en forma rápida y completa de los datos de los equipos y su relación con los usuarios. OCS Inventory está basado en software libre y publicado bajo la licencia GNU General Public License, versión 2.0.

OCS se basa en los estándares vigentes. El diálogo entre los equipos clientes y el servidor se fundamentan en HTTP (Hypertext Transfer Protocol) y el formato de los datos se realiza en XML.

Características

- Cuenta con una funcionalidad de búsqueda que permite filtrar datos como programas instalados, memoria RAM, redes, entre otros.
- El agente debe ser instalado y configurado en cada servidor o computadora a ser inventariada.
- Permite agrupar los servidores por diferentes criterios (similar a la búsqueda).
- El instalador del agente se encuentra disponible para Windows y Linux.
- Análisis de la red.
- La interfaz web muestra el detalle de cada servidor incluyendo:

✓ Hardware: CPU, RAM, red, placa madre, [video]], sonido, entre otros.
✓ Software: Sistema, programas instalados.
✓ Detalles de red o redes activas.
✓ Versión de BIOS.

1.2.6. *S*oftware Propietarios de Inventario de Hardware y Software.

- **Login Inventory**

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

Login Inventory solo se puede instalar en Windows NT, XP y 7, sin embargo, puede realizar también la recopilación de datos de ordenadores con sistema operativo GNU/Linux y Mac OS. Este presenta características importantes, como son (LPG, 2009):

- Realizar inventario de estaciones conectadas a la red en pocos minutos.

- Mostrar, a través de Microsoft Management Console o a través de la interfaz web de Login Inventory, toda la información de las estaciones a las que se les realiza inventario.

- No existe límite de dispositivos u ordenadores en la red.

- **Elementos a considerar en el módulo a desarrollar a partir del análisis de las soluciones existentes.**

Con el análisis realizado a los sistemas actualizadores y sistemas de inventarios, se acoplaron las características de cada uno de ellos y obtener una solución. Se determinó tomar algunas características de los mismos que son de gran utilidad para añadirlas al módulo a desarrollar, el cual permitirá obtener, analizar y mostrar los datos de las actualizaciones y parches de seguridad instalados y pendientes en los sistemas operativos. Estas características son:

- De los sistemas actualizadores se toma los datos con que se trabaja en las actualizaciones, que son en esencia lo que debe recoger en el inventario a realizar, estos datos son son:

- nombre de las actualizaciones
- la fecha de expiración de las actualizaciones
- si el sistema está o no actualizado
- si el sistema está o no habilitado para dicha actualización.

- De los sistemas de inventarios se observa los datos obtenidos de las actualizaciones se muestran mediante una tabla, que es en sí como se debe mostrar en el módulo. Además de almacenar rasgos en cuanto a:
  - No existe límite de dispositivos u ordenadores en la red (Login Inventory).
  - Facilita la prevención de ataques de seguridad (OCS Inventory).
  - Equipo de gestión de estado (GLPI)

1.3. Metodología de desarrollo de software

Las metodologías de desarrollo de software pueden considerarse como una base necesaria para la ejecución de cualquier proyecto de desarrollo de software que se considere serio, y que necesite sustentarse en algo más que la experiencia y capacidades de sus programadores y equipo.(Avison & Fitzgerald, 2003; Tinoco Gómez, Rosales López, & Salas Bacalla, 2010)

Para el desarrollo del proyecto se selecciona:

1.3.1. Metodología AUP versión para la UCI

AUP-UCI propone para el ciclo de vida de los proyectos en la UCI tener siete disciplinas, los flujos de trabajo son los siguientes: modelado de negocio, requisitos, análisis y diseño, implementación, pruebas internas, pruebas de liberación y pruebas de aceptación(T.R, 2015).

Además, está compuesta por cuatro escenarios:

- El **escenario uno** propone a los proyectos que modelan el negocio con casos de usos del negocio que solo pueden modelar el sistema con casos de usos del sistema.
- El **escenario dos** propone modelar el negocio con un modelo conceptual y el sistema con casos de uso del sistema.
- El **escenario tres** propone modelar el negocio con descripción de proceso de negocio, junto al modelo conceptual y el sistema mediante la descripción de requisitos por proceso.
- El **escenario cuatro** propone no modelar el negocio y describir el sistema mediante historias de usuario.

Se selecciona para esta investigación el escenario número dos ya que este se aplica a los proyectos que hayan evaluado el negocio a informatizar y como resultado obtengan que no sea necesario incluir las responsabilidades de las personas que ejecutan las actividades, de esta forma modelarían exclusivamente los conceptos fundamentales del negocio. Se recomienda este escenario para proyectos donde el objetivo primario es la gestión y presentación de información.

1.4. Lenguaje de programación

- **Python:**

Python es un lenguaje de programación poderoso y fácil de aprender. Cuenta con estructuras de datos eficientes y de alto nivel y un enfoque simple pero efectivo a la programación orientada a objetos(PyCharm, 2015).

Python es seleccionado como lenguaje para el desarrollo porque la infraestructura de XILEMA GRHS se basa en este lenguaje. También porque es posible usarlo de muchas maneras en el desarrollo de software, es capaz de realizar desde aplicaciones de servidores de red hasta páginas web. Aparte de admitir módulos y paquetes, lo que fomenta el modularidad del programa y la reutilización de código.

1.5. Lenguaje de Modelado.

- **UML**

UML facilita un vocabulario y reglas que permiten realizar una comunicación, cuyo objetivo principal es representar gráficamente un sistema.

Este tipo de modelado permite manejar la complejidad de los sistemas a analizar o diseñar un Diagrama es la representación gráfica de un conjunto de elementos con sus relaciones.

Se selecciona ULM como el lenguaje que se utilizará para modelar los diagramas y modelos necesarios en el desarrollo del software, tales como: modelo conceptual, modelo de base de datos, de clases y de despliegue propios del módulo.

1.6. Tecnologías utilizadas

En el presente trabajo se van a utilizar las mismas tecnologías con que está desarrollado el sistema XILEMA GRHS ya que la solución propuesta es un módulo integrado al sistema.

1.6.1. Marco de trabajo del lado del servidor

- **Django:**

Django como marco de trabajo Web esta implementado sobre el lenguaje de programación Python. Django brinda estructura al código fuente, impulsando las buenas prácticas de desarrollo Web, lo que fomenta un código comprensible y fácil de mantener(Holovaty, Kaplan-Moss, Jacob, & Dunck, 2008).

Se selecciona Django como el marco de trabajo Web, el cual junto al lenguaje de programación Python trabaja desde el Modelo del patrón MTV para la relación almacenamiento-obtención de datos entre el servidor y la base de datos.

Django pone énfasis en la reutilización, la conectividad y extensibilidad de componentes, y el desarrollo rápido(Django, 2014).

Cuenta con un potente gestor de plantillas HTML que permiten reutilizar la estructura establecida en una plantilla base, modificando solo los bloques necesarios en las plantillas hijas. Permite la conexión con el sistema gestor de base de datos PostgreSQL, que es de código abierto y libre de licencias comerciales. Permite la creación de múltiples aplicaciones dentro de un proyecto, lo que posibilita la inserción de nuevos módulos sin realizar cambios en los existentes. Junto al lenguaje de programación Python trabaja

desde el modelo del patrón MTV para la relación almacenamiento-obtención de datos entre el servidor y la base de datos.

1.6.2. Marco de trabajo del lado del cliente

- **Xilema Base Web:**

Xilema Base Web es un marco de trabajo desarrollado en el Centro de TLM que está constituido por Django como marco de trabajo base, además implementa las pautas de diseño de la Universidad. Tiene como propósito contribuir al desarrollo de nuevas aplicaciones web logrando una homogeneidad entre todos los productos del Centro TLM(GRHS, 2015).

1.7. Herramientas utilizadas

1.7.1. Visual Paradigm for UML

Visual Paradigm para UML es una herramienta para desarrollo de aplicaciones utilizando modelado UML.

Posibilita la representación gráfica de los diagramas permitiendo ver el sistema desde diferentes perspectivas como: componentes, despliegue, secuencia casos de uso; clase, actividad, estado, entre otros(Paradigm, 2013).

A continuación, se muestran las razones de elección:

- Disponibilidad en múltiples plataformas: Microsoft Windows (98, 2000, XP, o Vista) y Linux.

- Permite modelar diagramas de base de datos.

- Se pueden realizar diagramas de despliegue del producto, donde se plantea la distribución física que debe tener la solución.

1.7.2. Sistema Gestor de Base de Datos (PostgreSQL)

Un sistema gestor de bases de datos (SGBD) es un conjunto de programas que permiten crear y mantener una base de datos, asegurando su integridad, confidencialidad y seguridad(«¿Qué es un gestor de datos y para qué sirve?», 2018).

PostgreSQL es un sistema de administración de bases de datos de propósito general y objeto-relacional, el sistema de base de datos de código abierto más avanzado(«PostgreSQL-ES», 2018).

Entre sus principales ventajas destacan:

- Ampliamente popular e ideal para tecnologías web.
- Fácil de administrar.
- Su sintaxis SQL es estándar y fácil de aprender.
- Es multiplataforma.

Se selecciona PostgreSQL V.96, a partir de las ventajas descritas anteriormente, como el gestor de base de datos que se utilizará para el manejo y obtención de los datos referentes al módulo a desarrollar y principalmente porque la versión actual del software XILEMA GRHS funciona haciendo uso de dicho gestor.

- **pgAdmin**

PgAdmin 4 es una aplicación gráfica para gestionar el gestor de bases de datos PostgreSQL, siendo la más completa y popular con licencia a código abierto. Está diseñado para responder a las necesidades de los usuarios, desde escribir consultas SQL simples hasta desarrollar bases de datos complejas. La interfaz gráfica soporta todas las características de PostgreSQL y facilita la administración. La conexión al servidor puede hacerse mediante conexión TCP/IP y puede encriptarse mediante SSL (Secure Socket Layer), un protocolo de seguridad que hace que sus datos viajen de manera íntegra y segura, es decir, la transmisión de los datos entre un servidor y usuario web, y en retroalimentación, es totalmente cifrada o encriptada(PgAdmin, 2018).

1.7.3. Entorno de Desarrollo Integrado

PyCharm 2018.3.5

Es un IDE o entorno de desarrollo integrado multiplataforma utilizado para desarrollar en el lenguaje de programación Python. Proporciona análisis de código, depuración gráfica, además brinda soporte para el desarrollo web con Django(andrearrs, 2018).

1.8. Conclusiones parciales

En el presente capítulo se analizaron de los conceptos fundamentales relacionados con las actualizaciones y parches de seguridad del SO Windows, lo que posibilitó un mayor entendimiento del tema investigado. De los sistemas estudiados se determinó que no pueden ser integrados a la plataforma XILEMA GRHS porque sus funcionalidades no cumplen con las necesidades actuales que requiere el sistema para la solución propuesta, pero permitió identificar las características principales de las actualizaciones y parches de seguridad de SO. Se definieron las herramientas fundamentales para el desarrollo de dicho módulo y se seleccionó AUP-UCI como la metodología de desarrollo a utilizar. Se decidió utilizar Python 2.7 como lenguaje de programación, PyCharm como IDE de desarrollo y el marco de trabajo para el desarrollo web Django, por ser las tecnologías en las que se desarrolló el sistema XILEMA GRHS. Como sistema gestor de bases de datos se utilizó PostgreSQL y la notación UML para el modelado del proceso de negocio del módulo a implementar. Llevando a cabo todo lo planteado se logró determinar las funcionalidades a añadir al módulo.

Capitulo No. 2: Análisis y diseño del módulo de inventario de actualizaciones y parches de seguridad del sistema operativo Windows en XILEMA GRHS.

Introducción

Con el objetivo de establecer una visión general de lo que se debe realizar, se desarrolla el siguiente capítulo donde se exponen los principales aspectos relacionados con la solución propuesta. Se realizan el modelo conceptual y se elabora la descripción del mismo. Para definir las funcionalidades que tendrá dicha aplicación se generán los artefactos relacionados con la metodología utilizada para el desarrollo de la solución como son la especificación de los requisitos funcionales y no funcionales, así como la descripción y representación de los casos de uso del sistema. Estos artefactos incluyen la descripción de la arquitectura, el estilo arquitectónico y los patrones de diseño empelados.

2.1. Descripción de la solución propuesta

Como propuesta de solución se plantea realizar un módulo para el XILEMA GRHS que permita el inventario de los datos de las actualizaciones y parches de seguridad instalados y pendientes en el SO Windows, y comparar las actualizaciones pendientes que posee la máquina, es decir determinar si en servidor esta actualizado o no. Para esto se debe tener presente q XILEMA GRHS está compuesto por un servidor GRHS(gserver), un Cliente GRHS(gclient), y un Administrador(gadmin), donde para establecer la comunicación entre el gserver y el gclient se realiza mediante un administrador de cola de tareas (RabbitMQ).

En el gclient es donde se realiza el módulo, el cual será el encargado de trabajar en el inventario de las actualizaciones y parches de seguridad del SO. El inventario posibilitará el análisis de las actualizaciones realizadas, así como la comparación con el gserver y notificarlas. Para esto se crean 2 clases, una donde se pondrán los datos del módulo, o sea los atributos a mostrar, y otra donde se captura la información del SO referente a las actualizaciones y parches de seguridad del SO. Seguido se crea una tarea, y se envía al RabbitMQ, donde se ejecuta, se manda al gserver, y este guarda esa información en la Base de Datos(BD). En el gadmin cuando se acede al módulo, mediante un api obtienen los datos de las actualizaciones y parches de seguridad, y se muestran: el nombre de

dicha actualización, la fecha de expiración de las mismas, si el sistema está actualizado y si está habilitado.

2.2. Modelo conceptual

Un modelo conceptual tiene como propósito fundamental organizar y representar, de manera semi formal y uniforme, el conocimiento de un área o campo especifico asociado a un sistema de gestión o de información, y como objetivo comprender y describir solamente las clases más importantes dentro del contexto en el cual se desempeña el software(Larman, 2003).

Por esta razón para crear un modelo conceptual es suficiente con una buena definición y explicación de conceptos o entidades de negocio y de sus relaciones.

A continuación, en la figura se muestra el modelo conceptual desarrollado para describir desde una forma general la relación conceptual entre la UCI y sus centros de desarrollo, hasta llegar al punto particular: módulo Parche Security.

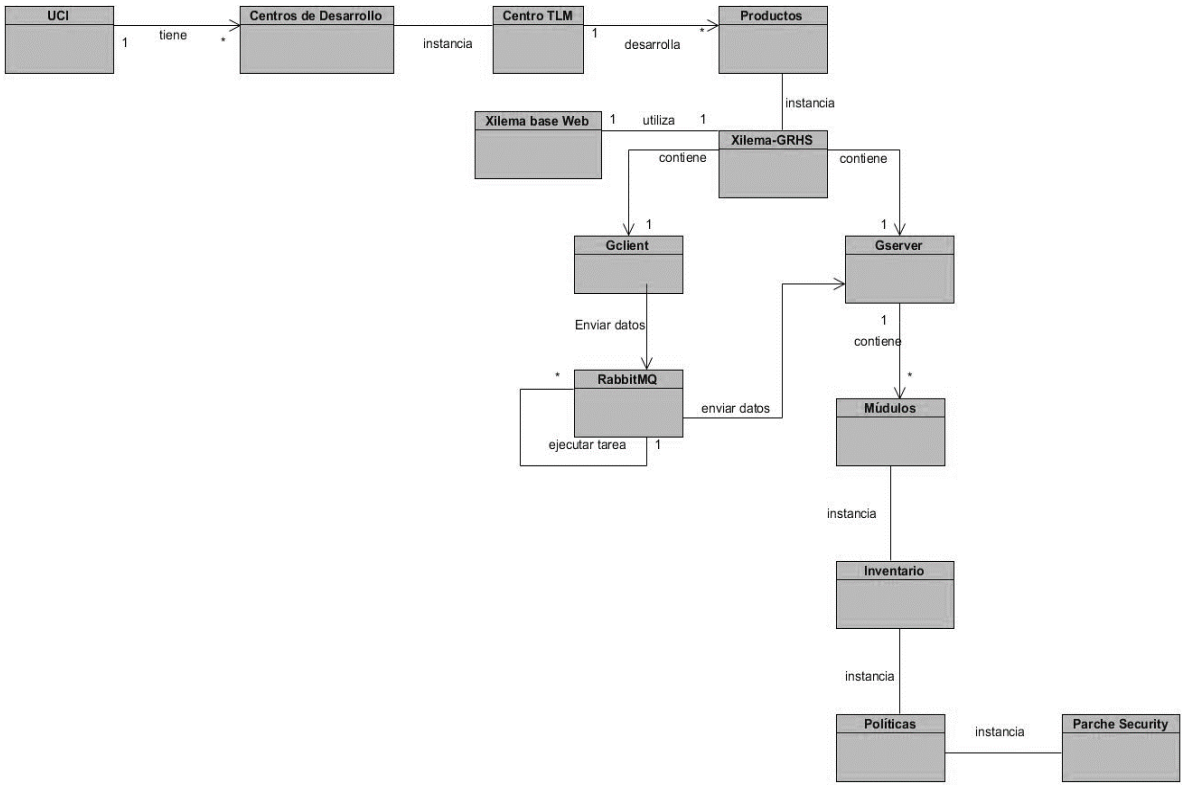


Figura 1 Modelo conceptual

2.2.1. Descripción del modelo conceptual

A continuación, se describen los conceptos empleados en el diagrama mostrado:

- gclient: Captura y envía los datos del inventario al Gserver mediante el RabbitMQ.
- gserver: Procesa los datos para generar el inventario.
- RabbitMQ: Establece la comunicación entre el Gserver y el Gclient, ejecuta las tareas creadas en el Gclient a partir de la capturara de los datos referentes a las actualizaciones y parches de seguridad en el SO.
- Parche Security: Nombre del módulo creado.

2.3. Especificación de requisitos de software

Los requisitos se clasifican en dos tipos; los requisitos funcionales y los requisitos no funcionales. Los requisitos funcionales describen lo que el sistema debe hacer. Mientras que los requisitos no funcionales son aquellos que no se refieren directamente a las funciones específicas que proporciona el sistema, sino a las propiedades emergentes de este como la fiabilidad, el tiempo de respuesta y la capacidad de almacenamiento(PRESSMAN, 2014).

2.3.1. Requisitos funcionales

Los requerimientos funcionales son declaraciones de los servicios que debe proporcionar el sistema, de la manera en que éste debe reaccionar a entradas particulares y de cómo se debe comportar en situaciones particulares.

A continuación, se presentan los requisitos funcionales que se deben cumplir para la solución planteada:

Tabla 1 Requisitos funcionales

	Requisitos generales
No.	**Nombre**
RF1	Mostrar los nombres de las actualizaciones existentes en el SO Windows para XILEMA GRHS.
RF2	Mostrar la fecha de expiración de las actualizaciones existentes en el SO Windows para XILEMA GRHS.
RF3	Determinar si el sistema operativo está actualizado o no
RF4	Determinar si el sistema operativo está habilitado o no
RF5	Filtrar la lista de elementos (¿nombre, fecha de expiración, actualizado?, habilitado?) a inventariar.

2.3.2. Requisitos no funcionales

"Los requisitos no funcionales son los requerimientos que no se refieren directamente a las funciones específicas que proporciona el sistema, sino a las propiedades emergentes de este, como la fiabilidad y el tiempo de respuesta". En resumen, los requisitos no funcionales representan aquellas cualidades que debe poseer el sistema pero que no son funcionalidades específicas.

Como el módulo se integrará en el Xilema -GRHS, los requerimientos con que debe cumplir son los mismos que el sistema tiene definidos.

A continuación, se muestran los requisitos no funcionales obtenidos:

- **Usabilidad**

- RNF. El módulo debe ser fácil de entender.

- RNF. El módulo debe realizar todas las operaciones solicitadas.

Este requisito no funcional posee varios sub-atributos, los cuales se especifican a continuación:

Tabla 2 Especificación del requisito no funcional Usabilidad: comprensibilidad

Atributo de Calidad	Usabilidad
Sub-atributos/Sub-características	Comprensibilidad
Objetivo	Lograr que el módulo sea entendible.
Origen	El cliente
Artefacto	Módulo

Entorno	El módulo está funcionando correctamente.
Estímulo	**Respuesta: Flujo de eventos (Escenarios)**
NA	NA
Medida de respuesta	
NA	

Tabla 3 Especificación del requisito no funcional Usabilidad: operabilidad

Atributo de Calidad	Usabilidad
Sub-atributos/Sub-características	Operabilidad
Objetivo	Lograr que se realicen todas las operaciones solicitadas por el cliente.
Origen	El cliente
Artefacto	Módulo
Entorno	El módulo está funcionando correctamente.
Estímulo	**Respuesta: Flujo de eventos (Escenarios)**
NA	NA
Medida de respuesta	
NA	

Tabla 4 Especificación del requisito no funcional Usabilidad: Navegabilidad

Atributo de Calidad	Usabilidad
Sub-atributos/Sub-características	Navegabilidad
Objetivo	Garantizar que no halla falla en la comunicación entre el gserver y el gclient, entre el gserver y el navegador.
Origen	El cliente
Artefacto	Módulo
Entorno	El módulo está funcionando correctamente.
Estímulo	**Respuesta: Flujo de eventos (Escenarios)**
NA	NA
Medida de respuesta	
NA	

- **Funcionalidad**

- RFN. El módulo tiene el objetivo principal mantener un inventario de las actualizaciones y parches de seguridad instalados y pendientes en el SO para las computadoras con XILEMA GRHS.

En este requisito no funcional existen diversos sub-atributos, que se describen a continuación

Tabla 5 Especificación del requisito no funcional Funcionabilidad: precisión

Atributo de Calidad	Funcionabilidad
Sub-atributos/Sub-características	Precisión
Objetivo	Verificar que el módulo proyecte resultados o efectos acordes a las necesidades para las cuales fue creado.
Origen	Cliente
Artefacto	Módulo
Entorno	El módulo está funcionando correctamente
Estímulo	**Respuesta: Flujo de eventos (Escenarios)**
NA	NA
Medida de respuesta	
NA	

- **Eficiencia**

La eficiencia se mide como los recursos gastados por el usuario en relación con la precisión y la integridad de los objetivos logrados. Se logra una alta eficiencia cuando el usuario alcanza sus objetivos mientras gasta la menor cantidad posible de recursos.

Tabla 6 Especificación del requisito no funcional Eficiencia: Utilización de recursos

Atributo de Calidad	Eficiencia
Sub-atributos/Sub-características	Utilización de recursos
Objetivo	La cantidad mínima de RAM es 256 MG. La capacidad mínima de disco duro a utilizar el servidor y servidor de base de datos es 20GB.
Origen	Externo al módulo
Artefacto	Hardware del cliente y los servidores
Entorno	Operación normal / Modo degradado
Estímulo	**Respuesta: Flujo de eventos (Escenarios)**
NA	NA

Medida de respuesta
NA

- **Portabilidad**

- RNF. El módulo debe ser adaptable a diferentes entornos especificados.

- RNF. El módulo debe poderse instalar en un entorno especificado.

Tabla 7 Especificación del requisito no funcional Portabilidad: Instalabilidad

Atributo de Calidad	Portabilidad
Sub-atributos/Sub-características	Instalabilidad
Objetivo	Lograr que el módulo se pueda este en un entorno especificado.
Origen	El cliente
Artefacto	Módulo
Entorno	Funcionando correctamente / Modo degradado
Estímulo	**Respuesta: Flujo de eventos (Escenarios)**
1.a Se instala el sistema en diferentes entornos.	
	El módulo funciona correctamente.
Medida de respuesta	
NA	

Tabla 8 Especificación del requisito no funcional Portabilidad: Adaptabilidad

Atributo de Calidad	Portabilidad
Sub-atributos/Sub-características	Adaptabilidad
Objetivo	Lograr que el módulo sea adaptable a diferentes entornos especificados.
Origen	El cliente
Artefacto	El módulo
Entorno	Funcionando correctamente
Estímulo	**Respuesta: Flujo de eventos (Escenarios)**
1.a Se prueba el sistema en diferentes entornos.	
	El módulo funciona correctamente.
Medida de respuesta	
2 segundos	

- **Mantenibilidad**

Tabla 9 Especificación del requisito no funcional Mantenibilidad: Cambiabilidad

Atributo de Calidad	Mantenibilidad
Sub-atributos/Sub-características	Cambiabilidad
Objetivo	Facilitar la correcta modificación del módulo en caso de ser necesario.
Origen	El desarrollador
Artefacto	El módulo
Entorno	Funcionando correctamente / Modo degradado
Estímulo	**Respuesta: Flujo de eventos (Escenarios)**
1.a El desarrollador realiza una modificación en el sistema	
	El módulo asimila correctamente la modificación realizada.
Medida de respuesta	
NA	

Tabla 10 Especificación del requisito no funcional Mantenibilidad: Estabilidad

Atributo de Calidad	Mantenibilidad
Sub-atributos/Sub-características	Estabilidad
Objetivo	Lograr el correcto funcionamiento del módulo ante un cambio que genere algún fallo.
Origen	El desarrollador
Artefacto	El módulo
Entorno	Funcionando correctamente / Modo degradado
Estímulo	**Respuesta: Flujo de eventos (Escenarios)**
1.a El desarrollador realiza una modificación en el sistema	
	El módulo notifica la ocurrencia de un fallo ante una modificación previamente realizada. El módulo se mantiene funcionando correctamente.
Medida de respuesta	
1 segundo	

Tabla 11 Especificación del requisito no funcional Mantenibilidad: Cumplimiento de mantenibilidad

Atributo de Calidad	Mantenibilidad
Sub-atributos/Sub-características	Cumplimiento de mantenibilidad
Objetivo	Garantizar la actualización de las tecnologías de desarrollo
Origen	El desarrollador
Artefacto	El módulo
Entorno	Funcionando correctamente / Modo degradado
Estímulo	**Respuesta: Flujo de eventos (Escenarios)**
1.a El desarrollador realiza una modificación en el sistema	
	El módulo asimila correctamente la modificación realizada.
Medida de respuesta	
NA	

- **Restricciones de diseño.**

- RFN. La codificación se regirá mediante el estilo de codificación establecido para XILEMA GRHS. Estándar definido por el proyecto para garantizar un mejor soporte.

Tabla 12 Especificación del requisito no funcional Restricciones de diseño.

Atributo de Calidad	No Aplica
Sub-atributos/Sub-características	Restricciones de diseño
Objetivo	Para el correcto funcionamiento del módulo se necesitan como tecnologías a utilizar: Python, Django, PostgreSQL, PyCharm. Navegadores: Chrome 65.0, Firefox 59.2.
Origen	Externo al módulo
Artefacto	Código del módulo
Entorno	Operación normal
Estímulo	**Respuesta: Flujo de eventos (Escenarios)**
NA	NA

- **Confiabilidad:**

- RNF. El módulo debe ser capaz de recuperarse ante fallos.

- RNF. El módulo debe ser capaz de prevenir el acceso no autorizado a los datos.

Tabla 13 Especificación del requisito no funcional Confiabilidad: Tolerancia a fallos

Atributo de Calidad	Confiabilidad
Sub-atributos/Sub-características	Tolerancia a fallos
Objetivo	Lograr que el módulo sea capaz de recuperarse ante fallos.
Origen	Interno al módulo
Artefacto	Servicios del módulo (cliente y servidor) / Canales de comunicación
Entorno	Operación normal / Modo degradado
Estímulo	**Respuesta: Flujo de eventos (Escenarios)**
1.a Interrupción de comunicaciones de red en la PC	
	Tratar de conectarse cada cierto tiempo para terminar el proceso de comunicación / Continuar funcionando en modo normal o degradado
Medida de respuesta	
2 minutos	
2.a Interrupción de comunicaciones de red en la PC servidor	
	Tratar de conectarse cada cierto tiempo para terminar el proceso de comunicación / Continuar funcionando en modo normal o degradado
Medida de respuesta	
2 minutos	
3.a Inactividad del cliente	
	El servidor se encargará de notificar los clientes inactivos / Continuar funcionando en modo normal o degradado
Medida de respuesta	
2 segundos	
4.a Ocurrencia de una excepción	
	Se notifica al usuario / Continuar funcionando en modo normal o degradado
Medida de respuesta	
2 segundos	

Tabla 14 Especificación del requisito no funcional Confiabilidad: Cumplimiento de fiabilidad

Atributo de Calidad	Confiabilidad
Sub-atributos/Sub-características	Cumplimento de fiabilidad
Objetivo	Prevenir el acceso no autorizado, ya sea accidental o premeditado, a los datos.
Origen	El cliente
Artefacto	El módulo
Entorno	El cliente desea obtener información sobre los datos de las actualizaciones y parches de seguridad procesados por el módulo.
Estímulo	**Respuesta: Flujo de eventos (Escenarios)**
1.a Persona no autorizada intenta acceder a los datos del sistema.	
	Bloquear el acceso al módulo
Medida de respuesta	
1 segundo	
2.a Persona con permisos restringidos intenta acceder a información a la que no tiene acceso.	
	Bloquear el acceso a estos datos.
Medida de respuesta	
1 segundo	

2.4. Definición de los casos de uso del Sistema

Los casos de uso son descripciones funcionales del sistema; describen cómo los actores pueden usar un sistema(Mediavilla, 2010).

2.4.1. Definiciones de actores

Actores del sistema.

Un actor es alguien o algo que interactúa con el sistema, pero que es externo al sistema(Mediavilla, 2010).

Un actor es una agrupación uniforme de personas, sistemas o máquinas externas al módulo que se está desarrollando, se relaciona con éste ya que solicita sus funcionalidades. En el Módulo interactúan dos actores, los cuales se define a continuación:

Actores	Justificación
gclient	Contiene parte del trabajo para realizar el inventario de actualizaciones y parches de seguridad en el SO Windows para XILEMA GRHS en el cual se ejecutarán varias tareas como: mostrar los nombres de las actualizaciones existentes, mostrar la fecha de expiración de las actualizaciones existentes, determinar si el SO está actualizado o no, determinar si el SO está habilitado o no para dicha actualización, filtrar la lista de elementos a inventariar y generar alertas de sistema desactualizado.

2.4.2. Diagrama de casos de uso del sistema

Un diagrama de casos de uso del sistema representa gráficamente a los procesos y sus interacciones con los actores.

Los casos de uso reflejan lo que los usuarios necesitan, lo cual se capta cuando se modela el negocio y se representa a través de los requisitos. Los mismos guían el proceso de desarrollo, ya que los modelos que se obtienen como resultado de los diferentes flujos de trabajo, representan la realización de estos casos.

Cada requisito funcional se encuentra identificado con un caso de uso.

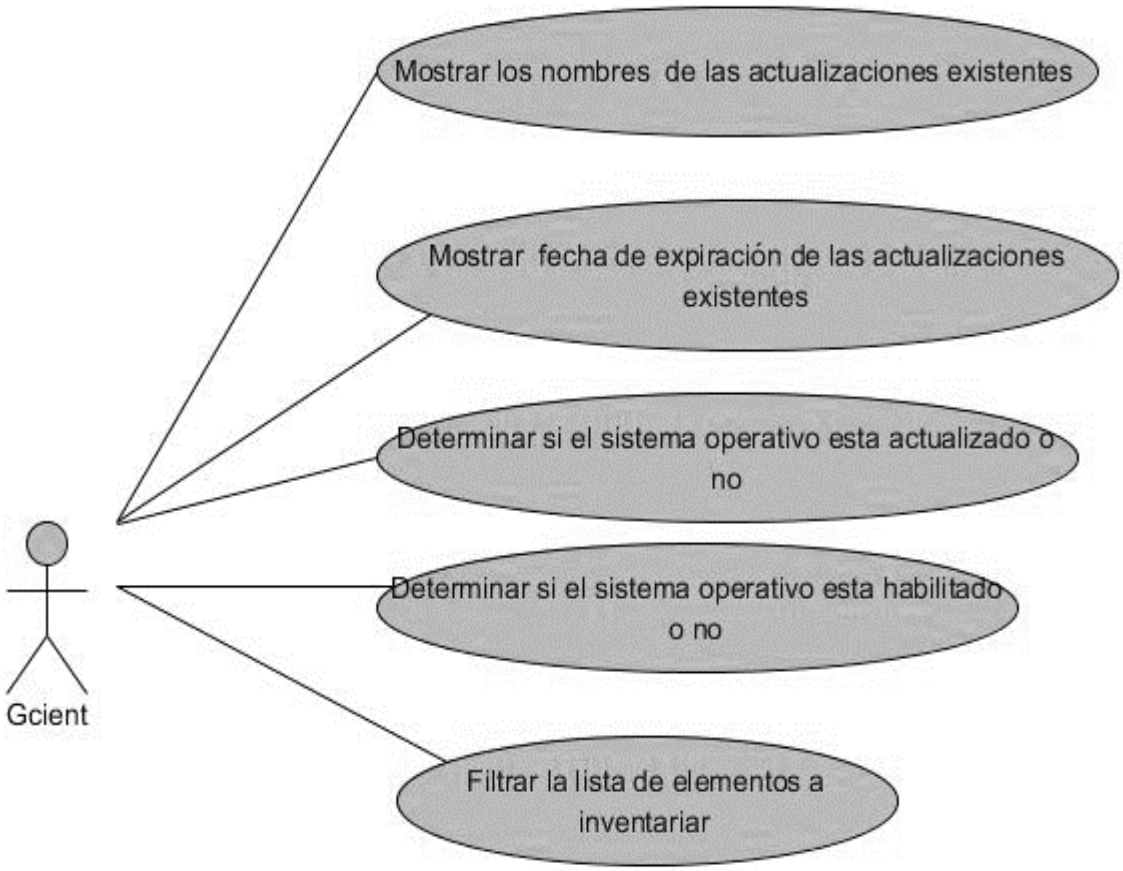


Figura 2 Caso de Uso del Sistema

2.4.2.1. Descripción de casos de uso del sistema

- **Mostrar el nombre de las actualizaciones existentes.**

Tabla 15 CU-Mostrar el nombre de las actualizaciones existentes.

Objetivo	Mostrar el nombre de las actualizaciones existentes en el SO Windows para XILEMA GRHS
Actores	gclient
Resumen	El caso de uso inicia cuando el gclient necesita mostrar el nombre de la actualización efectuada en el SO.
Complejidad	Alta
Prioridad	Crítico
Precondiciones	La PC debe estar encendida.
Postcondiciones	No procede.
Flujo de eventos	

Flujo básico: Mostrar nombre de actualizaciones		
	Actor	**Sistema**
1.	Solicita los nombres de las actualizaciones.	2. Recopila los nombres de las actualizaciones.
		3. Permite realizar una lista con los nombres de las actualizaciones.
		4. Termina el caso de uso.

- **Mostrar fecha de expiración de las actualizaciones existentes.**

Tabla 16 CU-Mostrar fecha de expiración de las actualizaciones existentes.

Objetivo	Mostrar fecha de expiración de las actualizaciones existentes en el SO Windows para XILEMA GRHS
Actores	gclient
Resumen	El caso de uso inicia cuando el gclient necesita mostrar la fecha de expiración de la actualización efectuada en el SO.
Complejidad	Alta
Prioridad	Crítico
Precondiciones	La PC debe estar encendida.
Postcondiciones	No procede.
Flujo de eventos	
Flujo básico: Mostrar fecha de expiración de actualizaciones	
Actor	**Sistema**
1. Solicita la fecha de expiración de las actualizaciones.	2. Guarda la fecha de expiración de las actualizaciones.
	3. Permite realizar una lista con las fechas de expiración de las actualizaciones.
	4. Termina el caso de uso.

- **Determinar si el sistema operativo está actualizado o no**

Tabla 17 CU-Determinar si el sistema operativo está actualizado o no

Objetivo	Determinar si el sistema operativo esta actualizado o no
Actores	gclient
Resumen	El caso de uso inicia cuando el gclient necesita notificar si el sistema está actualizado o no. Para esto se debe realizar un inventario en el sistema XILEMA GRHS donde se muestre la fecha de expiración de la actualización existente, para una posterior comparación con la fecha

	actual y así poder verificar el estado de actualización del sistema.
Complejidad	Alta
Prioridad	Crítico
Precondiciones	La PC debe estar encendida.
Postcondiciones	No procede.
Flujo de eventos	
Flujo básico: Determinar si el sistema operativo esta actualizado	

	Actor	**Sistema**
1.	Solicita comparar las fechas de expiración de las actualizaciones con la fecha actual.	1. Recopila los datos de las comparaciones realizadas.
		2. Permite realizar una lista señalando si el sistema está actualizado o no según el resultado de la comparación.
		3. Termina el caso de uso.

- **Determinar si el sistema operativo está Habilitado o no**

Tabla 18 CU-Determinar si el sistema operativo está Habilitado o no

Objetivo	Determinar si el sistema operativo está Habilitado o no
Actores	gclient
Resumen	El caso de uso inicia cuando el gclient necesita notificar si el sistema está habilitado o no. Para esto se debe realizar un inventario en el sistema XILEMA GRHS donde se muestre (nombre, estado en que se encuentra la maquina (actualizado-desactualizado), fecha de la actualización y fecha en q se actualizo, fecha de expiración), para una posterior comparación con la última actualización realizada y así encontrar cambios realizados en el ordenador.
Complejidad	Alta
Prioridad	Crítico
Precondiciones	La PC debe estar encendida.
Postcondiciones	No procede.
Flujo de eventos	
Flujo básico: Determinar si el sistema operativo está Habilitado	

	Actor	**Sistema**
1.	Solicita los datos de las actualizaciones.	1. Recopila los datos de las actualizaciones.

2.	Solicita comparar las fechas de expiración de las actualizaciones con la fecha actual.	3. Recopila los datos de las comparaciones realizadas.
		4. Permite realizar una lista señalando si el sistema está habilitado o no según el resultado de la comparación.
		5. Termina el caso de uso.

- **Filtrar la lista de elementos a inventariar**

Tabla 19 CU-Filtrar la lista de elementos a inventariar

Objetivo	Filtrar la lista de elementos a inventariar	
Actores	gclient	
Resumen	El caso de uso inicia cuando el gclient necesita Filtrar la lista de elementos a inventariar.Se realiza un filtrado del inventario realizado a los datos de las actualizaciones y así se informa acerca del estado actual del software del ordenador e incidencias ocurridas en dicha PC.	
Complejidad	Media	
Prioridad	-	
Precondiciones	Es necesario que exista una lista de elementos a inventariar.	
Postcondiciones	Lista de elementos a inventariar filtrada	
Flujo de eventos		
Flujo básico: Filtrar la lista de elementos a inventariar		
	Actor	**Sistema**
1.		1. Permite filtrar resultados obtenidos del inventario
		2. Termina el caso de uso.

2.4.3. Diagrama de Secuencia

Un diagrama de secuencia es un tipo de diagrama de interacción porque describe cómo, y en qué orden, un grupo de objetos funcionan en conjunto. Tanto los desarrolladores de

software como los profesionales de negocios usan estos diagramas para comprender los requisitos de un sistema nuevo o documentar un proceso existente. A los diagramas de secuencia en ocasiones se los conoce como diagramas de eventos o escenarios de eventos(Sánchez Guerra, 2008).

A continuación, se presenta el diagrama de secuencia correspondiente al caso de uso *Mostrar los nombres de las actualizaciones existentes.*

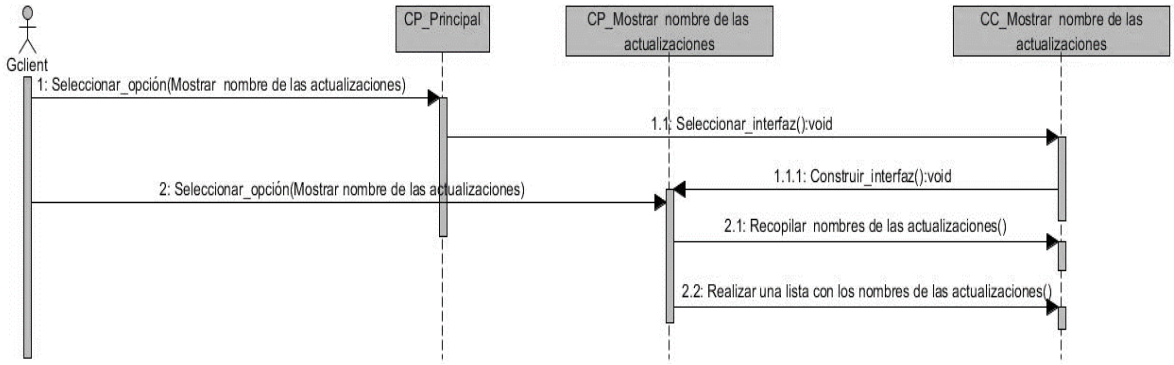


Figura 3 Diagrama de secuencia-CU(Mostrar los nombres de las actualizaciones existentes)

Para ver los restantes diagramas de secuencia consultar Anexo 3.

2.5. Descripción de la arquitectura

El diseño arquitectónico representa la estructura de datos y los componentes de un programa necesarios para construir un sistema computacional. Asume el estilo arquitectónico que adoptará el sistema, la estructura y las propiedades de los componentes que constituyen el sistema y las relaciones entre todos los componentes arquitectónicos de un sistema(Pressman, 2010).

La arquitectura de software es la organización fundamental de un sistema encarnada en sus componentes, las relaciones entre ellos y el ambiente y los principios que orientan su diseño y evolución(REYNOSO, 2004). Una arquitectura de software de un programa o un sistema computacional es la estructura del sistema, la cual comprende elementos de software, las propiedades externamente visibles de esos elementos, y las relaciones entre ellos(BASS, CLEMENTS, & KAZMAN, 2003).

2.5.1. Arquitectura Cliente Servidor

La modalidad Cliente-Servidor es un sistema distribuido entre múltiples procesadores donde hay clientes que solicitan servicios y servidores que los proporcionan. Es un modelo que implica productos y servicios enmarcados en el uso de la tecnología de punta y permite la distribución de la información en forma ágil y eficaz a las diversas áreas de una organización(Gómez Pérez, 2014).

En este tipo de arquitectura el servidor se caracteriza por estar operativo todo el tiempo, a la escucha de las peticiones que realizan los diferentes clientes y procesando dichas peticiones para enviar la respuesta correspondiente.

Es el cliente, por lo general, el que inicia el proceso de comunicación entre los dos sistemas, esto en dependencia de sus necesidades de actualización, información u otras. La Figura 4 muestra un esquema que simula este tipo de arquitectura.

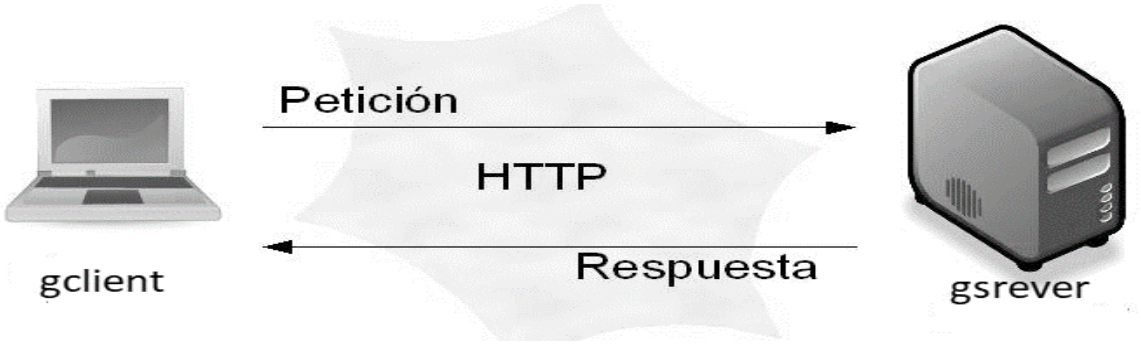


Figura 4 Arquitectura Cliente-Servidor(tomada de: https://edgarbc.wordpress.com/dos-capas/)

Se decide utilizar la modalidad o arquitectura Cliente-Servidor porque:
El sistema XILEMA GRHS cuenta con esta arquitectura para su correcto funcionamiento, la cual permite a los usuarios finales obtener acceso a la información en forma transparente aún en entornos multiplataforma. En la arquitectura cliente servidor, el cliente (gclient) envía un mensaje solicitando un determinado servicio al servidor (gserver), o lo que es lo mismo le hace una petición, y este envía uno o varios mensajes con la respuesta (provee el servicio).
En esta categoría se realizan funciones de software basándose en el hardware, pero en caso de no tener la capacidad de procesar los datos necesarios recurre al servidor y espera a que éste le brinde los servicios solicitados. El cliente es una estación de trabajo

o computadora que está conectada a una red a través de la cual puede acceder al servidor(CCM, 2017).

Entre las características fundamentales de esta arquitectura se encuentran que tanto el cliente como el servidor pueden realizar tareas en forma conjunta como separada ya que el cliente tiene sus propias aplicaciones, archivos y bases de datos y que, además, pueden estar en la misma plataforma o en plataformas diferentes. Por otra parte, el servidor puede brindar varios servicios a la vez, tanto al mismo cliente como a clientes múltiples.

Gserver es el encargado de manejar toda la comunicación entre los demás sub-sistemas que integran XILEMA GRHS.Las notificaciones realizadas van dirigidas a gserver para posteriormente ser mostradas en el gadmin q es un componente propio de gserver y se puede considerar la interfaz gráfica del módulo a desarrollar.

Permite mantener centralizada la información de todos los recursos de hardware de las computadoras donde se encuentren los clientes, para su posterior consulta por parte del administrador a través de una aplicación web. El módulo que se propone se encuentra en el cliente, brindando la información necesaria al servidor.

2.5.2. Patrón de arquitectónico

Los patrones arquitectónicos se utilizan para expresar una estructura de organización base o esquema para un software. Proporcionando un conjunto de subsistemas predefinidos, especificando sus responsabilidades, reglas, directrices que determinan la organización, comunicación, interacción y relaciones entre ellos(DS, 2015).

Modelo vista plantilla

Django se basa en el Modelo Vista Plantilla (MPV), que es una modificación del Modelo Vista Controlador (MVC). El controlador pasa a ser la vista y la vista pasa a denominarse plantilla. En Django, una vista describe los datos que se ofrecen al usuario, pero no necesariamente su aspecto. Una vista habitualmente delega los datos a una plantilla que describe la forma de presentarlos. El modelo es el encargado de las consultas a la base de datos(«El patrón de diseño MTV (El libro de Django 1.0)», 2015).

En el patrón MTV:

- M significa modelo (en inglés model), la capa de acceso a la base de datos. Esta capa contiene toda la información sobre los datos: cómo acceder a estos, cómo validarlos, cuál es el comportamiento que tiene, y las relaciones entre los datos.
- T significa plantilla (en inglés template), la capa de presentación. Esta capa contiene las decisiones relacionadas a la presentación: cómo algunas cosas son mostradas sobre una página web u otro tipo de documento.
- V significa vista (en inglés view), la capa de la lógica de negocios. Esta capa contiene la lógica que accede al modelo y la delega a la plantilla apropiada

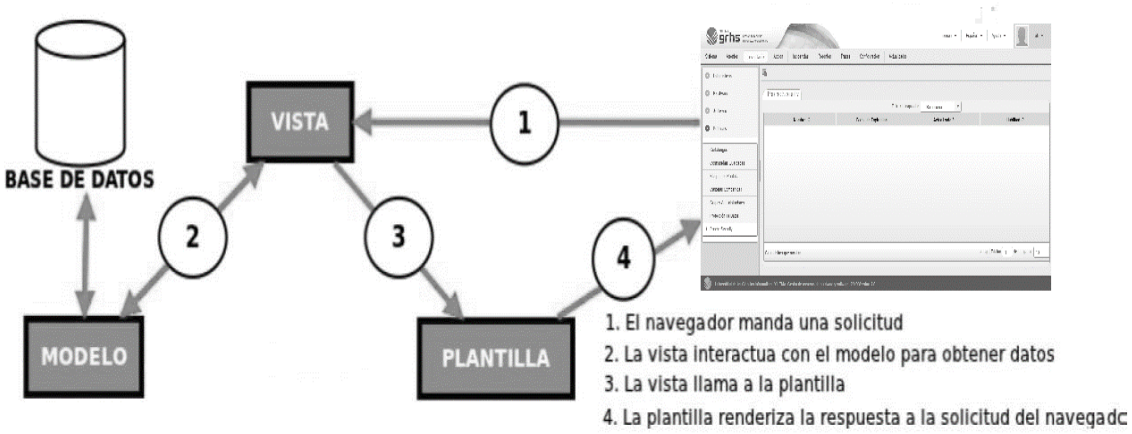


Figura 5 Patrón Model Template View (MTV) (Montero, 2015)

2.5.3. Patrones de Diseño

Un patrón es una unidad de información nombrada, instructiva e intuitiva que captura la esencia de una familia exitosa de soluciones probadas, a un problema recurrente dentro de un cierto contexto. El objetivo de los patrones es crear un lenguaje común para los desarrolladores con el objetivo de comunicar experiencia sobre los problemas y soluciones. Los patrones de diseño expresan esquemas para definir estructuras de diseño con las que se construye un sistema informático.

Los patrones de diseño son un conjunto de prácticas de óptimo diseño que se utilizan para abordar problemas recurrentes en la programación orientada a objetos. Los patrones de diseño que se especifican a continuación son los aplicados en la solución propuesta. Aunque no son de utilización obligatoria, seguir estos patrones garantiza mayor robustez en una aplicación informática y que el mantenimiento o ampliación de la misma pueda realizarse más fácilmente(Universidad Nacional de Colombia, 2015).

- **Patrones GRASP**

Los patrones GRASP describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en formas de patrones. GRASP es un acrónimo que significa General Responsibility Assignment Software Patterns. El nombre se eligió para indicar la importancia de captar estos principios, si se quiere diseñar eficazmente el software orientado a objetos(Visconti & Astudillo, 2012).

Experto

El patrón experto define que la responsabilidad de crear un objeto debe asignarse a la clase que mayor información tenga para la creación del mismo. Esto permite mayor cohesión y el encapsulamiento de la información.

- **Aplicación en el módulo**

Este patrón se observa en el módulo a la hora de distribuir las responsabilidades, cada uno de los plugin de acceso a dato hacen su trabajo en específico, pues son ellos los que tienen la información necesaria para realizar el inventario específico. Se encuentra presente en las clases parche_security.

Creador

Define quién debe crear o instanciar nuevos objetos, teniendo en cuenta que se cuente con toda la información necesaria para hacerlo.

- **Aplicación en el módulo**

En el módulo el patrón se pone de manifiesto porque es necesario crear un objeto de que mediante él se detecte si existen incidencias. Se evidencia en la clase ParcheSecuritySerializer.

Alta cohesión

Plantea que la información que almacena una clase debe estar lo más relacionada posible con la clase, evitando la implementación de funcionalidades que pueden ser heredadas de otras clases especializadas.

- **Aplicación en el módulo**

Este patrón es aplicable en el módulo puesto que los plugins de acceso a datos tienen pocos métodos y estos se encuentran relacionados, en estos solo se obtiene el nuevo objeto del inventario hecho y se compara con el anterior que se encuentra guardado en la

cache, son utilizados con un propósito específico, también colaborando con el bajo acoplamiento. Dentro del sistema, este patrón fue utilizado en las acciones que contienen varias funcionalidades que están relacionadas. Clases inventory_parche_security, parche_Windows.

Bajo acoplamiento

Consiste en reducir las dependencias entre clases, o reducir la mezcla de las mismas con el propósito de evitar que al modificar una clase sea necesaria la modificación del resto de las clases.

- **Aplicación en el módulo**

Este patrón se pone de manifiesto dentro de inventory ya que si se quiere agregar algún tipo nuevo de inventario no se verían afectados los demás, solo se agregaría a la interfaz y se implementaría en un plugins específico. Se evidencia en las clases utilizadas, de manera que al realizar un cambio en una de las clases no se afecten las demás, por esto la estructura de agrupar las clases según la parte del inventario en específico sobre la cual interactúa. Clase information_parche_security.

2.6. Diagrama de clases

Un Diagrama de Clases de Diseño muestra la especificación para las clases de software de una aplicación. A diferencia del Modelo Conceptual, un Diagrama de Clases de Diseño (DCD) muestra definiciones de entidades de software más que conceptos del mundo real. Se considera que constituye una aproximación a un CU. En él, la estructura de clases del sistema se especifica, con relaciones entre clases y estructuras de herencia.

A continuación, se muestran los diagramas de clases del diseño de los casos de uso *Mostrar el nombre de las actualizaciones existentes y Mostrar la fecha de expiración de las actualizaciones existentes.*

Para ver los restantes diagramas de clases del diseño, consultar Anexo 4

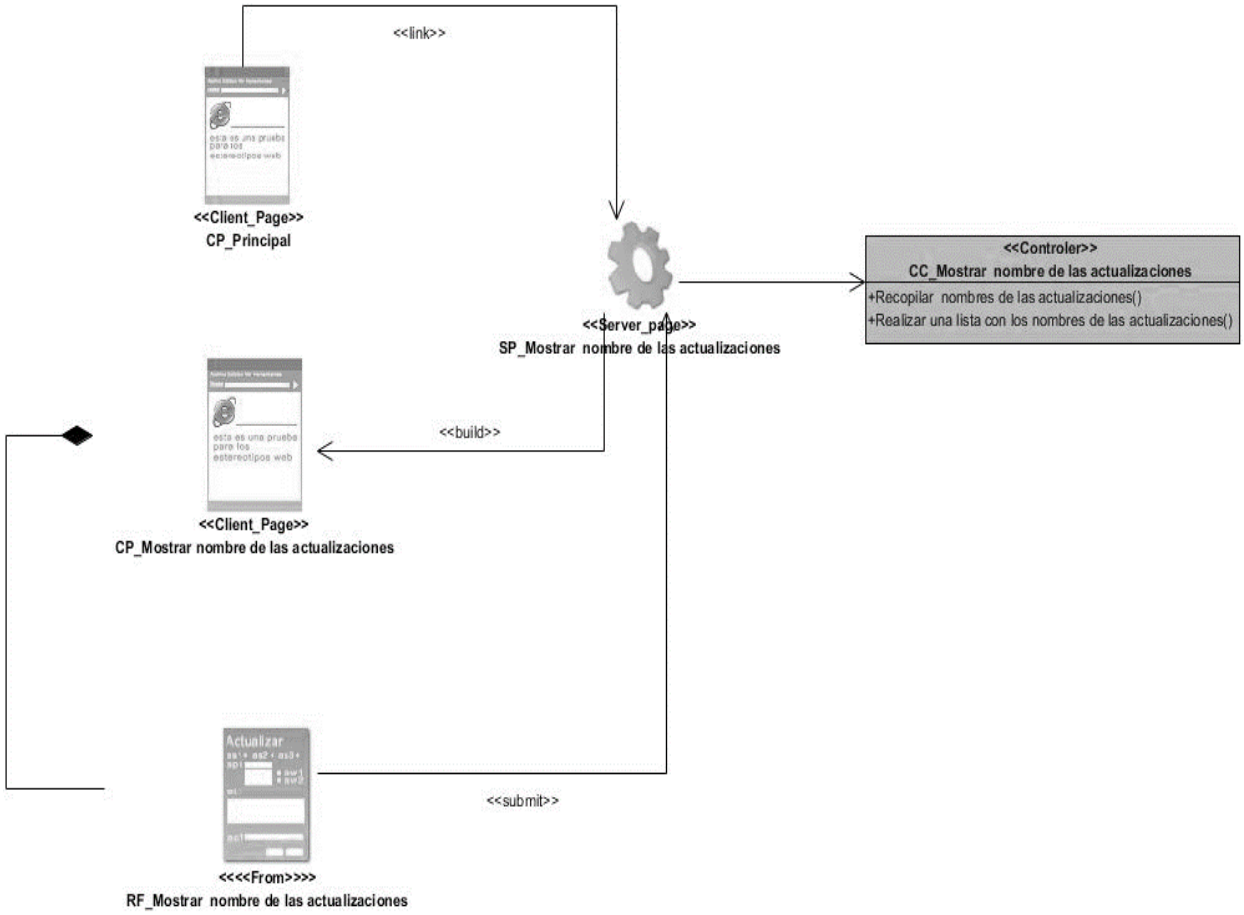


Figura 6 Diagrama de clases-Mostrar nombre de las actualizaciones .

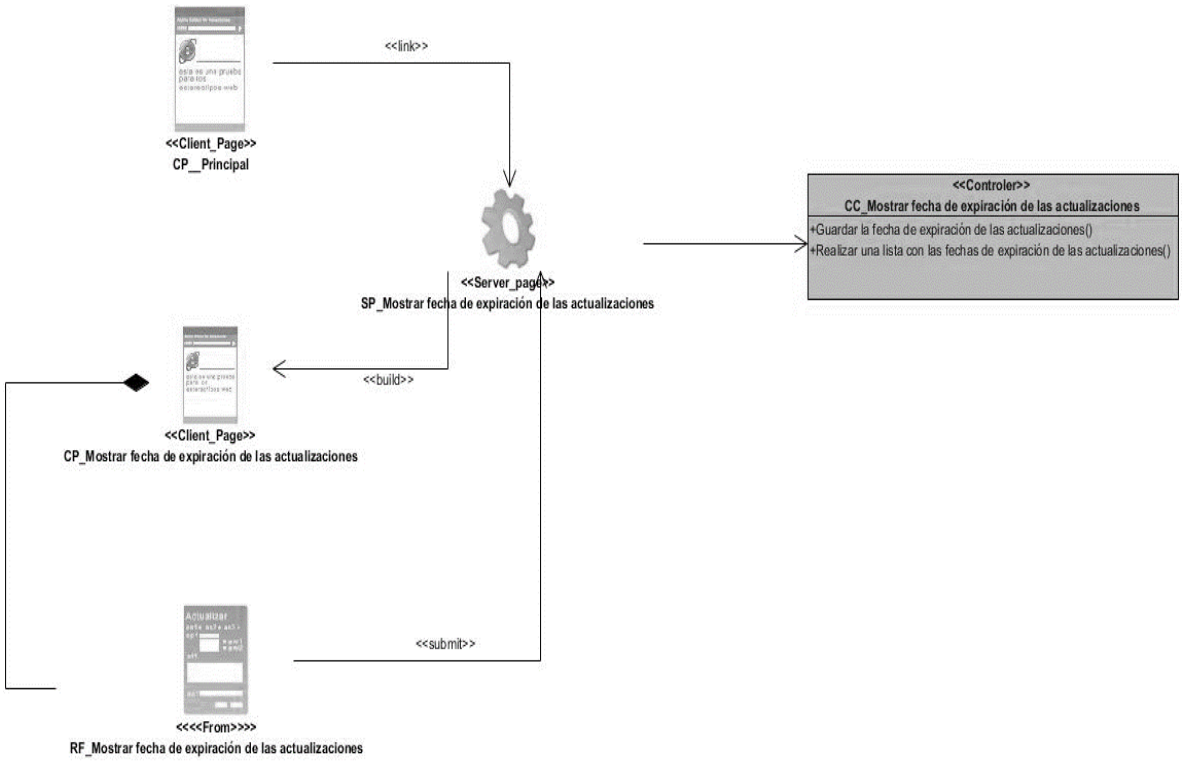


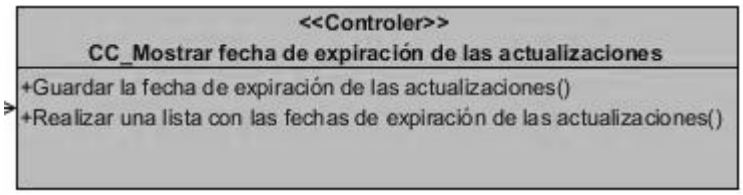
Figura 7 Diagrama de clases-Mostrar la fecha de expiración de las actualizaciones existentes.

2.6.1. Descripción de clases

Tabla 20 Descripción de clase-1

Número del módulo	1
Número de la clase	1
Clase	Mostrar nombre de las actualizaciones.
Propósito	Recopilar los nombres de las actualizaciones y mostrarlos mediante una lista.
Descripción	<<Controler>> CC_Mostrar nombre de las actualizaciones +Recopilar nombres de las actualizaciones() +Realizar una lista con los nombres de las actualizaciones()
Observaciones	• [Observación1.]

Tabla 21 Descripción de clase-2

Número del módulo	1
Número de la clase	2
Clase	Mostrar la fecha de expiración de las actualizaciones
Propósito	Guardar la fecha de expiración de las actualizaciones y realizar una listan donde se muestren.
Descripción	
Observaciones	• [Observación 2.]

2.7. Modelo de datos

El modelo de datos tiene gran importancia en el ciclo de desarrollo de software, y de manera particular para la fase de implementación, pues define formalmente las estructuras permitidas y las restricciones que se aplican con el fin de representar los datos del dominio de la aplicación. Está compuesto por objetos: entidades que existen y se manipulan; y atributos: características básicas de dichos objetos y relaciones: forma en que se enlazan los objetos entre sí(«Modelado de Datos», 2018; PIÑEIRO GOMEZ, 2013).

A continuación, se muestra el modelo de datos.

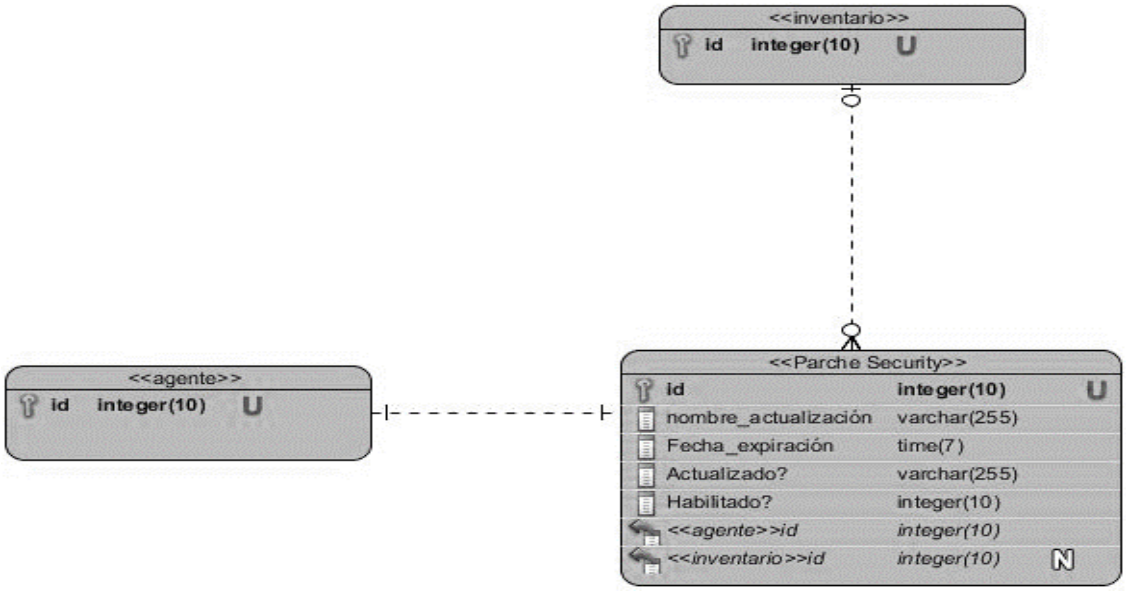


Figura 8 Modelo de datos .

La descripción de los componentes más relevantes de este modelo se muestra en la siguiente tabla, donde se especifica la función principal de cada modelo.

Tabla 22 Descripción de los componentes del modelo de datos

Modelo	Descripción
<<inventario>>	Almacena la información correspondiente a al módulo inventario. Establece una relación con la entidad Parche Security.
<<agente>>	Almacena la información correspondiente a un agente determinado.
<<Parche Security>>	Almacena la información del módulo correspondiente. Establece una relación de herencia con la entidad inventario, y también se relaciona con la entidad agente.

2.8. Conclusiones parciales

Los artefactos alcanzados en esta etapa dan cumplimiento a los requisitos funcionales y no funcionales definidos durante progreso de la solución propuesta. Mediante la representación del modelo conceptual fueron reconocidas las entidades relacionadas con el sistema y las relaciones entre ellas. Los diagramas de clases y de secuencia lograron que se obtuvieran de forma específica y cuidadosa las relaciones existentes entre las clases. El modelo de bases de datos ayudo a entender la relación entre las entidades en base de datos. Todos estos artefactos son considerados la entrada principal para las siguientes actividades de implementación y prueba.

Lo anteriormente planteado permitió un mejor entendimiento del funcionamiento del sistema, posibilitando la implementación del módulo de inventario de actualizaciones y parches de seguridad del SO Windows instalados y pendientes en el sistema XILEMA GRHS.

Capitulo No. 3: Implementación y Pruebas para el módulo.

Introducción

Con anterioridad se expusieron aspectos relacionados con las características que debe poseer la solución propuesta, de vital importancia para dar continuidad a la investigación. A continuación, se aborda la implementación del sistema.

Para lograr un producto con calidad es necesario trazarse un plan de pruebas desde el principio. Darles seguimiento a los cambios y desarrollarlos iterativamente. También controlar el funcionamiento de la aplicación y su despliegue. En este capítulo se describen los casos de pruebas a los que fue sometido el módulo para validar su correcto funcionamiento, plasmando los resultados.

3.1. Diagrama de despliegue

Un diagrama de despliegue puede ser usado para modelar la arquitectura física y (en el caso de las redes o sistemas distribuidos) la topología del sistema de software desarrollando. Se describen los dispositivos de hardware (conocidos como nodos), los componentes de software que se ejecutan en ellos (conocidos como artefactos), y los enlaces de comunicación entre los distintos nodos y artefactos(PRESSMAN, 2014).El mismo fue generado para facilitar la comprensión de la distribución de los componentes que integran la solución propuesta. A continuación, se muestra el diagrama de despliegue realizado.

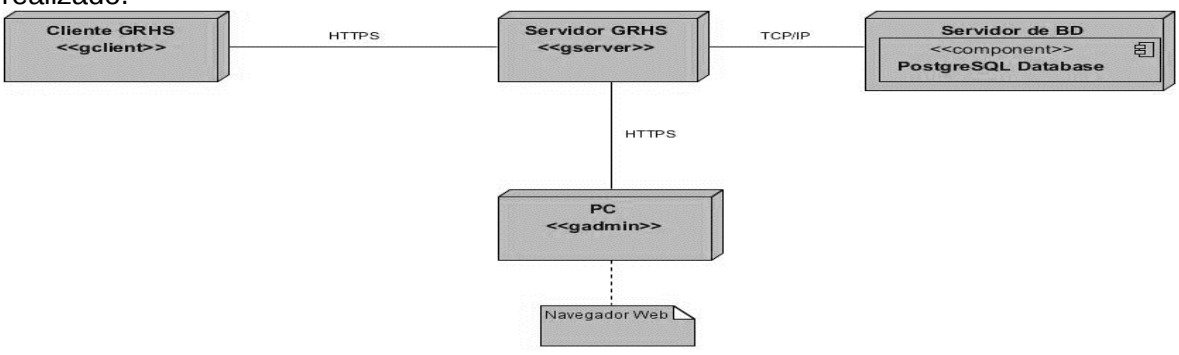


Figura 9 Diagrama de Despliegue

El diagrama de despliegue modela la configuración del tiempo de ejecución en una vista estática y visualiza la distribución de componentes en una aplicación. Como se muestra en la figura 9, la distribución física del sistema en tiempo de ejecución consta de 4 nodos. En uno se encuentra el gclient, el cual hace peticiones al gserver usando el protocolo de comunicación HTTPS. El gserver establece comunicación con el servidor de base dato PostrgresSQL utilizando el protocolo TCP/IP. Se comunica también con el gadmin usando el protocolo de comunicación HTTPS, y este se enlaza a un navegador web.

3.2. Diagrama de componentes

A partir de la búsqueda realizada sobre los diagramas de componentes los autores asumen los criterios de los Ing. Yaniel Alfredo Velázquez Bruceta y Sandy Fernando Pérez Matamoros cuando en su trabajo de diploma expresan que:

“El diagrama de componentes es utilizado para modelar los componentes de un sistema, incluyendo los artefactos que implementan dichos componentes. Un componente es la implementación física de un conjunto de elementos lógicos, como por ejemplo las clases. Un diagrama de componentes muestra la estructura de alto nivel del modelo de implementación, especificando los subsistemas de implementación y sus dependencias a la hora de importar código(Manager, 2014).

Sobre esta base se realizó el diseño del diagrama de componentes correspondiente al módulo de inventario de actualizaciones y parches de seguridad de sistemas operativos, el que se expresa en el esquema siguiente:

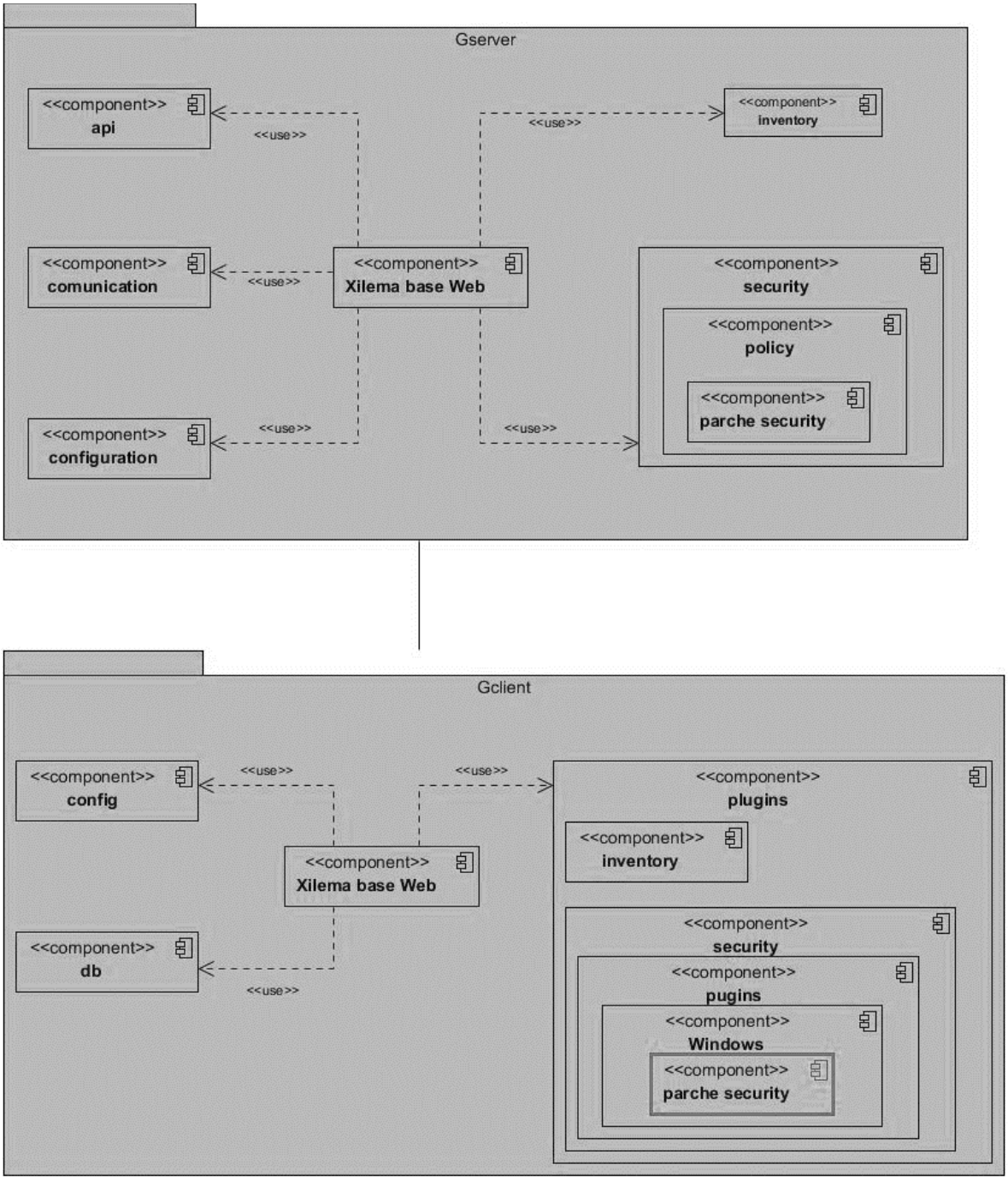


Figura 10 Diagrama de componentes

3.3. Implementación

La implementación que propone la metodología AUP-UCI, muestra cómo, a partir de los resultados del análisis y diseño se construye el sistema.

A partir de los resultados de diseño obtenidos fue realizada la implementación del módulo a través de scripts, ejecutables, etc. Esta disciplina explica cómo desarrollar, organizar e integrar el módulo implementado en el módulo Inventario del XILEMA GRHA, basándose en las especificaciones del diseño.

La fase de implementación de un sistema es de las más importantes para el desarrollo de un software. Esta fase materializa en forma de código la arquitectura, todos los artefactos y descripciones definidos en la anterior etapa de análisis y diseño con el objetivo de realizar el producto final que desea obtener el cliente. Al realizar pruebas a un software se garantiza de que el mismo cumpla con los requerimientos y funcionalidades que pide que el cliente. Estas pruebas se llevan a cabo en la etapa de validación donde se realizan un conjunto de pruebas, cada una con un objetivo específico.

3.3.1. Estándares de codificación

Se definen estándares de codificación ya que un estilo de programación uniforme en un proyecto permite que todos los participantes logren entenderlo en menos tiempo y que el código en resultado sea mantenible. A continuación, se representan los estándares de codificación para los archivos, selectores, clases, atributos, métodos y variables, definidos por python para el desarrollo de sus aplicaciones.

3.3.2. Nomenclatura de clases

Los nombres de las clases se escribirán con letra inicial mayúscula y si es combinado en la parte en el nombre iniciará con mayúscula cada una de las palabras, por ejemplo:
class Configuration class ServerConfig class Inventory

3.3.3. Nomenclatura de variables

Las variables se escribirán completamente en minúscula o mayúscula y si el nombre es compuesto se separarán las distintas palabras con guiones bajos **"_"**, por ejemplo:

Data gserver_address gclient_address

3.3.4. Imports

Los imports se colocan en distintas líneas, por ejemplo:

Sí:

```
import os
```

import sys

No:

import sys, os

Los imports se colocan siempre en la parte superior del archivo, justo después de cualquier comentario o cadena de documentación del módulo, y antes de las variables globales y las constantes del módulo.

Los imports se agrupan siguiendo el siguiente orden:

- ✓ imports de la librería estándar
- ✓ imports de proyectos de terceras partes relacionados
- ✓ imports de aplicaciones locales/imports específicos de la librería
- ✓ Se añade una línea en blanco después de cada grupo de imports.

Los nombres públicos se definen por el módulo con __all__ esto debería hacerse después de los imports.

3.3.5. Comentarios

Los comentarios son frases completas. La primera palabra se encuentra en mayúsculas, a menos que sea un identificador que comience con una letra en minúsculas

Si un comentario es corto, se omite el punto al final.

3.4. Interfaz

Para la interfaz gráfica se utilizará el estándar de diseño gráfico establecido por la UCI para la marca Xilema.

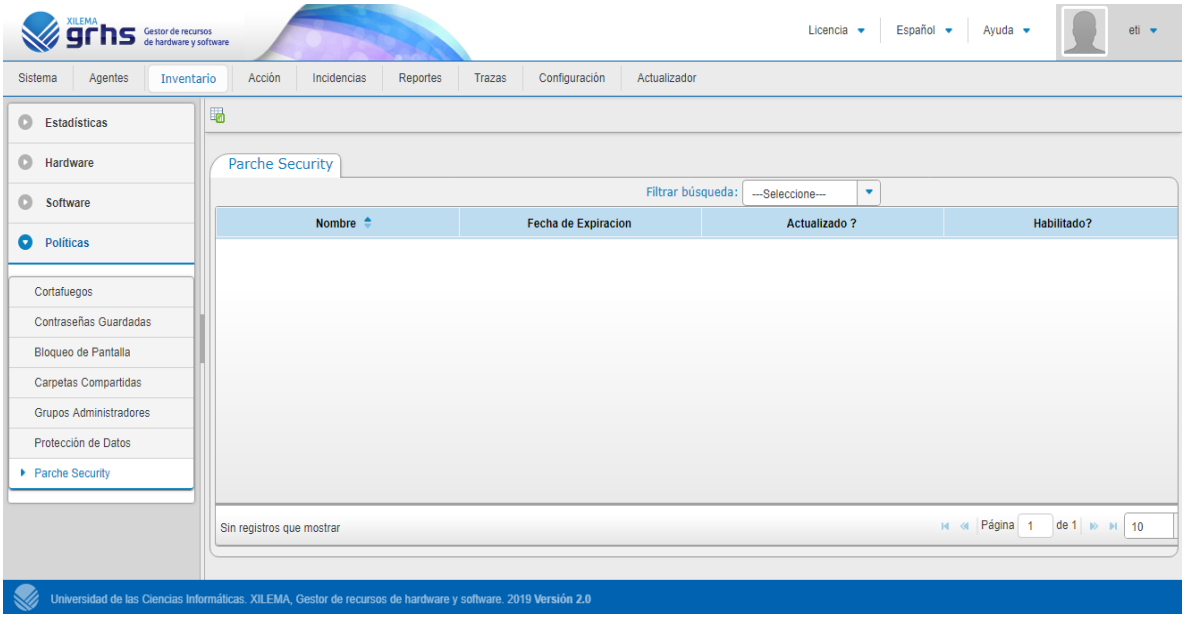


Figura 11 Interfaz

3.5. Pruebas

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

Se le denomina prueba aquellas acciones que se llevan a cabo sobre el software para verificar o revelar la calidad del producto, dando la posibilidad de identificar posibles fallos tanto de implementación como de calidad o usabilidad (Sommerville, 2005a).

3.5.1. Estrategias de pruebas

Las estrategias de pruebas de software proporcionan una guía que describe los pasos que deben realizarse como parte de la prueba, cuándo se planean y se llevan a cabo dichos pasos, y cuánto esfuerzo, tiempo y recursos se requerirán. Por tanto, cualquier estrategia de prueba debe incorporar la planificación, diseño, ejecución, recolección y evaluación de los casos de prueba (Pressman, 2010).

Elementos a tener en cuenta para que una prueba tenga éxito:

- Niveles de Prueba.
- Método de prueba.

- Tipo de prueba.
- Caso de prueba.

Para realizar una correcta estrategia de pruebas se deben tener en cuenta algunos criterios de importancia:

- Describe el enfoque y los objetivos generales de las actividades de prueba.
- Incluye los niveles de prueba a ser diseccionados, el tipo de prueba a ser ejecutada y los casos de prueba diseñados para lograr los objetivos.
- Define:

✓ Técnicas de pruebas (manual o automática) y herramientas a ser usadas.

✓ Criterios de éxitos y culminación de las pruebas.

✓ Consideraciones especiales relacionadas con los recursos necesarios para realizar las pruebas.

3.5.1.2. Niveles de Prueba

Las pruebas se aplican en diferentes tipos de objetivos, en diferentes escenarios o niveles de trabajo, para realizar las pruebas al sistema se escogieron los siguientes niveles(Ruiz & Vaillant, 2015):

Pruebas Unitarias: Se realizan para comprobar la correcta implementación de las funcionalidades. Son el proceso de probar los componentes individuales en el sistema. Este es un proceso de prueba de defecto, por lo que sus objetivos es encontrar defectos en estos componentes. Los desarrolladores de los componentes son los responsables de probarlos.

- Existen diferentes objetos que pueden probarse en esta etapa:
- Funciones individuales o métodos dentro de un objeto.
- Clases de objetos que tiene varios atributos y métodos.

Pruebas de Integración: El proceso de integración del sistema implica construir este a partir de sus componentes y probar el sistema resultante para encontrar problemas que puedan surgir debido a la integración de los componentes. Los componentes que se

integran pueden ser componentes reutilizables que han sido adaptados a un sistema en particular o componentes nuevos desarrollados.

Algunas veces, primero se desarrolla el esqueleto del sistema en su totalidad y se le añaden los componentes, esto de denomina integración descendente (Sommerville, 2005a).

Pruebas de aceptación: Las pruebas de aceptación representan algún tipo de resultado por parte del sistema. Los clientes son los responsables de verificar la exactitud de estas pruebas y de revisar los resultados para poder así priorizar las que fracasaron. Se comprueba que las funcionalidades solicitadas se han realizado satisfactoriamente. Este tipo de pruebas son comúnmente realizadas por usuarios, quienes deben informar todas las deficiencias o errores que encuentre antes de dar por aprobado el sistema definitivamente.

Para asegurarse que la aplicación desarrollada cumple sus requisitos, se definió realizar pruebas de aceptación, puesto que representan la satisfacción del cliente con el producto desarrollado. Estas pruebas conllevan a precisar lo que la aplicación debe hacer en determinadas circunstancias, por esto el cliente es la persona adecuada para diseñar las pruebas.

3.5.1.3. Método de prueba

Se explica a continuación los métodos de prueba a emplear en la solución del módulo:

Prueba de caja blanca

En ella se comprueban los caminos lógicos del módulo proponiendo casos de prueba donde se ejerciten conjuntos específicos de condiciones y/o bucles. Se puede examinar el estado del programa en varios puntos para determinar si el estado real coincide con el esperado.

Para desarrollar estas pruebas existen varias técnicas:

- Pruebas de flujo de control
- Pruebas de flujo de datos
- Pruebas de bifurcación
- Pruebas de caminos básicos

Prueba de caja negra

Pruebas que se llevan a cabo sobre la interfaz del software. El objetivo es demostrar que las funciones del software son operativas, que las entradas se aceptan de forma adecuada y se produce un resultado correcto, y que la integridad de la información externa se mantiene (no se ve el código).

Para desarrollar estas pruebas existen varias técnicas:

- Técnica de partición de equivalencia: esta técnica divide el campo de entrada en clases de datos que tienden a ejercitar determinadas funciones del software.
- Técnica de análisis de valores límites: esta técnica prueba la habilidad del programa para manejar datos que se encuentran en los límites aceptables.
- Técnica de Grafos de causa-efecto: es una técnica que permite al encargado de la prueba validar complejos conjuntos de acciones y condiciones.

La técnica a emplear para desarrollar las pruebas es la de partición de equivalencia. Esta técnica permite examinar los valores válidos e inválidos de las entradas existentes en el software, descubre de forma inmediata una clase de errores que, de otro modo, requerirían la ejecución de muchos casos antes de detectar el error genérico.

3.5.1.4. Tipos de pruebas

Existen varios tipos de pruebas que se deben tener en cuenta para un correcto de la misma:

- Funcionalidad (función, seguridad, volumen).
- Usabilidad.
- Fiabilidad (integridad, estructura, stress).
- Rendimiento (benchmark, contención, carga, performance profile).
- Soportabilidad (configuración, instalación).

El tipo de prueba a aplicar es Funcionalidad. Dentro de las pruebas de Funcionalidad se aplicarán pruebas de función. Estas pruebas fijan su atención en la validación de las funciones, métodos, servicios y casos de uso. Permiten comprobar el correcto funcionamiento de los requisitos funcionales del componente. Son descritas en artefactos

de la Ingeniería de Software conocidos como casos de prueba, los cuales son especificaciones de las entradas y la salida esperada por el sistema(Sommerville, 2005b).

Niveles de Pruebas	Tipos de Pruebas	Métodos
Integración	Funcionalidad Función	Caja Negra
Unitarias	Funcionalidad Función	Caja Blanca
Aceptación	Funcionalidad Función	Caja Negra

3.5.1.5. Descripción de casos de prueba

Un caso de prueba es un conjunto de entradas de pruebas, condiciones de ejecución y resultados esperados desarrollados para cumplir un objetivo en particular o una función esperada. Siempre es ejecutada como una unidad, desde el comienzo hasta el final [4]. Es una especificación de un caso para probar el sistema, incluyendo qué probar, con qué entradas y resultados y bajo qué condiciones. Su principal objetivo es obtener un conjunto de pruebas que tengan una mayor probabilidad de descubrir los defectos del software(JACOBSON, BOOCH, & RUMBAUGH, 2004).

Debe verificar:

- Si el producto satisface los requerimientos del usuario, tal y como se describe en las especificaciones de los requerimientos.
- Si el producto se comporta como se desea, tal y como se describe en las especificaciones funcionales del diseño.

Pruebas Unitarias

Prueba de: Unidad	
Nombre de la prueba: Mostrar los nombres de las actualizaciones existentes.	
Estado: Satisfactoria	**Ultima ejecución:**
Ejecutado por:	**Verificado por:**
Descripción: Se realiza una consulta HTTP al Gserver informando el nombre de la actualización.	
Entrada: Nombre de la actualización; versión que se ejecuta en el Gclient; descripción del error de haberse producido alguno; dirección IP del cliente.	

Criterio de aceptación: La funcionalidad se ejecuta satisfactoriamente. Grado de aceptación 99%.
Resultado: La funcionalidad se encuentra lista para ser integrada al sistema.

Tabla 23 Prueba Unitaria Mostrar los nombres de las actualizaciones existentes

Las demás pruebas unitarias se encuentran en el Anexo 1 de este documento.

Pruebas de Integración

La técnica utilizada para realizar esta estrategia fue la manual, realizando una tabla de integración para el módulo

Tabla 24 Pruebas de integración

Módulo integrado	Funcionalidad	Condiciones de ejecución	Escenarios de prueba	Resultado previsto	Resultado real
Módulo de inventario de actualizaciones y parches de seguridad del SO Windows para XILEMA GRHS	Mostrar los nombres de las actualizaciones existentes.	Se obtiene los nombres de las actualizaciones, en el SO. El módulo actual ha estado ejecutándose correctamente.	Mostrar los nombres de las actualizaciones existentes correctamente.	Se espera que llegue la orden de mostrar los nombres de las actualizaciones existentes.	Llega la orden de mostrar los nombres de las actualizaciones existentes .
Módulo de inventario de actualizaciones y parches de seguridad del SO Windows para XILEMA GRHS	Mostrar la fecha de expiración de las actualizaciones existentes.	Se obtiene la fecha de expiración de las actualizaciones, en el SO. El módulo actual ha estado ejecutándose correctamente.	Mostrar la fecha de expiración de las actualizaciones existentes correctamente.	Se espera que llegue la orden de mostrar la fecha de expiración de las actualizaciones existentes.	Llega la orden de mostrar la fecha de expiración de las actualizaciones existentes .
Módulo de inventario de	Determinar si el SO está	Se obtiene la	Mostrar la información	Se espera que llegue la	Llega la orden de mostrar la

actualizaciones y parches de seguridad del SO Windows para XILEMA GRHS	actualizado o no	información del estado de actualización en que se encuentra el SO. El módulo actual ha estado ejecutándose correctamente.	del estado de actualización en que se encuentra el SO correctamente	orden de mostrar la información del estado de actualización en que se encuentra el SO	información del estado de actualización en que se encuentra el SO
Módulo de inventario de actualizaciones y parches de seguridad del SO Windows para XILEMA GRHS	Determinar si el SO está habilitado o no	Se obtiene la información del SO en cuanto a si está habilitado o no. El módulo actual ha estado ejecutándose correctamente.	Mostrar la información del SO en cuanto a si está habilitado o no.	Se espera que llegue la orden de mostrar la información del SO en cuanto a si está habilitado o no.	Llega la orden de mostrar la información del SO en cuanto a si está habilitado o no.
Módulo de inventario de actualizaciones y parches de seguridad del SO Windows para XILEMA GRHS	Filtrar la lista de elementos a inventariar	La PC debe estar encendida. El sistema debe estar instalado en la computadora. El módulo debe funcionar correctamente.	Se espera que lleguen los datos de la lista de elementos que se desean inventariar para obtener correctamente el filtrado.	Obtener correctamente el filtrado de elementos a inventariar.	Se obtuvo correctamente el filtrado de la lista de elementos a inventariar

Pruebas de aceptación

Para realizar una prueba de aceptación se confecciona una tabla con las casillas Clases válidas, Clases inválidas, Resultado esperado, Resultado de la prueba y Observaciones.

Tabla 25 Prueba de aceptación , Mostrar los nombres de las actualizaciones existentes

Clases válidas	Clases inválidas	Resultado esperado	Resultado de la prueba	Observaciones
Se ejecuta la funcionalidad mientras el Gclient se encuentra operando correctamente.		Recibe la información del nombre de la actualización. Crea una lista con los nombres y ejecuta otras funcionalidades.	Satisfactorio	
	Se ejecuta la funcionalidad mientras el Gclient se encuentra operando correctamente.	Se reintenta realizar la solicitud de información al Cliente según se especifica. Al término de todos los intentos se detiene la ejecución del proceso.	Satisfactorio	

El resto de las pruebas de aceptación se encuentran en el Anexo 2 de este documento.

3.6. Resultado de las pruebas realizadas

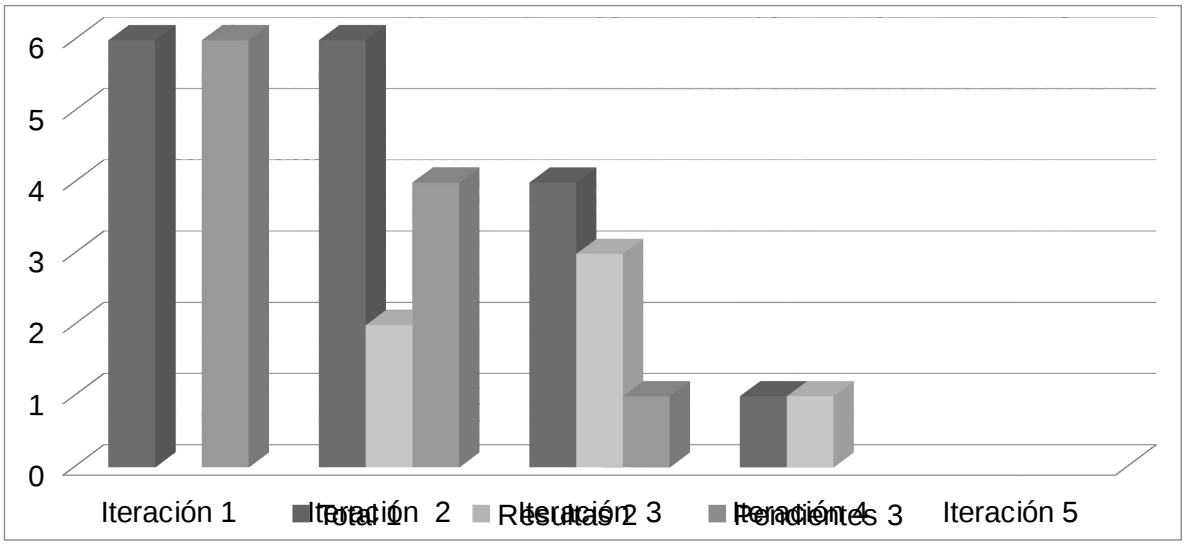
Resultados de las pruebas unitarias

Durante el proceso de pruebas de nivel unitario dividiéndose en 5 iteraciones para probar el correcto funcionamiento del sistema se detectaron 6 no conformidades que fueron satisfactoriamente corregidas. Estas no conformidades fueron:

- ✓ No se realiza correctamente la comprobación del estado (activo o inactivo) del Gclient en el sistema operativo Windows.

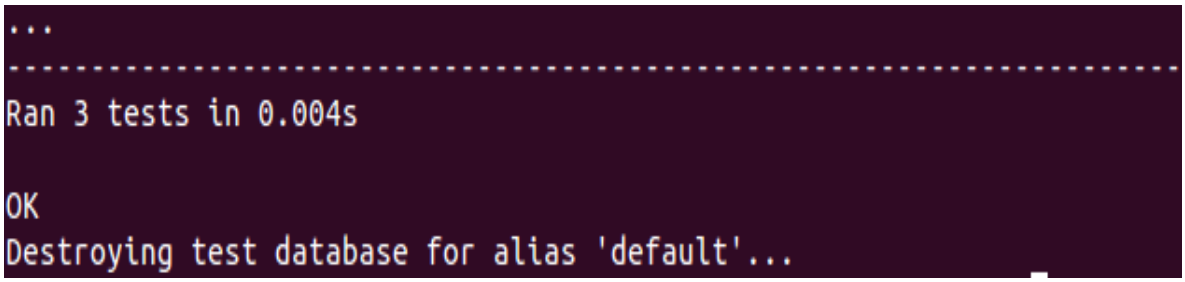
- ✓ El sistema no soporta conexión segura mediante el protocolo de comunicación HTTP.
- ✓ El sistema elimina la copia de seguridad sin comprobar que el Gclient esté ejecutándose satisfactoriamente.
- ✓ Ocurrencia de error en el sistema debido al intento de obtener algún inventario almacenado sin que este exista en el fichero de los datos.
- ✓ Problemas al filtrar.
- ✓ Se detectan errores de lógica algorítmica.

Gráfico 1 Resultados de las pruebas unitarias



La ejecución de las pruebas unitarias devino en el resultado mostrado en la siguiente imagen. Las pruebas unitarias automatizadas fueron ejecutadas utilizando PyUnit. El mismo se encuentra integrado a Django en la solución XILEMA GRHS.

Figura 12 Resultado de la ejecución de las pruebas unitarias

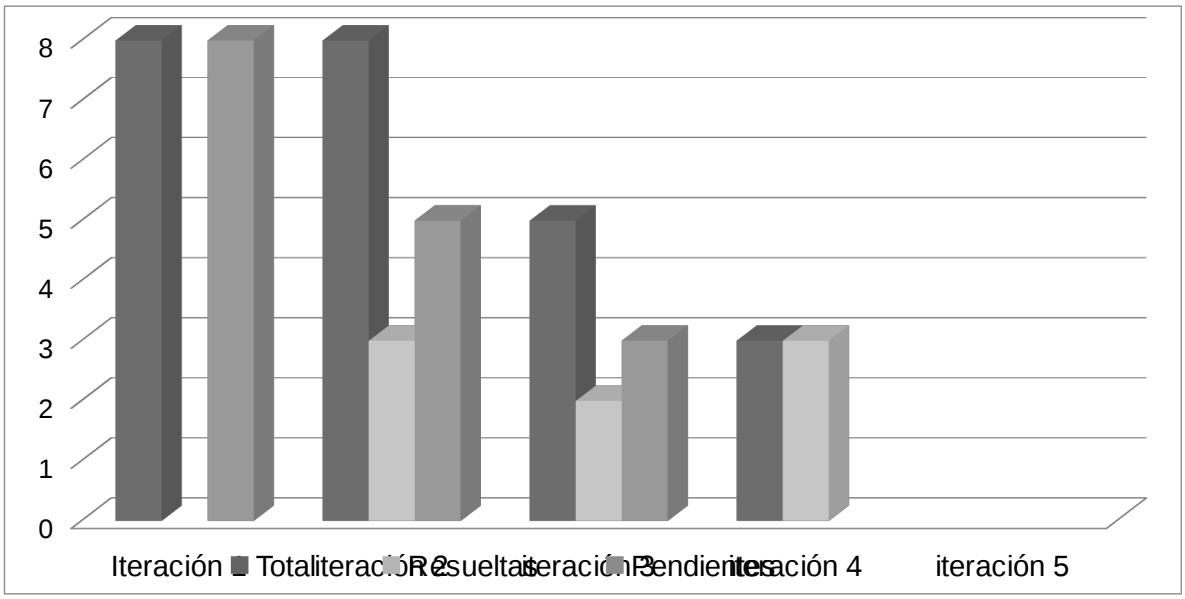


Resultados de las pruebas de integración

Durante el proceso de pruebas del nivel de integración dividiéndose en 5 iteraciones para probar el correcto funcionamiento del sistema se detectaron 8 no conformidades que fueron satisfactoriamente corregidas. Estas no conformidades fueron:

- ✓ Incompatibilidad para establecer conexiones por protocolo HTTPS utilizando el plugin *request_manager*.
- ✓ Incompatibilidad en el almacenamiento del fichero de actualización utilizando el componente *File* proporcionado por GRHS.
- ✓ Fallos en la interacción entre los componentes.
- ✓ Detección de incoherencias en el funcionamiento del módulo.
- ✓ Detección de deficiencias en el manejo de entidades dependientes por parte de los componentes, ya que se eliminaban datos que luego eran utilizados por otros elementos del sistema.
- ✓ El módulo no carga bien la pantalla principal.
- ✓ Retornos de datos incorrectos.
- ✓ Fallos en las funcionalidades al devolver los datos esperados

Gráfico 2 Resultados de las pruebas de integración

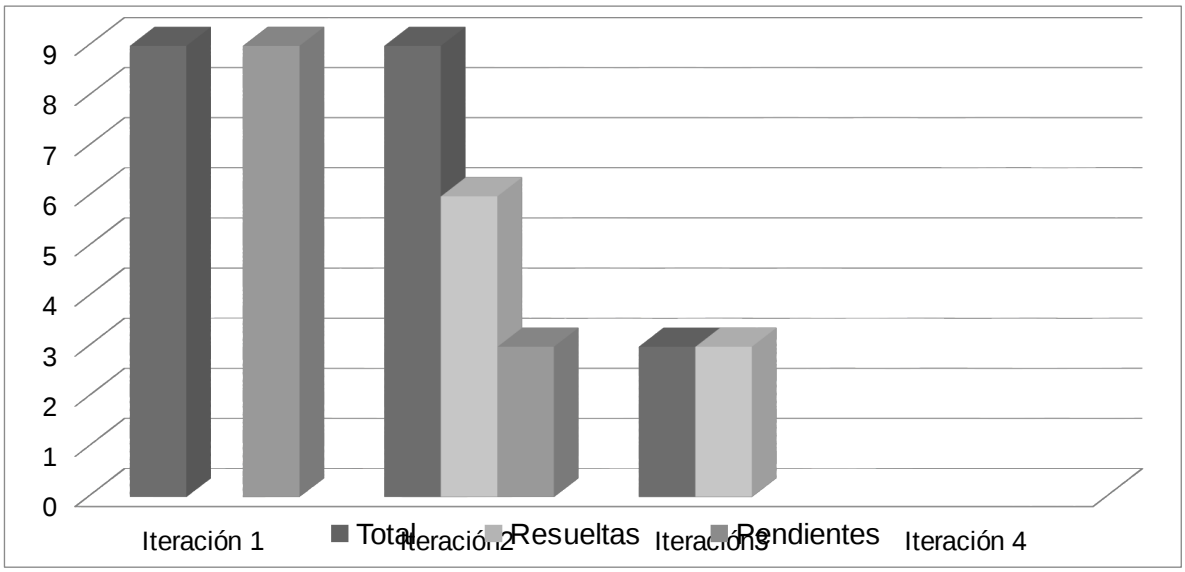


Resultados de las pruebas de aceptación

Durante el proceso de pruebas de nivel de aceptación dividiéndose en 4 iteraciones para probar el correcto funcionamiento del sistema se detectaron un total de dos no conformidades que fueron satisfactoriamente corregidas. Estas no conformidades fueron:

- ✓ No se puede definir el SO para el que estará disponible el inventario de actualizaciones y parches de seguridad.
- ✓ Errores ortográficos
- ✓ La versión de actualización registrada sólo admite números enteros.
- ✓ Errores de interfaz
- ✓ No se detiene la ejecución del inventario de actualizaciones si el servicio *gclient* no puede ser detenido.
- ✓ Intento innecesario de obtener datos del estado del SO.
- ✓ Errores ortográficos
- ✓ La versión de actualización registrada para el inventario sólo admite números enteros.
- ✓ Problemas al filtrar los datos de las actualizaciones y parches de seguridad.

Gráfico 3 Resultados de las pruebas de aceptación.



3.7. Conclusiones parciales

En este capítulo se abordó todo lo relacionado con la fase de implementación y prueba del sistema, realizando el diagrama de componentes el cual ayuda a un mejor entendimiento de los conceptos relacionados con el inventario de actualizaciones y parches de seguridad, y el diagrama de despliegue el cual se aplica con la culminación del sistema y facilitó mostrar la disposición de las particiones físicas del sistema y la asignación de los componentes de software a estas particiones. Se definieron las pruebas a aplicar para comprobar el correcto funcionamiento del sistema. También se mostraron los casos de prueba para los principales casos de uso, necesarios para determinar si el software cumple con los requisitos planteados.

Conclusiones generales

Se determinaron los elementos del diseño teórico de la investigación que constituyen el punto de partida y guían del proceso para garantizar el éxito de la investigación.

Se realizó un estudio de sistemas homólogos al módulo a desarrollar existentes a nivel mundial, donde se analizaron sus características, así como ventajas y desventajas para obtener una mejor comprensión y determinar qué características se podían aplicar a la aplicación desarrollada.

El uso de patrones de diseño y arquitectónicos brindó como resultado un diseño sólido y flexible para la codificación del módulo.

Se desarrolló la propuesta de solución basada en las necesidades del cliente y el estudio realizado sobre los sistemas similares.

A la aplicación le fueron realizadas varias pruebas, entre las que se encuentran las pruebas de aceptación, pruebas de integración y pruebas unitarias obteniendo resultados satisfactorios lo que permitió validar el correcto funcionamiento del sistema propuesto.

De manera general se desarrolló el Módulo de inventario de actualizaciones y parches de seguridad en el Gestor de Recursos de Hardware y Software (GRHS) permitiendo mayor funcionalidad y posibilidades de éxito en el proceso de llevar un inventario de los datos de las actualizaciones y parches de seguridad instalados y pendientes en el sistema operativo, concluyendo que se ha cumplido satisfactoriamente con los objetivos propuestos en el presente trabajo.

Bibliografía

1. ARIAS CHAVES, Michael. La ingeniería de requerimientos y su importancia en el desarrollo de proyectos de software. In: InterSedes: Revista de las Sedes Regionales [online]. 2005, Vol. VI, no. 10. [Accessed 23 enero 2019]. Available from: http://www.redalyc.org/resumen.oa?id=66612870011.

2. MANAGERS, Net. Administración de recursos IT: OCS-Inventory NG & GLPI. In: Net Managers [online]. [Accessed 13 noviembre 2018]. Available from: http://www.netmanagers.com.ar/stories/ocs_glpi/.

3. 5.2. Recopilar Requisitos. In: Project Management | Gladys Gbegnedji [online]. 31 octubre 2016. [Accessed 1 febrero 2019]. Available from: https://www.gladysgbegnedji.com/recopilar-requisitos/.

4. PyCharm: uno de los mejores IDE para Python. In: Escuela de Python [online]. 31 enero 2018. [Accessed 3 junio 2019]. Available from: https://www.escuelapython.com/pycharm-uno-de-los-mejores-ide-para-python/.

5. Software de gestión de inventarios [online]. S.l.: s.n., 2018. [Accessed 23 enero 2019]. Available from: https://es.wikipedia.org/w/index.php?title=Software_de_gesti%C3%B3n_de_inventarios&oldid=109280982. Page Version ID: 109280982.

6. PATRONES DE ARQUITECTURA Y DISEÑO DE SOFTWARE. In: DESARROLLO DE PÁGINAS WEB - DISEÑO DE PÁGINAS WEB [online]. 14 agosto 2018. [Accessed 7 mayo 2019]. Available from: https://www.desarrollodepaginasweb.com.mx/patrones-de-arquitectura-de-software/.

7. Inventario [online]. S.l.: s.n., 2018. [Accessed 6 diciembre 2018]. Available from: https://es.wikipedia.org/w/index.php?title=Inventario&oldid=111652075. Page Version

Referencias Bibliográficas

1. Alassafi, M. O., Hussain, R. K., Ghashgari, G., Walters, R. J., & Wills, G. B. (2017). Security in Organisations: Governance, Risks and Vulnerabilities in Moving to the Cloud. En *Lecture Notes in Computer Science*. *Enterprise Security* (pp. 241-258). https://doi.org/10.1007/978-3-319-54380-2_11

Bibliografía

1. ARIAS CHAVES, Michael. La ingeniería de requerimientos y su importancia en el desarrollo de proyectos de software. In: InterSedes: Revista de las Sedes Regionales [online]. 2005, Vol. VI, no. 10. [Accessed 23 enero 2019]. Available from: http://www.redalyc.org/resumen.oa?id=66612870011.

2. MANAGERS, Net. Administración de recursos IT: OCS-Inventory NG & GLPI. In: Net Managers [online]. [Accessed 13 noviembre 2018]. Available from: http://www.netmanagers.com.ar/stories/ocs_glpi/.

3. 5.2. Recopilar Requisitos. In: Project Management | Gladys Gbegnedji [online]. 31 octubre 2016. [Accessed 1 febrero 2019]. Available from: https://www.gladysgbegnedji.com/recopilar-requisitos/.

4. PyCharm: uno de los mejores IDE para Python. In: Escuela de Python [online]. 31 enero 2018. [Accessed 3 junio 2019]. Available from: https://www.escuelapython.com/pycharm-uno-de-los-mejores-ide-para-python/.

5. Software de gestión de inventarios [online]. S.l.: s.n., 2018. [Accessed 23 enero 2019]. Available from: https://es.wikipedia.org/w/index.php?title=Software_de_gesti%C3%B3n_de_inventarios&oldid=109280982. Page Version ID: 109280982.

6. PATRONES DE ARQUITECTURA Y DISEÑO DE SOFTWARE. In: DESARROLLO DE PÁGINAS WEB - DISEÑO DE PÁGINAS WEB [online]. 14 agosto 2018. [Accessed 7 mayo 2019]. Available from: https://www.desarrollodepaginasweb.com.mx/patrones-de-arquitectura-de-software/.

7. Inventario [online]. S.l.: s.n., 2018. [Accessed 6 diciembre 2018]. Available from: https://es.wikipedia.org/w/index.php?title=Inventario&oldid=111652075. Page Version

2. Ali, S. M. (2014). Integration of information security essential controls into information technology infrastructure library-A proposed framework. *International Journal of Applied Science and Technology*, *4*(1). https://doi.org/10.30845/ijast

3. Amin, A., & Valverde, R. (2017). Using Dashboards to Reach Acceptable Risk in Statistics Data Centers Through Risk Assessment and Impact. En *Service Science: Research and Innovations in the Service Economy*. *Engineering and Management of Data Centers* (pp. 41-72). https://doi.org/10.1007/978-3-319-65082-1_3

Bibliografía

1. ARIAS CHAVES, Michael. La ingeniería de requerimientos y su importancia en el desarrollo de proyectos de software. In: InterSedes: Revista de las Sedes Regionales [online]. 2005, Vol. VI, no. 10. [Accessed 23 enero 2019]. Available from: http://www.redalyc.org/resumen.oa?id=66612870011.

2. MANAGERS, Net. Administración de recursos IT: OCS-Inventory NG & GLPI. In: Net Managers [online]. [Accessed 13 noviembre 2018]. Available from: http://www.netmanagers.com.ar/stories/ocs_glpi/.

3. 5.2. Recopilar Requisitos. In: Project Management | Gladys Gbegnedji [online]. 31 octubre 2016. [Accessed 1 febrero 2019]. Available from: https://www.gladysgbegnedji.com/recopilar-requisitos/.

4. PyCharm: uno de los mejores IDE para Python. In: Escuela de Python [online]. 31 enero 2018. [Accessed 3 junio 2019]. Available from: https://www.escuelapython.com/pycharm-uno-de-los-mejores-ide-para-python/.

5. Software de gestión de inventarios [online]. S.l.: s.n., 2018. [Accessed 23 enero 2019]. Available from: https://es.wikipedia.org/w/index.php?title=Software_de_gesti%C3%B3n_de_inventarios&oldid=109280982. Page Version ID: 109280982.

6. PATRONES DE ARQUITECTURA Y DISEÑO DE SOFTWARE. In: DESARROLLO DE PÁGINAS WEB - DISEÑO DE PÁGINAS WEB [online]. 14 agosto 2018. [Accessed 7 mayo 2019]. Available from: https://www.desarrollodepaginasweb.com.mx/patrones-de-arquitectura-de-software/.

7. Inventario [online]. S.l.: s.n., 2018. [Accessed 6 diciembre 2018]. Available from: https://es.wikipedia.org/w/index.php?title=Inventario&oldid=111652075. Page Version

4. andrearrs. (2018, del 06 del). PyCharm hipertextual. Recuperado de http://hipertextual.com/archivo/2014/06/pycharm-ide-python/.
5. Avison, D., & Fitzgerald, G. (2003). *Information systems development: methodologies, techniques and tools (3rd edition)*. Maidenhead: McGraw Hill.
6. BASS, L., CLEMENTS, P., & KAZMAN, R. (2003). *Software architecture in practice, SEI Series in Software Engineering.* (Second). Addison Wesley.
7. CCM. (2017). *Entorno cliente/servidor.*

Bibliografía

1. ARIAS CHAVES, Michael. La ingeniería de requerimientos y su importancia en el desarrollo de proyectos de software. In: InterSedes: Revista de las Sedes Regionales [online]. 2005, Vol. VI, no. 10. [Accessed 23 enero 2019]. Available from: http://www.redalyc.org/resumen.oa?id=66612870011.

2. MANAGERS, Net. Administración de recursos IT: OCS-Inventory NG & GLPI. In: Net Managers [online]. [Accessed 13 noviembre 2018]. Available from: http://www.netmanagers.com.ar/stories/ocs_glpi/.

3. 5.2. Recopilar Requisitos. In: Project Management | Gladys Gbegnedji [online]. 31 octubre 2016. [Accessed 1 febrero 2019]. Available from: https://www.gladysgbegnedji.com/recopilar-requisitos/.

4. PyCharm: uno de los mejores IDE para Python. In: Escuela de Python [online]. 31 enero 2018. [Accessed 3 junio 2019]. Available from: https://www.escuelapython.com/pycharm-uno-de-los-mejores-ide-para-python/.

5. Software de gestión de inventarios [online]. S.l.: s.n., 2018. [Accessed 23 enero 2019]. Available from: https://es.wikipedia.org/w/index.php?title=Software_de_gesti%C3%B3n_de_inventarios&oldid=109280982. Page Version ID: 109280982.

6. PATRONES DE ARQUITECTURA Y DISEÑO DE SOFTWARE. In: DESARROLLO DE PÁGINAS WEB - DISEÑO DE PÁGINAS WEB [online]. 14 agosto 2018. [Accessed 7 mayo 2019]. Available from: https://www.desarrollodepaginasweb.com.mx/patrones-de-arquitectura-de-software/.

7. Inventario [online]. S.l.: s.n., 2018. [Accessed 6 diciembre 2018]. Available from: https://es.wikipedia.org/w/index.php?title=Inventario&oldid=111652075. Page Version

8. Django. (2014, noviembre 3). Comunidad en español de Django. Recuperado de http://www.django.es/.

9. DS, I. (2015, enero 24). Patrones Arquitectónicos. Recuperado de https://ingeniods.wordpress.com/2013/09/16/patrones-arquitectonicos/.

10. El patrón de diseño MTV (El libro de Django 1.0). (2015, abril 24). Recuperado de https://librosweb.es/libro/django_1_0/capitulo_5/el_patron_de_diseno_mtv.html.

Bibliografía

1. ARIAS CHAVES, Michael. La ingeniería de requerimientos y su importancia en el desarrollo de proyectos de software. In: InterSedes: Revista de las Sedes Regionales [online]. 2005, Vol. VI, no. 10. [Accessed 23 enero 2019]. Available from: http://www.redalyc.org/resumen.oa?id=66612870011.

2. MANAGERS, Net. Administración de recursos IT: OCS-Inventory NG & GLPI. In: Net Managers [online]. [Accessed 13 noviembre 2018]. Available from: http://www.netmanagers.com.ar/stories/ocs_glpi/.

3. 5.2. Recopilar Requisitos. In: Project Management | Gladys Gbegnedji [online]. 31 octubre 2016. [Accessed 1 febrero 2019]. Available from: https://www.gladysgbegnedji.com/recopilar-requisitos/.

4. PyCharm: uno de los mejores IDE para Python. In: Escuela de Python [online]. 31 enero 2018. [Accessed 3 junio 2019]. Available from: https://www.escuelapython.com/pycharm-uno-de-los-mejores-ide-para-python/.

5. Software de gestión de inventarios [online]. S.l.: s.n., 2018. [Accessed 23 enero 2019]. Available from: https://es.wikipedia.org/w/index.php?title=Software_de_gesti%C3%B3n_de_inventarios&oldid=109280982. Page Version ID: 109280982.

6. PATRONES DE ARQUITECTURA Y DISEÑO DE SOFTWARE. In: DESARROLLO DE PÁGINAS WEB - DISEÑO DE PÁGINAS WEB [online]. 14 agosto 2018. [Accessed 7 mayo 2019]. Available from: https://www.desarrollodepaginasweb.com.mx/patrones-de-arquitectura-de-software/.

7. Inventario [online]. S.l.: s.n., 2018. [Accessed 6 diciembre 2018]. Available from: https://es.wikipedia.org/w/index.php?title=Inventario&oldid=111652075. Page Version

11. Gartner. (2019). Recuperado 16 de febrero de 2019, de Best Client Management Tools of 2018 as Reviewed by Customers website: https://www.gartner.com/reviews/customers-choice/client-management-tools
12. Gómez Pérez, L. F. (2014, julio 14). Definición arquitectura cliente servidor. Recuperado de https://www.monografias.com/trabajos24/arquitectura-cliente-servidor/arquitectura-cliente-servidor.shtml

Bibliografía

1. ARIAS CHAVES, Michael. La ingeniería de requerimientos y su importancia en el desarrollo de proyectos de software. In: InterSedes: Revista de las Sedes Regionales [online]. 2005, Vol. VI, no. 10. [Accessed 23 enero 2019]. Available from: http://www.redalyc.org/resumen.oa?id=66612870011.

2. MANAGERS, Net. Administración de recursos IT: OCS-Inventory NG & GLPI. In: Net Managers [online]. [Accessed 13 noviembre 2018]. Available from: http://www.netmanagers.com.ar/stories/ocs_glpi/.

3. 5.2. Recopilar Requisitos. In: Project Management | Gladys Gbegnedji [online]. 31 octubre 2016. [Accessed 1 febrero 2019]. Available from: https://www.gladysgbegnedji.com/recopilar-requisitos/.

4. PyCharm: uno de los mejores IDE para Python. In: Escuela de Python [online]. 31 enero 2018. [Accessed 3 junio 2019]. Available from: https://www.escuelapython.com/pycharm-uno-de-los-mejores-ide-para-python/.

5. Software de gestión de inventarios [online]. S.l.: s.n., 2018. [Accessed 23 enero 2019]. Available from: https://es.wikipedia.org/w/index.php?title=Software_de_gesti%C3%B3n_de_inventarios&oldid=109280982. Page Version ID: 109280982.

6. PATRONES DE ARQUITECTURA Y DISEÑO DE SOFTWARE. In: DESARROLLO DE PÁGINAS WEB - DISEÑO DE PÁGINAS WEB [online]. 14 agosto 2018. [Accessed 7 mayo 2019]. Available from: https://www.desarrollodepaginasweb.com.mx/patrones-de-arquitectura-de-software/.

7. Inventario [online]. S.l.: s.n., 2018. [Accessed 6 diciembre 2018]. Available from: https://es.wikipedia.org/w/index.php?title=Inventario&oldid=111652075. Page Version

13. GRHS. (2015, mayo 12). Plugin para Xilema Base Web-Centro de Telemática. Recuperado de http://10.128.50.236/grhs-doc/index.php/Plugin_para_Xilema_Base_Web.
14. Holovaty, A., Kaplan-Moss, Jacob, & Dunck, J. (2008). El libro de Django.
15. Hsu, T. (2018). *Hands-On Security in DevOps: Ensure continuous security, deployment, and delivery with DevSecOps*. Packt Publishing Ltd.

Bibliografía

1. ARIAS CHAVES, Michael. La ingeniería de requerimientos y su importancia en el desarrollo de proyectos de software. In: InterSedes: Revista de las Sedes Regionales [online]. 2005, Vol. VI, no. 10. [Accessed 23 enero 2019]. Available from: http://www.redalyc.org/resumen.oa?id=66612870011.

2. MANAGERS, Net. Administración de recursos IT: OCS-Inventory NG & GLPI. In: Net Managers [online]. [Accessed 13 noviembre 2018]. Available from: http://www.netmanagers.com.ar/stories/ocs_glpi/.

3. 5.2. Recopilar Requisitos. In: Project Management | Gladys Gbegnedji [online]. 31 octubre 2016. [Accessed 1 febrero 2019]. Available from: https://www.gladysgbegnedji.com/recopilar-requisitos/.

4. PyCharm: uno de los mejores IDE para Python. In: Escuela de Python [online]. 31 enero 2018. [Accessed 3 junio 2019]. Available from: https://www.escuelapython.com/pycharm-uno-de-los-mejores-ide-para-python/.

5. Software de gestión de inventarios [online]. S.l.: s.n., 2018. [Accessed 23 enero 2019]. Available from: https://es.wikipedia.org/w/index.php?title=Software_de_gesti%C3%B3n_de_inventarios&oldid=109280982. Page Version ID: 109280982.

6. PATRONES DE ARQUITECTURA Y DISEÑO DE SOFTWARE. In: DESARROLLO DE PÁGINAS WEB - DISEÑO DE PÁGINAS WEB [online]. 14 agosto 2018. [Accessed 7 mayo 2019]. Available from: https://www.desarrollodepaginasweb.com.mx/patrones-de-arquitectura-de-software/.

7. Inventario [online]. S.l.: s.n., 2018. [Accessed 6 diciembre 2018]. Available from: https://es.wikipedia.org/w/index.php?title=Inventario&oldid=111652075. Page Version

16. ISO/IEC. (2005). *“ISO/IEC 27001: Information technology - Security techniques - Information security management systems - Requirements*. International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC.
17. JACOBSON, I., BOOCH, G., & RUMBAUGH, J. (2004). *El proceso unificado de desarrollo de software.* Editorial Félix Varela.
18. Larman. (2003). *Craig. Uml y Patrones* (2da Edición).

Bibliografía

1. ARIAS CHAVES, Michael. La ingeniería de requerimientos y su importancia en el desarrollo de proyectos de software. In: InterSedes: Revista de las Sedes Regionales [online]. 2005, Vol. VI, no. 10. [Accessed 23 enero 2019]. Available from: http://www.redalyc.org/resumen.oa?id=66612870011.

2. MANAGERS, Net. Administración de recursos IT: OCS-Inventory NG & GLPI. In: Net Managers [online]. [Accessed 13 noviembre 2018]. Available from: http://www.netmanagers.com.ar/stories/ocs_glpi/.

3. 5.2. Recopilar Requisitos. In: Project Management | Gladys Gbegnedji [online]. 31 octubre 2016. [Accessed 1 febrero 2019]. Available from: https://www.gladysgbegnedji.com/recopilar-requisitos/.

4. PyCharm: uno de los mejores IDE para Python. In: Escuela de Python [online]. 31 enero 2018. [Accessed 3 junio 2019]. Available from: https://www.escuelapython.com/pycharm-uno-de-los-mejores-ide-para-python/.

5. Software de gestión de inventarios [online]. S.l.: s.n., 2018. [Accessed 23 enero 2019]. Available from: https://es.wikipedia.org/w/index.php?title=Software_de_gesti%C3%B3n_de_inventarios&oldid=109280982. Page Version ID: 109280982.

6. PATRONES DE ARQUITECTURA Y DISEÑO DE SOFTWARE. In: DESARROLLO DE PÁGINAS WEB - DISEÑO DE PÁGINAS WEB [online]. 14 agosto 2018. [Accessed 7 mayo 2019]. Available from: https://www.desarrollodepaginasweb.com.mx/patrones-de-arquitectura-de-software/.

7. Inventario [online]. S.l.: s.n., 2018. [Accessed 6 diciembre 2018]. Available from: https://es.wikipedia.org/w/index.php?title=Inventario&oldid=111652075. Page Version

19. LPG, L. P. G. (2009). Login Inventory. Recuperado de http://www.linux-party.com/modules.php?name=News&file=article&sid=4468.
20. Manager, N. (2014, noviembre 6). NetSupport Manager. Recuperado de http://www.netsupportmanager.com/es/features.asp.
21. Mediavilla, E. (2010). *Programacion orientada a objetos. Master de computacion.Modelado de casos de uso*.

Bibliografía

1. ARIAS CHAVES, Michael. La ingeniería de requerimientos y su importancia en el desarrollo de proyectos de software. In: InterSedes: Revista de las Sedes Regionales [online]. 2005, Vol. VI, no. 10. [Accessed 23 enero 2019]. Available from: http://www.redalyc.org/resumen.oa?id=66612870011.

2. MANAGERS, Net. Administración de recursos IT: OCS-Inventory NG & GLPI. In: Net Managers [online]. [Accessed 13 noviembre 2018]. Available from: http://www.netmanagers.com.ar/stories/ocs_glpi/.

3. 5.2. Recopilar Requisitos. In: Project Management | Gladys Gbegnedji [online]. 31 octubre 2016. [Accessed 1 febrero 2019]. Available from: https://www.gladysgbegnedji.com/recopilar-requisitos/.

4. PyCharm: uno de los mejores IDE para Python. In: Escuela de Python [online]. 31 enero 2018. [Accessed 3 junio 2019]. Available from: https://www.escuelapython.com/pycharm-uno-de-los-mejores-ide-para-python/.

5. Software de gestión de inventarios [online]. S.l.: s.n., 2018. [Accessed 23 enero 2019]. Available from: https://es.wikipedia.org/w/index.php?title=Software_de_gesti%C3%B3n_de_inventarios&oldid=109280982. Page Version ID: 109280982.

6. PATRONES DE ARQUITECTURA Y DISEÑO DE SOFTWARE. In: DESARROLLO DE PÁGINAS WEB - DISEÑO DE PÁGINAS WEB [online]. 14 agosto 2018. [Accessed 7 mayo 2019]. Available from: https://www.desarrollodepaginasweb.com.mx/patrones-de-arquitectura-de-software/.

7. Inventario [online]. S.l.: s.n., 2018. [Accessed 6 diciembre 2018]. Available from: https://es.wikipedia.org/w/index.php?title=Inventario&oldid=111652075. Page Version

22. Microsoft. (2015). Microsoft Windows. Windows Update. Recuperado de http://windows.microsoft.com/es-419/windows7/products/features/windows-update.
23. Modelado de Datos. (2018). Recuperado de https://www.tecnologias-informacion.com/modeladodatos.html
24. Montesino Perurena, R. (2012). *MODELO PARA LA GESTIÓN AUTOMATIZADA E INTEGRADA DE CONTROLES DE SEGURIDAD INFORMÁTICA* (Dotorado). Universidad de las Ciencias Informáticas, La Habana, Cuba.

Bibliografía

1. ARIAS CHAVES, Michael. La ingeniería de requerimientos y su importancia en el desarrollo de proyectos de software. In: InterSedes: Revista de las Sedes Regionales [online]. 2005, Vol. VI, no. 10. [Accessed 23 enero 2019]. Available from: http://www.redalyc.org/resumen.oa?id=66612870011.

2. MANAGERS, Net. Administración de recursos IT: OCS-Inventory NG & GLPI. In: Net Managers [online]. [Accessed 13 noviembre 2018]. Available from: http://www.netmanagers.com.ar/stories/ocs_glpi/.

3. 5.2. Recopilar Requisitos. In: Project Management | Gladys Gbegnedji [online]. 31 octubre 2016. [Accessed 1 febrero 2019]. Available from: https://www.gladysgbegnedji.com/recopilar-requisitos/.

4. PyCharm: uno de los mejores IDE para Python. In: Escuela de Python [online]. 31 enero 2018. [Accessed 3 junio 2019]. Available from: https://www.escuelapython.com/pycharm-uno-de-los-mejores-ide-para-python/.

5. Software de gestión de inventarios [online]. S.l.: s.n., 2018. [Accessed 23 enero 2019]. Available from: https://es.wikipedia.org/w/index.php?title=Software_de_gesti%C3%B3n_de_inventarios&oldid=109280982. Page Version ID: 109280982.

6. PATRONES DE ARQUITECTURA Y DISEÑO DE SOFTWARE. In: DESARROLLO DE PÁGINAS WEB - DISEÑO DE PÁGINAS WEB [online]. 14 agosto 2018. [Accessed 7 mayo 2019]. Available from: https://www.desarrollodepaginasweb.com.mx/patrones-de-arquitectura-de-software/.

7. Inventario [online]. S.l.: s.n., 2018. [Accessed 6 diciembre 2018]. Available from: https://es.wikipedia.org/w/index.php?title=Inventario&oldid=111652075. Page Version

25. Nabber. (2014). Appupdater Project. Recuperado de http://www.nabber.org/projects/appupdater/
26. OCS. (2010). OCS Inventory NG. Recuperado de http://www.ocsinventory-ng.org/en/
27. Paradigm, V. (2013). *Visual paradigm for uml. Visual Paradigm for UML-UML tool for software application development*.

Bibliografía

1. ARIAS CHAVES, Michael. La ingeniería de requerimientos y su importancia en el desarrollo de proyectos de software. In: InterSedes: Revista de las Sedes Regionales [online]. 2005, Vol. VI, no. 10. [Accessed 23 enero 2019]. Available from: http://www.redalyc.org/resumen.oa?id=66612870011.

2. MANAGERS, Net. Administración de recursos IT: OCS-Inventory NG & GLPI. In: Net Managers [online]. [Accessed 13 noviembre 2018]. Available from: http://www.netmanagers.com.ar/stories/ocs_glpi/.

3. 5.2. Recopilar Requisitos. In: Project Management | Gladys Gbegnedji [online]. 31 octubre 2016. [Accessed 1 febrero 2019]. Available from: https://www.gladysgbegnedji.com/recopilar-requisitos/.

4. PyCharm: uno de los mejores IDE para Python. In: Escuela de Python [online]. 31 enero 2018. [Accessed 3 junio 2019]. Available from: https://www.escuelapython.com/pycharm-uno-de-los-mejores-ide-para-python/.

5. Software de gestión de inventarios [online]. S.l.: s.n., 2018. [Accessed 23 enero 2019]. Available from: https://es.wikipedia.org/w/index.php?title=Software_de_gesti%C3%B3n_de_inventarios&oldid=109280982. Page Version ID: 109280982.

6. PATRONES DE ARQUITECTURA Y DISEÑO DE SOFTWARE. In: DESARROLLO DE PÁGINAS WEB - DISEÑO DE PÁGINAS WEB [online]. 14 agosto 2018. [Accessed 7 mayo 2019]. Available from: https://www.desarrollodepaginasweb.com.mx/patrones-de-arquitectura-de-software/.

7. Inventario [online]. S.l.: s.n., 2018. [Accessed 6 diciembre 2018]. Available from: https://es.wikipedia.org/w/index.php?title=Inventario&oldid=111652075. Page Version

28. PgAdmin. (2018, noviembre 13). PgAdmin IV - Guía Ubuntu. Recuperado de https://www.guia-ubuntu.com/index.php?title=PgAdmin_IV
29. PIÑEIRO GOMEZ, J. M. (2013). *BASES DE DATOS RELACIONALES Y MODELADO DE DATOS*. Recuperado de https://www.casadellibro.com/libro-bases-de-datos-relacionales-y-modelado-de-datos/9788428333566/2130244
30. PostgreSQL-ES. (2018, noviembre 13). Recuperado de https://e-mc2.net/es/postgresql-es#sobre-postgresql

Bibliografía

1. ARIAS CHAVES, Michael. La ingeniería de requerimientos y su importancia en el desarrollo de proyectos de software. In: InterSedes: Revista de las Sedes Regionales [online]. 2005, Vol. VI, no. 10. [Accessed 23 enero 2019]. Available from: http://www.redalyc.org/resumen.oa?id=66612870011.

2. MANAGERS, Net. Administración de recursos IT: OCS-Inventory NG & GLPI. In: Net Managers [online]. [Accessed 13 noviembre 2018]. Available from: http://www.netmanagers.com.ar/stories/ocs_glpi/.

3. 5.2. Recopilar Requisitos. In: Project Management | Gladys Gbegnedji [online]. 31 octubre 2016. [Accessed 1 febrero 2019]. Available from: https://www.gladysgbegnedji.com/recopilar-requisitos/.

4. PyCharm: uno de los mejores IDE para Python. In: Escuela de Python [online]. 31 enero 2018. [Accessed 3 junio 2019]. Available from: https://www.escuelapython.com/pycharm-uno-de-los-mejores-ide-para-python/.

5. Software de gestión de inventarios [online]. S.l.: s.n., 2018. [Accessed 23 enero 2019]. Available from: https://es.wikipedia.org/w/index.php?title=Software_de_gesti%C3%B3n_de_inventarios&oldid=109280982. Page Version ID: 109280982.

6. PATRONES DE ARQUITECTURA Y DISEÑO DE SOFTWARE. In: DESARROLLO DE PÁGINAS WEB - DISEÑO DE PÁGINAS WEB [online]. 14 agosto 2018. [Accessed 7 mayo 2019]. Available from: https://www.desarrollodepaginasweb.com.mx/patrones-de-arquitectura-de-software/.

7. Inventario [online]. S.l.: s.n., 2018. [Accessed 6 diciembre 2018]. Available from: https://es.wikipedia.org/w/index.php?title=Inventario&oldid=111652075. Page Version

31. Pressman, R. S. (2010). *Ingeniería del Software: Un Enfoque Práctico.* Madrid: Sexta. : McGraw-Hill.
32. PRESSMAN, R. S. (2014). *Ingeniería del software. Un enfoque práctico*.
33. PyCharm, J. (2015, febrero 10). Python IDE & Django IDE for Web developers.
34. ¿Qué es un gestor de datos y para qué sirve? (2018, noviembre 13). Recuperado de https://blog.powerdata.es/el-valor-de-la-gestion-de-datos/que-es-un-gestor-de-datos-y-para-que-sirve.

Bibliografía

1. ARIAS CHAVES, Michael. La ingeniería de requerimientos y su importancia en el desarrollo de proyectos de software. In: InterSedes: Revista de las Sedes Regionales [online]. 2005, Vol. VI, no. 10. [Accessed 23 enero 2019]. Available from: http://www.redalyc.org/resumen.oa?id=66612870011.

2. MANAGERS, Net. Administración de recursos IT: OCS-Inventory NG & GLPI. In: Net Managers [online]. [Accessed 13 noviembre 2018]. Available from: http://www.netmanagers.com.ar/stories/ocs_glpi/.

3. 5.2. Recopilar Requisitos. In: Project Management | Gladys Gbegnedji [online]. 31 octubre 2016. [Accessed 1 febrero 2019]. Available from: https://www.gladysgbegnedji.com/recopilar-requisitos/.

4. PyCharm: uno de los mejores IDE para Python. In: Escuela de Python [online]. 31 enero 2018. [Accessed 3 junio 2019]. Available from: https://www.escuelapython.com/pycharm-uno-de-los-mejores-ide-para-python/.

5. Software de gestión de inventarios [online]. S.l.: s.n., 2018. [Accessed 23 enero 2019]. Available from: https://es.wikipedia.org/w/index.php?title=Software_de_gesti%C3%B3n_de_inventarios&oldid=109280982. Page Version ID: 109280982.

6. PATRONES DE ARQUITECTURA Y DISEÑO DE SOFTWARE. In: DESARROLLO DE PÁGINAS WEB - DISEÑO DE PÁGINAS WEB [online]. 14 agosto 2018. [Accessed 7 mayo 2019]. Available from: https://www.desarrollodepaginasweb.com.mx/patrones-de-arquitectura-de-software/.

7. Inventario [online]. S.l.: s.n., 2018. [Accessed 6 diciembre 2018]. Available from: https://es.wikipedia.org/w/index.php?title=Inventario&oldid=111652075. Page Version

35. RAE. (2010). Real Academia Española. Inventario. Recuperado de http://buscon.rae.es/draeI/SrvltConsulta?TIPO_BUS=3&LEMA=inventario.
36. REYNOSO, C. B. (2004, junio 15). Introducción a la Arquitectura de Software. Recuperado de http://sunshine.prod.uci.cu/book/4e73a4c305717427e9000003/
37. Ruiz, & Vaillant. (2015). *Ingeniaria del Software.*
38. Sánchez Guerra, sther. (2008). Diagrama de secuencia en UML. Recuperado de http://Tutorial de diagrama de secuencia UML%20%20%20Lucidchart.htm

Bibliografía

1. ARIAS CHAVES, Michael. La ingeniería de requerimientos y su importancia en el desarrollo de proyectos de software. In: InterSedes: Revista de las Sedes Regionales [online]. 2005, Vol. VI, no. 10. [Accessed 23 enero 2019]. Available from: http://www.redalyc.org/resumen.oa?id=66612870011.

2. MANAGERS, Net. Administración de recursos IT: OCS-Inventory NG & GLPI. In: Net Managers [online]. [Accessed 13 noviembre 2018]. Available from: http://www.netmanagers.com.ar/stories/ocs_glpi/.

3. 5.2. Recopilar Requisitos. In: Project Management | Gladys Gbegnedji [online]. 31 octubre 2016. [Accessed 1 febrero 2019]. Available from: https://www.gladysgbegnedji.com/recopilar-requisitos/.

4. PyCharm: uno de los mejores IDE para Python. In: Escuela de Python [online]. 31 enero 2018. [Accessed 3 junio 2019]. Available from: https://www.escuelapython.com/pycharm-uno-de-los-mejores-ide-para-python/.

5. Software de gestión de inventarios [online]. S.l.: s.n., 2018. [Accessed 23 enero 2019]. Available from: https://es.wikipedia.org/w/index.php?title=Software_de_gesti%C3%B3n_de_inventarios&oldid=109280982. Page Version ID: 109280982.

6. PATRONES DE ARQUITECTURA Y DISEÑO DE SOFTWARE. In: DESARROLLO DE PÁGINAS WEB - DISEÑO DE PÁGINAS WEB [online]. 14 agosto 2018. [Accessed 7 mayo 2019]. Available from: https://www.desarrollodepaginasweb.com.mx/patrones-de-arquitectura-de-software/.

7. Inventario [online]. S.l.: s.n., 2018. [Accessed 6 diciembre 2018]. Available from: https://es.wikipedia.org/w/index.php?title=Inventario&oldid=111652075. Page Version

39. Sommerville, I. (2005a). *Ingeniaria del Softwar* (7ma ed.). Madrid: PEARSON EDUCACION,.
40. Sommerville, I. (2005b). *Ingeniería del software.* Pearson Educación, S.A.
41. Soomro, Z. A., Shah, M. H., & Ahmed, J. (2016). Information security management needs more holistic approach: A literature review. *International Journal of Information Management*, *36*(2), 215-225. https://doi.org/10.1016/j.ijinfomgt.2015.11.009

Bibliografía

1. ARIAS CHAVES, Michael. La ingeniería de requerimientos y su importancia en el desarrollo de proyectos de software. In: InterSedes: Revista de las Sedes Regionales [online]. 2005, Vol. VI, no. 10. [Accessed 23 enero 2019]. Available from: http://www.redalyc.org/resumen.oa?id=66612870011.

2. MANAGERS, Net. Administración de recursos IT: OCS-Inventory NG & GLPI. In: Net Managers [online]. [Accessed 13 noviembre 2018]. Available from: http://www.netmanagers.com.ar/stories/ocs_glpi/.

3. 5.2. Recopilar Requisitos. In: Project Management | Gladys Gbegnedji [online]. 31 octubre 2016. [Accessed 1 febrero 2019]. Available from: https://www.gladysgbegnedji.com/recopilar-requisitos/.

4. PyCharm: uno de los mejores IDE para Python. In: Escuela de Python [online]. 31 enero 2018. [Accessed 3 junio 2019]. Available from: https://www.escuelapython.com/pycharm-uno-de-los-mejores-ide-para-python/.

5. Software de gestión de inventarios [online]. S.l.: s.n., 2018. [Accessed 23 enero 2019]. Available from: https://es.wikipedia.org/w/index.php?title=Software_de_gesti%C3%B3n_de_inventarios&oldid=109280982. Page Version ID: 109280982.

6. PATRONES DE ARQUITECTURA Y DISEÑO DE SOFTWARE. In: DESARROLLO DE PÁGINAS WEB - DISEÑO DE PÁGINAS WEB [online]. 14 agosto 2018. [Accessed 7 mayo 2019]. Available from: https://www.desarrollodepaginasweb.com.mx/patrones-de-arquitectura-de-software/.

7. Inventario [online]. S.l.: s.n., 2018. [Accessed 6 diciembre 2018]. Available from: https://es.wikipedia.org/w/index.php?title=Inventario&oldid=111652075. Page Version

42. Symantec Corporation. (2014). LiveUpdate. Recuperado de http://www.symantec.com/about/news/release/article.jsp?prid=19960618_01
43. Tinoco Gómez, O., Rosales López, P. P., & Salas Bacalla, J. (2010). Criterios de selección de metodologías de desarrollo de software. *Sistema e Informática*, *3*(1), 70-74.
44. TLM, C. T. (2015). *Expediente de Sistema Xilema GRHS*. Recuperado de http://10.128.50.236/grhs-doc/index.php/GRHS

Bibliografía

1. ARIAS CHAVES, Michael. La ingeniería de requerimientos y su importancia en el desarrollo de proyectos de software. In: InterSedes: Revista de las Sedes Regionales [online]. 2005, Vol. VI, no. 10. [Accessed 23 enero 2019]. Available from: http://www.redalyc.org/resumen.oa?id=66612870011.

2. MANAGERS, Net. Administración de recursos IT: OCS-Inventory NG & GLPI. In: Net Managers [online]. [Accessed 13 noviembre 2018]. Available from: http://www.netmanagers.com.ar/stories/ocs_glpi/.

3. 5.2. Recopilar Requisitos. In: Project Management | Gladys Gbegnedji [online]. 31 octubre 2016. [Accessed 1 febrero 2019]. Available from: https://www.gladysgbegnedji.com/recopilar-requisitos/.

4. PyCharm: uno de los mejores IDE para Python. In: Escuela de Python [online]. 31 enero 2018. [Accessed 3 junio 2019]. Available from: https://www.escuelapython.com/pycharm-uno-de-los-mejores-ide-para-python/.

5. Software de gestión de inventarios [online]. S.l.: s.n., 2018. [Accessed 23 enero 2019]. Available from: https://es.wikipedia.org/w/index.php?title=Software_de_gesti%C3%B3n_de_inventarios&oldid=109280982. Page Version ID: 109280982.

6. PATRONES DE ARQUITECTURA Y DISEÑO DE SOFTWARE. In: DESARROLLO DE PÁGINAS WEB - DISEÑO DE PÁGINAS WEB [online]. 14 agosto 2018. [Accessed 7 mayo 2019]. Available from: https://www.desarrollodepaginasweb.com.mx/patrones-de-arquitectura-de-software/.

7. Inventario [online]. S.l.: s.n., 2018. [Accessed 6 diciembre 2018]. Available from: https://es.wikipedia.org/w/index.php?title=Inventario&oldid=111652075. Page Version

45. T.R, S. (2015). *Metodología de desarrollo para la Actividad productiva de la UCI.*
46. Universidad Nacional de Colombia. (2015, marzo 14). Patrones de Diseño. Recuperado de https://msdn.microsoft.com/es-es/library/bb972240.aspx.
47. Visconti, M., & Astudillo, H. (2012). *Fundamentos de Ingeniería de Software*. Recuperado de http://roa.ult.edu.cu/bitstream/123456789/401/1/08-Patrones.pdf
48. Wolden, M., Valverde, R., & Talla, M. (2015). The effectiveness of COBIT 5 Information Security Framework for reducing Cyber Attacks on Supply Chain

Bibliografía

1. ARIAS CHAVES, Michael. La ingeniería de requerimientos y su importancia en el desarrollo de proyectos de software. In: InterSedes: Revista de las Sedes Regionales [online]. 2005, Vol. VI, no. 10. [Accessed 23 enero 2019]. Available from: http://www.redalyc.org/resumen.oa?id=66612870011.

2. MANAGERS, Net. Administración de recursos IT: OCS-Inventory NG & GLPI. In: Net Managers [online]. [Accessed 13 noviembre 2018]. Available from: http://www.netmanagers.com.ar/stories/ocs_glpi/.

3. 5.2. Recopilar Requisitos. In: Project Management | Gladys Gbegnedji [online]. 31 octubre 2016. [Accessed 1 febrero 2019]. Available from: https://www.gladysgbegnedji.com/recopilar-requisitos/.

4. PyCharm: uno de los mejores IDE para Python. In: Escuela de Python [online]. 31 enero 2018. [Accessed 3 junio 2019]. Available from: https://www.escuelapython.com/pycharm-uno-de-los-mejores-ide-para-python/.

5. Software de gestión de inventarios [online]. S.l.: s.n., 2018. [Accessed 23 enero 2019]. Available from: https://es.wikipedia.org/w/index.php?title=Software_de_gesti%C3%B3n_de_inventarios&oldid=109280982. Page Version ID: 109280982.

6. PATRONES DE ARQUITECTURA Y DISEÑO DE SOFTWARE. In: DESARROLLO DE PÁGINAS WEB - DISEÑO DE PÁGINAS WEB [online]. 14 agosto 2018. [Accessed 7 mayo 2019]. Available from: https://www.desarrollodepaginasweb.com.mx/patrones-de-arquitectura-de-software/.

7. Inventario [online]. S.l.: s.n., 2018. [Accessed 6 diciembre 2018]. Available from: https://es.wikipedia.org/w/index.php?title=Inventario&oldid=111652075. Page Version Management System. *15th IFAC Symposium onInformation Control Problems inManufacturingINCOM 2015*, *48*(3), 1846-1852. https://doi.org/10.1016/j.ifacol.2015.06.355