MÓDULO DE REGISTRO DE EVENTOS PARA NOVA-LTSP

Trabajo de Diploma para Optar por el Título de

Ingeniero en Ciencias Informáticas

Autor:

Ana Isabel Marí Martín

Tutores:

Ing. Yasiel Pérez Villazón

Ing. María Leisy González Carrera

Ing. Lexys Manuel Díaz Alonso

La Habana, junio 2018

“Año 60 de la Revolución”

Nuestra recompensa se encuentra en el esfuerzo y no el resultado. Un esfuerzo total es una victoria completa.

Mahatma Gandhi

DECLARACIÓN JURADA DE AUTORÍA

Declaro por este medio que yo Ana Isabel Marí Martín, con carné de identidad 94051230973 soy la autora principal del trabajo de diploma titulado *"Módulo de registro de eventos para Nova-LTSP"* y autorizo a la Universidad de las Ciencias Informáticas a hacer uso de la misma en su beneficio, así como los derechos patrimoniales con carácter exclusivo.

Para que así conste se firma la presente, a los 19 días del mes de junio del año 2018.

Autor
Ana Isabel Marí Martín

Tutor
Ing. Yasiel Pérez Villazón

Tutor
Ing. María Leisy González Carrera

Tutor
Ing. Lexys Manuel Díaz Alonso

*A mi **mamá** y **papá**, por guiarme y darme apoyo en los momentos más duros de toda la carrera universitaria, a ustedes va dedicado este trabajo. Los quiero.*

*A mi **hermana**, por ser mi compañera en este viaje y darme todo su apoyo. Te quiero.*

*A mis **abuelas**, por su cariño y amor incondicional, siempre pendientes a mis estudios.*

*A mi **abuelo**, sé que donde quiera que estés en este momento te sientes muy orgulloso de mí.*

*A toda mi **familia** por creer en mí en este proyecto que hoy se hace realidad.*

A ustedes, con todo mi corazón

Ana Isabel Marí Martín

Hoy es uno de los días más importantes de mi vida, hoy me graduó de ingeniera en ciencias informáticas, por ello no puedo dejar de agradecer a todas esas personas que contribuyeron a que este momento tan especial se cumpliera, por tal motivo agradezco.

A mi mamá por ser la guía en todo momento de mi vida, por su apoyo incondicional, por su amor y cariño, por demostrarme que este día podía ser posible con esfuerzo y esmero. Y hoy te digo, tenías razón mami.

A mi papá por enseñarme a luchar por las cosas que uno desea, por tu amor y apoyo constante.

A mis abuelas, Nora y Chabela que me han ayudado mucho a lo largo de mi vida y hoy están más orgullosas que nunca de esta victoria.

A mi hermana por su apoyo incondicional.

A mi tía Thais por su cariño y estar siempre pendiente de mí.

A toda mi familia, por amarme y guiarme por el camino correcto.

A mi novio Gustavo, por todo su apoyo, compresión y paciencia durante todo este proceso.

A todos mis compañeros de aula quienes han sido mi familia en la UCI durante estos cinco años, en especial a Maylenis, Bia, Grettel, Ada y Laura.

A mis otros compañeros de aula con los que pase mis primeros momentos en la universidad Ortelio, Osmel, Carlos, Aimet, María, Arlene, Alejandro.

A mis tutores Yasiel, María Leisy y Lexys, por confiar en mí y por su ayuda incondicional en el desarrollo de esta investigación.

A todos mis profesores que hoy han contribuido en mi formación como profesional.

En general a todas las personas que hicieron posible que hoy este aquí graduándome de Ingeniera en Ciencias Informáticas, A todos mis más sinceros agradecimientos.

RESUMEN

La Distribución Cubana GNU/Linux Nova, cuenta con la plataforma de administración de clientes ligeros Nova-LTSP, que brinda un conjunto de funcionalidades para administrar estos clientes ligeros. Actualmente, la herramienta carece del monitoreo de los registros de eventos de la aplicación, elemento fundamental para llevar a cabo el control y seguimiento de las acciones que los usuarios realizan sobre el sistema. Debido a esto, la presente investigación tiene como objetivo principal desarrollar un módulo que permita el registro de eventos para la plataforma Nova-LTSP. La propuesta de solución es guiada por la metodología de desarrollo de software variación AUP-UCI en su escenario Historias de usuario. Para la implementación se selecciona Python 2.7 como lenguaje de programación sobre el *framework* Django en su versión 1.10. Como sistema gestor de base de datos se utiliza PostgreSQL 9.4 y como entorno integrado de desarrollo PyCharm2017 en su versión 2.2. Para la validación de la herramienta se define una estrategia de prueba que permite asegurar que el módulo es funcional y que este cumple con los requisitos del cliente tales como: integración, funcionales, de carga y estrés, aceptación, garantizando un correcto funcionamiento del módulo. Al culminar con el desarrollo de la investigación se obtiene un módulo para la plataforma Nova-LTSP que permite el registro de eventos.

Palabras clave: eventos, módulo, monitoreo, Nova-LTSP, registro.

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

INTRODUCCIÓN

INTRODUCCIÓN

Las Tecnologías de la Información y las Comunicaciones (TICs) han tenido un desarrollo explosivo en el transcurso del tiempo, al punto de que han dado forma a lo que se denomina “Sociedad del Conocimiento” o de la Información. La informatización de la sociedad se define en Cuba como el proceso de utilización ordenada y masiva de las TIC para satisfacer las necesidades de información y conocimiento de todas las personas y esferas de la sociedad (Medina Carbó, 2016). Sin embargo, este proceso se ha visto afectado debido al bloqueo económico y financiero impuesto por los Estados Unidos de América a Cuba, lo cual resulta una barrera para mantenerse a la par del modelo consumista impuesto por las economías capitalistas en el área de las TIC. Teniendo en cuenta la dependencia de productos extranjeros y el alto riesgo que implican en la seguridad nacional, surge la necesidad de apostar por la utilización de software libre para lograr la soberanía tecnológica del país (Rodríguez Figueredo, y otros, 2015).

A partir de la necesidad de llevar la tecnología a diversos sectores de la sociedad cubana y desarrollar la industria del software manteniendo los principios de soberanía tecnológica, seguridad, socio adaptabilidad y sostenibilidad, en el año 2004 el Consejo de Ministros decide dar inicio a un proceso de migración a plataformas libres y de código abierto (Pérez Villazón, y otros, 2013). Actualmente la Universidad de las Ciencias Informáticas (UCI) representa un eslabón principal en dicho proceso, siendo uno de los grandes logros de la Revolución cubana en aras de consolidar y fomentar el desarrollo de la informática.

Dentro de su infraestructura docente-productiva se encuentra el Centro de Software Libre (CESOL) de la Facultad 1. El mismo tiene como objetivo desarrollar la distribución cubana GNU/Linux Nova y conducir los procesos de migración a aplicaciones de código abierto, desde un modelo de integración de la formación, investigación y el postgrado, contribuyendo así a la formación integral de profesionales comprometidos con la revolución que respondan a las necesidades del progreso científico-técnico y socio-económico.

La distribución de GNU/Linux Nova surge como un proyecto desarrollado por estudiantes y profesores, con la participación de miembros de otras instituciones, para apoyar la migración a tecnologías de software libre y código abierto en Cuba (Pierra, 2011). “La obsolescencia tecnológica ocasionada por las rápidas trasformaciones que se producen en el hardware y las limitaciones económicas que debe enfrentar Cuba y en particular el Ministerio de Educación Superior (MES) han ocasionado la búsqueda de alternativas que permitan alargar la vida útil de las PC adquiridas sin limitar el uso de las TICs; una

de estas alternativas es conocida internacionalmente como Cliente Ligero" (Porro Santos, y otros, 2015). "Un cliente liviano o cliente ligero es una computadora cliente o un software de cliente en una arquitectura de red cliente-servidor que depende primariamente del servidor central para las tareas de procesamiento, y principalmente se enfoca en transportar la entrada y la salida entre el usuario y el servidor remoto" (Medina Rojas, 2012).

Durante los últimos años el uso de clientes ligeros en el país se ha convertido en una solución factible y económica. Desde el arribo a Cuba de la primera computadora sin disco duro, en la distribución cubana de GNU/Linux Nova se ha estado buscando una solución que permita la fácil administración e implementación de esta tecnología (Peña Escalona, 2013). La distribución cubana de GNU/Linux Nova cuenta con la plataforma de administración de clientes ligeros Nova-LTSP.

Nova-LTSP presenta módulos para la gestión de los clientes ligeros, imágenes de los sistemas operativos y perfiles de comportamiento de *hardware* y software, así como la administración del protocolo de configuración dinámica de host, en inglés *Dynamic Host Configuration Protocol* (DHCP) y del sistema. Constituye también una herramienta que permite automatizar la administración de los clientes ligeros, a través de la tecnología *Linux Terminal Server Project* (LTSP), la cual provee una manera simple de utilizar estaciones de trabajo de bajo costo tanto como terminales gráficas o bien como terminales de caracteres sobre un servidor GNU/Linux (Díaz Alonso, 2017).

Actualmente en este sistema no se registran las acciones ejecutadas sobre la aplicación. Estos caminos representan las trazas en un sistema informático y constituyen un mecanismo de registro de eventos y datos, donde cada uno debe proveer un conjunto de información que permita obtener una trazabilidad del evento ocurrido. Una traza debe proveer la información necesaria sobre qué, cuándo, cuál, quién y dónde ocurre un evento. (Porven Rubier, y otros, 2015). Esto constituye una desventaja que impide conocer el estado del sistema, lo que trae consigo que no se pueda identificar con posterioridad como recuperar o revertir algún cambio realizado sobre los datos o configuraciones y/o quién fue el responsable de algún cambio o problema en específico en un momento dado. Desde el punto de vista de la seguridad, es fundamental poder establecer la trazabilidad de un suceso determinado. Esto permite descubrir las causas de un incidente mediante el análisis forense o detectarlo en el momento de su ocurrencia.

Actualmente el departamento encargado de brindar servicios integrales en migración, asesoría y soporte SIMAYS del centro CESOL requiere de una aplicación web que permita de forma genérica el registro de eventos de la aplicación y evite los problemas antes mencionados sin tener que implementar una solución particular para cada uno de ellos.

INTRODUCCIÓN

Teniendo en cuenta la problemática anteriormente descrita, se identificó como **problema científico:** ¿Cómo garantizar el registro de los eventos que realizan los usuarios sobre la plataforma Nova-LTSP?

Para la realización de la investigación se define como **objeto de estudio**: el registro de eventos en aplicaciones web y el **campo de acción** se encuentra enmarcado en el registro de eventos en aplicaciones web para Nova-LTSP.

Para dar solución al problema planteado se define el siguiente **objetivo general**: desarrollar un módulo de registro de eventos que permita llevar el control en la plataforma Nova-LTSP.

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

1. Elaborar el marco teórico conceptual respecto a las herramientas de registro de eventos.
2. Diseñar un módulo de registro de eventos para Nova-LTSP que permita registrar los eventos realizados por los usuarios en la aplicación.
3. Implementar el módulo de registro de eventos para Nova-LTSP.
4. Evaluar el correcto funcionamiento del módulo.

Para establecer un orden de trabajo y guiar el ciclo de vida de la solución, se plantean las siguientes **preguntas científicas**:

1. ¿Cuáles son los presupuestos teóricos que sustentan la implementación de un módulo de registro de eventos para Nova-LTSP?
2. ¿Qué características tienen las herramientas informáticas existentes para la implementación del módulo de registro de eventos para Nova-LTSP?
3. ¿Qué aspectos deben tenerse en cuenta para realizar el análisis y diseño del módulo de registro de eventos para Nova-LTSP?
4. ¿Cómo implementar, a partir del análisis y diseño realizado, el módulo de registro de eventos para Nova-LTSP?
5. ¿Qué resultados se obtendrán al evaluar, a través de una estrategia de prueba de software, el módulo de registro de eventos para Nova-LTSP?

INTRODUCCIÓN

Con el propósito de dar cumplimiento a los objetivos se trazan las siguientes **tareas de la investigación:**

1. Análisis de las características de las herramientas existentes para el registro de eventos.
2. Selección de las herramientas, estándares, metodología y tecnologías que se necesitan para el desarrollo de la propuesta de solución.
3. Elaboración de los artefactos requeridos por la metodología de desarrollo seleccionada.
4. Implementación de las funcionalidades del módulo de registro de eventos para Nova-LTSP.
5. Validación, mediante una estrategia de pruebas de software, del módulo de registro de eventos para Nova-LTSP.

Para el desarrollo de las tareas científicas se han combinado diferentes métodos teóricos y empíricos de la investigación en la búsqueda y procesamiento de la información. Estos son:

Métodos teóricos:

- **Analítico-Sintético:** se emplea con el fin de analizar un conjunto de libros, artículos y documentos sobre las variantes existentes sobre el registro de eventos y lograr obtener de manera sintetizada el contenido necesario y suficiente para la ejecución de la investigación.
- **Inductivo-Deductivo:** se emplea para arribar a razonamientos que puedan ser aplicables al problema a resolver luego de adquirir una serie de elementos referentes al registro de eventos.

Métodos empíricos:

- **Modelación:** utilizado en la representación, mediante el uso de diagramas, de las características del sistema y relaciones entre objetos que intervienen en los procesos implementados por la propuesta de solución.
- **Entrevista:** se emplea para obtener los requisitos funcionales de la solución propuesta y para interactuar con personal familiarizado con la plataforma Nova-LTSP (Ver anexo 1).

Estructura de la investigación

El presente trabajo de diploma está estructurado de la siguiente manera: introducción, tres capítulos, conclusiones generales, recomendaciones y referencias bibliográficas empleadas durante el desarrollo de la investigación. A continuación, se realiza una breve descripción de cada uno de los capítulos.

INTRODUCCIÓN

Capítulo1. Marco teórico del módulo de registro de eventos para Nova-LTSP

En este capítulo se realiza un estudio de las bibliotecas para el registro de eventos. Mediante un análisis bibliográfico se determinan las tecnologías, herramientas, lenguajes de modelado y de programación a utilizar para implementar la solución propuesta.

Capítulo 2. Diseño del módulo de registro de evento para Nova-LTSP

A lo largo de este capítulo se especifican los requisitos, tanto funcionales como no funcionales del sistema para lograr que el módulo de registro de eventos se integre de forma satisfactoria a la plataforma Nova-LTSP. También se describen las Historias de usuario correspondientes a los requisitos funcionales, la arquitectura y los patrones de diseño que se utilizan durante la implementación, auxiliados por el modelado de diagramas.

Capítulo 3. Implementación y pruebas del módulo de registro de eventos para Nova-LTSP

En este capítulo se muestra el modelo de implementación como resultado del diseño anteriormente realizado. Se define el estándar de codificación que sirve de guía para la implementación de la solución propuesta, así como la estrategia de pruebas a utilizar y los resultados obtenidos de estas.

CAPÍTULO 1

CAPÍTULO 1 Marco teórico del módulo de registro de eventos para Nova-LTSP

En el marco teórico se hace una síntesis de los resultados alcanzados en la revisión bibliográfica relacionado con el tema. Se presentan organizadamente los conocimientos científicos acumulados hasta la fecha y los principales conceptos que trabajan la temática. También se describen lenguajes y herramientas que serán empleadas en el desarrollo de la solución.

Para lograr un mejor entendimiento del problema a investigar se tuvieron en cuenta conceptos importantes tales como: evento, campo de evento, productor de evento y registro de evento. Estos elementos guían el desarrollo del módulo a implementar. A continuación, se definen los mismos:

1.1 Conceptos fundamentales

1.1.1 Evento

"Un evento es una acción que es detectada por un programa; éste, a su vez, puede hacer uso del mismo o ignorarlo. Por lo general, una aplicación cuenta con uno o más hilos de ejecución dedicados a atender los distintos eventos que se le presenten. Entre las fuentes más comunes de eventos se encuentran las acciones del usuario con el teclado o el ratón. Cabe mencionar que cualquier programa tiene el poder de disparar sus propios eventos, como puede ser comunicar al sistema que ha completado una función en particular". De acuerdo con la definición anteriormente expuesta, la autora del presente trabajo considera que los eventos son sucesos que ocurren en los sistemas operativos y aplicaciones como resultado de la ejecución de un proceso interno o la interacción con otros sistemas (Pérez Porto, y otros, 2009).

1.1.2 Campo de evento

"Un campo de evento, describe una característica de un evento. Los ejemplos de un campo de evento incluyen fecha / hora, nombre de usuario, dirección IP[1] de origen, identificadores de dispositivo, nombres de función en código fuente e identificadores de puerta de edificio" (Fitgerald, y otros, 2010).

1.1.3 Productor de evento

[1] IP: Protocolo de internet por sus siglas en español. Se trata de un estándar que se emplea para el envío y recepción de información mediante una red que reúne paquetes conmutados.

“Es un dispositivo o componente de software que observa un evento y genera un registro de eventos que contiene los detalles sobre el mismo” (Fitgerald, y otros, 2010).

1.1.4 Registro de evento

“Un registro de evento es una colección de campos que en su conjunto describen un evento. Sinónimos de registro de eventos son registro de auditoría, traza y log[2]. Un registro de eventos, o simplemente un registro, es una colección ordenada de eventos. Términos como "registro de datos", "registro de actividad" y "archivo de registro" se utilizan a menudo para significar "registro de eventos". Los términos como "registro de auditoría" y "pista de auditoría" se utilizan para describir tipos específicos de registros de eventos que contienen conjuntos de eventos relevantes para la seguridad. Los registros de eventos pueden ser persistentes, como en un archivo almacenado en el disco o una copia impresa, o pueden ser efímeros, como en una secuencia de registros de eventos proporcionados a un suscriptor a través de una red” (Fitgerald, y otros, 2010).

Después de haber analizado los conceptos fundamentales para entender el tema de la investigación; se explica el estándar para los formatos de archivo de registro y el porqué de su uso, con el objetivo de especificar qué datos deben incluirse en el módulo.

1.2 Estándar para los formatos de archivo de registro

De acuerdo con Gasteiz (2012) define que, en el ámbito de las tecnologías, un estándar es una norma que regula la realización de ciertos procesos o la fabricación de componentes para garantizar la uniformidad de sus características, junto a su calidad. Concretamente su definición precisa es: “Una norma o estándar es una especificación técnica aprobada por un organismo reconocido de actividad normativa para aplicación repetida o continua, cuya observancia no es obligatoria”.

Los ficheros de registro contienen información de eventos internos ocurridos diariamente en los sistemas y aplicaciones, la información registrada es solo legible por programas capaces de interpretarla y mostrarla en un entorno amigable, por lo que es necesario estandarizar el formato de los archivos de registro para que puedan ser interpretados por una gran variedad de programas de análisis de registros. El Consorcio WWW, en inglés: *World Wide* Web *Consortium* (W3C), es un consorcio internacional que genera recomendaciones y estándares que aseguran el crecimiento de la *World Wide* Web (internet) a

2 Log: Registro por sus siglas en español. Es un archivo de texto en el que constan cronológicamente los acontecimientos que han ido afectando a un sistema informático.

largo plazo, el cual presenta un formato mejorado para los archivos de registro utilizado por los servidores web. Este patrón de registro tiene como objetivo establecer una línea base para el formato de los archivos de registro, a fin de garantizar su interpretación por la mayor parte de los sistemas de análisis de eventos. El formato es extendido, lo que permite capturar una mayor variedad de datos, es un formato de texto ASCII personalizable. Puede seleccionar qué campos incluir en el archivo de registro, lo que le permite mantener los archivos lo más pequeños posible (Hallam Baker, y otros, 2017).

El formato de archivo de registro extendido está diseñado para satisfacer las siguientes necesidades (Hallam Baker, y otros, 2017):

- Permitir el control de los datos grabados.
- Soporte de necesidades de los *proxy*[3], clientes y servidores en un formato común.
- Permitir que los archivos de registro personalizados se graben en un formato legible por herramientas de análisis genéricas.
- Permitir el intercambio de datos demográficos.

El formato de W3C registra los siguientes datos para cada solicitud HTTP[4] o FTP[5] basado en los campos que ha seleccionado: fecha y hora, en hora universal coordinada; dirección IP de origen, nombre de usuario, si se conoce; nombre del sitio de destino y nombre del equipo; número de puerto y dirección IP[6] de destino, tipo de solicitud, destino URL solicitado, consulta URL, códigos de estado y subestado HTTP, código de estado de Windows, número de bytes en el servidor de envío y la recepción, tiempo que tarda el proceso en milisegundos, versión del protocolo, nombre de host, agente de usuario, cookies y el

[3] Proxy: Es un servidor (un programa o sistema informático), que sirve de intermediario en las peticiones de recursos que realiza un cliente (A) a otro servidor (C).

[4] HTTP: Protocolo de trasferencia de hipertexto por sus siglas en español permite las transferencias de información en la World Wide Web (internet).

[5] FTP (Protocolo de Transferencia de Archivos): Es un protocolo de red para la transferencia de archivos entre sistemas conectados a una red TCP (Protocolo de Control de Transmisión), basada en la arquitectura cliente-servidor.

[6] Del inglés *Uniform Resource Locator* (Localizador De Recursos Uniforme).

referente (Hallam Baker, y otros, 2017).

Para el desarrollo de la solución, la autora considera que los campos principales a tener en cuenta del formato extendido para el registro de evento son: usuario, dirección IP del usuario, fecha, hora, acción, entidad e id del objeto.

A continuación, se realiza un estudio de la plataforma Nova-LTSP, sus principales funcionalidades, tecnologías y herramientas como base para el desarrollo del módulo.

1.3 Plataforma Nova-LTSP

Nova-LTSP es una plataforma web de administración centralizadas de todas las tecnologías utilizadas para el despliegue de los clientes ligeros. Enfocada a lograr una máxima compatibilidad entre un servidor LTSP desplegado en Nova Servidor 6.0 y estaciones de clientes ligeros con Nova Ligero 6.0. Esta plataforma está desarrollada en el lenguaje de programación Python y como marco de trabajo utiliza Django para una administración rápida, efectiva y segura de los sistemas GNU/Linux. Relativo a la seguridad, la plataforma presenta un baneador dinámico ante intentos de acceso fallido por los diversos protocolos (*fail2ban*[7]), cortafuegos con política *deny all*, para la administración remota utiliza el protocolo SSH (*Secure Shell*) que facilita las comunicaciones seguras entre dos sistemas usando una arquitectura cliente/servidor que permite a los usuarios conectarse a un *host* remotamente (Díaz Alonso, 2017).

Esta plataforma brinda un conjunto de funcionalidades como:

- Diagnóstico del sistema.
- Gestión de usuarios del sistema.
- Gestión de clientes ligeros.
- Gestión de perfiles personalizados para los clientes ligeros.
- Gestión de asignaciones dinámicas y estáticas.
- Creación y configuración de imágenes de sistemas operativos.
- Administración de los clientes ligeros.

7 Fail2ban: Es una aplicación para la prevención de intrusos en un sistema, actúa penalizando o bloqueando las conexiones remotas que intentan accesos por fuerza bruta.

1.4 Análisis de los sistemas que realizan el registro de eventos

1.4.1 Sistemas existentes a nivel internacional

Proxmox VE (por sus siglas en inglés *Proxmox virtual environment*) es un entorno de virtualización de servidor de código abierto. Está distribuido bajo la licencia GNU/Linux basado en Debian[8]. Proxmox VE incluye una consola web y herramientas de línea de comandos, y proporciona una API REST[9] para herramientas de terceros. Contiene un sitio web basado en la interfaz de administración, el cual presenta un archivo de registro para llevar el control de las acciones que se realizan en el sitio cuyos datos son: tiempo en cuanto a fecha y hora de los procesos ejecutados, el nodo creado, el servicio, ID del proceso, nombre de usuario, la gravedad y el mensaje generado o respuesta que da el sistema a las tareas de virtualización (proxmox, 2018).

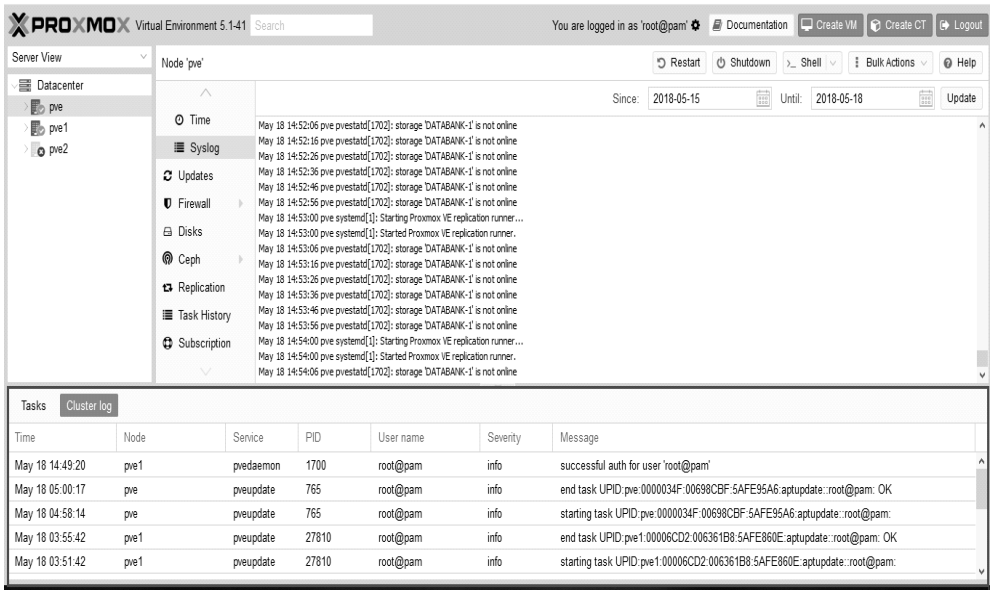


Figura 1: Herramienta proxmox

[8] Debian: Sistema operativo basado en GNU/Linux y software libre.

[9] API REST: Es un marco que facilita la creación de servicios HTTP disponibles para una amplia variedad de clientes, entre los que se incluyen exploradores y dispositivos móviles. Utiliza la arquitectura REST (*Representational State Transfer*) que se apoya totalmente en el estándar HTTP.

1.4.2 Sistemas existentes a nivel nacional

XAVIA HIS constituye un sistema integral para la gestión hospitalaria que tiene como atributo fundamental una historia clínica electrónica (HCE) única por paciente, que incluye toda la documentación, imágenes e información que se genere en torno al mismo. Dicho sistema presenta un componente de software que permite generar el registro de eventos del sistema (Orellana García, y otros, 2015).

Para formalizar la estructura de los registros de eventos a utilizar en la minería de proceso utiliza: XES[10], su propósito es proporcionar un formato generalmente reconocido para el intercambio de datos de registro de eventos entre las herramientas y los dominios de la aplicación. Su objetivo principal es la minería de procesos, es decir, el análisis de procesos operativos basados en sus registros de eventos. (Orellana García, y otros, 2015).

Este requiere como entradas, para generar el registro de eventos, un nombre de proceso, esta opción está definida para que el usuario identifique el proceso según desee, luego selecciona el proceso que necesita analizar, una vez seleccionado este, delimita el registro con una fecha inicial y final, seguidamente genera el registro con el formato requerido para su análisis en la herramienta para la minería de proceso *ProM*. La interfaz muestra también un listado con las ejecuciones previas, permitiendo su persistencia en el tiempo y la reutilización de los ficheros de formato XES (Orellana García, y otros, 2015).

[10] XES: *Extensible Event Stream* es un formato basado en XML para el intercambio de registros de eventos. El objetivo de este estándar es proporcionar un formato XML generalmente reconocido para el intercambio de datos de eventos entre sistemas de información en muchos dominios de aplicaciones, por un lado, y herramientas de análisis para tales datos, por otro.

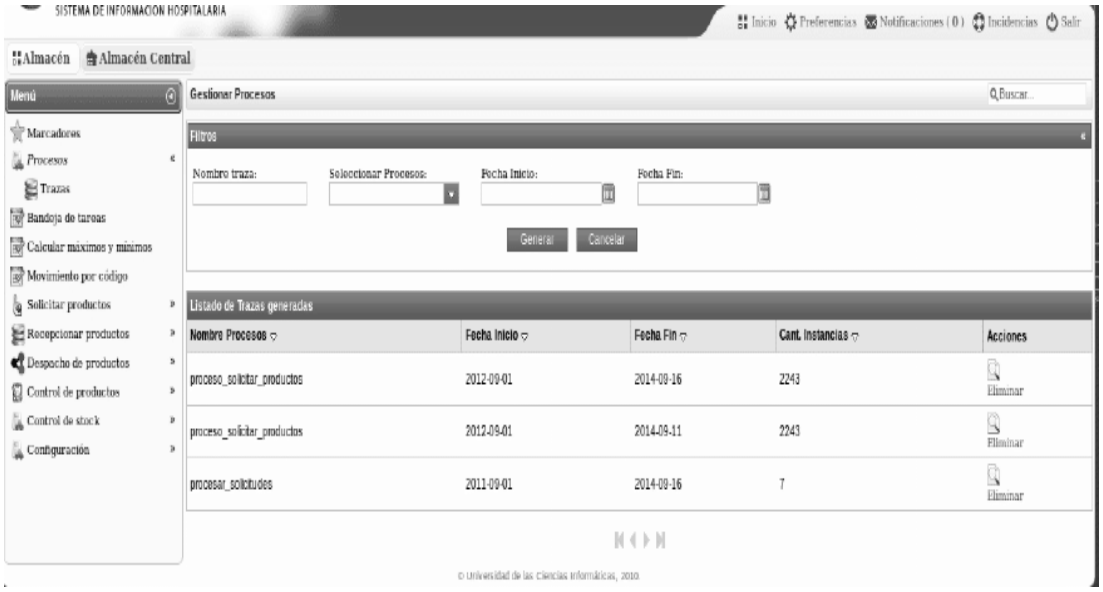


Figura 2 Herramienta XAVIA HIS para el registro de eventos

Sistema de Gestión Integral de Aduanas (GINA)

GINA presenta un componente de Autenticación, Autorización y Auditoría (AAA) que es capaz de gestionar usuarios, sus roles, así como los permisos correspondientes a cada rol. El sistema permite la autenticación: proceso de verificación de la identidad digital de un remitente de una comunicación que hace una petición para conectarse a un sistema, y la autorización: proceso por el cual se autoriza al usuario identificado a acceder a determinados recursos del sistema, se comprueba que los usuarios con identidad válida solo tengan acceso a aquellos recursos sobre los cuales tienen privilegios (Suárez Fabre, y otros, 2016).

Este sistema además registra eventos sobre las acciones ejecutadas en los sistemas administrados. El registro de evento guarda la operación realizada, usuario que realizó la operación, dirección IP desde donde se realizó la operación, fecha y hora, id del tipo de objeto del negocio, además de un texto con una breve descripción de la operación. Cuenta además con la facilidad de definir de manera sencilla las acciones que generan eventos. La consulta a los eventos generados, permite auditar objetos de negocio, usuarios, roles, direcciones ip, y cada uno de los elementos registrado en los eventos (Suárez Fabre, y otros, 2016).

Sistema de gestión de rentas de autos y limusinas (SIGREX)

SIGREX es un sistema para la gestión de autos y limusinas desarrollado en la Universidad de las Ciencias Informáticas por el centro de identificación y seguridad digital (CISED), que permite registrar las acciones ejecutadas por los usuarios. Está desarrollado sobre el lenguaje de programación Python y

el entorno de desarrollo PyCharm acompañado por el marco de trabajo Django, para el registro de los eventos utiliza la biblioteca *LogEntry,* el cual almacena dichos eventos en la base de datos y cada vez que se realice una acción sobre la plataforma se llama a dicho método. Los registros de eventos se llevan a cabo sobre las acciones que realizan los usuarios en los recursos, guardando la hora, fecha, usuario, modelo, acción realizada, el objeto y la ubicación de donde se realizó la acción, permite un filtrado y búsqueda de los registros, luego de creado el registro permite eliminarlo a través de un rango de fecha (Ver Anexo 2).

Autor: centro CISED
Entrevistado: Ing. Osay González Fuentes

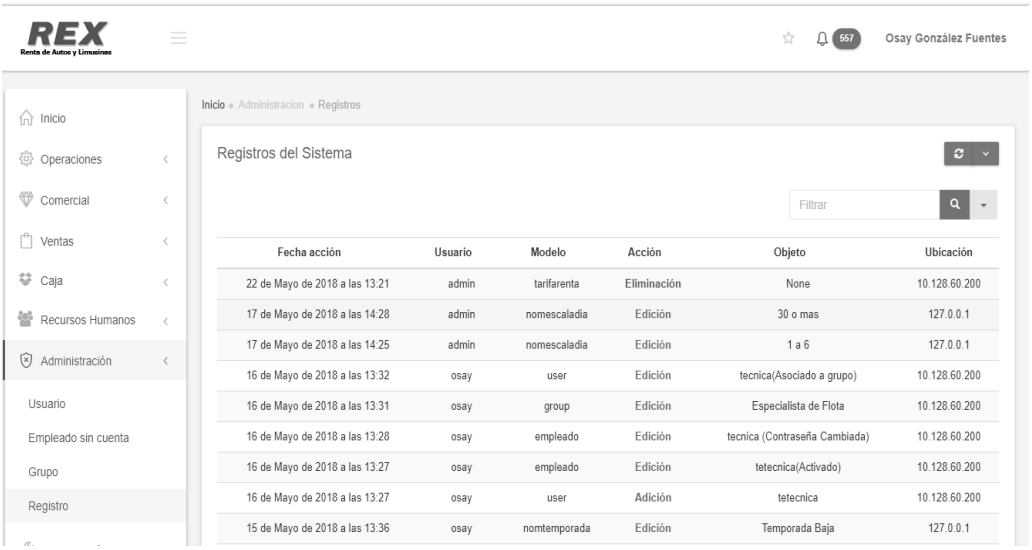


Figura 3. Herramienta SIGREX para el registro de eventos

Resultados del análisis de las herramientas para el registro de evento:

La investigación sobre la existencia de estos sistemas arrojó como resultado que la herramienta Proxmox no cumplen con los requerimientos del cliente ya que el registro de evento es específico del negocio de este sistema. El sistema XAVIA HIS para formalizar la estructura de los registros de eventos en la minería de proceso utiliza el formato XES, como resultado no es el utilizado para el desarrollo de la propuesta de solución. El sistema GINA estudiado ofrece un mecanismo de seguridad mediante los procesos de autenticación, autorización y auditoría estrechamente vinculados entre sí, en el que las funcionalidades de registro de eventos dependen de los procesos anteriores. La plataforma Nova-LTSP cuenta con un mecanismo de autenticación y autorización propia del negocio, por lo que no puede ser sustituido por un sistema externo y por consiguiente no se pueden aprovechar las funcionalidades del

sistema GINA. Por último, el sistema SIGREX no constituye una solución a la problemática planteada ya que la información que registra es propia del negocio. Sin embargo, aunque ninguno de estos sistemas constituye una solución integral al problema en cuestión, su estudio permitió obtener una visión más amplia de la solución y proporcionó funcionalidades de interés para enriquecer la solución propuesta, como por ejemplo el proceder utilizado en cuanto al modo en que se genera un filtrado de los eventos ocurridos y en la forma en que registran los eventos con la biblioteca *LogEntry*.

Lo mencionado anteriormente enfatizó la necesidad de desarrollar un módulo de registro de eventos para Nova-LTSP, que responda a las necesidades del cliente.

Después de haber realizado un estudio de las herramientas para el registro de eventos; se hace necesario la definición de las herramientas, tecnologías y metodología que se emplearán para el desarrollo del módulo de registro de eventos para Nova-LTSP.

1.5 Selección del entorno de desarrollo para la construcción de la solución

El entorno de desarrollo de un proyecto de desarrollo de software es el término que cubre todo lo que necesita el proyecto para desarrollar y desplegar el sistema, como las herramientas, directrices, procesos, plantillas e infraestructura.

1.5.1 Metodología de desarrollo de software

Según varios especialistas como Arias, Blanco y Bagarotti han referido que: “Desde el punto de vista informático una metodología de desarrollo de software es el conjunto de procedimientos, técnicas, herramientas y un soporte documental a la hora de desarrollar un producto de software, que indica quién, cuándo y cómo hacer algo. La presencia de una metodología en el desarrollo de un proyecto garantiza un producto con más calidad y reduce el tiempo de entrega del producto ya que al tener una mejor planificación la producción permite cumplir con la fecha establecida”.

Para guiar el proceso de desarrollo de software de la investigación se usará la variación de la metodología de Proceso Unificado Ágil (AUP por sus siglas en inglés), AUP-UCI por ser la empleada en el centro CESOL en el desarrollo de proyectos. Esta logra estandarizar el proceso de desarrollo de software en la UCI siendo el documento rector por el que se rige la institución (Rodríguez Sánchez, 2015).

Esta metodología es una versión simplificada del Proceso Racional Unificado (RUP) y describe de una manera sencilla y fácil de comprender las formas de desarrollar aplicaciones de software de negocio utilizando técnicas ágiles y conceptos que son válidos tanto para RUP como para AUP. La UCI le ha realizado modificaciones con el fin de adaptarlo al ciclo de vida de los proyectos que la institución ha

estado proponiendo; de las 4 fases que encierra la metodología AUP se simplificaron a (Ambler, 2014):

- Inicio: El objetivo de esta fase es llevar a cabo las actividades relacionadas con la planeación del proyecto. En ella se realiza un estudio inicial de la organización cliente y se obtiene información crucial acerca del alcance del proyecto, se realizan estimaciones de tiempo, esfuerzo, costo y finalmente se decide si se ejecuta o no el proyecto.
- Ejecución: En esta fase se unifican las actividades que desarrolla AUP de elaboración, construcción y transición en una sola. Se ejecutan las actividades requeridas para desarrollar el software, incluyendo el ajuste de los planes del proyecto considerando los requisitos y la arquitectura.
- Cierre: En esta fase se analizan tanto los resultados del proyecto como su ejecución y se realizan las actividades formales de cierre del proyecto.

Para el desarrollo de la solución, la autora considera que para el desarrollo del sistema se utiliza el escenario 4 para la disciplina requisitos y como forma de encapsular los requisitos las Historias de suario. Este escenario aplica a los proyectos que hayan evaluado el negocio a informatizar y como resultado obtengan un negocio muy bien definido.

La metodología AUP en la versión simplificada se ajusta al ambiente que presenta el negocio y da la posibilidad al cliente de siempre acompañar al equipo de desarrollo para convenir los requisitos y así poder implementarlos. Además, es una metodología que se adapta a cualquier proyecto productivo de la UCI.

1.5.2 Marco de trabajo

Es una estructura de soporte definida en la cual otro proyecto de software puede ser organizado y desarrollado donde suelen incluir bibliotecas, soporte de programas y unir diferentes componentes de un proyecto de desarrollo de programas (Alegsa, 2016).

Django

Es un marco de trabajo (*framework)* para el desarrollo web de código abierto. Posee la característica de que es fácil de instalar y puede ser usado tanto para *Windows* como para Linux. Es compatible con varios sistemas gestores de base de datos como son PostgreSQL, Mysql, Sqlite3 y Oracle. Este *framework* promueve el desarrollo rápido permitiendo crear aplicaciones muy rápidas y potentes, sigue el principio DRY (conocido también como una vez y solo una) utiliza esta filosofía para no crear bloques de código iguales y fomentar la reutilización del mismo. Usa una modificación de la arquitectura Modelo-

Vista-Controlador (MVC), llamada MTV (*Model-Template-View*), que sería Modelo-Plantilla-Vista, está forma de trabajar permite que sea práctico (Infante Montero, 2012).

Para el desarrollo del módulo de registro de eventos en Nova-LTSP se decidió utilizar este marco de trabajo en su versión 1.10, ya que es implementado sobre Python con el que se pueden crear y mantener aplicaciones de alta calidad.

1.5.3 Lenguajes

Un lenguaje de programación es un lenguaje diseñado para describir el conjunto de acciones consecutivas que un equipo debe ejecutar, es un modo práctico para que los seres humanos puedan dar instrucciones a un equipo. Se utilizan principalmente para controlar como se comporta una máquina o crear programas informáticos (Areatecnologia, 2016).

A continuación, se presentan los lenguajes que son utilizados para la realización del módulo.

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. Es un lenguaje ideal para el desarrollo rápido de aplicaciones en diversas áreas y sobre la mayoría de las plataformas de acuerdo a su elegante sintaxis y su tipado dinámico, junto con su naturaleza interpretada, (Van Rossum, 2009).

Ventajas del uso de Python (Martínez Estévez, y otros, 2014).

- **Portabilidad:** Es un lenguaje multiplataforma, puede usarse prácticamente de la misma manera en cualquier sistema operativo.
- **Fácil:** Es más fácil de aprender. Su sintaxis es muy elegante y permite la escritura de programas cuya lectura resulta más fácil que si se utilizaran otros lenguajes de programación.
- **Poder:** Posee extensiones escritas en el propio lenguaje de Python que permiten, entre otras cosas, el acceso a la base de datos, la edición de audio y video, interfaz gráfica de usuario y el desarrollo web.
- **Código abierto:** Python se puede utilizar libremente y está incluido en la gran mayoría de las distribuciones de Linux.

Se selecciona el lenguaje de programación Python, en su versión 2.7, debido a que es el utilizado por la plataforma de administración de clientes ligeros Nova-LTSP a la que se debe integrar la

propuesta de solución, por lo que su utilización contribuye a la homogeneidad y compatibilidad del código

Bootstrap

Bootstrap es un *framework* que tiene como objetivo facilitar el diseño web. Permite crear de forma sencilla webs de diseño adaptable, es decir, que se ajusten a cualquier dispositivo y tamaño de pantalla y siempre se vean igual de bien. Es de código abierto, por lo que se puede usar de forma gratuita y sin restricciones. El marco de trabajo contiene plantillas de diseño basadas en HTML y CSS con tipografías, formularios, botones, gráficos, barras de navegación y demás componentes de interfaz, así como extensiones opcionales de JavaScript[11] (Domínguez, 2016).

Para el desarrollo de la aplicación se utilizó Bootstrap en su versión 3.0, porque se integra con la biblioteca JQuery y está basado en herramientas actuales y potentes como CSS3 y HTML5.

JQuery

JQuery es una biblioteca o *framework* de JavaScript que permite simplificar la manera de interactuar con los documentos HTML, manejar eventos que generan los periféricos de entrada como el mouse o ratón y el teclado, desarrollar animaciones y agregar interacción con la tecnología *Asynchronous JavaScript and XML*(AJAX) (técnica de desarrollo web para crear aplicaciones interactivas) a páginas web. Al igual que otras bibliotecas, ofrece una serie de funcionalidades basadas en JavaScript que de otra manera requerirían de mucho más código. JQuery es soportado por la mayoría de los navegadores como Explorer, Chrome, Firefox permitiendo hacer más amplia sus funcionalidades (Álvarez, 2009).

Este *framework* JavaScript, ofrece una infraestructura con la cual se tiene mayor facilidad para la creación de aplicaciones complejas del lado del cliente. Con JQuery se obtiene ayuda en la creación de interfaces de usuario, efectos dinámicos, aplicaciones que hacen uso de Ajax (Álvarez, 2009).

Para el desarrollo de la aplicación se decidió utilizar JQuery en su versión 1.10.3 para las validaciones no funcionales en la parte del cliente, por ser un marco de trabajo que facilita la selección de elementos HTML, la creación de animaciones y evita la implementación de funcionalidades comunes.

Además se empleó la estrategia marcaria de los productos que son desarrollados en los diferentes centros de la Universidad de las Ciencias Informáticas, específicamente la línea de desarrollo Xilema

[11] Lenguaje de programación interpretado

que permite elaborar interfaces gráficas con JQuery y Bootstrap.

HTML

HTML, siglas en inglés de *HyperText Markup Language* (lenguaje de Marcado para Hipertextos), es el elemento de construcción más básico de una página web y se usa para crear y representar visualmente una página web. El lenguaje HTML es muy sencillo y fácil de aprender, este permite el enlace que conecta una página web con otra, ya sea dentro de una página o entre diferentes sitios web. HTML usa "*markup*" o marcado para anotar textos, imágenes, y otros contenidos que se muestran en el Navegador Web. El lenguaje de marcado HTML incluye elementos especiales tales como las etiquetas para el desarrollo del diseño gráfico de la página web (Barakat, 2017).

HTML 5 es la última versión de HTML y se utiliza para el desarrollo de este sistema, es una versión que contiene todos los elementos de sus versiones anteriores y mejora la compatibilidad entre los navegadores, dispositivos móviles y plataformas (Guiu, 2015).

CSS

CSS u hojas de estilo en cascada (en inglés *Cascading Style Sheets*) es un lenguaje usado para definir la presentación de un documento estructurado escrito en HTML. Es la mejor forma de separar los contenidos y su presentación y es imprescindible para crear páginas web complejas. Este lenguajes es usado para definir los estilos de los contenidos, es decir, el color, tamaño y tipo de letra de los párrafos de texto, la tabulación con la que se muestran los elementos de una lista y la separación entre titulares y párrafos para agregar belleza, diseño, orden a nuestras páginas web ya estructuradas con HTML y asi hacer más agradable la experiencia de uso de una página web (Navaja Ojeda, 2012). La versión de CSS que se utiliza para el desarrollo del módulo de registro de eventos para Nova-LTSP es CSS3.

1.5.4 Servidor web

Un servidor web es un programa utilizado para la distribución de contenido web en redes internas o en Internet. Como parte de una red de ordenadores, un servidor web transfiere documentos a los clientes. Así pues, un servidor web puede considerarse como un software que posibilita, a los clientes, acceder a las diferentes páginas web, interpretando las demandas, y respondiendo a éstas a través del protocolo de comunicación HTTP, el cual permite establecer la comunicación con el software del cliente, es decir, con el navegador web (Vilanova Blanco, 2017).

Posteriormente, se describe el servidor web y herramientas que serán utilizados para la implementación de la propuesta de solución.

Apache

Apache es uno de los servidores web más utilizados en la red, es un software de código abierto para la creación de páginas y servicios web. Es un servidor multiplataforma, gratuito, muy robusto y que destaca por su seguridad y rendimiento. Entre sus características se encuentra la capacidad de manejar más de un millón de visitas al día, la extensibilidad por módulos lo hace extremadamente flexible y fácil de usar, así como de configurar (Fumás Cases, 2015).

Para el desarrollo del sistema se utiliza el servidor web Apache en su versión 2.4, con el módulo mod_python 3.0. Dicho módulo permite que Apache pueda interpretar código Python.

1.5.5 Herramientas

Las herramientas son programas, aplicaciones o simplemente instrucciones usadas para efectuar tareas de otro modo más sencillo. En un sentido amplio del término, una herramienta es cualquier programa o instrucción que facilita una tarea. A continuación, se muestran todas las herramientas utilizadas en el desarrollo de la solución propuesta.

1.5.5.1 Entorno de desarrollo integrado

Un entorno de desarrollo integrado, o bien conocido como IDE (por sus siglas en inglés *Integrated Development Environment*), es un software compuesto por un conjunto de herramientas de programación, es decir, consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica (Quiroz, y otros, 2012).

PyCharm

PyCharm es un entorno de desarrollo integrado que ofrece las siguientes características (Pycharm, 2017)

- Compatibilidad con múltiples marcos de desarrollo web como Django y otros, lo que lo convierte en un competo IDE de desarrollo de aplicaciones rápidas.
- Soporte de bases de datos.
- Multiplataforma, utilizado para desarrollar en el lenguaje Python.
- Mantiene el código bajo control de chequeos, asistencia de pruebas, refactorizaciones y un conjunto de inspecciones que posibilitan codificar de forma limpia y sostenible.

Para llevar a cabo la implementación del componente se utiliza PyCharm2017 en su versión 2.2.

1.5.5.2 Herramientas CASE

Las herramientas CASE (*Computer Aided* Software *Engineering*, Ingeniería de Software Asistida por Computador) agilizan y facilitan la optimización de un producto software, ofreciendo apoyo permanente al grupo de desarrollo. Estas herramientas ayudan a los gestores y practicantes de la ingeniería del software en todas las actividades asociadas a los procesos de software. Gestionan todos los productos de los trabajos elaborados a través del proceso y ayudan a los ingenieros en el trabajo de análisis, diseño y codificación. Las herramientas CASE se pueden integrar dentro de un entorno sofisticado (Leyva, y otros, 2017).

Visual Paradigm

Visual Paradigm es una herramienta UML que permite modelar el ciclo de vida completo del desarrollo de un software. Ayuda a una rápida construcción de aplicaciones de calidad a un menor coste. Permite dibujar todos los tipos de diagramas de clases, generar código desde diagramas y documentación (Leyva, y otros, 2017).

Para el proceso de desarrollo del sistema se emplea Visual Paradigm en su versión 8.0, por ser una herramienta de software libre que permite realizar ingeniería tanto inversa como directa, además de brindar las siguientes ventajas (Leyva, y otros, 2017).

- Usa un lenguaje estándar común a todo el equipo de desarrollo y facilita la comunicación.
- Tiene modelos y códigos que permanecen sincronizados en todo el ciclo de desarrollo.
- Presenta disponibilidad de múltiples versiones, para cada necesidad y múltiples plataformas.

1.5.5.3 Lenguaje unificado de modelado

UML es el acrónimo de Lenguaje Unificado de Modelado es el lenguaje estándar especificado por el *Object Management Group* (OMG) para visualizar, especificar, construir y documentar los artefactos de un sistema, incluyendo su estructura y diseño. Brinda a desarrolladores de software las herramientas para el análisis, el diseño y la implementación de sistemas basados en software, así como para el modelado de procesos de negocios y similares. Permite el modelado de procesos de negocio y el modelado de requisitos apoyándose en el análisis orientado a objetos (Pinelo, 2009).

Para llevar a cabo el diseño del módulo se utiliza UML, en su versión 2.0, por las siguientes características (López, y otros, 2011):

- **Visualizar:** Permite expresar de una forma gráfica un sistema de forma tal que otro lo pueda entender.
- **Especificar:** Permite especificar cuáles son las características de un sistema antes de su construcción.
- **Construir:** A partir de los modelos especificados se pueden construir los sistemas diseñados.
- **Documentar:** Los propios elementos gráficos sirven como documentación del sistema desarrollado que pueden servir para su futura revisión.

1.5.5.4 Sistema gestor de bases de datos

Un sistema gestor de base de datos es el software que permite la creación, gestión y administración de bases de datos, así como la elección y manejo de las estructuras necesarias para el almacenamiento y búsqueda de la información del modo más eficiente posible (Iruela, 2016).

PostgreSQL

Es un sistema de gestión de bases de datos objeto-relacional, distribuido bajo licencia BSD (*Berkeley* Software *Distribution*) y con su código fuente disponible libremente. Este potente gestor soporta distintos tipos de datos, tipos fecha, monetarios, elementos gráficos y cadenas de *bits*. Permite la creación de tipos propios e incorpora una estructura de datos *array* (PostgreSQL, 2017).

Se selecciona el SGBD[12] PostgreSQL9.4 debido a que fue el utilizado en la plataforma de administración de clientes ligeros Nova-LTSP a la cual se va a integrar el presente módulo, por lo que su utilización contribuye a la homogeneidad de la información, además de las facilidades antes mencionadas.

1.5.5.5 Formato de intercambio de datos

JSON, acrónimo de *JavaScript Object Notation,* es un formato de texto ligero para el intercambio de datos. Es un subconjunto de la notación literal de objetos de *JavaScript,* aunque hoy en día*,* debido a su adopción como alternativa a XML[13], se considera un formato de lenguaje independiente. Una de las

[12] Sistema Gestor de Base de Datos

[13] XML: lenguaje que permite la organización y el etiquetado de documentos. XML no es un lenguaje en sí mismo, sino un sistema que permite definir lenguajes de acuerdo a las necesidades.

ventajas de *JSON* como formato de intercambio de datos es la sencillez para escribir un analizador sintáctico (*parser*) de *JSON*. En JavaScript, un texto *JSON* se puede analizar fácilmente usando la función *eval* (), lo cual ha sido fundamental para que *JSON* haya sido aceptado por parte de la comunidad de desarrolladores *AJAX*, debido a la ubicuidad de *JavaScript* en casi cualquier navegador web (Gutiérrez, 2009).

Después de haber realizado un estudio de las principales herramientas, tecnologías y metodología a utilizar en el desarrollo del módulo; como parte del marco teórico de la investigación se hace necesario realizar un análisis de la biblioteca para el registro de eventos a utilizar.

1.6 Biblioteca para el registro de eventos

En ciencias de la computación, una biblioteca (del inglés *library*) es un conjunto de subprogramas utilizados para desarrollar software. Las bibliotecas contienen código y datos, que proporcionan servicios a programas independientes, es decir, pasan a formar parte de estos. Esto permite que el código y los datos se compartan y puedan modificarse de forma modular (Alegsa, 2017).

Existen distintas bibliotecas para el registro de eventos. La investigación se enmarca en la biblioteca *LogEntry* desarrollada en el marco de trabajo Django la cual permite registrar los cambios realizados por los usuarios a la base de datos, eslabón principal para la implementación del módulo de registro de eventos para Nova-LTSP.

1.6.1 LogEntry

Es una biblioteca nativa de Django para el registro de eventos. *LogEntry* presenta una función de acceso directo para crear entradas de registro para los cambios realizados por los usuario a la base de datos. Esta presenta atributos como, la fecha y la hora de la acción realizada por el usuario, el contenido del objeto modificado, el usuario que realizó la acción, la representación textual de la clave principal del objeto modificado, la representación del objeto después de la modificación y el tipo de acción registrada, tales como adicionar, modificar y eliminar (django, 2017).

1.7 Conclusiones parciales del capítulo

- El análisis del formato de archivo de registro permitió seleccionar los campos personalizables para el registro de los eventos.
- El análisis de la biblioteca *LogEntry* posibilitó determinar las características que constituyen la base para el diseño de las funcionalidades que se definen en la propuesta de solución.

- Las herramientas existentes para el registro de evento, se identificó que son sistemas que cada uno responde a las necesidades particulares; aun así fue posible reconocer a partir de ellos, nuevos elementos que contribuyen al perfeccionamiento de la propuesta de solución.
- La propuesta de solución se realizará utilizando la metodología de desarrollo de software variación de AUP para la UCI, herramientas y tecnologías libres.

CAPÍTULO 2

CAPÍTULO 2 Diseño del módulo de registro de evento para Nova-LTSP

En el presente capítulo se aborda sobre el diseño del módulo de registro de eventos en Nova-LTSP. Se identifican sus características a través de los requisitos funcionales y no funcionales y se describen las Historias de usuario que especifican cada requisito funcional. Además, se incluye la arquitectura del software, los patrones de diseño y los diferentes artefactos de ingeniería de software correspondiente a las funcionalidades.

A continuación se expone la propuesta del módulo, donde se describen las principales funcionalidades.

2.1 Propuesta del módulo de registro de eventos para Nova-LTSP

Dada las necesidades planteadas en la situación problemática de la investigación, la propuesta de solución constituye un módulo de registro de eventos para Nova-LTSP basado en la arquitectura Modelo-Vista-Plantilla que propone el marco de desarrollo Django. Este módulo permite adicionar un registro de evento el cual cuenta con los siguientes campos: usuario, acción, ip, entidad, fecha, hora e id_objeto, esto sucede de forma automática cuando el usuario interactúa con la plataforma y realiza una acción de eliminar, adicionar o actualizar. Luego de realizada la adición se muestra un listado de todos los registros ocurridos en la plataforma Nova-LTSP con el objetivo que se pueda llevar un seguimiento y control de las acciones realizadas por los usuarios. También da la posibilidad de realizar una búsqueda y eliminar uno o más registro de eventos. Otra de las opciones que brinda el módulo es la creación de consultas basado en la selección de los campos antes descritos, los operadores tales como: Empieza por, Termina por, Contiene, No contiene, Mayor igual, Menor Igual, Igual, Mayor, Menor, Diferente y el valor que permite los valores generados por el registro de eventos, después de seleccionado estos campos dan la opción de ejecutar y salvar dicha consulta, al ejecutar se muestra en la vista la consulta creada y al salvar te guarda lo creado. Las consultas dinámicas permiten generar consultas de información que se encuentre en las diferentes tablas de la aplicación, de esta forma el usuario puede estructurar los datos de acuerdo a las necesidades en el momento deseado a la información actualizada. Luego el usuario puede eliminar una consulta creada y listar las consultas generadas hasta el momento.

Descripción de clases de la propuesta de solución

Usuario: persona que interactúa con una PC_cliente.

PC_cliente: computadora que se conecta a la plataforma Nova-LTSP vía url.

Plataforma Nova-LTSP: sistema que administra al servidor LTSP y registra los eventos.

Servidor LTSP: representa al servidor LTSP de la plataforma.

Evento: contiene las entidades y este se registra en la base de datos.

Base de Datos: entidad que almacena los eventos.

Entidad: objeto que representa a los clientes ligeros tales como, usuarios, clientes ligeros, ámbito dinámico, ámbito estático, notificación, imágenes de clientes ligeros, servidores de dominio, perfiles de clientes ligeros y repositorio.

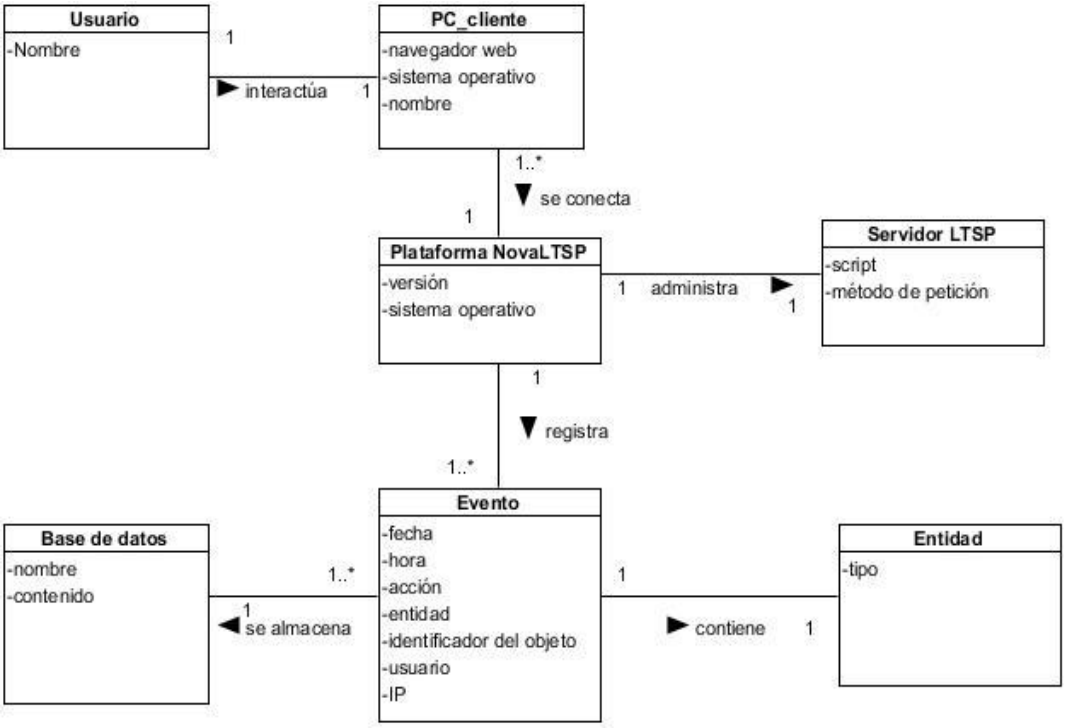


Figura 4: Representación de la propuesta de solución

Fuente: (Creación del propio autor)

2.2 Requisitos

Los requisitos para un sistema son las descripciones de lo que el sistema debería hacer, los servicios que proporciona y las limitaciones de su funcionamiento. Estos requisitos reflejan las necesidades de los clientes de un sistema que cumple un determinado propósito, como controlar un dispositivo, realizar un pedido o buscar información (Sommerville, 2011).

2.2.1 Fuentes para la obtención de requisitos

Las fuentes de obtención de requisitos utilizadas fueron:

- Propuesta de solución del módulo de registro de eventos para Nova-LTSP (Ver figura 1)

- Análisis de las herramientas existentes (Ver epígrafe 1.8)
- Especialistas de CESOL

2.2.2 Técnicas para la identificación de requisitos

Las técnicas de identificación de requisitos de software son mecanismos que se utilizan para recolectar la información necesaria en la obtención de los requisitos de una aplicación, permiten investigar aspectos generales para posteriormente ser especificados con un mayor detalle, requieren ser adecuadamente orientadas para cubrir la información que se requiere capturar (Pressman, 2010).

Entrevista

La entrevista es una técnica muy utilizada que permite recolectar opiniones, criterios o descripciones sobre las actividades. Se lleva a cabo mediante una conversación estructurada donde es fundamental la relación que se establezca durante el proceso (Pérez Escobar, 2010). Se realizó una entrevista al cliente para obtener la información acerca de las funcionalidades que debía cumplir el módulo, en el cual el cliente plantea que actualmente en la plataforma Nova-LTSP no se registran los eventos ocurridos en la aplicación de ninguna forma y que se pretende con ello garantizar la seguridad del sistema (Ver Anexo 1).

2.3 Especificación de requisitos de software

El propósito de la definición de requisitos es especificar las condiciones o capacidades con que el sistema informático debe contar y las restricciones bajo las cuales debe operar, logrando un acuerdo entre el equipo de desarrollo y el cliente y especificando las necesidades reales de forma que satisfaga sus expectativas (Jacobson, y otros, 2004).

2.3.1 Especificación de requisitos funcionales

Los requisitos funcionales definen las acciones que debe realizar el sistema, son capacidades o condiciones que debe cumplir, como debe comportarse en situaciones específicas. En algunos casos también pueden plantear explícitamente qué no debe hacer el sistema (Sommerville, 2011).

La siguiente tabla muestra el listado de los requisitos funcionales identificados para el desarrollo del módulo y la descripción de los mismos:

Tabla 1: Listado de requisitos funcionales.

Fuente: (Creación del propio autor)

No.	Nombre	Descripción	Complejidad
RF_1	Adicionar registro de evento	El sistema debe permitir adicionar un registro de evento teniendo en cuenta los parámetros: usuario, ip, entidad, fecha, hora, acción, id objeto.	Alta
RF_2	Eliminar registro de evento	El sistema debe permitir eliminar un registro de eventos.	Media
RF_3	Eliminar todos los registros de eventos	El sistema debe permitir eliminar todos los registros de eventos.	Media
RF_4	Listar registros de eventos	El sistema debe permitir mostrar los registros de eventos que han sido adicionados previamente.	Media
RF_5	Buscar registro de evento	El sistema debe permitir buscar uno o más registro de eventos.	Baja
RF_6	Crear consultas dinámicas	El sistema debe permitir crear consultas dinámicas a partir de filtros, teniendo en cuenta los campos: Usuario, IP, Entidad, Fecha, Hora, Acción, id Objeto.	Alta
RF_7	Eliminar consultas dinámicas	El sistema debe permitir eliminar una o más consultas dinámicas.	Media
RF_8	Listar consultas dinámicas	El sistema debe permitir mostrar las consultas dinámicas.	Media
RF_9	Ejecutar consultas dinámicas	El sistema debe permitir ejecutar las consultas dinámicas previamente creadas.	Alta

2.3.2 Especificación de requisitos no funcionales

Los requisitos no funcionales son aquellos requisitos que no describen información a guardar, ni funciones a realizar, sino que son propiedades que hacen al producto usable, rápido o confiable. Además, se conocen como un conjunto de características de calidad, que es necesario tener en cuenta al diseñar e implementar el software (Sommerville, 2011).

Restricciones del diseño e implementación

RnF1: Se utilizan el lenguaje de programación Python y el *framework* Django.

RnF2: La interfaz visual debe mantener el estilo y diseño de la plataforma de administración de clientes ligeros Nova-LTSP.

RnF3: El sistema cumple con los patrones de diseño, alta cohesión, bajo acoplamiento, experto, creador y controladora.

Usabilidad

RnF4: La aplicación debe adaptarse a las estrategias marcarias de la UCI.

Eficiencia

RnF5: El sistema debe permitir que los usuarios interactúen con él de manera concurrente.

Portabilidad

RnF6: El sistema se puede desplegar en un servidor que disponga de 4 GB de memoria RAM y 320 GB de disco duro o superior. Dependiendo de la carga de trabajo que genera cada cliente se necesitará más o menos RAM por lo que será recomendable tener una gran cantidad de memoria RAM en el servidor.

RnF7: Se puede acceder al sistema desde navegadores web que soporten HMTL5, CSS3 y JavaScript.

Mantenibilidad

RnF8: Se debe hacer uso de los estándares de codificación definidos para la plataforma Nova-LTSP.

RnF9: Se permitirá realizar modificaciones posteriores para adaptar mejoras al sistema.

Seguridad

RnF10: La seguridad de los datos se garantiza mediante conexiones seguras empleando el protocolo SSH.

2.4 Descripción de requisitos funcionales: Historias de usuario

La metodología AUP-UCI, en su escenario 4 para la disciplina requisitos, genera como uno de sus artefactos las Historias de usuario (HU). Estas consisten en una técnica para encapsular los requisitos del software, a través de un conjunto de tablas en las cuales el cliente describe brevemente las características que el sistema debe poseer. Cada historia de usuario es lo suficiente comprensible y delimitada para que los programadores puedan implementarla en un corto período de tiempo (Penádes, y otros, 2006).

Para el diseño de la propuesta de solución se generaron un total de 9 Historias de usuario, a continuación, se muestran tres de estas, correspondientes a los RF-1 y RF-6 y RF-9 respectivamente, detallados en el sub-epígrafe 2.3.1.

Tabla 2: Historia de usuario Adicionar registro de evento

Fuente: (Creación del propio autor)

Número: HU_1	**Nombre del requisito:** Adicionar registro de evento
Programador: Ana Isabel Marí Martín	**Iteración Asignada:** 1
Prioridad: Alta	**Tiempo Estimado:** 20 horas
Riesgo en Desarrollo: N/A	**Tiempo Real:** 15 horas
Descripción: El sistema permite almacenar los eventos cada vez que se realice una acción por parte de los usuarios teniendo en cuenta los siguientes campos: Usuario (Obligatorio): es un campo de texto, valores aceptados: *root*, administrador, *ltsp*. IP (Obligatorio): es un campo de texto, valores aceptados: x.x.x.x || x.x || x.x.x || x (x va entre 0 y 255). Entidad (Obligatorio): es un campo de texto, valores aceptados: usuarios, clientes ligeros, ámbito dinámico, ámbito estático, notificación, imágenes de clientes ligeros, servidores de dominio, perfiles de clientes ligeros, repositorio). Fecha (Obligatorio): es un campo de texto, valores aceptados: día/mes/año || día/mes || || mes/ año, día, mes, año). Hora (Obligatorio): es un campo de texto, valores aceptados: 00:00).	

Acción (Obligatorio): es un campo de texto, valores aceptados: Eliminar, Editar, Adicionar Id Objeto (Obligatorio): es un campo de texto, valores aceptados: valores contenidos en los campos.
Observaciones: El sistema crea por defecto los eventos una vez que los usuarios realicen una acción.

Tabla 3: Historia de usuario Crear consultas dinámicas.

Fuente: (Creación del propio autor)

Número: HU_6	**Nombre del requisito:** Crear consultas dinámicas
Programador: Ana Isabel Marí Martín	**Iteración Asignada:** 1
Prioridad: Alta	**Tiempo Estimado:** 20 horas
Riesgo en Desarrollo: N/A	**Tiempo Real:** 15 horas
Descripción: El sistema permite crear consultas dinámicas a partir de los campos seleccionados por el usuario y reutilizar consultas previamente creadas. La consulta consiste en una sentencia SQL con los distintos campos a seleccionar. A continuación, se describen dichos campos: Usuario (Obligatorio): es un campo de texto, valores aceptados: *root*, administrador, *ltsp*. IP (Obligatorio): es un campo de texto, valores aceptados: x.x.x.x || x.x || x.x.x || x (x va entre 0 y 255). Entidad (Obligatorio): es un campo de texto, valores aceptados: usuarios, clientes ligeros, ámbito dinámico, ámbito estático, notificación, imágenes de clientes ligeros, servidores de dominio, perfiles de clientes ligeros, repositorio). Fecha (Obligatorio): es un campo de texto, valores aceptados: día/mes/año || día/mes || mes/ año, día, mes, año). Hora (Obligatorio): es campo de texto, valores aceptados: 00:00). Acción (Obligatorio): es un campo de texto, valores aceptados: Eliminar, Editar, Adicionar. Id Objeto (Obligatorio): es un campo de texto, valores aceptados: caracteres alfanuméricos.	

Operador (Obligatorio): es un campo de texto, valores aceptados: Empieza por, Termina por, Contiene, No contiene, Mayor igual, Menor Igual, Igual, Mayor, Menor, Diferente. Valor (Obligatorio): es un campo de texto, valores aceptados: valores contenido en los campos.
Observaciones: Si la consulta fue creada correctamente se muestra un mensaje de éxito, si las consultas son iguales se muestra un mensaje de que la consulta ya está creada. Si ocurre un error mostrará un mensaje: “Error interno del servidor. Consulte las notificaciones”.
Prototipo: 

Tabla 4: Historia de usuario Ejecutar consultas dinámicas.

Fuente: (Creación del propio autor)

Número: HU_9	**Nombre del requisito:** Ejecutar consultas dinámicas
Programador: Ana Isabel Marí Martín	**Iteración Asignada:** 1
Prioridad: Alta	**Tiempo Estimado**: 15 horas
Riesgo en Desarrollo: N/A	**Tiempo Real:** 15 horas
Descripción: Se ejecuta un registro de evento a partir de los filtros seleccionados. A continuación, se describen dichos filtros:	

Campo (Obligatorio): es un campo de texto, valores aceptados: acción, entidad, usuario, dirección ip, id objeto, fecha, hora.

Operador (Obligatorio): es un campo de texto, valores aceptados: Empieza por, Termina por, Contiene, No contiene, Mayor igual, Menor Igual, Igual, Mayor, Menor, Diferente.

Valor (Obligatorio): es un campo de texto, valores aceptados: caracteres alfanuméricos.

Observaciones: Si la consulta fue ejecutada correctamente se muestra el registro de la misma.

Prototipo:

2.5 Análisis y diseño

En esta disciplina, los requisitos pueden ser refinados y estructurados para conseguir una comprensión más precisa de estos, y una descripción que sea fácil de mantener y ayude a la estructuración del sistema, incluyendo su arquitectura. Además, en esta disciplina se modela el sistema y su forma para que soporte todos los requisitos, incluyendo los requisitos no funcionales (Rodríguez Sánchez, 2015).

2.5.1 Diseño arquitectónico

La arquitectura que se utiliza para el desarrollo del módulo es la empleada por Django, el cual sigue una arquitectura Modelo-Vista-Controlador, solo que hace una adaptación de esta Modelo-Vista-Plantilla (a partir de ahora MTV por sus siglas en inglés, Model Template View). Por tanto, el sistema propuesto hereda una arquitectura MTV, debido a que se desarrolla sobre el marco de trabajo Django.

A partir de lo planteado se explica su funcionamiento el cual se observa en la Figura 4:

1. El navegador web envía una solicitud
2. La URLConf interpreta la solicitud y ubica la vista apropiada
3. La vista interactúa con el modelo para obtener datos
4. La vista llama a la platilla
5. La plantilla envia la respuesta a la solicitud del navegador

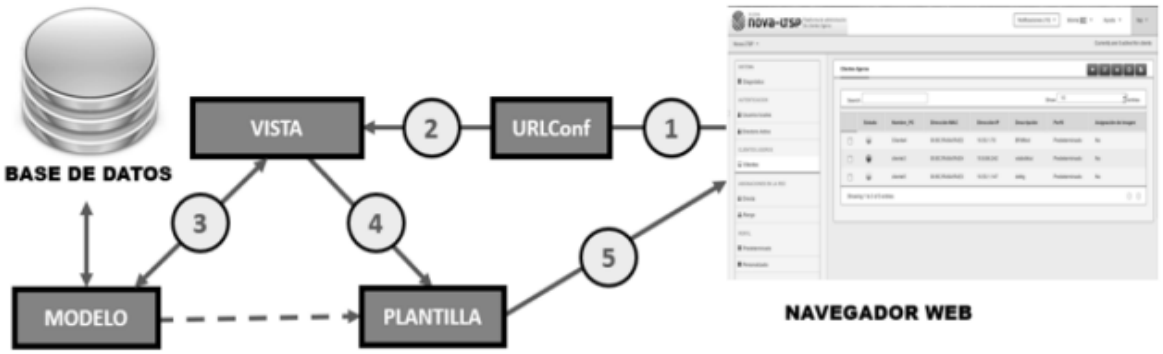


Figura 5: Funcionamiento del Modelo Vista Plantilla de Django.
Fuente: (Creación del propio autor)

Del funcionamiento del Modelo-Vista-Plantilla de Django se deriva lo siguiente (Infante Montero, 2012):

Modelo: Contiene toda la información sobre los datos. Cada una de las entidades de la base de datos se encuentra en el modelo en forma de clases de Python, y sus atributos se almacenan en variables con ciertos parámetros. También estos archivos poseen métodos, lo que permite indicar y controlar el comportamiento de los datos.

Vista: Es la capa de la lógica de negocios, contiene la lógica que accede al modelo y la delega a la plantilla apropiada. Esta capa sirve de “puente” entre el modelo y la plantilla, se presenta en forma de funciones de Python y su función principal es determinar qué datos serán visualizados en las plantillas.

Plantilla: Recibe los datos de la vista y luego los organiza para la presentación al navegador web. Básicamente es una página HTML (*HyperText Markup Language*) con algunas etiquetas extras que son propias del marco de trabajo Django.

La imagen de la figura 6 muestra la organización en carpetas de la propuesta de solución a partir de la

utilización del framework Django.

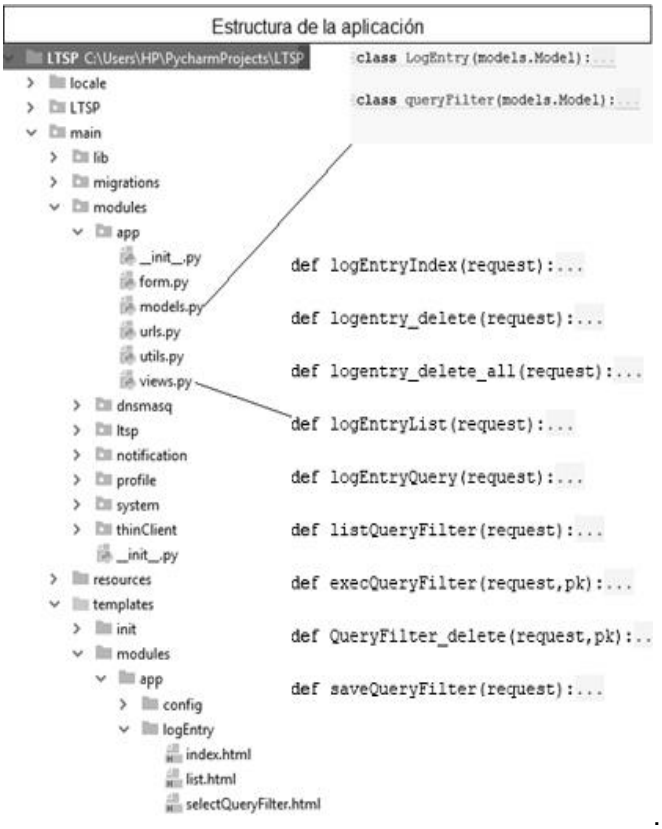


Figura 6: Estructura de la propuesta de solución.

Fuente: (Creación del propio autor)

Estructura de la aplicación

model.py: Contiene la clase *LogEntry* y *QueryFilter* que son las encargadas de brindar información acerca de los registro de eventos y las consultas.

view.py: Contiene todas la vistas encargadas de acceder a la clase del *models.py* y llamar a las plantillas contenidas en la carpeta templates.

templates: Es la carpeta que contiene todas las plantillas utilizadas en la aplicación y llamadas desde las vistas.

2.5.2 Modelo de datos

Un modelo de datos es un sistema formal y abstracto que permite describir los datos de acuerdo con reglas y convenios predefinidos o podríamos decir que es un conjunto de conceptos que permiten describir, a distintos niveles de abstracción, la estructura de una base de datos (Redondo González, 2017).

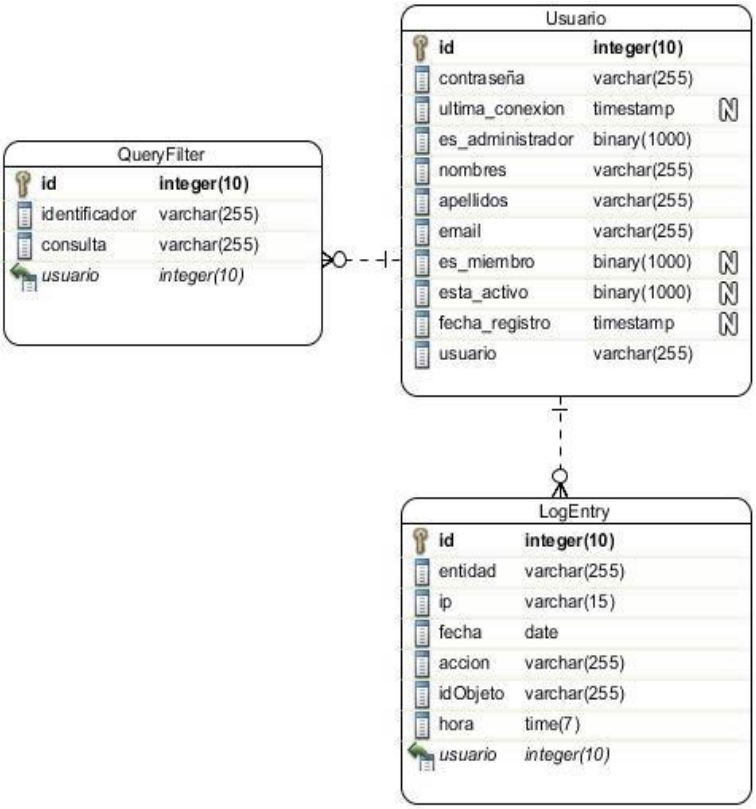


Figura 7: Modelo de datos del módulo de registro de eventos en Nova-LTSP.

Fuente: (Creación del propio autor)

2.5.3 Modelado del diseño

A continuación, se presenta el modelado del diseño de la propuesta de solución:

Diagrama de clases del diseño

Los diagramas de clase (DC) pueden usarse cuando se desarrolla un modelo de sistema orientado a objetos para mostrar las clases en un sistema y las asociaciones entre dichas clases (Sommerville, 2011).

Los diagramas de clases del diseño con estereotipos *web*, describen gráficamente las especificaciones del modelo, la vista y la plantilla de las Historias de usuario descritas en el sub-epígrafe 2.4. Estas representaciones contienen información acerca de las clases, asociaciones, atributos, métodos y dependencias.

Para el diseño de la propuesta de solución fueron generadas un total 9 diagramas de clases, a continuación, se muestran dos (2) de estos correspondientes a los RF-3 y RF-6 respectivamente, detallados en el sub-epígrafe 2.3.1.

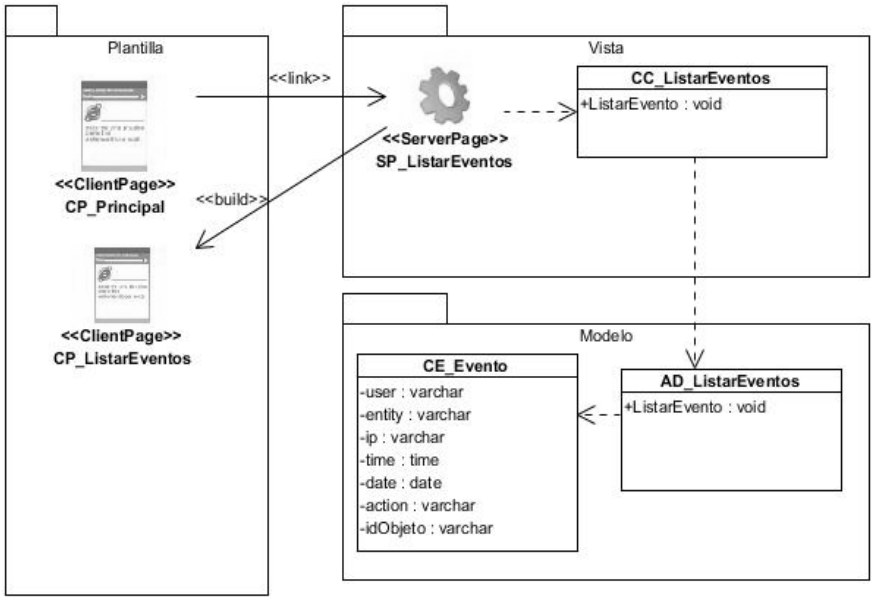


Figura 8: Diagrama de clases del diseño de la HU Listar registro de eventos.

Fuente: (Creación del propio autor)

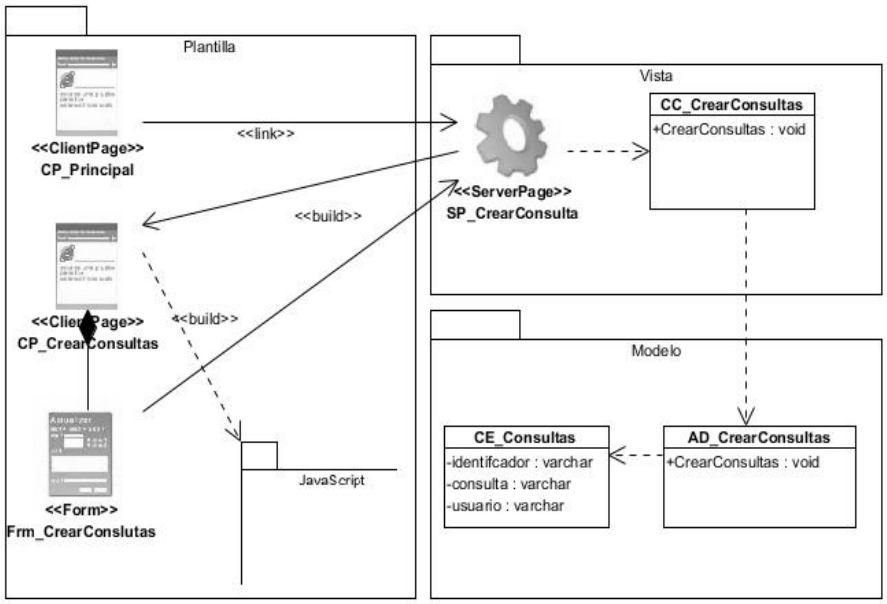


Figura 9: Diagrama del diseño de la HU Crear consultas dinámicas.

Fuente: (Creación del propio autor)

2.5.4 Patrones de diseño

Un patrón de diseño es una buena práctica documentada de la solución de un problema que ha sido aplicado satisfactoriamente en múltiples entornos. Es una solución recurrente a un problema común observado o descubierto durante el estudio o construcción de numerosas aplicaciones. Su principal objetivo es incrementar la calidad del software en términos de reusabilidad, mantenimiento y extensibilidad (Larman, 2004).

Los Patrones Generales de Software para Asignar Responsabilidades (GRASP del inglés General *Responsibility Assignment* Software *Patterns*) tienen una importante utilidad en el diseño de una aplicación, al igual que los *Gang-of-Four* o Pandilla de los Cuatro (*GoF* por sus siglas en inglés). A continuación, se muestra una selección de estos patrones los cuales son utilizados durante el diseño del módulo:

Experto

Este patrón tiene como objetivo principal asignar una responsabilidad determinada a la clase que tenga la mayor cantidad de información para hacer esta tarea. Es la razón anterior la que le da su apellido Experto “en información” (Larman, 2004). Este patrón se evidencia en la clase *LogEntry* y *QueryFilter* que son las que tienen acceso a todas las entidades necesarias y por ello a la información, por lo cual se le asigna la responsabilidad de generar todos los eventos y consultas que se requieren.

```
class LogEntry(models.Model):

class queryFilter(models.Model):
```

Figura 10: Clases expertas en información

Fuente: (Creación del propio autor)

Creador

Se asigna la responsabilidad a una clase de crear cuando contiene, agrega, compone, almacena o usa otra clase, lo que brinda una alta posibilidad de reutilizar la clase creadora (Larman, 2004).

Este patrón se pone de manifiesto en las vistas, donde se implementan las clases controladoras para crear objetos del modelo de datos, permitiendo acceder a estos de forma directa en cada una de las vistas para enviar la información necesaria hacia las plantillas, respondiendo a las peticiones del usuario a través de un navegador web.

```
def execQueryFilter(request,pk):
    query=queryFilter.objects.get(pk=pk)
    listLogEntry = LogEntry.objects.raw(query.query)
    return render(request, 'modules/app/logEntry/list.html', {"listLogEntry": listLogEntry})
```

Figura 11: Código fuente para ejecutar una consulta

Fuente: (Creación del propio autor)

Bajo acoplamiento

Es una medida de la fuerza con que una clase está conectada a otras clases, con que las conoce y con que recurre a ellas. Una clase con bajo (o débil) acoplamiento no depende de muchas otras clases. Este patrón ya viene incluido con Django que permite un bajo acoplamiento entre las piezas, lo que evita las dependencias, por ejemplo, a la hora de realizar cambios en las configuraciones de las *URL*, en la Base de Datos y las plantillas *HTML*, basta solo con realizarlo una sola vez (Cárdenas Escalante, 2014). Este patrón se evidencia en la clase *ActionLogEntry* la cual para cumplir con sus funciones solo necesita relacionarse con la clase *LogEntry*.

Alta Cohesión

Asigna responsabilidades de manera que una clase no tenga muchas funcionalidades no relacionadas o no realice un trabajo excesivo. Este patrón incrementa la claridad y facilita la comprensión del diseño. Una de las características de Django es la organización del trabajo en cuanto a la estructura del proyecto, lo cual permite crear y trabajar con clases con una alta cohesión (Cárdenas Escalante, 2014).

```
def new_d(request):
    clear_breadcrumb()
    add_breadcrumb(_("Listado de asignaciones dinamicas"), reverse('listDinamicScope'))
    add_breadcrumb(_("Adicionar asignacion dinamica"), request.path)
    if request.method == "POST":
        rangeFirst=request.POST['rangeFirst']
        rangeLast=request.POST['rangeLast']
        dinamicScope=DinamicScope(rangeFirst,rangeLast)
        utils.addDinamicScope(dinamicScope)
        notify.add(0,"El nuevo rango "+str(dinamicScope)+" se ha creado con exito.")
        logentry_register(request, ActionLogEntry.ADDITION, DinamicScope, rangeFirst)
        return redirect('listDinamicScope')

    else:
        return render(request, 'modules/dnsmasq/dinamicScope/new.html')


def logentry_register(request,action,entity,idObjeto):
    try:
        LogEntry(entity=entity.__name__, ip=get_client_ip(request), date=datetime.datetime.now().date(),
            time=datetime.datetime.now().time(), action=action, user=request.user,
            idObjeto=idObjeto).save()
    except Exception.e:
        notify.add(1,"Error al guardar el registro de evento: "+str(e))
```

Figura 12: Código fuente para adicionar asignación dinámica

Fuente: (Creación del propio autor)

Este patrón se evidencia en la propuesta de solución en la funcionalidad *new_d ()* para adicionar rangos dinámicos, dado que esta funcionalidad luego de crear el nuevo rango registra dicha acción, para ello hace uso de la funcionalidad *logentry_register ()* ubicada en la clase *Utils*.

Controladora

Asigna la responsabilidad de administrar un mensaje de evento del sistema a una clase que representa el sistema global, dispositivo, subsistema o representa un escenario de caso de uso en el que tiene lugar el evento del sistema. Este patrón es empleado en la funcionalidad *logentry_delete_all ()* que es el encargado de definir funcionalidades que controlan las vistas que se muestran en la aplicación (Bermúdez, y otros, 2013). La imagen a continuación muestra la vista que elimina todos los registros de eventos y muestra el mensaje “Se han eliminado todos los registros de evento con éxito”.

```
def logentry_delete_all(request):
    LogEntry.objects.all().delete()
    notify.add(0, _('Se han eliminado todos los registros de eventos con exito.'))
    return redirect('logEntryList')
```

Figura 13: Código fuente para eliminar todos los registros de eventos.

Fuente: (Creación del propio autor)

2.5.5 Diagrama de despliegue

El diagrama de despliegue se utiliza para mostrar la estructura física del sistema, incluyendo las relaciones entre el *hardware* y el software que se despliega. Las relaciones que existen entre nodos representan los protocolos de comunicación que se utilizan para acceder a cada uno. (SparxSystems, 2014).

Este diagrama se considera necesario para lograr un despliegue exitoso de la aplicación. En este se definen:

Figura 14: Diagrama de despliegue de la solución propuesta.

Fuente: (Creación del propio autor)

Dispositivo Cliente: Se refiere a las estaciones de trabajo (Cliente: PC, tableta, teléfono celular) que el usuario utilizará para conectarse, vía *HTTPS*, con el servidor de aplicaciones web.

Servidor de Bases de Datos: Almacena toda la información que brinda la Plataforma hospedada en el servidor ltsp. La comunicación con el servidor ltsp es a través del protocolo *TCP* empleando el puerto 5432.

Servidor LTSP: Es el encargado de brindar la interfaz de la plataforma para que los usuarios puedan hacer uso de esta, almacena todo el código fuente del sistema. Se comunica por medio del protocolo seguro TCP empleando el puerto 5432 con el servidor de bases de datos y por el protocolo seguro *SSH* empleando el puerto 443 con los clientes ligeros.

2.6 Conclusiones parciales del capítulo

Luego de haber realizado el análisis y diseño del módulo de registro de eventos para Nova-LTSP y haber generado los artefactos que dispone la metodología AUP, se puede concluir lo siguiente:

- La entrevista con el cliente permitió identificar los principales requisitos funcionales y no funcionales del módulo de registro de eventos en Nova-LTSP, los que fueron agrupados y categorizados en sus respectivas Historias de usuario.
- La identificación de los patrones de diseño y el estilo arquitectónico de la solución propuesta permite disminuir el impacto de los cambios futuros en el código fuente de la misma.
- El diseño de los diagramas de clase facilitó el enfoque en cuanto a composición lógica y física de la propuesta de solución.
- La generación de todos los artefactos requeridos por el modelo de desarrollo documentan la solución propuesta, lo cual facilita su posterior mantenimiento (actualización o adición de funcionalidades).
- La elaboración del diagrama de despliegue permitió identificar la disposición física de los artefactos del módulo de registro de eventos para Nova-LTSP.

CAPÍTULO 3

CAPÍTULO 3 Implementación y pruebas del módulo de registro de eventos para Nova-LTSP

En el presente capítulo se muestran los diferentes artefactos que se utilizan para la implementación y pruebas del sistema, así como los estándares de codificación que debe seguir el equipo de desarrollo para un mejor entendimiento y organización del código. De acuerdo a la metodología utilizada se especifican las pruebas que son realizadas al módulo, para darle validez a los requisitos funcionales y no funcionales.

3.1 Modelo de implementación

Según Pressman (2010) el modelo de implementación es comprendido por un conjunto de componentes y subsistemas que constituyen la composición física de la implementación del sistema. Entre los componentes se encuentran datos, archivos, ejecutables, código fuente y los directorios. Fundamentalmente, se describe la relación que existe desde los paquetes y clases del modelo de diseño a subsistemas y componentes físicos.

3.1.1 Diagrama de componente

El diagrama de componente muestra las relaciones estructurales entre los componentes de un sistema. Los componentes se consideran unidades autónomas encapsuladas dentro de un sistema o subsistema que proporcionan una o más interfaces. Los diagramas de componentes son generalmente dirigidos al personal de aplicación de un sistema, que presenta una comprensión temprana del sistema global que se está construyendo (Bell, 2004).

Descripción de los componentes del sistema

Plantillas: Contiene las interfaces con las que interactúa el usuario.

Vista: Archivo que agrupa a todos los componentes que interactúan con la clase modelo. Estos componentes permiten trabajar con algunas utilidades sobre los formularios y la renderización de la información en las plantillas apropiadas.

Url: Archivo que contiene todas las *url* (*Uniform Resource Locator*, Localizador Uniforme de Recursos).

Modelo: Archivo que contiene todas las clases del modelo y permite la interacción directa con el archivo Vista. Contiene las clases entidades que almacenan los registros de los eventos y las consultas dinámicas.

Util: Archivo que contiene la clase encargada de registrar los eventos.

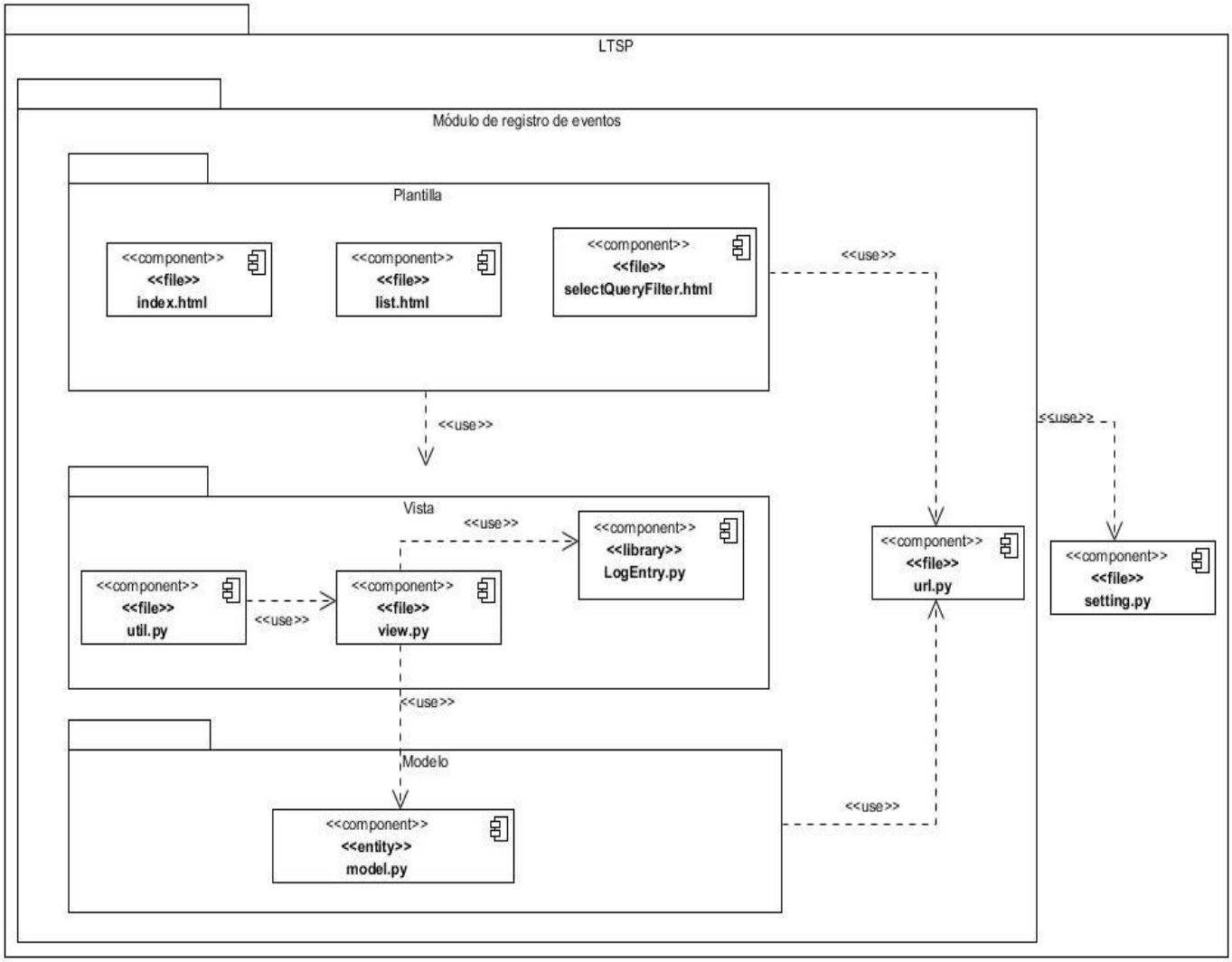


Figura 15: Diagrama de componente.

Fuente: (Creación del propio autor)

3.2 Estándares de codificación utilizados

Un estándar de codificación comprende todos los aspectos a tener en cuenta en la generación de código para que este sea legible y asegurarse que todos los programadores del proyecto trabajen de forma coordinada (Hernández, y otros, 2015). Usar técnicas de codificación sólidas y realizar buenas prácticas de programación con vistas a generar un código de alta calidad es de gran importancia para la calidad del software y para obtener un buen rendimiento.

Para la codificación de la propuesta de solución se utiliza el estándar de codificación para Python basado en la guía de estilo del código Python por Guido Van Rossum y Barry Warsaw. En este

documento se listan distintas convenciones utilizadas en el código Python comprendido en la librería estándar de la distribución principal de Python (recursospython.com, 2013).

Tabla 5: Estándar de codificación a utilizar en la implementación del sistema.

Tipo de estándar	Descripción
Organización de código	• Usa cuatro espacios por cada nivel en la codificación. • Las líneas de continuación deben alinearse verticalmente con el carácter que se ha utilizado (paréntesis, llaves, corchetes). • No se mezclan tabuladores y espacios en la codificación. El método para establecer un orden en el código en Python es con espacios. El segundo más popular es con tabulaciones, sin mezclar unos con otros. Cualquier código que utilice este método con una mezcla de espacios y tabulaciones debe ser convertido a espacios exclusivamente.
Máxima longitud de las líneas	• Todas las líneas deben estar limitadas a un máximo de 79 caracteres. • Dentro de paréntesis, corchetes o llaves se puede utilizar la continuación implícita para cortar las líneas largas. • En cualquier circunstancia se puede utilizar el carácter "\" para cortar líneas largas.
Líneas en blanco	• Separar funciones de alto nivel y definiciones de clase con dos líneas en blanco. • Definiciones de métodos dentro de una clase son separadas por una línea en blanco. • Se puede utilizar líneas en blanco en funciones, escasamente para indicar secciones lógicas.

Codificaciones	• Utilizar la codificación UTF-8. • Se pueden incluir caracteres que no correspondan a esta codificación utilizando “\x”, “\u” o “\U” para cadenas (*strings*).
Importación	• Las importaciones deben estar en líneas separadas. • Las importaciones siempre se colocan al comienzo del archivo, simplemente luego de cualquier comentario o documentación del módulo, y antes de globales y constantes. • Deben quedar agrupadas de la siguiente forma: 1. Importaciones de la librería estándar. 2. Importaciones terceras relacionadas. 3. Importaciones locales de la aplicación / librerías. • Cada grupo de importaciones debe de estar separado por una línea en blanco.
Espacios en blanco en expresiones y sentencias	• Evitar usar espacios en blanco en las siguientes situaciones: 1. Inmediatamente dentro de paréntesis, corchetes o llaves. 2. Inmediatamente antes de una coma, un punto y coma o dos puntos. 3. Inmediatamente antes de un corchete que empieza una indexación. 4. Más de un espacio alrededor de un operador de asignación u otro para alinearlo con otro. • Deben rodearse de espacios en blanco los siguientes operadores: 1. Asignación (“=”) 2. Asignación de aumentación (“+=”, “-=”, “*=”, “/=”) 3. Comparación (“==”, “<”, “>”, “>=”, “<=”, “! =”, “<>”, “*in”*, “*not in”*, “*is not”*)

	4. Expresiones lógicas • Si se utilizan operadores con prioridad diferente se aconseja rodear con espacios a los operadores de menor prioridad. • No utilizar espacios alrededor del igual (=) cuando es utilizado para indicar un argumento de una función o un parámetro con un valor por defecto.
Comentarios	• Los comentarios deben ser oraciones completas. • Si un comentario es una frase u oración su primera palabra debe comenzar con mayúscula a menos que sea un identificador que comience con minúscula. • Si un comentario es corto el punto final puede omitirse. • Los comentarios de una línea para aclaraciones del código aparecerán seguidos de los caracteres "//" en caso de código *JavaScript* mientras que en *Python* por el carácter "#" y deben ubicarse en la misma línea que se desea comentar. • Los comentarios de varias línea para organización del código aparecerán dentro de los caracteres "/** … **/" en caso de código *JavaScript*, mientras que en *Python* se hará con los caracteres "'…'"
Convecciones de nombramiento	• Nunca se deben utilizar como simple caracteres para nombres de variables los caracteres ele minúscula "l", o mayúscula "O", ele mayúscula "L" ya que en algunas fuentes son indistinguibles de los números uno (1) y cero (0). • Los módulos deben tener un nombre corto y en minúscula. • Los nombres de clases deben utilizar la convención "*CapWords*" (palabras que comienzan con mayúscula).

	• Los nombres de las excepciones deben estar escrito también en la convención “CapWords” utilizando el sufijo “Error”. • Los nombres de las funciones deben estar escritos en minúscula separando las palabras con un guión bajo (_). • Las constantes deben quedar escritas con letras mayúsculas separando las palabras con un guión bajo (_).

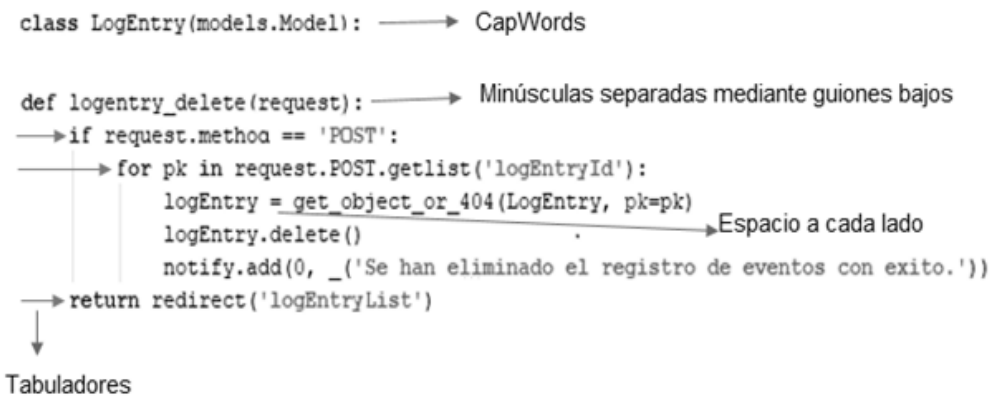


Figura 16: Aplicación de los estándares de codificación.

Fuente: (Creación del propio autor)

Después de realizar el modelado de implementación y definido los estándares de codificación utilizados para comprender todos los aspectos a tener en cuenta en la generación de código se realizan las pruebas de software y se define su estrategia para una mejor organización.

3.3 Pruebas de software

Las pruebas de software son un conjunto de actividades para proporcionar información objetiva e independiente sobre la calidad del producto. Se centra en los procesos lógicos internos del software asegurando que todas las sentencias se han comprobado, y en los procesos externos funcionales, es decir, la realización de las pruebas para la detección de errores. Además, son utilizadas para identificar posibles fallos de implementación, calidad o usabilidad de un programa (Pressman, 2010).

3.3.1 Estrategias de prueba

Según Pressman (2010) una estrategia de prueba proporciona una guía que describe los pasos que deben realizarse como parte de las pruebas, cuándo se planean y se llevan a cabo dichos pasos, y cuánto esfuerzo, tiempo y recursos se requerirán. Dicha estrategia debe ser lo suficientemente flexible para promover un enfoque personalizado, y al mismo tiempo lo adecuadamente rígido para originar una planeación razonable y un seguimiento administrativo del avance del producto.

En este epígrafe se muestran los resultados de la estrategia de prueba diseñada para la propuesta de solución (ver Tabla 6), en función de garantizar y validar la calidad del módulo.

Tabla 6: Estrategia de Pruebas.

Fuente: (Creación del propio autor)

Tipo de Prueba	**Método y técnica de prueba**	**Validación**
Integración	Tipo de integración ascendente • Ruta lógicas y partición de equivalencia • Caja Blanca y Caja Negra	Valida a partir de sus características, la capacidad del módulo de integrarse al sistema.
Funcional	• Partición de equivalencia • Caja Negra	Valida las funcionalidades diseñadas para el sistema.
Carga y estrés	Software *Apache JMeter*	Valida el comportamiento del sistema con distintos niveles de usuario concurrentes y el consumo excesivo de sus recursos.
Aceptación	• Partición de equivalencia • Caja Negra	Valida la aceptación del módulo por parte del cliente.

CAPÍTULO 3

A continuación, se describen los resultados de las pruebas definidas anteriormente del módulo de registro de eventos para Nova-LTSP.

3.4 Aplicación de las pruebas de software

3.4.1 Pruebas de integración

La prueba de integración es una técnica sistemática para construir la arquitectura del software mientras se llevan a cabo pruebas para descubrir errores asociados con la interfaz. El objetivo es tomar los componentes probados de manera independiente y construir una estructura de programa que determine el diseño (Pressman, 2010). Es una forma de verificar la correcta interrelación de los distintos componentes del sistema, en el caso de la solución desarrollada es la verificación de una correcta interoperabilidad entre el módulo desarrollado y Nova-LTSP.

Partiendo que el módulo debe integrarse a una plataforma base, se emplea una estrategia de integración ascendente, donde los componentes se integran de abajo hacia arriba. En esta prueba de integración ascendente se realiza la prueba de regresión que permite ejecutar nuevamente el mismo subconjunto de pruebas que ya se ha aplicado para asegurar que los cambios no han propagado efectos colaterales indeseables (Pressman, 2010).

Resultados de las pruebas de integración

Para ello se integró el módulo, que contiene el registro de eventos implementado a la plataforma Nova-LTSP, esta operación arrojó un total de dos (2) errores, a continuación, se muestran los resultados obtenidos en cada iteración de pruebas, así como la corrección de cada uno de los errores. En la primera iteración se encontró una (1) no conformidad que fue migrar la base de datos para PostgreSQL ya que Nova-LTSP la tiene implementada en Sqlite3[14]. La segunda iteración arrojó una (1) no conformidad donde se identificó una URL duplicada en diferentes módulos de la plataforma Nova-LTSP. Luego de solucionar esta no conformidad, el módulo de registro de eventos se integró correctamente con dicha plataforma.

[14] Sqlite: Sistema de gestión de bases de dato relacional.

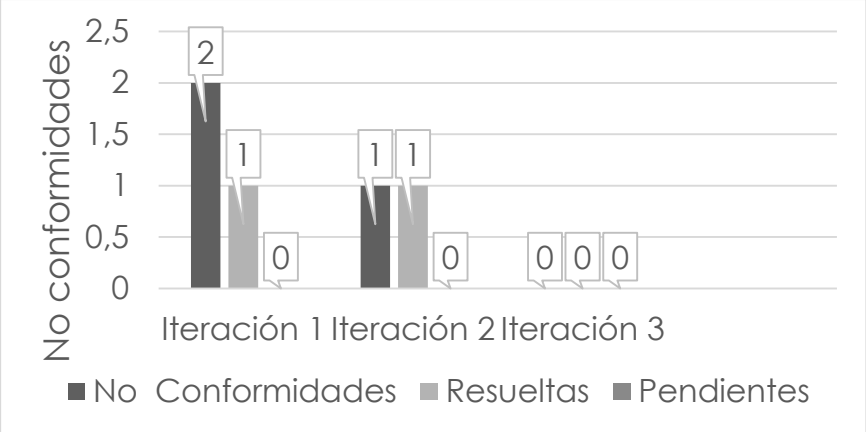


Figura 17: Resultado de la prueba de integración

Fuente: (Creación del propio autor)

3.4.2 Pruebas funcionales

Las pruebas funcionales son aquellas que se llevan a cabo sobre la interfaz del software sin prestar atención al código, por lo que los casos de prueba son creados con el objetivo de demostrar que la entrada es aceptada de forma adecuada y que se produce una salida correcta. El diseño de esta prueba se realiza con la intención de detectar funciones incorrectas o ausentes, errores en accesos a bases de datos externas, errores de interfaz, errores de rendimiento, y errores de inicialización y de terminación (Pressman, 2010).

Dentro de la prueba se incluyen la técnica de partición de equivalencia que será la empleada en la validación. La partición de equivalencia es una técnica de prueba de caja negra que divide el dominio de entrada de un programa en clases de datos a partir de las cuales pueden derivarse casos de prueba (Pressman, 2010).

Se realizaron los diseños de casos de prueba para validar la propuesta de solución implementada, con el fin de comprobar que la herramienta crea correctamente una consulta dinámica de los registros de eventos. A continuación, en la Tabla 7 se muestran las variables empleadas en el caso de prueba “Crear consultas dinámicas”.

Tabla 7: Variables empleadas en el caso de prueba "Crear consultas dinámicas"

Fuente: (Creación del propio autor)

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
1	**Campo**	Campo de selección	No	Contiene los campos de los eventos: usuario, ip, entidad, fecha, hora, acción e id del objeto.
2	**Operador**	Campo de selección	No	Contiene los operadores para la selección: Empieza por, Termina por, Contiene, No contiene, Mayor igual, Menor Igual, Igual, Mayor, Menor, Diferente.
3	**Valor**	Campo de texto	No	Contiene caracteres alfanuméricos.
4	**Identificador de la consulta**	Campo de texto	No	Contiene caracteres alfanuméricos.

Las celdas de la tabla contienen V, I, o N/A. V indica válido, I indica inválido, y N/A que no es necesario proporcionar un valor del dato en este caso, ya que es irrelevante.

Tabla 8: Caso de prueba para el escenario "Crear consultas dinámicas"

Fuente: (Creación del propio autor)

Escenario	Descripción	Variable 1	Variable 2	Variable 3	Variable 4	Respuesta del sistema	Flujo central
EC 1.1 Crear consultas dinámicas. Datos correctos	El sistema permite crear consultas dinámicas en caso de que no exista ningún dato incorrecto.	V usuario	V Igual	V ana	V Usuario ana	Muestra el mensaje “La consulta se registró con éxito”	1- El usuario selecciona la opción crear filtro. 2- El usuario rellena los campos de forma correcta. 3- El usuario da clic en el botón salvar.
		V	I	V	I		

EC 1.2 Crear consultas dinámicas. Datos incorrectos	El sistema permite verificar si existen datos incorrectos y no crea la consulta.	fecha	Vacío	2018-03-05	Vacío	Muestra el mensaje *"Existen campos obligatorios vacíos "*	1- El usuario selecciona la opción crear filtro. 2- El usuario no rellena los campos de forma correcta. 3- El usuario da clic en el botón salvar.

Resultado de las pruebas funcionales

Las pruebas funcionales se realizaron en tres iteraciones donde se aplicaron los casos de prueba diseñados. A continuación, se muestran los resultados obtenidos en cada iteración de pruebas al módulo de registro de eventos para Nova-LTSP, así como la corrección de cada uno de los errores.

Figura 18: Resultados de la prueba funcional

Fuente: (Creación del propio autor)

En la figura 18 se muestra el comportamiento de las no conformidades encontradas durante el proceso e prueba donde se observa que en la primera iteración se detectaron dieciséis (16) no conformidades, de ellas cuatro (4) errores de interfaz tales como vínculos defectuosos, la función "recargar página" corrompe la entrada de datos en el formulario, el formulario no establece salvaguardas adecuadas provocando que el usuario ingrese cadenas de texto más largas que cierto máximo predefinido y los guiones que realizan la comprobación de errores en los datos ingresados no funcionan de manera

adecuada. También se encontró cuatro (4) validaciones incorrectas, cuatro (4) errores ortográficos y cuatro (4) de idioma, en la segunda iteración se encontraron tres (3) no conformidades, de ellas una (1) validación incorrecta, dos (2) errores de interfaz, y en la tercera iteración no se encontraron no conformidades.

3.4.3 Pruebas de carga y estrés

Las pruebas de rendimiento o también pruebas de carga y estrés se diseñan para poner a prueba el rendimiento del software en tiempo de corrida, dentro del contexto de un sistema integrado. Esto ocurre a lo largo de todos los pasos del proceso de prueba. Incluso en el nivel de unidad, puede accederse al rendimiento de un módulo individual conforme se realizan las pruebas (Pressman, 2010).

Para la realización de las pruebas de carga y estrés se utiliza la herramienta *Apache JMeter* en su versión 2.12. *JMeter* es una aplicación de código abierto desarrollada en JAVA, diseñada para medir el rendimiento y el comportamiento de los sistemas ante las pruebas de sobrecarga. Puede ser utilizado para probar el rendimiento tanto de los recursos estáticos como dinámicos (archivos, bases de datos y consultas, servidores FTP y más). También se utiliza para simular una sobrecarga en un servidor, una red o un objeto, para poner a prueba su resistencia o para analizar el rendimiento global para diferentes tipos de carga. Puede usarse para hacer un análisis gráfico de rendimiento o para probar su servidor con sobrecargas concurrentes (Díaz, y otros, 2010).

Hadware:

- Microprocesador: Intel(R) Core (TM) i3-4030U CPU @1.90 GHz
- Memoria RAM: 4GB

Software:

- Sistema Operativo: GNU/Linux Nova 6.0 Arquitectura de 64 bit

A continuación, se describen las variables que miden el resultado de las pruebas de carga y estrés realizadas al módulo:

Muestra: Cantidad de peticiones realizadas para cada *URL*.
Media: Tiempo promedio en milisegundos en el que se obtienen los resultados.
Mediana: Tiempo en milisegundos en el que se obtuvo el resultado que ocupa la posición central.
Min: Tiempo mínimo que demora un hilo en acceder a una página.
Max: Tiempo máximo que demora un hilo en acceder a una página.

% Error: Por ciento de error de las páginas que no se llegaron a cargar de manera satisfactoria.
Rendimiento (Rend): El rendimiento se mide en cantidad de solicitudes por segundo.
Kb/s: Velocidad de carga de las páginas.

Como se muestra en la tabla 9 se simularon las peticiones realizadas al módulo por un total de 100, 500 y 10000 usuarios simultáneamente, realizando hasta 5 peticiones por segundo, obteniéndose los siguientes resultados:

Tabla 9: Resultados de las pruebas de carga y estrés.

Fuente: (Creación del propio autor)

Usuarios	Muestras	Media	Mediana	Min	Max	% Error	Rend	Kb/s
100	500	1632	1122	93	4715	0.00%	75.5	136.06
500	2500	1075	1089	159	3940	0.49%	210.4	536.3
10000	5000	1230	1032	69	1610	2.71%	357.7	974.2

Las pruebas realizadas muestran que el módulo es capaz de responder a 500 peticiones de 100 usuarios conectados simultáneamente en un tiempo promedio de 1632 milisegundos (1.6 segundos aproximadamente) con 0 % de error, esto evidencia que el módulo puede procesar la carga esperada para esta cantidad de usuarios.

Por otra parte, se realizaron 2500 peticiones iniciadas por 500 usuarios y en este caso el módulo respondió en 1075 milisegundos (1.1 segundos aproximadamente) como tiempo promedio. Esto demuestra que el módulo es capaz de procesar la carga esperada para este número de usuarios, aunque fallo en un 0.49% de las peticiones.

Por último, y con el objetivo de analizar el comportamiento del módulo en condiciones extremas, se realizó una prueba de estrés para un total de 1000 usuarios conectados simultáneamente. En este caso, el módulo responde a las 5000 peticiones en un tiempo promedio de 1230 milisegundos (1.2 segundos aproximadamente) siendo este un buen resultado de respuesta de las peticiones realizadas, pero falla el 2.71%, esto significa que no se satisfacen todos los requisitos de rendimiento para la cantidad de usuarios accediendo de forma concurrente a los servicios brindados, esto no constituye una inconveniente ya que este sistema no está determinado para 1000 usuarios.

3.4.4 Pruebas de aceptación

Esta es la etapa final en el proceso de pruebas, antes de que el sistema sea aceptado para uso operacional. El sistema se pone a prueba con datos suministrados por el cliente del sistema, en vez de datos de prueba simulados. Las pruebas de aceptación revelan los errores y las omisiones en la definición de requerimientos del sistema, ya que los datos reales ejercitan el sistema en diferentes formas a partir de los datos de prueba (Sommerville, 2011). A continuación, se muestran los casos de prueba realizados a algunas de las Historias de usuario.

Tabla 10: Caso de prueba de aceptación para la historia de usuario “Adicionar registro de eventos”

Fuente: (Creación del propio autor)

Caso de Prueba de Aceptación
Nombre de la historia de usuarios: Adicionar registro de eventos.
Nombre de la persona que realiza la prueba: Yasiel Pérez Villazón
Descripción de la prueba: El sistema permite adicionar un registro de eventos al realizar los usuarios determinadas acciones sobre las entidades de la plataforma.
Condiciones de ejecución: El usuario debe estar autenticado en la plataforma Nova-LTSP. Debe realizar una acción.
Entrada/Pasos de ejecución: 1. Acceder al módulo registro de eventos. 2. Realizar acción sobre entidades de la plataforma tales como: Editar, Eliminar, Adicionar.
Resultado esperado: El sistema adiciona un registro de los eventos realizados por los usuarios en donde se muestra los campos: acción, entidad, usuario, dirección ip, id del objeto, fecha y hora.
Evaluación de la prueba: Satisfactoria

Tabla 11: Caso de prueba de aceptación para la historia de usuario “Crear consultas dinámicas”

Fuente: (Creación del propio autor)

Caso de Prueba de Aceptación
Nombre de la historia de usuarios: Crear consultas dinámicas
Nombre de la persona que realiza la prueba: Yasiel Pérez Villazón
Descripción de la prueba: El sistema permite crear consultas dinámicas a partir de filtros, teniendo en cuenta los campos: Usuario, IP, Entidad, Fecha, Hora, Acción, id Objeto.
Condiciones de ejecución: El usuario debe estar autenticado en la plataforma Nova-LTSP. Debe seleccionar cada filtro antes de realizar la consulta.
Entrada/Pasos de ejecución: 1. Acceder al módulo registro de eventos. 2. Seleccionar la opción Crear filtros. 3. Seleccionar los campos. 4. Salvar los cambios.
Resultado esperado: El sistema crea una consulta y luego la guarda.
Evaluación de la prueba: Satisfactoria

Resultados de las pruebas de aceptación:

Al finalizar estas pruebas el cliente fue capaz de verificar el cumplimiento del objetivo planteado. Los resultados obtenidos para cada uno de los casos de prueba efectuados fueron satisfactorios como se aprecia en el Anexo 3.

3.5 Evaluación del objetivo de la investigación

Con la creación de la propuesta de solución se realizan encuestas a los usuarios con el objetivo de tener una mayor retroalimentación acerca de la opinión de estos. Esta información es necesaria para conocer

las debilidades de la misma y profundizar en sus fortalezas. De acuerdo a esto, la técnica de Iadov es un instrumento que ayuda a conocer el grado de satisfacción de los potenciales usuarios (Silega, 2014).

Esta técnica de criterio de usuario debe usarse como vía para valorar resultados en aquellos casos en que los evaluadores son usuarios de lo que se propone, es decir que además de tener dominio del problema en estudio, están contextualizados, inmersos en el contexto en el que se aplica el resultado (Fernández de Castro Fabre, y otros, 2014).

Este método calcula el Índice de satisfacción grupal (ISG) el cual se implementa mediante un cuestionario (Ver anexo 4) en donde se le incluyen tres preguntas cerradas que se intercalan dentro de un cuestionario de cinco preguntas y cuya relación el encuestado desconoce (Montielo Torrado, y otros, 2016).

Estas tres preguntas se relacionan a través del "Cuadro Lógico de Iadov" el cual permite ubicar a cada encuestado, según el cuadro lógico en una escala de satisfacción, para luego calcular el ISG. La escala de satisfacción la cual toma valores de 1 a 6 es la siguiente 1 -Clara satisfacción, 2-Más satisfecho que insatisfecho, 3-No definida, 4-Más insatisfecho que satisfecho, 5-Clara insatisfacción y 6-Contradictoria (Montielo Torrado, y otros, 2016).

Tabla 12: Cuadro lógico de Iadov

Fuente: (Creación del propio autor)

4- Luego de haber mostrado los resultados de la solución refleje en qué medida le gusta la solución desarrollada.	2. ¿Considera usted correcta la forma en que se registran los eventos generados por la plataforma Nova-LTSP?								
	No			**No sé**			**Sí**		
	3. ¿Considera usted factible la implementación de un módulo que permita conocer los eventos de la plataforma Nova-LTSP?								
	Sí	**No sé**	**No**	**Sí**	**No sé**	**No**	**Sí**	**No sé**	**No**
Me gusta mucho	1	2	6	2	6	6	6	6	6
Me gusta más de lo que me disgusta	2	2	3	2	3	3	6	3	3
Me da lo mismo	3	3	3	3	3	3	3	3	3
Me disgusta más de lo que me gusta	6	3	6	3	4	4	3	3	4
No me gusta nada	6	6	6	6	4	4	6	4	5

No sé decir	2	3	6	3	3	3	6	3	4

El número resultante de la interrelación de las tres preguntas indica la posición en la escala de satisfacción siguiente: clara satisfacción (A), más satisfecho que insatisfecho (B), no definida (C), más insatisfecho que satisfecho(D), clara insatisfacción (E) y contradictoria (C).

A partir de la cantidad de respuestas por categoría es posible calcular el Índice de Satisfacción Grupal (ISG) siguiendo la siguiente fórmula:

ISG = A (+1) + B (+0.5) + C (0) + D (-0.5) + E (-1) / N, donde N es la cantidad total de respuestas

El valor del ISG permite identificar las siguientes categorías grupales:

- Máxima insatisfacción: desde -1 hasta -0,49
- Más insatisfecho que satisfecho: desde -0,5 hasta -0,1
- No definido y contradictorio: 0
- Más satisfecho que insatisfecho: desde 0,1 hasta 0,49
- Máximo de satisfacción: desde 0,5 hasta 1

El índice general arroja valores entre + 1 y - 1. Los valores que se encuentran comprendidos entre -1 y -0,5 indican insatisfacción; los comprendidos entre - 0,49 y + 0,49 evidencian contradicción y los que están entre 0,5 y 1 indican que existe satisfacción.

Resultados obtenidos

Los resultados obtenidos de la aplicación de la encuesta se presentan en la Tabla 14.

Tabla 13: Resultados de la escala de satisfacción

Fuente: (Creación del propio autor)

Categorías grupales de satisfacción	N = 6	Escala
Clara satisfacción	5	A
Más satisfecho que insatisfecho	1	B
No definida	0	C
Más insatisfecho que satisfecho	0	D

Clara insatisfacción	0	E
Contradictoria	0	C

2. Cálculo del índice de satisfacción grupal

ISG = A (+1) + B (+0.5) + C (0) + D (-0.5) + E (-1) / N

ISG = (5(+1) + 1(+0.5)) / 6 = 0.91

3. Interpretación del resultado de ISG

El valor obtenido del ISG fue 0.91 lo que indica máxima satisfacción de los usuarios con respecto a la módulo de registro de eventos para Nova-LTSP. Se puede afirmar que se cumplió el objetivo de la investigación. Las respuestas a las preguntas abiertas brindadas por los encuestados reafirman los beneficios que traerá la utilización del sistema propuesto.

3.6 Interfaz principal

Una vez finalizado el desarrollado del software es posible visualizar la pantalla principal del módulo de registro de eventos para Nova-LTSP, donde se observa el resultado obtenido durante la implementación de las Historias de usuario descritas en el capítulo anterior.

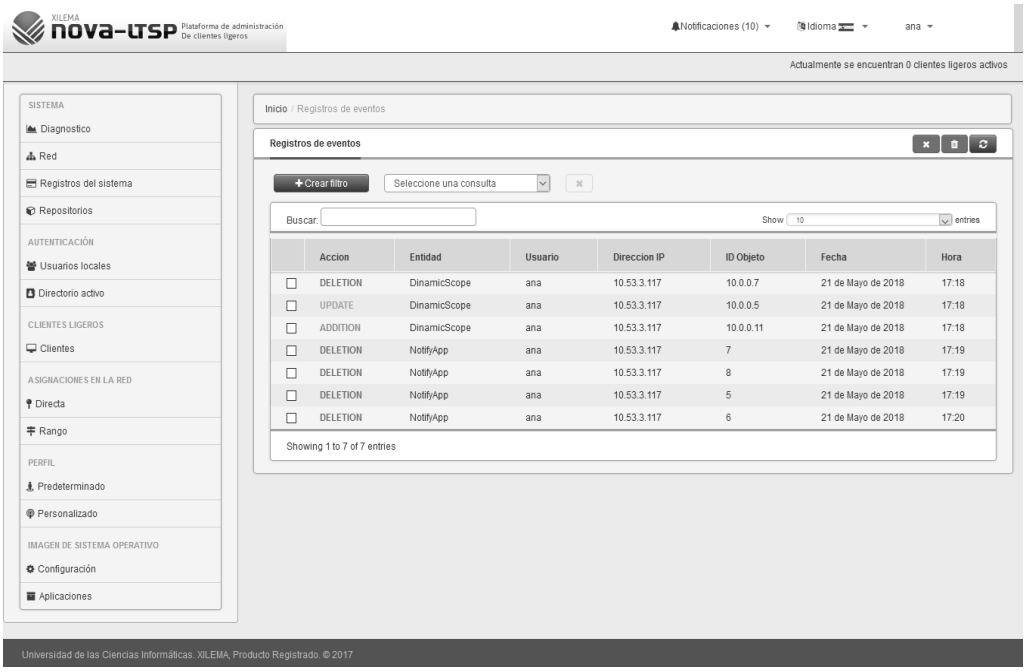


Figura 19: Interfaz Principal del módulo de registro de eventos para Nova-LTSP

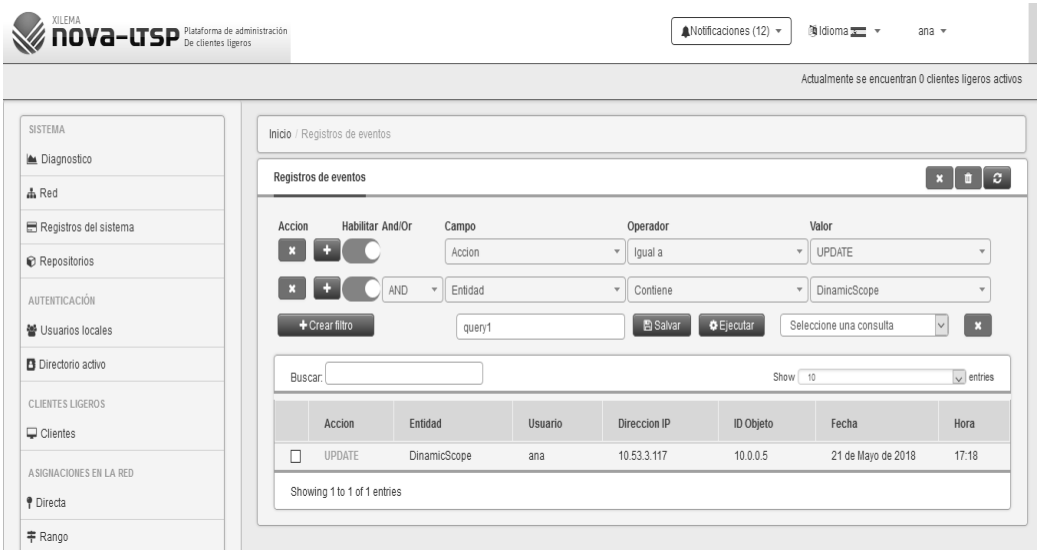


Figura 20: Interfaz Principal del requisito Crear consultas dinámicas

3.7 Conclusiones parciales del capítulo

En este capítulo se abordaron los elementos de la implementación del módulo de registro de eventos para Nova-LTSP, así como las pruebas realizadas al mismo y los resultados obtenidos; lo que permite concluir:

- La elaboración del diagrama de componentes permitió una mejor comprensión de la estructura de los componentes de la propuesta de solución.
- El empleo de un estándar de codificación permitió garantizar la alta calidad, minimización errores y mayor limpieza en el código fuente. Lo que logra así que pueda ser mantenido fácilmente y reutilizado por otros desarrolladores que lo necesiten.
- La implementación de la propuesta de solución facilitó la obtención de una aplicación funcional lista para su uso.
- La aplicación de las pruebas de integración, funcionales, de carga y estrés y aceptación permitieron detectar errores que afectaban el funcionamiento del sistema, lo que posibilitó corregirlos a tiempo para que el mismo cumpliese totalmente con los requisitos funcionales definidos en la etapa de análisis.
- La aplicación de la técnica de Iadov propició la evaluación satisfactoria del módulo del registro de eventos para Nova-LTSP.

CONCLUSIONES GENERALES

CONCLUSIONES GENERALES

Una vez completada la investigación esta cumple con los objetivos planteados mediante el desarrollo del módulo de registro de eventos para Nova-LTSP por lo que se concluye que:

- El análisis de los referentes teóricos y de los sistemas informáticos estudiados evidenció la necesidad de desarrollar un módulo de registro de eventos para Nova-LTSP.
- La selección de herramientas, lenguajes y tecnologías permitió la implementación del módulo de registro de eventos para Nova-LTSP.
- La utilización de la estrategia de pruebas garantizó la identificación temprana de las deficiencias en el módulo desarrollado, corrigiéndose las misma logrando un producto más seguro y funcional.
- La aplicación de la técnica de Iadov propició la evaluación satisfactoria del módulo del registro de eventos para Nova-LTSP.

RECOMENDACIONES

RECOMENDACIONES

Para contribuir a la continuidad de la investigación, se hace la siguiente recomendación:

- Realizar un estudio acerca de la detección de cambios no autorizados a la base de datos y si es posible incluir nuevas funciones que permitan realizar controles automatizados en tiempo real a través de una solución de seguridad de base de datos.

REFERENCIAS

Alegsa, Leandro. 2017. ¿Qué significa biblioteca? *Alegsa.com.ar.* [En línea] 2017. [Citado el: 14 de marzo de 2018.] http://www.alegsa.com.ar/Dic/biblioteca.php.

Alegsa, Leandro. 2016. ¿Qué significa Framework? *Alegsa.com.ar.* [En línea] 2016. [Citado el: 7 de noviembre de 2017.] http://www.alegsa.com.ar/Dic/framework.php.

Álvarez, Miguel Angel. 2009. Manual de JQuery. [En línea] 2009. [Citado el: 8 de noviembre de 2017.] http://www.cav.jovenclub.cu/comunidad/datos/descargas/jQuery.pdf.

Ambler, Scott. 2014. The Agile Unified Process (AUP). *Ambysoft .* [En línea] 2014. [Citado el: 6 de noviembre de 2017.] http://www.ambysoft.com/unifiedprocess/agileUP.html.

Areatecnologia. 2016. Areatecnologia.com. *Tecnologias. Lenguajes de programación.* [En línea] 2016. [Citado el: 8 de noviembre de 2017.] http://www.areatecnologia.com/informatica/lenguajes-de-programacion.html.

Arias Guerra, Y, Blanco Salinas, R y Bagarotti Acebo, Y. 2013. Metodología para el desarrollo e implantación de sistemas de información geográfica. *Serie Científica de la Universidad de las Ciencias Informáticas.* 2013. Vol. 6, no.7. ISSN: 2306-2495.

Barakat, Jose. 2017. HTML. [En línea] 2017. [Citado el: 8 de noviembre de 2017.] https://developer.mozilla.org/es/docs/Web/HTML.

Bell, Donald. 2004. UML basic. *The component diagram.* [En línea] 2004. [Citado el: 14 de marzo de 2018.] https://www.ibm.com/developerworks/rational/library/dec04/bell/index.html.

Bermúdez, G. S, Jiménez, I. M y Rodríguez, L. R. 2013. Uso de patrones de diseño: Un caso Práctico. *Journal of Tropical Engineering 22.2.* 2013.

Cárdenas Escalante, Lain. 2014. Aplicación de patrones de diseño para garantizar alta flexibilidad en el software. *Tecnología y desarrollo. Universidad César Vallejo.* [En línea] 2014. [Citado el: 29 de enero de 2018.] http://revistas.ucv.edu.pe/index.php/RTD/article/view/696/541.

Díaz Alonso, Lexys Manuel. 2017. Módulo para el monitoreo en tiempo real de los clientes ligeros desde Nova-LTSP. *Trabajo de diploma para optar por el Título de Ingeniero en Ciencias Informáticas.* 2017.

Díaz, Francisco Javier, y otros. 2010. Evaluación de herramientas Free/Open Source para pruebas de software. *Laboratorio de Investigación de Nuevas Tecnologías Informáticas, Facultad de Informática, Universidad de La Plata, Buenos Aires.* [En línea] 2010. [Citado el: 3 de abril de 2018.] https://www.linti.unlp.edu.ar/uploads/docs/evaluacion_de_herramientas_open_source_para_pruebas_de_software.pdf.

django. 2017. The Django admin site. *Django Software Foundation.* [En línea] 2017. [Citado el: 30 de noviembre de 2017.] https://docs.djangoproject.com/es/1.9/ref/contrib/admin/#logentry-objects.

REFERENCIAS

Domínguez, María. 2016. ¿Qué es Bootstrap y cuáles son sus ventajas? *Bootstrap.* [En línea] 2016. [Citado el: 7 de noviembre de 2017.] http://puntoabierto.net/blog/que-es-bootstrap-y-cuales-son-sus-ventajas.

Fernández de Castro Fabre, Astrid y López Padrón, Alexander. 2014. Validación mediante criterio de usuarios del sistema de indicadores para prever, diseñar y medir el impacto en los proyectos de investigación del sector agropecuario. *Revista Ciencias Técnicas Agropecuarias.* [En línea] 2014. [Citado el: 5 de abirl de 2018.] http://scielo.sld.cu/scielo.php?script=sci_arttext&pid=S2071-00542014000300012. ISSN 2071-0054.

Fitgerald, Eric y Heinbockel, Bill. 2010. Common Event Expression (Cee) Overview. 2010.

Fumás Cases, Eduard. 2015. Apache HTTP Server: ¿Qué es, cómo funciona y para qué sirve? *consultoría Marketing online.* [En línea] 2015. [Citado el: 9 de noviembre de 2017.] http://www.ibrugor.com/blog/apache-http-server-que-es-como-funciona-y-para-que-sirve/.

Guiu, David. 2015. Social Media, Empresas & TIC. [En línea] 2015. [Citado el: 8 de noviembre de 2017.] https://www.socialetic.com/que-es-html5.html.

Gutiérrez, E. 2009. JavaScript: Conceptos básicos y avanzados. *Barcelona, España: Informática Técnica.* [En línea] 2009. [Citado el: 19 de marzo de 2018.]

Hernández, B, Rodríguez, O y Mar, O. 2015. Sistema para la gestión de impresiones del Vicedecanato de la Facultad 6 de la Universidad de las Ciencias Informáticas. [En línea] 2015.

Infante Montero, Sergio. 2012. Curso Django para perfeccionistas con deadlines. [En línea] 2012. [Citado el: 7 de noviembre de 2017.] http://www.academia.edu/9510572/maestrosdelweb_curso_django.

Iruela, Frank. 2016. Los gestores de bases de datos más usados. *Revista digital INESEM.* [En línea] 2016. [Citado el: 29 de noviembre de 2017.] https://revistadigital.inesem.es/informatica-y-tics/los-gestores-de-bases-de-datos-mas-usados/.

Jacobson, Ivar, Booch, Grady y Rumbaugh, James. 2004. El Proceso Unificado del desarrollo del software. 2004. pág. 464.

Larman, Craig. 2004. UML y Patrones. Introducción al análisis y diseño orientado a objetos. [En línea] 2004. [Citado el: 29 de enero de 2018.] http://www.fmonje.com/UTN/ADES%20-%20208/UML%20y%20Patrones%20%202da%20Edicion.pdf.

Leyva, Viana de la Cruz, Ortiz Pacheco, Yoel Adrián y Enamorado Selema, Noel. 2017. Sistemas informáticos para el diseño y gestión de recorridos virtuales para Tourdroid. *3C TIC: Cuadernos de desarrollo aplicado a las TIC.* 2017. Vol. 6, no. 2, págs. 38-59. ISSN: 2254 – 6529.

López, Patricia y Ruíz, Francisco. 2011. Lenguaje Unificado de Modelado - UML. 2011.

Martínez Estévez, Rafael, Pereira Rosa, Manuel y Gónzalez Fernández, Raimundo. 2014. Viabilidad de Python en la enseñanza de la programación. [En línea] 2014. [Citado el: 8 de noviembre de 2017.] http://mendive.upr.edu.cu/index.php/MendiveUPR/article/view/659/658. ISSN 1815-7696.

Medina Carbó, Yosvany. 2016. Cuba y el impacto de las TIC en la informatización de la sociedad. *monografía.com.* [En línea] 2016. [Citado el: 30 de septiembre de 2017.]

REFERENCIAS

http://www.monografias.com/trabajos109/cuba-y-impacto-tic-informatizacion-sociedad/cuba-y-impacto-tic-informatizacion-sociedad.shtml#cubayelusa.

Medina Rojas, Ferley. 2012. Sistemas de clientes livianos con licenciamiento libre para entornos universitarios. [En línea] 2012. [Citado el: 28 de octubre de 2017.] http://www.laccei.org/LACCEI2012-Panama/RefereedPapers/RP124.pdf.

Montielo Torrado, Jordan, González Castro, Yoandry y Carbonell Tamayo, Arianna. 2016. Análisis y diseño de un sistema para la gestión de la información de los procedimientos del servicio hermodinámica . [En línea] 2016. [Citado el: 5 de abril de 2018.] http://www.informaticahabana.cu/sites/default/files/ponencias/SLD09.pdf.

Navaja Ojeda, Antonio. 2012. Guía completa de CSS3. *Guía completa de CSS3.* [En línea] 2012. [Citado el: 9 de noviembre de 2017.] https://openlibra.com/es/book/guia-completa-de-css3.

Orellana García, Arturo, Damian, Pérez y Larrea Armenteros, Osvaldo. 2015. ResearchGate. [En línea] 2015. [Citado el: 22 de mayo de 2018.] https://www.researchgate.net/publication/282000093_Generador_de_Registros_de_Eventos_para_el_analisis_de_procesos_en_el_Sistema_de_Informacion_Hospitalaria_xavia_HIS.

Penádes, Maria del Carmen y Letelier Torres, Patricio. 2006. Métodologías ágiles para el desarrollo de software eXtreme Programming (XP). 2006. Vol. 5, no. 26. ISSN 1666-1680.

Peña Escalona, Yannier. 2013. Módulo para la administración de clientes ligeros en Nova para Servidores. *Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas.* La Habana : s.n., 2013.

Pérez Escobar, Carlos Javier. 2010. ¿Qué significa CMMI? *Aspro Tech.* [En línea] 2010. [Citado el: 23 de enero de 2018.] http://asprotech.blogspot.com/2010/10/tecnicas-para-identificacion-de.html.

Pérez Porto, Julian y Gardey, Ana. 2009. Definición.De. *Definición de Evento.* [En línea] 2009. [Citado el: 19 de octubre de 2017.] https://definicion.de/evento/.

Pérez Villazón, Yoandy y otros. 2013. El proceso de migración a aplicaciones de código abierto en Cuba desde un enfoque metodológico. *Revista Cubana de Ciencias Informáticas.* 2013. Vol. Vol 7, no. 4. ISSN 2227-1899.

pgAdmin. 2016. pgAdmin, PostgreSQL Tools. [En línea] 29 de septiembre de 2016. [Citado el: 3 de diciembre de 2016.] https://www.pgadmin.org/.

Pierra, Allan. 2011. Nova, Distribución Cubana de GNU/Linux. Reestructuración estratégica de su proceso de desarrollo. [En línea] 2011. [Citado el: 7 de noviembre de 2017.] https://repositorio.uci.cu/jspui/handle/ident/8710.

Pinelo, David. 2009. Introducción a UML. [En línea] 2009. [Citado el: 8 de noviembre de 2017.] http://moodle2.unid.edu.mx/dts_cursos_mdl/pos/TI/IS/AM/10/Introduccion_uml.pdf.

Porro Santos, Ernesto, Vigoa Machin, Lilian y Alonso Reyes, Reynaldo. 2015. Guía de implementación de LTSP sobre Lubuntu 12.04 para los clientes ligeros de. *Revista Publicando.* 2015. Vol. 2, no.5, págs. 36-46. ISSN 1390-9304.

REFERENCIAS

Porven Rubier, Joelsy y Montesino Perurena, Raydel. 2015. Marco de trabajo para la gestión centralizada de trazas de seguridad usando herramientas de cógido abierto. *Revista Cubana de Ciencias Informáticas.* 2015. Vol. 9, no. 3, págs. 18-32. ISSN: 2227-1899.

PostgreSQL. 2017. The PostgreSQL Global Development Group . [En línea] 2017. [Citado el: 29 de noviembre de 2017.] https://www.postgresql.org/?&.

Pressman, Roger. 2010. Ingenieria del Software. Un enfoque práctico. Séptima edición. México D.F: McGraw-Hill : s.n., 2010. ISBN: 978-607-15-0314-5.

Producción, Dirección Técnica de la. 2010. *Programa de mejoras. Estándar de codificación para python. Universidad de las Ciencias Informáticas.* La Habana : s.n., 2010.

proxmox. 2018. Proxmox Server Solutions GmbH. [En línea] 2018. [Citado el: 22 de mayo de 2018.] https://www.proxmox.com/en/.

Pycharm. 2017. Pycharm. [En línea] 2017. [Citado el: 10 de noviembre de 2017.] https://www.jetbrains.com/pycharm/.

Quiroz, Patricio, Muñoz, Roberto y Noél, René. 2012. Desarrollo de un lenguaje de programación y entorno de desarrollo que facilite la programación de robots LEGO mindstorms. [En línea] 2012. [Citado el: 9 de noviembre de 2017.] http://www.tise.cl/volumen8/TISE2012/68.pdf.

Redondo González, Miguel. 2017. Estudio con modelos de datos para la automatización en redes eléctricas inteligentes. *Estudio con modelos de datos para la automatización en redes eléctricas inteligentes.* [En línea] 2017. [Citado el: 20 de febrero de 2018.] http://hdl.handle.net/10396/14506.

Reguant-Álvarez, Mercedes y Torrado-Fonseca, Mercedes. 2016. El método Delphi. *REIRE, Revista d'Innovació i Recerca en Educació.* 2016. Vol. 9, no. 1, págs. 87-102. ISSN:2013-2255.

Rodríguez Figueredo, Héctor y otros. 2015. El movimiento del Software Libre. *Revista Cubana de Ciencias Informáticas.* 2015. Vol. 3, págs. 5-12.

Rodríguez Sánchez, Tamara. 2015. *Metodología de desarrollo para la actividad productiva de la UCI. Universidad de las Ciencias Informáticas.* La Habana. Cuba : s.n., 2015.

Silega, Nemury. 2014. Método para la transformación automatizada del modelo de procesos de negocio a modelos de componente para sistemas de gestión empresarial. *Tesis presentada en opción al grado de Doctor en Ciencias Técnicas.* [En línea] 2014. [Citado el: 5 de abril de 2018.] https://repositorio_institucional.uci.cu/jspui/bitstream/ident/8584/3/Tesis_Nemury_V14.pdf.

Sommerville, Ian. 2011. Ingeniería de Software. 9na s.l. : Addison-Wesley, 2011. ISBN 978-0-13-703515-1.

SparxSystems. 2014. Diagrama de despliegue UML 2. *Sparx Systems-Tutorial UML 2-Diagrama de despliegue.* [En línea] 2014. [Citado el: 21 de febrero de 2018.] http://www.sparxsystems.com.ar/resources/tutorial/uml2_deploymentdiagram.html.

Suárez Fabre, Joyce, Cabrera Martínez, Maurice y Madelis, Pérez Gil. 2016. *Ventanilla única desde la experiencia del departamento de soluciones para la aduana del centro CEIGE.* [En línea] 2016. http://ecorfan.org/handbooks/Ciencias%20Tecnologia%20Innovacion%20T-I/Handbook%20Universidad%20Tecnol%C3%B3gica%20de%20Quer%C3%A9taro_2.pdf.

REFERENCIAS

Van Rossum, Guido. 2009. Tutorial Python version 2.6.2. [En línea] 2009. [Citado el: 7 de noviembre de 2017.] http://ralsina.me/static/tutorial-8.pdf.

Vilanova Blanco, Luis. 2017. Kyocera. *¿Qué es un servidor web?* [En línea] 2017. [Citado el: 9 de noviembre de 2017.] https://smarterworkspaces.kyocera.es/blog/que-es-un-servidor-web/.

ANEXOS

Anexo 1

Entrevista al líder del proyecto de Nova-LTSP en el centro CESOL:

Estimado especialista: Se necesita de su cooperación en una investigación para una tesis de pregrado. Por ello, sería de gran ayuda que respondiera lo siguiente:

1. ¿Existe en la actualidad algún mecanismo para registrar los eventos en la Plataforma Nova-LTSP? En caso positivo ¿De qué manera lo hace?
2. ¿Por qué usted considera necesario almacenar los registros de eventos de las acciones que realizan los usuarios sobre la plataforma Nova-LTSP?
3. ¿Qué requerimientos le gustaría a usted que brindara el software a desarrollar?
4. ¿Qué otras características considera que deba presentar el sistema, en cuanto a la usabilidad, seguridad, interfaz u otro aspecto que garantice su calidad?

Muchas gracias por su colaboración.

Anexo 2

Entrevista realizada al Ing Osay González Fuentes. Jefe del centro de Identificación y Seguridad Digital CISED de la UCI

Estimado especialista: Se necesita de su cooperación en una investigación para una tesis de pregrado. Por ello, sería de gran ayuda que respondiera lo siguiente:

¿Qué es el sistema de gestión de rentas de autos y limusinas (SIGREX)?

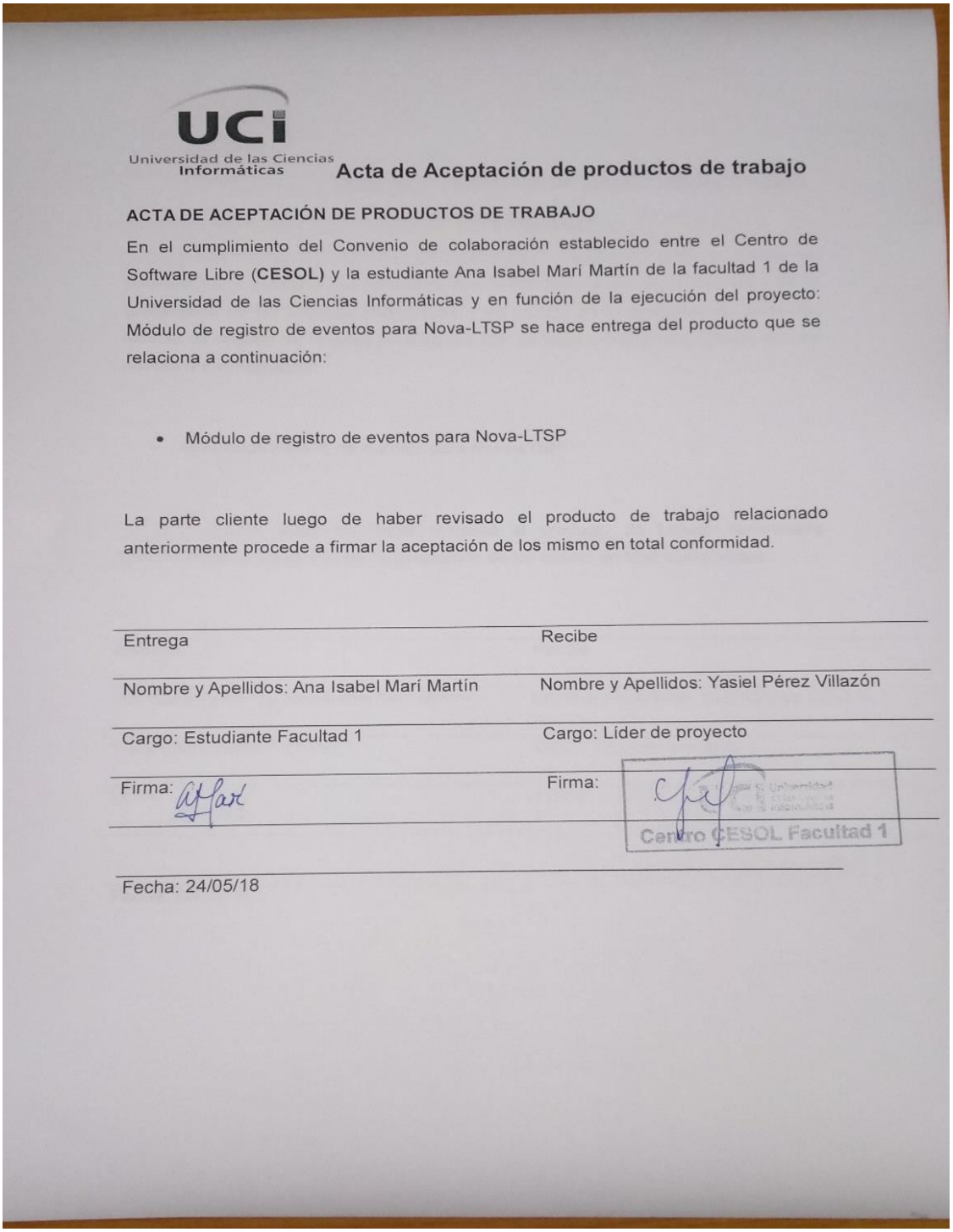
SIGREX es un sistema destinado a la renta de autos y limusinas el cual incluye un registro de eventos para así auditar las acciones que en el sistema se realizan evitando luego errores y desinformación en el proceso.

¿Cómo se realiza el registro de eventos en el sistema?

El registro de eventos se llevan a cabo sobre las acciones que realizan los usuarios en los recursos, guardando la hora, fecha, usuario, modelo, acción realizada, el objeto y la ubicación de donde se realizó la acción. Estas solo pueden ser gestionadas por los administradores del sistema. Utiliza para

el registro de evento la biblioteca *LogEntry* encargada de registrar las acciones que se realizan a la base de dato.

Anexo 3: Acta de Aceptación



UCI
Universidad de las Ciencias Informáticas

Acta de Aceptación de productos de trabajo

ACTA DE ACEPTACIÓN DE PRODUCTOS DE TRABAJO

En el cumplimiento del Convenio de colaboración establecido entre el Centro de Software Libre (**CESOL)** y la estudiante Ana Isabel Marí Martín de la facultad 1 de la Universidad de las Ciencias Informáticas y en función de la ejecución del proyecto: Módulo de registro de eventos para Nova-LTSP se hace entrega del producto que se relaciona a continuación:

- Módulo de registro de eventos para Nova-LTSP

La parte cliente luego de haber revisado el producto de trabajo relacionado anteriormente procede a firmar la aceptación de los mismo en total conformidad.

Entrega	Recibe
Nombre y Apellidos: Ana Isabel Marí Martín	Nombre y Apellidos: Yasiel Pérez Villazón
Cargo: Estudiante Facultad 1	Cargo: Líder de proyecto
Firma:	Firma:

Centro CESOL Facultad 1

Fecha: 24/05/18

Anexo 4

Encuesta inicial de la investigación

Especialista, le invito a responder el siguiente cuestionario con el fin de conseguir su colaboración en la presente investigación, solicito que exprese en sus respuestas criterios verídicos que guíen al autor de la investigación. Marque con una X en una sola opción y en el caso de la 5 responda brevemente. Por el tiempo brindado, muchas gracias.

1. ¿Considera importante la implementación de un sistema que permita registrar los eventos en la plataforma Nova-LTSP?

Sí ___ No ___ sé___

2. ¿Considera usted correcta la forma en que se registran los eventos generados por la plataforma Nova-LTSP?

Sí ___ No ___ sé___

3. ¿Considera usted factible la implementación de un módulo que permita conocer los eventos de la plataforma Nova-LTSP?

Sí ___ No ___ sé___

4. Luego de haber mostrado los resultados de la solución refleje en qué medida le gusta la solución desarrollada.

__Me gusta mucho __Me disgusta más de lo que me gusta __Me gusta más de lo que me disgusta __No me gusta nada __Me da lo mismo __No sé decir

5. ¿Qué opina usted acerca de los beneficios que traería para las instituciones del país disponer del sistema propuesto para el registro de eventos en Nova-LTSP?