Universidad de las Ciencias Informáticas

TRABAJO DE DIPLOMA PARA OPTAR POR EL TÍTULO DE INGENIERO EN CIENCIAS INFORMÁTICAS

Título**:** Plataforma svntoweb para la administración integrada de servidores Apache y Subversion en la XETID

Autor:

Juan Carlos Matos del Rio

Tutores:

Ing. Dany Esquijarosa Bonilla

Ing. Yanet Jiménez Escobar

Ing. Leiny Amel Pons Flores

La Habana, 2018

"Año 60 de la Revolución"

"Me he dado cuenta que incluso las personas que dicen que todo está predestinado y que no podemos hacer nada para cambiar nuestro destino, siguen mirando a ambos lados antes de cruzar la calle"

Stephen Hawking

DECLARACIÓN DE AUTORÍA

Declaración de autoría:

Declaro ser autor de la presente tesis y reconozco a la XETID, Empresa de Tecnologías de la Información para la Defensa, los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmo la presente a los __ días del mes _____ del año _____

_____

Firma del Autor

Juan Carlos Matos del Rio

_____

Firma del Tutor

Yanet Jiménez Escobar

_____

Firma del Tutor

Dany Esquijarosa Bonilla

_____

Firma del Tutor

Leiny Amel Pons Flores

DATOS DEL CONTACTO

Datos del contacto

Síntesis de Tutor(es):

Yanet Jiménez Escobar
Administrador de red y sistemas con 2 años de experiencia. Graduada UCI.
Email: yjescobar@xetid.cu

Dany Esquijarosa Bonilla
Administrador de red y sistemas con 9 años de experiencia. Graduado UCI.
Email: dbonilla@xetid.cu

Leiny Amel Pons Flores
Subdirector del Centro de Ideoinformática (CIDI) con 4 años de experiencia. Graduado UCI.
Email: lenny@uci.cu

Agradecimientos

Tal vez no sean las palabras más bonitas, pero si les garantizo que están escritas con el corazón, le agradezco a:

Mi mamá, mi luz, gracias por estar siempre ahí sin pedir nada a cambio, por reír conmigo en los buenos momentos y llorar en los duros. Gracias por apoyarme a toda hora, por sacrificarte por mí y por mi hermana y soportar nuestras malacrianzas.

A mi peleón favorito, mi papá, tal vez nunca te he dicho lo orgulloso que me siento de ser tu hijo, gracias a ti he comprendido lo que es ser un hombre de bien y un buen hijo.

Mi hermana tú sabes que después de mi mamá eres la segunda mujer de mi vida como te lo dijo aquella vez el señor Max. Viki no sabes cuánto te quiero y como me esfuerzo cada fin de semana que voy a la casa por ser un mejor hermano. Te agradezco cada momento en el que me diste tu apoyo. Gracias Lachy.

Mis abuelos por velar por mí en toda ocasión y darme consejos que me sirvieron para seguir adelante, cito el de mi abuela María: ¡Usted es un hombre, ajústese los pantalones y tire pa`lante! y el de mi abuela Raquel con: ¡El hombre no se mide por cuantas veces tropieza sino por cuantas veces se levanta! Gracias a los cuatro.

A todos mis familiares y amistades, de Oriente y del exterior, aunque estemos lejos, su apoyo lo sentí en cada momento.

A mis tutores Dany, Yanet y Leiny, más que tutores amigos. Perdón por cada dolor de cabeza que les di y gracias por su apoyo y disposición.

A los cuatro miembros del tribunal, por la paciencia que tuvieron conmigo en mis momentos difíciles.

AGRADECIMIENTOS

A la directiva de la Facultad 1, a los profesores que durante estos cinco años tuve el honor de recibir sus clases, a las instructoras de la residencia, peleonas pero buenas personas.

A todos mis compañeros de aula, a los que se fueron y a los que están, desde la brigada 1103 hasta la 1503, sin duda alguna la mejor.

A mi trío favorito de chicas: Cristy, Daniela y Elena. Gracias a cada una por estar siempre ahí como una hermana más, por preocuparse por mí en los momentos más complicados y ayudarme a salir de estos, sin dejar atrás los buenos momentos que compartimos juntos.

Por último y no menos importante, para mis hermanos del Bakano, en el orden en que los voy a mencionar no significa preferencia alguna:
Carlos, mi nakama (compañero) en esta aventura durante estos cinco años, gracias por tus chistes y tu música.

Mauricio, de usted me quedo con su enorme corazón. Te agradezco por hacerme reír así fuera en los momentos más tristes y por los consejos que me diste como el hermano que nunca tuve.

Manu, aunque el fútbol les debe un mundialito a los Titanes como voy a extrañar nuestras discusiones en los juegos, tal vez como jugadores, el fútbol no es para nosotros, pero como personas nos unió más aún.

Michel, a ti te veo como un hermano mayor, gracias por levantar mis ánimos cuando estaban por el piso, por estar ahí en cada momento bueno y malo soportando mis defectos.

Ramón, no sabes cómo me gusta escuchar tu frase: ¿Quién tiene hambre? Gracias hermano por compartirlo todo con nosotros sin pedir nada a cambio.

AGRADECIMIENTOS

Amado, gracias por tu locura, tus ocurrencias y tu transparencia. Fuiste el primero que conocí desde que entré a la UCI y desde un comienzo siempre has sido el mismo hasta hoy, no cambies.

Papi Rauli, te agradezco por ser un excelente compañero, alguien con quien contar a toda hora.

A el Llillo por esforzarse en la difícil tarea de enseñarnos a bailar.

A David, Vladimir, Diwel, Osvaldo y Alfred por compartir buenos momentos.

A todos los presentes de la sala que gracias a su granito de arena hoy puedo estar aquí.

Dedicatoria

Este trabajo va dedicado a:

Mis padres y mi hermana, mis tres pilares, que sería de mí sin ustedes ¡Los quiero!

Mi más preciado tesoro, mi familia y mis amigos, motivos me sobran para decir cuánto los aprecio.

Los hermanos y hermanas que conocí en estos cinco años de universidad, mientras mi corazón siga latiendo, presente siempre estarán.

A mi abuela María que dentro de unas horas estará cumpliendo años ¡felicidades mami!

A todas aquellas personas que contribuyeron para que hoy estuviera aquí.

Resumen

El presente trabajo de diploma presenta la propuesta de una plataforma web para la Empresa de Tecnologías de la Información para la Defensa (XETID) que garantice la administración de servidores web Apache y control de versiones Subversion de forma integrada. Para el desarrollo de la plataforma se realiza un estudio de los sistemas homólogos encargados de administrar Apache y Subversion; además de un análisis de las herramientas y tecnologías adecuadas para su construcción. La solución está guiada por el Proceso de Desarrollo de Software (PRODESOFT). El sistema se evalúa a través de la técnica Partición de Equivalencias para realizar pruebas funcionales, mientras que se utiliza la aplicación JMeter para hacer pruebas de rendimiento. Para la validación de la solución se emplea el método Opinión de Expertos.

Palabras clave: Apache, administración, despliegue, Subversion.

Índice de contenidos

Índice de Figuras

ÍNDICE DE TABLAS

Índice de Tablas

Introducción

La llamada Web 2.0, la movilidad y computación en la nube, entre otros, hacen que la tendencia de desarrollo se encuentre orientada, sobre todo, a la utilización de tecnologías web según el entorno y las necesidades de los usuarios.

Con diferentes perspectivas, las instituciones y/o empresas, utilizan Entornos Controlados de Desarrollo (ECD) (UCID, 2009), para definir los flujos de trabajo, tecnologías, estándares y mecanismos de seguridad a utilizar. Los servidores, tanto web como de control de versiones, son imprescindible en este escenario. Los servidores web, procesan cualquier aplicación del lado del servidor realizando conexiones bidireccionales y/o unidireccionales y síncronas o asíncronas, generando o cediendo una respuesta en cualquier lenguaje o aplicación del lado del cliente. El uso de sistemas de control de versiones, permite conocer los cambios de un archivo o grupo de archivos determinados en el proceso de elaboración, así como el control de un registro histórico de los cambios realizados por cada miembro (Donato, 2009).

En un ECD, cada desarrollador cuenta con una copia local de código fuente, a la cual le van realizando modificaciones y actualizaciones periódicamente. Antes de subir una modificación al servidor de control de versiones, transita por un servidor de Integración Continua (IC), para garantizar la calidad y limpieza del código, además de verificar problemas de integración con el trabajo de otro desarrollador. El proceso de IC garantiza que las actualizaciones en el código fuente no generen retrasos o anomalías en el desarrollo de una aplicación, quedando una vez subida la actualización, el código más limpio y funcional posible en el repositorio para posteriormente ser publicado en un servidor web.

Según las estadísticas que brinda la compañía Netcraft[1], el servidor web Apache es el más utilizado en la comunidad de desarrollo, entre otras, debido a la abundante documentación y la compatibilidad con los diferentes sistemas operativos. Su soporte de *host virtuales* basados en direcciones IP o por nombre permite servir cientos de sitios web clientes en un único servidor y su sistema de módulos lo hace adaptable a diferentes entornos y necesidades de los clientes.

Entre los servidores de control de versiones, se encuentra Subversion (SVN), que mantiene popularidad, a pesar de otras tendencias, especialmente en la comunidad de desarrolladores de software libre. Es

[1] Es una compañía de servicios de Internet, ofrece análisis de cuota de mercado de servidores y alojamiento web.

centralizado y aporta un respaldo sólido para los proyectos, pero no sólo de la última versión, sino de todas las versiones que han sido subidas al repositorio. Es una fuente valiosa de información, permite saber qué han estado haciendo otras personas, qué ficheros modificaron y si se documentan mínimamente bien los progresos. Dispone de un historial detallado de cómo se ha construido el proyecto, mejorando la coordinación sin necesidad de comunicarse directamente entre miembros del equipo de desarrollo (Ben Collins-Sussman, 2011).

Como parte del ECD de la XETID, los desarrolladores de esta entidad, trabajan constantemente en el perfeccionamiento y adaptabilidad de sus productos para satisfacer las necesidades de sus clientes. A la hora de confeccionar una aplicación, se desarrollan, integran y/o reutilizan tecnologías y componentes de diferentes áreas temáticas o divisiones según las necesidades, mediante un proceso iterativo e incremental (UCID, 2009). Los productos pasan por revisiones de calidad antes del despliegue y finalmente se les brinda soporte de nivel uno, dos y tres, de forma permanente.

En el flujo de trabajo descrito, debido a los bajos niveles de automatización de los despliegues y de la administración, existe ineficiencia en los desarrollos. La inexistencia de una herramienta que integre Apache y Subversion para su gestión, dificulta el trabajo colaborativo, la unidad entre sistemas y la reutilización de códigos. Además, los procesos de despliegue, configuración y administración se realizan de manera independiente ocasionando falta de integración y pérdida de tiempo en los desarrollos.

A partir de lo descrito, se desea garantizar la administración de forma integrada de Subversion y Apache mediante el desarrollo de una plataforma web, por lo que se identifica como **problema de la investigación**: ¿Cómo garantizar la administración de forma integrada de los servidores Apache y Subversion para incrementar el trabajo colaborativo y reducir los tiempos de desarrollo en un entorno controlado?

Se define como **objeto de estudio:** la administración de servidores Apache y Subversion y como **campo de acción:** la administración integrada de servidores Apache y Subversion en el entorno de desarrollo de la XETID.

Para darle solución a la problemática descrita se plantea como **objetivo general:** Desarrollar una plataforma web que administre de forma integrada Apache y Subversion para incrementar el trabajo colaborativo y reducir los tiempos de desarrollo en un entorno controlado.

El objetivo general se puede desglosar en los siguientes **objetivos específicos:**

- ✓ Realizar el estudio de las tecnologías y los sistemas existentes.
- ✓ Elaborar el análisis y diseño de la propuesta de solución.
- ✓ Implementar la plataforma web.
- ✓ Realizar las pruebas al sistema.
- ✓ Evaluar el resultado de la propuesta de solución.

A partir de lo anteriormente expuesto se plantea como **idea a defender**: el desarrollo de una plataforma web permitirá integrar la administración de Apache y Subversion para incrementar el trabajo colaborativo y reducir los tiempos de desarrollo en un entorno controlado.

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

- ✓ Elaboración del diseño teórico – metodológico de la investigación.
- ✓ Realización del estado del arte sobre las herramientas de administración de Apache y Subversion.
- ✓ Selección de las tecnologías, herramientas y estándares necesarios para la implementación de la propuesta de solución.
- ✓ Selección de la metodología de desarrollo de software.
- ✓ Elaboración de los artefactos requeridos por la metodología de desarrollo seleccionada.
- ✓ Implementación de la propuesta de solución.
- ✓ Realización de pruebas funcionales y de rendimiento para evitar vulnerabilidades en la plataforma web.
- ✓ Evaluación de la propuesta de solución.

Los métodos de investigación utilizados se exponen a continuación:

Métodos teóricos:

- ✓ Analítico–sintético: Con el objetivo de analizar la información y la documentación relevante para el desarrollo del software, enfatizando en los elementos más importantes que se relacionan con el objeto de estudio.

- ✓ Inductivo-Deductivo: A partir de los conocimientos adquiridos este método permitió arribar a conclusiones lógicas, así como definir el objetivo general, la idea a defender y los objetivos específicos a tener en consideración.

Métodos Empíricos:

- ✓ Modelación: Se realiza una modelación simplificada de la realidad permitiendo establecer esquemas, modelar ideas de forma gráfica, además de posibilitar la utilización de diagramas para expresar información de manera organizada.
- ✓ Encuesta: Mediante un cuestionario previamente elaborado, se conoce la opinión o valoración de determinadas personas sobre el resultado obtenido.

Estructura del documento:

La presente investigación se encuentra estructurada por tres capítulos y anexos, en los cuales se exponen todo el trabajo investigativo y práctico realizado:

- ✓ **Capítulo 1**: **Fundamentación teórica**. Se describen los conceptos fundamentales asociados al dominio del problema. Se hace un análisis del estado del arte sobre los principales sistemas de administración de Apache y Subversion. Se describen las herramientas, tecnologías y metodología a partir de las cuales se construye la solución.
- ✓ **Capítulo 2**: **Propuesta de solución**. Se realiza la descripción de los procesos de negocio a automatizar y las características generales de la aplicación. Se definen los requisitos funcionales y no funcionales que guían el desarrollo y se describen los elementos del diseño que sirven de apoyo en la fase de implementación.
- ✓ **Capítulo 3**: **Implementación y evaluación.** Se abordan los elementos más significativos de la implementación como los estándares de la codificación utilizados y algunas de sus interfaces de usuario. Se muestra el diagrama de despliegue. Se realizan las pruebas correspondientes a la plataforma. Se muestra el resultado de la validación del sistema obtenido a través de la Opinión de Expertos.

Capítulo 1: Fundamentación teórica

En el presente capítulo se describen los conceptos fundamentales asociados al dominio del problema. Se hace un análisis del estado del arte sobre los principales sistemas de administración de Apache y Subversion y se describen las herramientas, tecnologías y metodología a partir de las cuales se construye la solución.

1.1 Sistemas similares

En el siguiente apartado se realiza un análisis de los diferentes sistemas de administración, con el objetivo de exponer sus principales características y el enfoque que tienen sobre el objetivo planteado en la presente investigación.

1.1.1 Herramientas de administración de Subversion

WebSVN

WebSVN a través de una interfaz interesante, ofrece una vista sobre los repositorios Subversion que se han diseñado para reflejar la metodología de Subversion. Puede ver el registro de cualquier archivo o directorio y una lista de todos los archivos cambiados, añadido o eliminado en cualquier revisión dada. También puede ver las diferencias entre dos versiones de un archivo para ver exactamente lo que fue cambiado en una particular revisión (Nápoles, 2015).

Desde su escrito usando PHP, WebSVN es también portable y fácil de instalar. Ésta herramienta web es liberada bajo la licencia GNU (*General Public License*).

WebSVN ofrece las siguientes características:

- ✓ Interfaz fácil de usar.
- ✓ Multi-lenguaje.
- ✓ Sistema de plantillas personalizables.
- ✓ Archivo coloreado de anuncios.
- ✓ Autoría de vista.
- ✓ Mensaje de registro de la búsqueda (Nápoles, 2015).

El sistema muestra información útil para el administrador sobre el servidor de control de versiones, por vistas y a través de plantillas, pero no posibilita gestionar los niveles de permisos y no posee integración efectiva con el servidor Apache.

VisualSVN

VisualSVN Server es un paquete de instalación que incluye Subversion y el servidor web Apache ya configurados y listos para funcionar desde el primer momento. Su panel de control utiliza el aspecto familiar de las consolas de Windows, con un resumen de los repositorios activos y la dirección del servidor. Los repositorios se pueden crear o importar desde el mismo panel de VisualSVN Server. En las propiedades de cada elemento presenta una pestaña para cambiar los permisos de seguridad (Nápoles, 2015).

Está orientado hacia plataformas Windows y carece de prototipos web. Se administra y gestiona a través de una aplicación de escritorio.

USVN

USVN es una herramienta que presenta una interfaz web escrita en PHP utilizada para configurar los repositorios de Subversion. Su objetivo es facilitar la creación de nuevos proyectos sin tener que usar la interfaz de línea de comandos, por lo tanto, si no se tiene un acceso privilegiado al servidor se genera una lista de usuarios que permite el acceso a su código fuente (Nápoles, 2015).

Carece de integración con el servidor Apache para establecer una sincronización efectiva entre un control de versiones y un *virtualhost* publicado.

SVN Access Manager

SVN *Access Manager* es una herramienta utilizada para gestionar el acceso a los repositorios de Subversion. La herramienta proporciona administración de usuario y de grupo y los derechos de acceso (lectura/escritura) a un repositorio. SVN Access Manager soporta bases de datos MySQL, PostgreSQL y Oracle. Además utiliza proyectos para ofrecer a los usuarios el derecho a administrar sus propios módulos en un repositorio (Nápoles, 2015).

Carece de integración con el servidor Apache para establecer una sincronización efectiva entre un control de versiones y un *virtualhost* publicado.

UberSVN

UberSVN es una suite de administración de Subversion. Desarrollado para ampliar el alcance y la funcionalidad de Subversion. Es un programa gratuito, producto de software desarrollado por WANdisco[2].

Como características de UberSVN se encuentran las siguientes:

1. Una interfaz de codificación social que permite que reveladores colaboren en tiempo real alrededor del código que destinan al depósito.
2. Una interfaz web para crear nuevos depósitos, usuarios gerentes y permisos y administrar ajustes comunes del servidor.
3. Integración con un servidor LDAP[3] para autenticación delegada de usuarios.
4. Actualización en línea automatizada que hace la instalación rápida y confiable.
5. Interfaz gráfica de permisos de usuarios gerentes y acceso al depósito (Nápoles, 2015).

Carece de integración con el servidor Apache para establecer una sincronización efectiva entre un control de versiones y un *virtualhost* publicado. Además de necesitar altos valores de procesamiento y RAM para funcionar de forma efectiva, debido a la utilización de la máquina virtual de java.

1.1.2 Herramientas de administración de Apache

Webmin

WebMin es una herramienta de administración web gráfica e intuitiva. Fue escrita por Jaime Cameron en lenguaje Perl. Es fácil ampliar WebMin y darle nuevas posibilidades, es abierto y está bien documentado. Provee de una interfaz gráfica para prácticamente cualquier tarea que se haga sobre un sistema basado en UNIX. Es importante aclarar que WebMin no modifica opciones desconocidas para él en los archivos de configuración y que no es necesario tener un entorno gráfico porque se puede utilizar con navegadores en modo texto. Las principales características de este entorno administrativo son el cambio de contraseñas de usuarios, planificación de copias de seguridad de directorios importantes, asignación y administración de

[2] Proveedor líder de software de infraestructuras para alta disponibilidad, escalabilidad y duplicación, y un socio corporativo del proyecto de código abierto Subversion.

[3] Protocolo ligero de acceso a directorios (del inglés, Lightweight Directory Access Protocol).

cuotas de disco, configuración de Samba, administración y configuración de servidores POP, servidor web Apache, Servidor Proxy Squid, análisis de históricos, monitorización del ancho de banda, etc. (Sosa, 2010).

Carece de integración efectiva con el servidor de control de versiones Subversion para establecer una sincronización.

Rapache

Rapache es una herramienta gráfica de configuración del servidor web Apache para sistemas GNU/Linux basados en Debian. Mediante una interfaz gráfica sencilla, permite modificar las principales opciones y realizar tareas comunes con el servidor, de una forma mucho más agradable para el usuario, sin necesidad de usar el terminal y editar los ficheros de configuración. La aplicación está escrita en Python, integrada en el escritorio Gnome y utiliza el protocolo de comunicación SSH para realizar las modificaciones en la configuración de Apache y sus módulos. Como principales características se pueden mencionar que administra VirtualHost, administra módulos, edita archivos de configuración e inicia y detiene el servidor (Sosa, 2010).

Carece de integración efectiva con el servidor de control de versiones Subversion.

Herramienta para la Migración y Administración de los Servicios Telemáticos (HMAST)

Esta herramienta creada por el Centro de Soluciones Libres (CESOL), de la Universidad de las Ciencias Informáticas, permite administrar los servidores de forma remota, contemplando las funcionalidades necesarias para administrar los usuarios, las tareas programadas y los servicios. El módulo web de HMAST permite la administración de los servidores web en dicha herramienta, asegurando su adaptabilidad en las diferentes empresas e instituciones cubanas. El módulo administra Apache, por lo que el sistema está estructurado basándose en su funcionamiento. Permite la administración de Apache en diferentes PC servidoras de forma remota. Se puede administrar las secciones de configuración de Apache, donde la configuración del entorno global afecta tanto al servidor principal como a los *hosts* virtuales, ya sean basados en nombres o basados en IP. Facilita un mejor entendimiento por parte de los usuarios para la gestión de los *hosts* virtuales debido a que se muestra de forma detallada la estructura de almacenamiento lógico de los mismos (Delgado, 2014).

El módulo web permite realizar las siguientes funcionalidades de Apache2:

- **Gestionar *hosts* virtuales asignados a Apache2**: Permite al usuario mostrar, adicionar, eliminar, modificar, habilitar y deshabilitar *hosts* virtuales.
- **Permite Manejar los estados de Apache2:** Se manejan los estados del servidor Apache2 realizando diferentes acciones como: instalar, iniciar, reiniciar, detener y recargar.
- **Modificar puertos para las conexiones en Apache2:** Permite modificar los puertos para las conexiones en Apache2.
- **Especificar los módulos a usar en Apache2:** Permite al usuario mostrar, habilitar, y deshabilitar módulos.
- **Especificar parámetros en el servidor principal de Apache2:** Muestra y modifican algunos parámetros en el servidor principal de Apache2.
- **Especificar parámetros en el MPM *prefork* de Apache2:** Permite mostrar y modificar los parámetros de configuración pertenecientes al MPM *prefork.*
- **Especificar parámetros para el manejo de conexiones en Apache2:** Permite manejar el estado de las conexiones en Apache2 (Delgado, 2014).

Carece de integración con el servidor de control de versiones Subversion para establecer una sincronización efectiva.

1.1.3 Herramientas en XETID

En la empresa XETID actualmente existen dos herramientas encargadas de la actualización del código fuente de los componentes de los distintos proyectos.

1. La primera herramienta emplea Subversion como control de versiones y Apache como servidor web. En un servidor de aplicaciones se encuentran publicadas las ramas de los códigos fuentes de los proyectos y la aplicación web que permite las actualizaciones de código. En el repositorio Subversion se encuentran los *hooks*[4] o ganchos, con la configuración para que una vez se actualice, busque el número de revisión actualizado y en dónde se produjo, simultáneamente se ejecuta un script para actualizar la revisión publicada en el servidor web. Cuando se realiza el intercambio de código entre el desarrollador

[4] Un directorio lleno de plantillas de ganchos, una vez que se hayan instalados algunos.

y Subversion, la aplicación web alojada en el servidor de aplicación, ejecuta el script descrito con anterioridad, publicando en el servidor web las modificaciones realizadas por el desarrollador subidas para el SVN.

2. La segunda herramienta al igual que la primera utiliza Subversion y Apache como control de versiones y servidor web respectivamente. Cada desarrollador tiene una copia local a las que le realizan *commit* [5] y posteriormente la suben al repositorio SVN, que tiene configurado los ganchos por cada control de versiones para publicar en servidores web. Está constituido por dos partes, un cliente y otro servidor. La cliente se encarga de configurar la gestión del servidor de control de versiones y la parte servidor de establecer comunicación entre un servidor web con el Apache, de tal forma que pueda desplegarse en servidores diferentes.

1.2 Valoración de los sistemas estudiados

El estudio realizado sobre estas herramientas aportó una serie de elementos a tener en cuenta para el desarrollo de la propuesta de solución. Dentro de estos elementos destacan en primer lugar por especificaciones del cliente, utilizar como servidor web a Apache y como control de versiones a Subversion, aunque existan otras tendencias. En segundo lugar, la administración debe ser de manera integrada para ambos sistemas.

Por tanto, las herramientas analizadas gestionan de manera independiente un único sistema o carecen de una integración efectiva que ayuden a reducir los tiempos de desarrollo y el trabajo colaborativo. Las concebidas y empleadas en XETID a pesar de usar los sistemas anteriores, ninguna permite el despliegue de forma automatizada a través de un instalador, la configuración y administración se realizan a través de líneas de comando, no permiten administrar a través de interfaces gráficas las funcionalidades más importantes de Apache y Subversion, ni el registro de los eventos ocurridos. Aunque poseen la facilidad de mantener actualizada de forma automática las publicaciones del control de versiones en los directorios raíces de los servidores web.

[5] Enviar cambios desde la copia de trabajo local al repositorio.

1.3 Integración entre Apache y Subversion

Apache es un servidor web que Subversion aprovecha a través de un módulo propio y permite servir repositorios Subversion por el protocolo WebDAV/DeltaV.

Subversion puede conectarse al servidor HTTP Apache como un módulo de extensión. Esto proporciona una gran ventaja en estabilidad e interoperabilidad y acceso instantáneo a las características existentes que ofrece este servidor: autenticación, autorización, compresión de la conexión, etcétera (Ben Collins-Sussman, 2011). Mediante el módulo **mod_dav_svn**, Apache puede hacer que el repositorio SVN esté disponible a los usuarios mediante la red.

1.3.1 Script Gancho

Gancho: es un programa activado por algún evento del repositorio, como la creación de una nueva revisión o la modificación de una propiedad no versionada. A cada gancho se le da suficiente información para que sepa de qué evento se trata, cuál es su objetivo y el nombre de usuario de la persona que ejecutó el evento. Dependiendo de la salida del gancho o del estado de su salida, el programa de enganche puede continuar la acción, pararla, o suspenderla de alguna manera. Existe una plantilla por cada gancho que implementa el repositorio Subversion y examinando los contenidos de dichas plantillas de *scripts*, puede ver qué disparador ejecuta cada *script* y qué datos se le pasan. También se encuentran en muchas de estas plantillas ejemplos de cómo debería usar dicho *script*, conjuntamente con otros programas provistos por Subversion, para realizar tareas comunes y útiles (Ben Collins-Sussman, 2011).

- **pre-commit:** Se ejecuta cuando se completa la transacción antes de ser enviados los datos al repositorio. Este gancho se usa normalmente como protección contra envíos que no se permiten debido a contenido o ubicación (por ejemplo, el sitio puede requerir que todos los cambios sobre una rama determinada incluyan un número de ticket del seguimiento de fallos, o que el mensaje de informe de cambios entrante no esté vacío). El repositorio pasa dos argumentos a este programa: la ruta al repositorio y el nombre de la transacción que va a sufrir el cambio. Si el programa devuelve una salida que no sea cero, el envío se aborta y la transacción se borra (Ben Collins-Sussman, 2011).
- **post-commit:** Se ejecuta después de que la transacción se haya confirmado y una nueva revisión haya sido creada. La mayoría de las personas usan este gancho para enviar correos descriptivos

acerca de la confirmación o para hacer una copia de seguridad del repositorio. El repositorio pasa dos argumentos a este programa: la ruta al repositorio y el número de la nueva revisión creada. El código de salida del programa es ignorado (Ben Collins-Sussman, 2011).

Subversion trata de ejecutar los ganchos como el mismo usuario que posee el proceso que está accediendo al repositorio Subversion. En la mayoría de los casos, el repositorio se accede a través del servidor HTTP Apache y mod_dav_svn, así que este usuario es el mismo que el usuario con el que se ejecuta Apache. Los mismos ganchos necesitan ser configurados con permisos a nivel de sistema operativo que les permitan ser ejecutados por dicho usuario. Asimismo, esto significa que cualquier fichero o programas (incluyendo el repositorio mismo) al que acceda directa o indirectamente el gancho, será a través del mismo usuario (Ben Collins-Sussman, 2011).

1.4 Guía para el proceso de desarrollo

Un proceso de desarrollo de software tiene como objetivo la producción eficiente de un producto de software que satisfaga los requisitos de un especialista funcional con una planificación y una estimación de recursos predecibles (UCID, 2009). Por otro lado, una metodología de software, comprende un conjunto de técnicas, herramientas, métodos y un soporte documental que ayuda a los desarrolladores.

Las metodologías se basan en una combinación de los modelos de proceso de desarrollo genéricos (cascada, incremental...) mediante las cuales se definen roles, actividades, prácticas y técnicas recomendadas. Asimismo, permiten optimizar el proceso y producto, hacer una guía en la planificación y en el desarrollo del software, por último, definir qué hacer, cómo y cuándo durante el desarrollo y mantenimiento del proyecto (Ramírez, 2013).

Para el desarrollo de la solución, se definió como guía de trabajo el Proceso de Desarrollo y Gestión de Proyectos de Software, por sus siglas (PRODESOFT), adoptado por la empresa XETID, facilitando la construcción, instalación y mantenimiento de sus sistemas. El modelo de desarrollo de software propuesto, como característica fundamental, combina los modelos basados en componentes y el iterativo e incremental, describiéndolos como aparece a continuación:

- **Desarrollo basado en componentes:** pretende alcanzar un mayor nivel de reutilización de software, aún en contextos distintos a aquellos para los que fue diseñado. Permite que las pruebas sean ejecutadas en cada uno de los componentes antes de probar el conjunto completo de estos

ensamblados.

Cuando existe un débil acoplamiento entre componentes, el desarrollador es libre de actualizar y/o agregar otros según sea necesario, sin afectar terceras partes del sistema. Dado que un componente puede ser construido y luego mejorado continuamente, la calidad de una aplicación que siga este modelo

mejorará con el paso del tiempo.

- **Desarrollo iterativo e incremental:** es un enfoque en el que el ciclo de vida está compuesto por iteraciones, estas son pequeños procesos compuestos de varias actividades cuyo objetivo es entregar una parte del sistema parcialmente completo, probado, integrado y estable. Todo el software es integrado en cada entrega de cada iteración hasta obtener el producto de software completo en la última iteración. En cada iteración se obtiene como resultado un incremento (UCID, 2009).

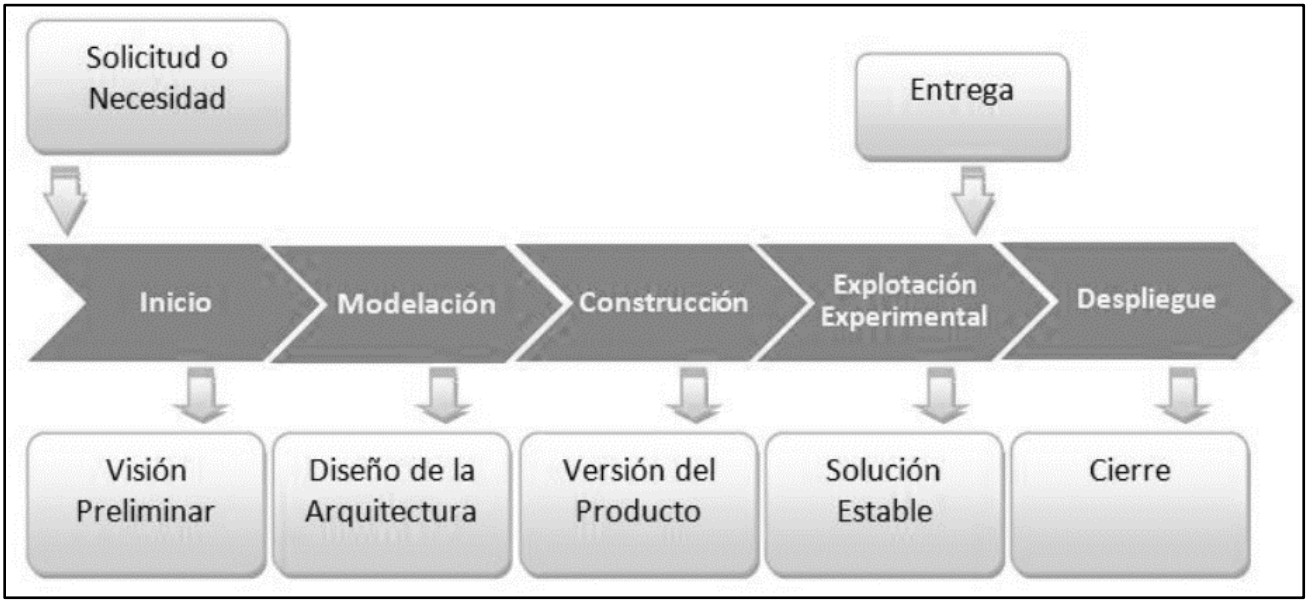


Figura 1: Etapas del ciclo de vida de un proyecto

1.5 Elementos a utilizar en el desarrollo de la solución

En la actualidad debido al desarrollo alcanzado en la informática son innumerables las tecnologías que pueden ser usadas para el desarrollo de un software informático. Es importante conocer estas tecnologías,

sus ventajas y sus particularidades, para poder seleccionar cuáles son las que más pueden aportar a la aplicación que se quiere desarrollar.

A continuación, se muestran las tecnologías que por sus ventajas se utilizan en el desarrollo de la plataforma web.

1.5.1 Herramienta de modelado

Visual Paradigm 8.0

Es una suite de aplicaciones, herramienta profesional que soporta el ciclo completo de desarrollo de software. Soporta el modelado visual con UML, que ayuda a una rápida construcción de aplicaciones de calidad, mejores y con menor costo. Permite generar documentación en HTML/PDF con los diagramas realizados. Permite la generación de bases de datos, conversión de diagramas entidad-relación a tablas de base de datos, mapeos de objetos y relaciones, ingeniería inversa desde gestores de bases de datos. Proporciona una plataforma de modelado colaborativo para el trabajo en equipo. Con las características de colaboración en equipo, los miembros del equipo pueden ver y editar el mismo proyecto, o el mismo esquema, incluso de forma simultánea. Todos los cambios se almacenan en el servidor de Visual Paradigm en función de revisión.

Se caracteriza por:

- Ser software libre.
- Presentar una licencia: gratuita y comercial.
- Estar disponible para múltiples plataformas (Windows, Linux).
- Permitir el uso de un lenguaje estándar común a todo el equipo de desarrollo que facilita la comunicación.
- Permitir la realización de ingeniería directa e inversa.
- Modelado colaborativo con CVS y Subversion (Tarragó, 2012).

Lenguaje Unificado de Modelado (UML 2.0)

Lenguaje Unificado de Modelado (*Unified Modeling Lenguaje*, UML por sus siglas en inglés), es un lenguaje basado en una notación gráfica la cual permite: especificar, construir, visualizar y documentar los objetos

de un sistema programado. Es un lenguaje para la modelación de sistemas, con características orientado a objetos. Aplicable para tratar asuntos inherentes a sistemas complejos de misión crítica, tiempo real y cliente/servidor. Por otra parte facilita a los integrantes de un equipo multidisciplinario participar e intercomunicarse fácilmente (Pérez, 2014).

1.5.2 Herramienta de base de datos

PostgreSQL 9.4

Es un gestor de bases de datos orientadas a objetos muy conocido y usado en entornos de software libre porque cumple los estándares SQL92 y SQL99 y también por el conjunto de funcionalidades avanzadas que soporta, lo que lo sitúa al mismo a un mejor nivel que muchos SGBD (Sistema de Gestión de Base de Datos) comerciales. Funciona en múltiples plataformas. Cuenta con un rico conjunto de tipos de datos, permitiendo además su extensión mediante tipos y operadores definidos y programados por el usuario. Su administración se basa en usuarios y privilegios. Puede extenderse con librerías externas para soportar encriptación y búsquedas por similitud fonética. Controla la concurrencia multi-versión, lo que mejora sensiblemente las operaciones de bloqueo y transacciones en sistemas multi-usuario (Herrera, 2015).

PgAdmin III 1.22.0

PgAdmin III es una aplicación gráfica para gestionar el gestor de bases de datos PostgreSQL, con licencia Open Source. Está escrita en C++, lo que permite que se pueda usar en varios sistemas operativos como Linux, FreeBSD, Solaris, Mac OS X y Windows. Incluye un editor SQL con resaltado de sintaxis, un editor de código de la parte del servidor y un agente para lanzar scripts programados. La conexión al servidor puede hacerse mediante conexión TCP/IP (Pérez, 2014).

1.5.3 IDE de desarrollo

Un entorno de desarrollo integrado o IDE (acrónimo en inglés de *Integrated Development Environment*), es un programa informático compuesto por un conjunto de herramientas de programación que puede dedicarse en exclusiva a un sólo lenguaje de programación o varios (Pérez, 2014).

Pycharm 2017.3

PyCharm es un IDE (Entorno de desarrollo integrado) desarrollado por la compañía Jetbrains utilizado específicamente para programar en el lenguaje Python. Cuenta con una versión gratuita llamada

"Community Edition" liberada bajo la licencia Apache que presenta entre sus características: autocompletado, depuración de código gráfico, refactorización, inspección de código, integración de control de versiones, ejecución de "tests", soporte para el desarrollo web con Django además de contar con un editor inteligente y una consola python integrada. La versión seleccionada es la v2017.3 pues esta incluye mejoras para reducir el tiempo de espera para que se complete el proceso de indexación, también indexar el código JavaScript es más rápido; mayor velocidad y facilidad de configuración del intérprete remoto SSH e incluye un cliente REST construido para PyCharm 2017.3 que permite enviar una solicitud compleja para probar su funcionalidad (JetBrains).

1.5.4 Lenguajes, notaciones y tecnologías de programación

Lenguaje de Marcas de Hipertexto (HTML 5.0)

Es un lenguaje que se utiliza para crear documentos que muestren una estructura hipertexto. Además, permite crear documentos de tipo multimedia, es decir, que contengan información más allá de la simplemente textual como, por ejemplo: imágenes, videos, sonido, entre otros. Los documentos HTML se conforman como documentos de texto planos, en los que todo el formato del texto se especifica mediante marcas de texto (nombradas etiquetas), que delimitan los contenidos a los que afecta la etiqueta (Cera, 2015).

JavaScript 7

JavaScript es un lenguaje de programación interpretado, que puede usarse sin necesidad de adquirir una licencia que permita a los desarrolladores añadir y crear interactividad en el desarrollo y diseño de sitios web. No requiere de compilación puesto que el lenguaje funciona del lado del cliente y los navegadores son los encargados de interpretar estos códigos (Pérez, 2014).

Hojas de estilo en cascada (CSS) 3.0

Las hojas de estilo en cascada (*Cascading Style Sheets*), es la tecnología desarrollada por el *World Wide Web Consortium* (W3C) con el fin de separar la estructura de la presentación, permite crear páginas web de una manera más exacta, gracias a CSS el desarrollador es mucho más dueño de los resultados finales de la página, pudiendo hacer muchas cosas que no se podía hacer utilizando solamente HTML, como fuentes, colores, márgenes, líneas, altura, anchura, imágenes de fondo, posicionamiento avanzado y muchos otros temas (Pérez, 2014).

Python 3.5

Python es un lenguaje de programación multipropósito de alto nivel, su filosofía de diseño enfatiza la productividad del programador y la legibilidad del código. Tiene un núcleo sintáctico minimalista con unos pocos comandos básicos y simple semántica, pero además tiene una enorme y variada librería estándar, que incluye una Interfaz de Programación de Aplicaciones (API) para muchas de las funciones en el nivel del Sistema Operativo. El código Python, aunque minimalista, define objetos incorporados como listas enlazadas (list), tuplas (tuple), tablas hash (dict) y enteros de longitud arbitraria (long). Python soporta múltiples paradigmas de programación, incluyendo programación orientada a objetos (class), programación imperativa (def) y funcional (lambda). Python tiene un sistema de tipado dinámico y manejo automatizado de memoria utilizando conteo de referencias (similar a Perl, Ruby y Scheme). Python fue publicado por primera vez por Guido Van Rossum en 1991. El lenguaje tiene un modelo abierto de desarrollo basado en la comunidad, administrado por la organización sin fines de lucro Python Software Foundation. Existen varios intérpretes y compiladores que implementan el lenguaje Python, incluyendo uno en Java (Jython) (Rossum, 2009)**.**

1.5.5 Marcos de trabajo (Frameworks)

Un framework es una estructura conceptual y tecnológica de soporte, definida normalmente con artefactos o módulos de software concretos, en base a la cual otro proyecto de software puede ser organizado y desarrollado. Típicamente, puede incluir soporte de programas, bibliotecas, un lenguaje interpretado, entre otros programas para ayudar a desarrollar y unir los diferentes componentes de un proyecto (Pérez, 2014).

jQuery 3.2.1

Es un framework JavaScript rápido, pequeño y rico en funciones que permite la manipulación, manejo de eventos y animación mucho más simple que funciona a través de una multitud de navegadores. Es el framework más utilizado por su fácil manejo en el tratamiento de los objetos del Modelo de Objetos del Documento (DOM por sus siglas en inglés Document Object Model). Es un software libre y de código abierto permitiendo su uso en proyectos libres y privativos. Muy aconsejable para la integración de efectos y animaciones personalizadas (Duarte, 2013).

Ext JS 4.1

Aprovecha las características de HTML5 en los navegadores modernos. Cuenta con más de 115 componentes de interfaz de usuarios de alto rendimiento, aprobados e integrados que incluyen calendario, cuadrículas, gráficos y más. El amplio conjunto de herramientas y temas de Ext JS ayuda a mejorar la productividad de desarrollo y a acelerar la entrega de aplicaciones web de excelente apariencia. Las herramientas están disponibles para ayudar con el diseño, desarrollo, tematización y depuración de aplicaciones, así como también la optimización e implementación de compilación (sencha.com).

Django 2.0.2

Es un framework web de alto nivel que permite el desarrollo de sitios web rápido y seguro. Desarrollado por programadores experimentados, Django se encarga de gran parte de las complicaciones del desarrollo web. Es gratuito y de código abierto, tiene una comunidad próspera y activa, una gran documentación y muchas opciones de soporte gratuito y de pago. Cada lanzamiento ha añadido nuevas funcionalidades y solucionado errores, que van desde soporte de nuevos tipos de bases de datos, motores de plantillas, hasta la adición de funciones genéricas y clases de visualización (que reducen la cantidad de código que los desarrolladores tiene que escribir para numerosas tareas de programación). Django permite crear formularios para la recolección de datos, la autenticación y permisos de los usuarios, sitios de administración, socialización de datos, etc. (Moreno, 2014).

1.5.6 Servidor Web

Apache 2.2

De manera sintetizada, por lo descrito anteriormente acerca de Apache, se elige porque es un servidor web HTTP de código abierto, para plataformas *Unix, Windows, MAC OS* y otras, que implementa el protocolo HTTP/1.1 y la noción de sitio virtual, presenta entre otras características altamente configurables, bases de datos de autenticación y negociado de contenido.

1.5.7 Control de Versiones

Subversion 1.7

Es un controlador de versiones empleado en la administración de archivos utilizados en el desarrollo de software o contenido. CVS considerado su antecesor como uno de los controladores de versiones más

utilizados en proyectos de software libre, a pesar de su amplio uso, el mismo diseño resultó ineficiente para diversos grupos de usuarios y ante estas inconformidades se dio inicio al proyecto que hoy es conocido como Subversion (SVN), el mismo que ha empezado a socavar el dominio de CVS (Abalde, 2010).

1.5.8 Navegador Web

Mozilla Firefox 56 o superior

Es un navegador de Internet libre y de código abierto. Es usado para visualizar páginas web. Incluye corrector ortográfico, búsqueda progresiva, marcadores dinámicos y un sistema de búsqueda integrado que utiliza el motor de búsqueda que desee el usuario. Además, se pueden añadir funciones a través de complementos desarrollados por terceros. Firefox es uno de los navegadores más fácilmente personalizados con soporte para extensiones del mismo, *plugins* y temas que alteran fundamentalmente la función y apariencia del navegador para adecuarse mejor a tus necesidades (Guerrero, 2014).

1.6 Conclusiones del capítulo

En este capítulo se realizó un estudio sobre los principales conceptos relacionados al tema de la investigación. El análisis de las herramientas de administración de Apache y Subversion, ratificó la necesidad de desarrollar una solución integradora. Se determinó la guía de desarrollo, así como los lenguajes, tecnologías y herramientas definidas que respalden el desarrollo de una solución que cumpla con el objetivo propuesto.

Capítulo 2: Propuesta de solución

Introducción

En el presente capítulo se detalla el funcionamiento del sistema propuesto. Se confecciona un conjunto de artefactos correspondientes al modelado de los procesos de negocio. Se describen los requisitos que el sistema debe cumplir para satisfacer las necesidades del cliente.

2.1 Modelación de los Procesos del Negocio

La modelación de proceso de negocio permite realizar una exploración del dominio del problema, con el fin de lograr comprensión por parte del equipo de desarrollo de los procesos que se realizan actualmente en la entidad y la relación que existe entre estos. Modelar el proceso de negocio es una parte esencial de cualquier proceso de desarrollo de software. El modelado de procesos de negocio es la base para comprender mejor la operación de una organización, documentar y publicar los procesos buscando una estandarización en la organización, buscar eficiencias en la operación e integrar soluciones en arquitecturas orientadas a servicios (Sparx Systems, 2013).

2.1.1 Mapas de procesos del negocio

El mapa de procesos de negocio es una representación de los procesos de negocio, sus relaciones y sus interacciones. Un proceso de negocio es un conjunto de tareas relacionadas lógicamente que se llevan a cabo para lograr como resultado un negocio definido. Los procesos describen como es realizado el trabajo en la organización y se caracterizan por ser observables, medibles, mejorables y repetitivos (Aiteco, 2013).

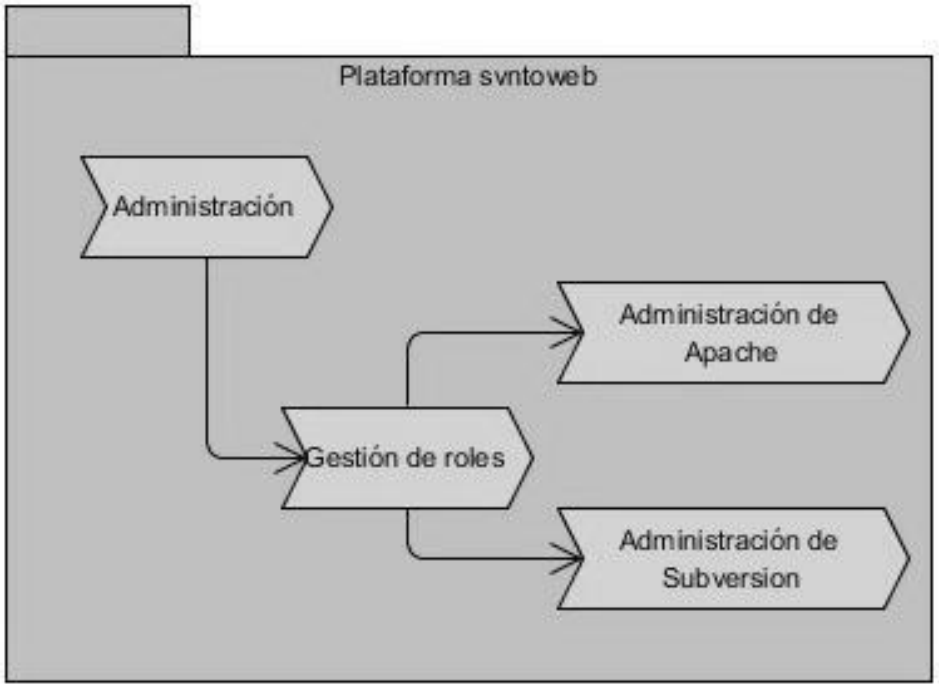


Figura 2: Mapa de proceso del negocio Administración de Apache y Subversion

2.1.2 Descripción del proceso de negocio

La descripción de los procesos de negocio debe ser precisa y de fácil comprensión. En la misma se deben especificar las restricciones, donde son definidas las reglas del negocio que rigen la actividad. Éstas describen las políticas, normas, operaciones, definiciones y restricciones presentes en una organización y que son de vital importancia para alcanzar los objetivos (UCID, 2009).

A continuación, se detalla el funcionamiento del proceso de negocio: Administración.

Una vez que un usuario es autenticado en la aplicación, según su rol, será el modo de administración que pueda realizar. El sistema tiene determinada jerarquía, algunas personas podrán ejecutar tareas que otros no. En caso de que el rol sea Administrador, puede realizar cualquier acción en el sistema, además de crear usuarios y asignarles que rol desempeñará, seleccionando entre Gestor de Configuración o Arquitecto, otorgándoles los permisos respectivos. Cuando a determinado sujeto le corresponda ser Gestor de Configuración efectuará gran parte del proceso, desde la administración de Apache, ya sea la gestión de VirtualHosts, hasta la administración de Subversion con la gestión de repositorios, el otorgamiento de

permisos de lectura y escritura y la publicación de un repositorio en un VirtualHost. Por último, si es Arquitecto, sus funciones serán chequear los registros y restaurar alguna salva de un repositorio.

Tabla 1: Descripción del proceso de negocio Administración de la plataforma web

Objetivos	Realizar la descripción del proceso de administración en la plataforma.
Evento(s) que lo generan	No procede.
Precondiciones	El usuario debe estar autenticado.
Post-condiciones	El proceso de administración empieza a realizarse dependiendo del rol realizado.
Marco Jurídico	No procede.
Clientes Internos	Administrador, Gestor de Configuración, Arquitecto.
Clientes Externos	No procede.
Entradas	Necesidad de un usuario para la administración de Apache y Subversion.
Salidas	Necesidad cumplida y usuario satisfecho.

2.1.3 Diagrama del proceso de negocio

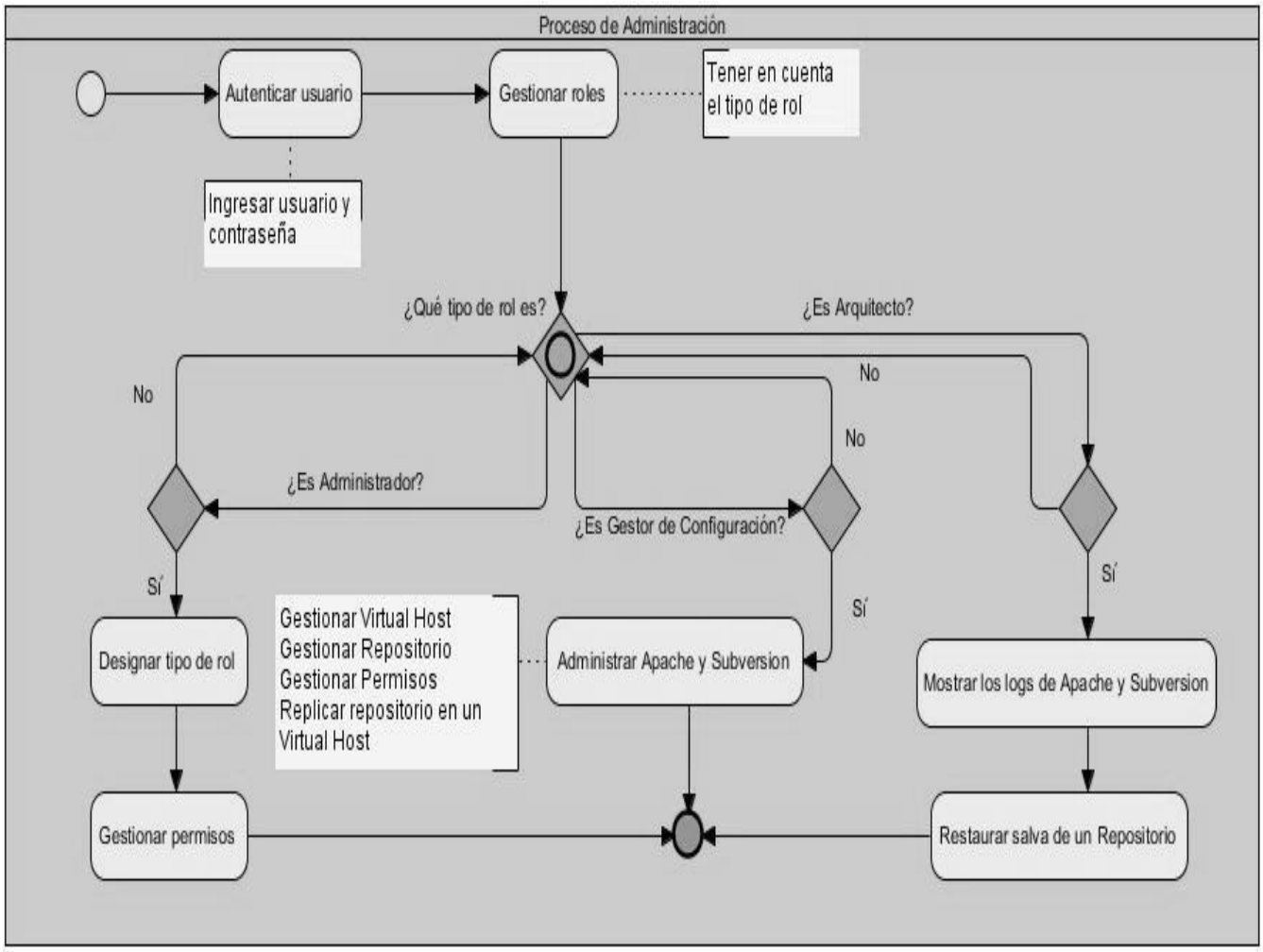


Figura 3: Diagrama del proceso de negocio Administración en la plataforma web

2.2 Modelo conceptual

El modelo conceptual explica los conceptos más significativos en el dominio del problema, teniendo como cualidad esencial representar elementos del mundo real. Un concepto en el Modelo de Dominio o Conceptual es un elemento lógico o físico que ayuda a entender el problema, es parte del lenguaje utilizado por el cliente y generalmente se nombra como sustantivo. Los conceptos pueden o no tener atributos que lo caractericen en el mundo real (UCID, 2009).

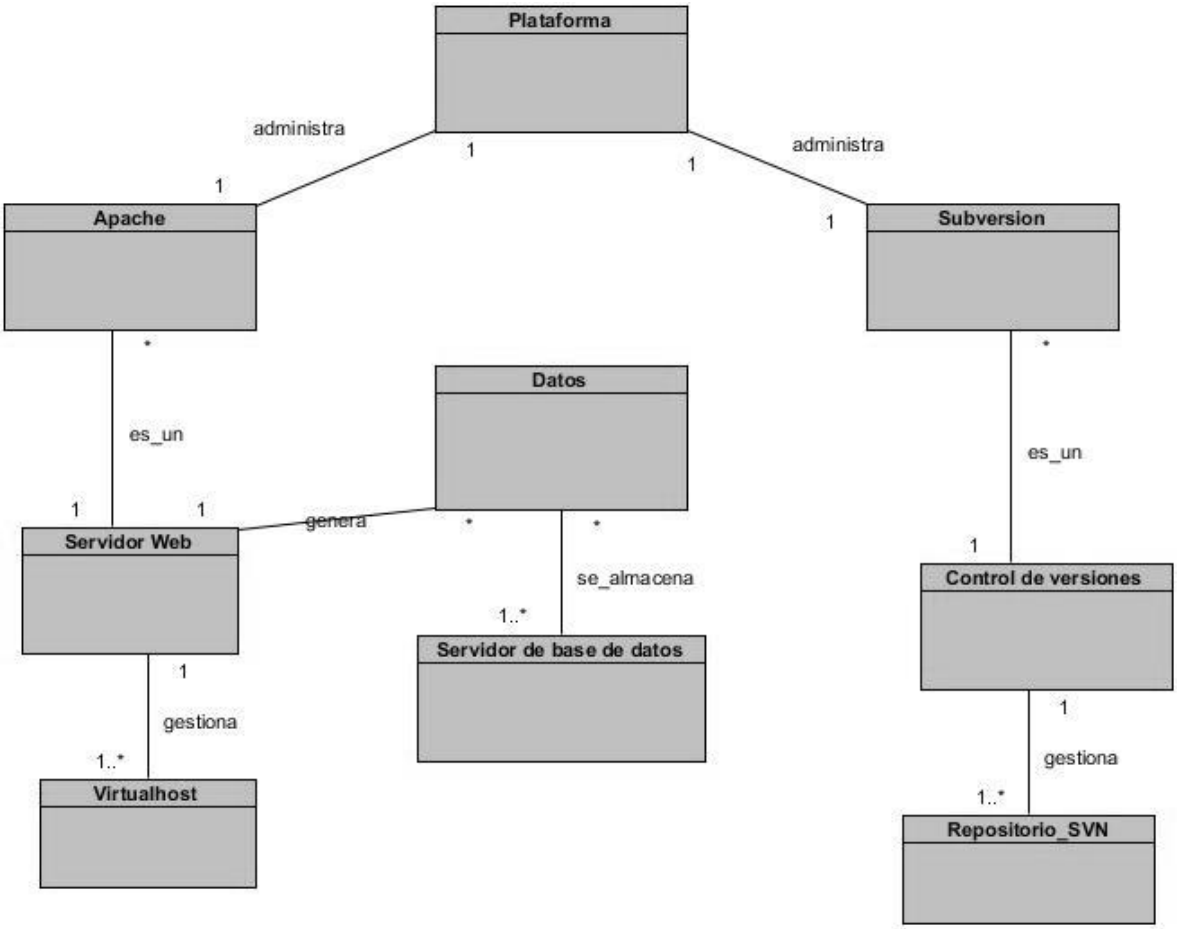


Figura 4: Modelo conceptual Administración de servidores Apache y Subversion

Tabla 2: Tabla de conceptos

Conceptos	Descripción
Servidor de Base de Datos	Servidor donde se almacena toda la información generada.
Servidor web	Servidor para realizar publicaciones en la web.
Plataforma	Aplicación web que se encargará de visualizar el funcionamiento del sistema instalado.
Datos	Es una representación simbólica (numérica, alfabética, algorítmica, espacial, etc.) de un atributo o variable cuantitativa o cualitativa.

Subversion	Control de versiones para administrar (Gestión de repositorios, mostrar los *logs*, publicar repositorios, etc.).
Apache	Servidor web para administrar (Gestión de VirtualHost, mostrar los *logs*, etc.).
Repositorio_SVN	Tipo de almacén donde se guardan los proyectos y las actualizaciones realizadas a sus componentes.
VirtualHost	Archivo de configuración para realizar publicaciones mediante Apache.
Control de Versiones	Gestiona los cambios que se le realizan a un archivo o grupo de archivos.

2.3 Propuesta de solución

Se propone desarrollar una plataforma web que garantice la administración de forma integrada de servidores Apache y Subversion. Se automatiza el despliegue de todos los componentes, a partir de un instalador, facilitando los procesos de instalación y configuración de la plataforma, reduciendo el trabajo mediante líneas de comando y el margen de errores. A través del navegador web, se realizan los procesos de administración según el rol del usuario autenticado, mediante interfaces gráficas intuitivas y con un registro de las acciones realizadas, que ayudan a reducir los tiempos de desarrollo e incrementan el trabajo colaborativo.

2.4 Definición de los requisitos de software

El propósito de la definición de requisitos es especificar las condiciones o capacidades que el sistema debe cumplir y las restricciones bajo las cuales debe operar, logrando un entendimiento entre el equipo de desarrollo y el especialista funcional y especificando las necesidades reales de forma que satisfaga sus expectativas (UCID, 2009).

2.4.1 Técnicas empleadas para la captura de requisitos

- **Entrevista:** Se le realizan entrevistas a los participantes del proyecto e interesados, mediante preguntas previamente elaboradas que cubran todos los aspectos del proyecto, para posteriormente obtener definiciones de riesgos que luego serán analizadas (UCID, 2009).

- **Tormenta de ideas:** Mediante esta técnica de trabajo en grupo, los miembros del equipo de desarrollo realizan rondas donde en cada una se propone una idea, siendo analizadas al final (UCID, 2009).

2.4.2 Requisitos funcionales

Los requisitos funcionales definen las condiciones o capacidades que el sistema será capaz de realizar. Estos describen las transformaciones que el sistema realiza sobre las entradas para producir salidas. Estos requisitos, al tiempo que avanza el proyecto de software, se convierten en los algoritmos, la lógica y gran parte del código del sistema (UCID, 2009). A continuación, se especifican los requisitos funcionales que responden a la propuesta solución.

Tabla 3: Requisitos Funcionales

No	Requisitos Funcionales
1	Autenticar usuario
Rol	
2	Adicionar rol
3	Modificar rol
4	Eliminar rol
5	Buscar rol
Publicar un repositorio en un VirtualHost	
6	Adicionar publicación
7	Buscar publicación
8	Eliminar publicación
9	Modificar publicación
10	Mostrar los *logs* de las acciones de Apache
Permisos por la web	
11	Adicionar permisos
12	Modificar permisos
13	Eliminar permisos

Repositorio	
14	Adicionar repositorio
15	Eliminar repositorio
16	Buscar repositorio
17	Realizar salvas a un repositorio
18	Restaurar una salva de un repositorio
VirtualHost	
19	Adicionar VirtualHost
20	Eliminar VirtualHost
21	Buscar VirtualHost
22	Cambiar el puerto de un VirtualHost de Apache
23	Habilitar un VirtualHost de Apache
24	Deshabilitar un VirtualHost de Apache
25	Autenticar por nombre en un VirtualHost de Apache
26	Autenticar por dirección (IP) en un VirtualHost de Apache
27	Confeccionar el instalador de la aplicación

2.4.3 Especificación de los requisitos funcionales

Contiene los conceptos tratados, precondiciones y post-condiciones, una descripción detallada del flujo del requisito, las validaciones, la complejidad, prioridad del requisito y una propuesta de los prototipos de interfaz de usuario.

A continuación, se muestra la descripción del requisito funcional Adicionar VirtualHost. El resto de las especificaciones del requisito *"Gestionar VirtualHost"* y sus prototipos de interfaz de usuario se ubican en el **Anexo 1**.

Tabla 4: Especificación del requisito "Adicionar VirtualHost"

Conceptos tratados	Conceptos	Atributos

	VirtualHost	id, nombre_vh, fecha_creacion, IP, puerto, ssl, DocumentRoot, logs, servername, serveralias.
Precondiciones	**Precondiciones**	**Pre-requisito**
	1. El usuario debe estar autenticado.	1. Autenticar usuario.
Descripción	1. Se ubica el **Menú de Opciones/Apache**. 2. Se despliega el árbol de Apache y se hace clic en la opción VirtualHost. 3. Se muestra la interfaz de VirtualHost. 4. Se selecciona el botón Adicionar. 5. Se muestra la ventana con los campos para Adicionar VirtualHost. 6. Se llenan los campos obligatorios y en dependencia los campos opcionales. 7. Se presiona el botón Guardar si están completos y validados los campos. 8. Se guardan los cambios realizados en el sistema. 9. Concluye el requisito	
Complejidad	Media	
Validaciones	6.1 El sistema le hace saber al usuario los campos obligatorios. 7.1 Si el usuario ingresa caracteres incorrectos en determinados campos, el sistema le notifica.	
Post-condiciones	No procede.	

Post-requisito	No procede.

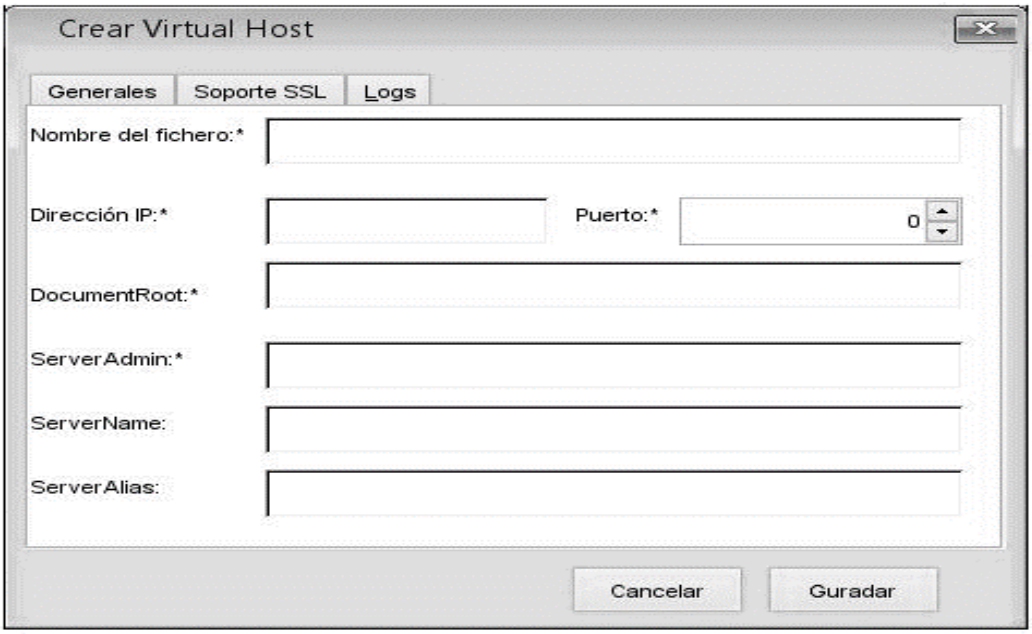


Figura 5: Prototipo de interfaz de usuario

2.4.4 Requisitos no funcionales

Los requisitos no funcionales son propiedades o cualidades que el producto debe tener. Debe pensarse en estas propiedades como las características que hacen al producto atractivo, usable, rápido o confiable. En muchos casos los requisitos no funcionales son fundamentales en el éxito del producto. Normalmente están vinculados a requisitos funcionales, es decir, una vez se conozca lo que el sistema debe hacer se puede determinar cómo ha de comportarse, qué cualidades debe tener o cuán rápido o grande debe ser (UCID, 2009).

Requisitos de Usabilidad

- Se requiere que el sistema sea sencillo, brindando un acceso rápido y fácil a todas las funcionalidades, permitiendo una adecuada interacción con el usuario.
- La aplicación solo podrá ser utilizada por los usuarios que estén registrados en el sistema.

Requisitos de Interfaz

- La aplicación debe tener una única vista donde todas las acciones que se ejecuten deben realizarse ahí mismo.
- Mantendrá en todo momento un diseño sencillo, sin uso excesivo de imágenes y gráficos, con el objetivo de elevar el nivel de respuesta del sistema ante las peticiones del usuario.

Requisitos de Soporte

- Se debe confeccionar el instalador de la aplicación en un único archivo con extensión .iso, .sh[6] o **.**run[7].

Requisitos de Software

- Las PC cliente deben tener instalado el navegador web Mozilla Firefox 56 en lo adelante y como sistema operativo Linux o Windows.
- Se debe emplear herramientas como Apache y Subversion en sus versiones 2.2 y 1.7 respectivamente.
- Se utiliza una base de datos implementada en PostgreSQL versión 9.4 o superior.

Requisitos de Hardware

Para garantizar un buen funcionamiento del sistema, el servidor donde estará desplegado el sistema debe contar con:

- RAM: 4GB
- CPU: 4 núcleos a 2.4Gz
- Almacenamiento: 30GB

Requisitos de Confiabilidad

- Para tener un mejor conocimiento de las acciones que se ejecutan en el sistema, deberán mostrarse en el panel de registros (*logs*) las mismas.
- A ciertos usuarios se les otorgan determinados permisos especiales, mientras que a otros se les restringe para garantizar la integridad de los datos.

[6] Son archivos implementados en el lenguaje BASH, en su interior cuentan con una secuencia de comandos a ejecutar.

[7] Son archivos para instalar aplicaciones desarrolladas para el sistema operativo Linux.

2.4.5 Validación de los requisitos

La validación de requisitos examina las especificaciones para asegurar que todos los requisitos del sistema han sido establecidos sin ambigüedad, inconsistencia, sin omisiones; que los errores detectados hayan sido corregidos y el resultado del trabajo se ajuste a los estándares establecidos para el proceso, el proyecto y el producto (Pressman, 2010).

Par la validación de los requisitos existen algunas técnicas, por ejemplo:

- Revisiones
- Auditorías
- Matrices de trazabilidad
- Prototipos

A continuación, se explican las técnicas de validación de requisitos empleadas en la investigación:

- ✓ Prototipos: A partir de la definición de los requisitos, se realizan prototipos que, sin tener el completo funcionamiento del sistema, permitan hacerse idea de la estructura de la interfaz del sistema, así como su funcionamiento.
- ✓ Revisiones: Se van realizando revisiones a los requisitos con el objetivo de verificar que cumplan con la calidad requerida, en otras palabras, que sean consistentes y no ambiguos.

2.5 Arquitectura del sistema

La **IEEE**[8] define que arquitectura de software es “la organización fundamental de un sistema encarnada en sus componentes, las relaciones entre ellos y el ambiente y los principios que orientan su diseño y evolución.”

2.5.1 Patrón arquitectónico Modelo-Vista-Controlador (MVC)

Es un estilo arquitectónico usado principalmente en aplicaciones que manejan gran cantidad de datos y transacciones complejas donde se requiere una mejor separación de los conceptos para que el desarrollo

[8] International Electrical and Electronics Engineers

esté estructurado adecuadamente. Facilita la programación en diferentes capas de forma paralela e independiente. MVC sugiere la separación del software en 3 estratos:

- ✓ **Modelo:** Es la representación de la información que maneja la aplicación. El modelo en sí son los datos puros que puestos en un contexto del sistema proveen de información al usuario o a la aplicación misma.
- ✓ **Vista:** Es la representación del modelo en forma gráfica disponible para la interacción con el usuario. En el caso de una aplicación web la "Vista" es la página HTML con contenido dinámico sobre el cual el usuario puede realizar operaciones.
- ✓ **Controlador:** Es la capa encargada de manejar y responder las solicitudes del usuario, procesando la información necesaria y modificando el Modelo en caso de ser necesario (Fernández, 2012).

El marco de trabajo Django utiliza una variante de patrón MVC conocido como Modelo-Vista-Plantilla por sus siglas MTV. Una analogía entre el MVC y el MTV es:

- El modelo sigue siendo modelo.
- La vista en Django se llama Plantilla (*Template*).
- El controlador en Django se llama Vista (View).

Representación del MTV:

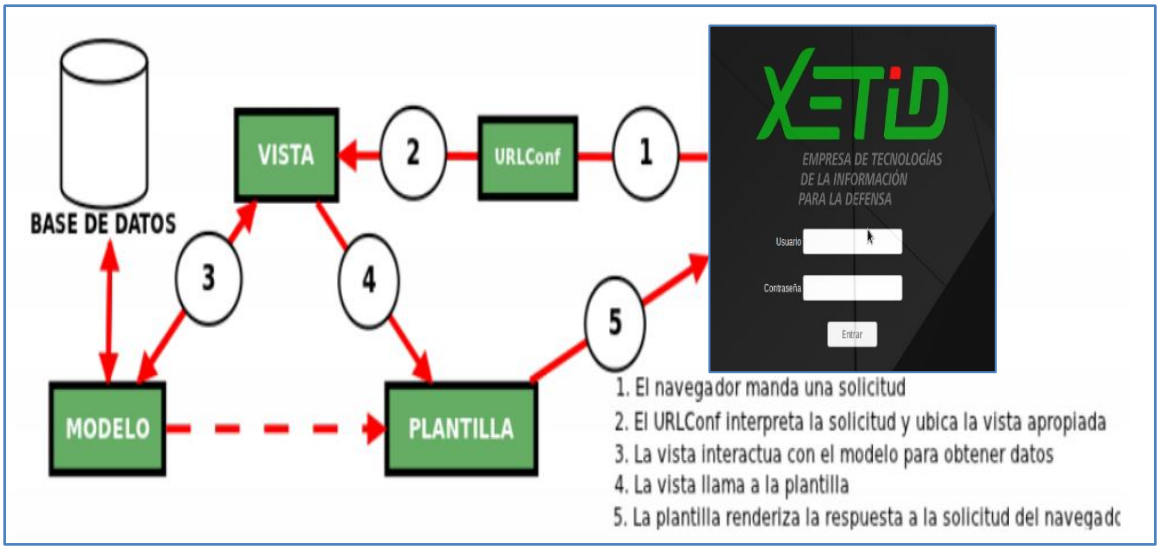


Figura 6: Funcionamiento del MTV de Django

Descripción del MTV:

URLConf (urls.py): Constituye una particularidad de MTV que lo diferencia del MVC. Permite controlar el despliegue de las vistas. Se encarga de leer la URL que el usuario solicitó, encontrar la vista apropiada para la solicitud y pasar cualquier variable que la vista necesite para completar su trabajo.

Modelo (models.py): Es la capa de acceso a la base de datos. Se encarga de encapsular toda la lógica de negocio de la aplicación. Contiene toda la información sobre los datos: cómo acceder a estos, cómo validarlos, cuál es el comportamiento que tiene y las relaciones entre los datos.

***Template* (fichero.html):** Es la capa de presentación (Vista). Es la respuesta de cada controlador y lo que se le presenta al usuario final, se puede comunicar con el controlador.

Vista (view.py): Es la capa de la lógica de negocios (Controlador). Constituye el eje central de la arquitectura, encargada de gestionar todas las peticiones, validar los datos recibidos y dirigir cualquier petición de cualquier tipo. Esta capa contiene la lógica que accede al modelo y la delega a la plantilla apropiada: se puede pensar en esto como un puente entre el modelo y las plantillas (Holovaty, 2008).

2.5.2 Modelo de datos

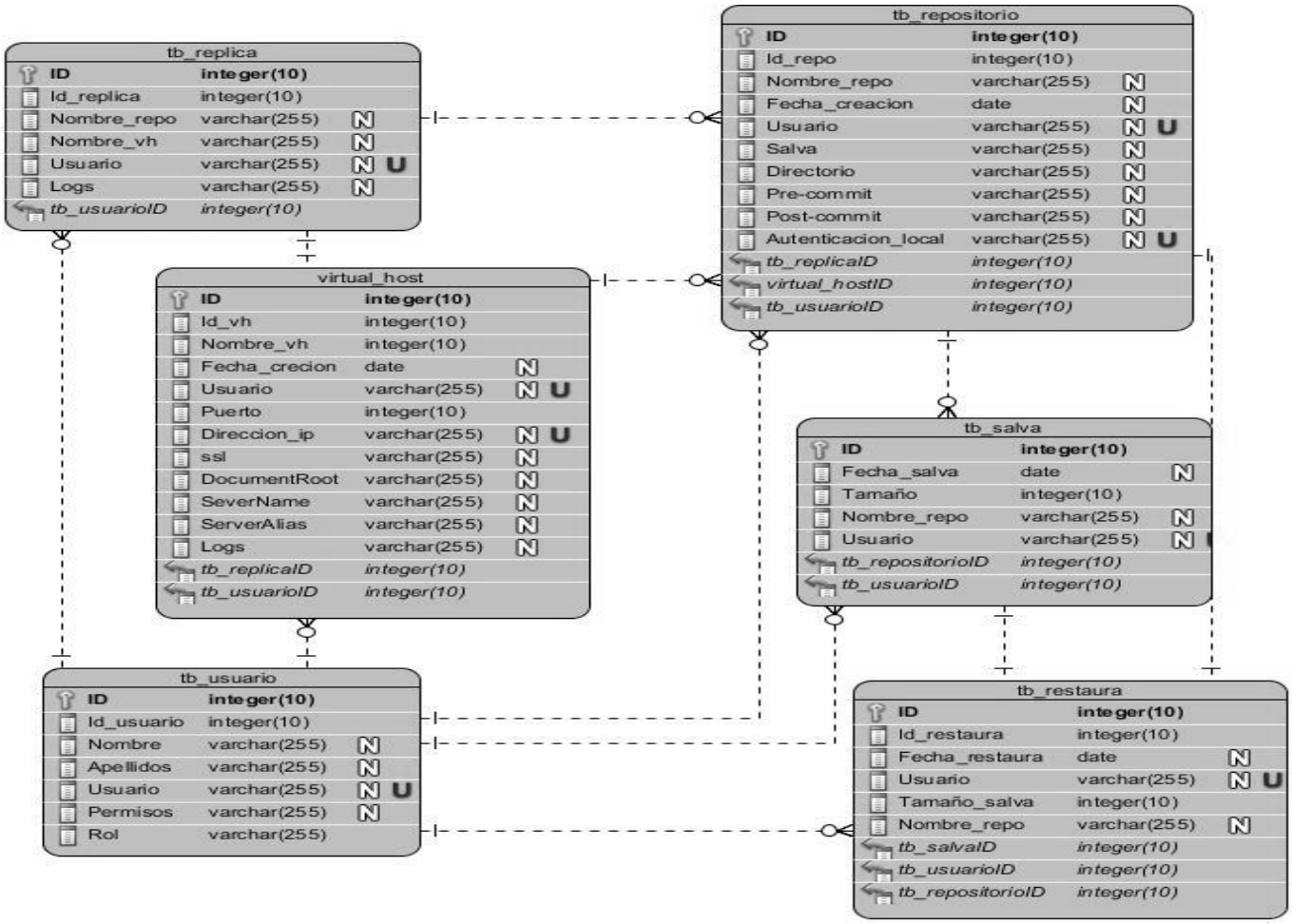


Figura 7: Modelo de datos

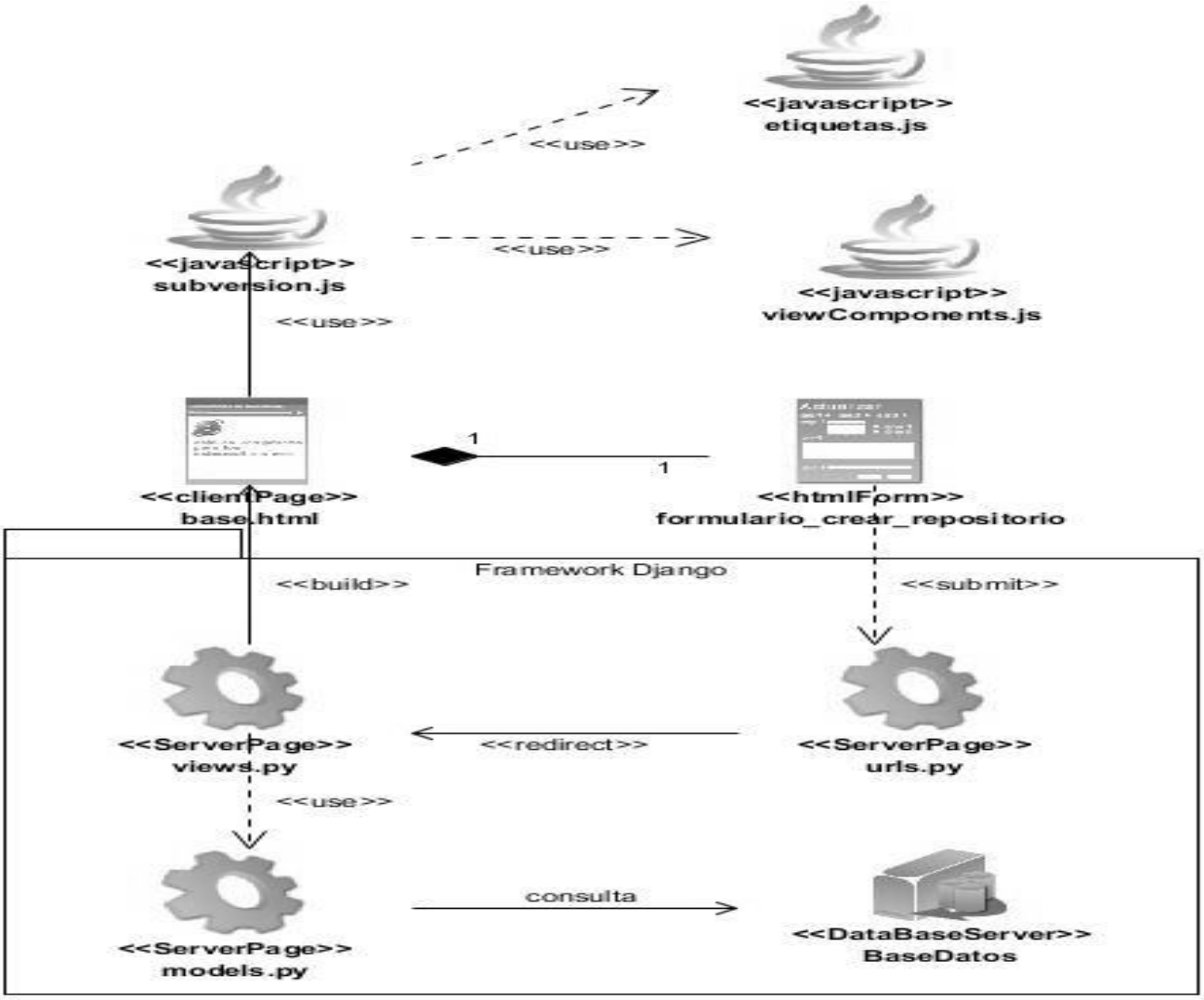
2.6 Diseño del sistema

En el diseño del sistema se realiza una modelación de los artefactos correspondientes, entre los cuales se encuentra el modelo de diseño.

2.6.1 Diagrama de clases del diseño

El diagrama de clases del diseño de la plataforma web describe la estructura del mismo, mostrando las clases y las relaciones entre ellas. Para un mayor entendimiento del comportamiento de la capa de presentación del subsistema se utilizaron estereotipos web. A continuación, se muestra dicho diagrama

correspondiente al requisito funcional: Crear Repositorio. El resto de los diagramas se encuentra en el **Anexo 2**.



Descripción de las clases:

Clase<<subversion.js>>: Contiene el código JavaScript necesario para realizar todas las validaciones de los formularios del lado del cliente.

Clase<<etiquetas.js>>: Define los textos que se usan en la aplicación.

Clase <<viewComponents.js>>Contiene los componentes genéricos que se usan en la aplicación.

Clase<<base.html>>: Contiene la vista principal con la cual interactúa el usuario.

Clase<<urls.py>>: Constituye el punto de entrada principal de todas las peticiones web.

Clase<<views.py>>: Se refiere al controlador, que es la capa que se encarga del negocio. Esta capa interactúa con el modelo para obtener los datos**.**

Clase<<models.py>>: Contiene los modelos de la aplicación, los cuales representan tablas de la base de datos.

2.6.2 Diagrama de clases persistentes

Teniendo en cuenta lo descrito en el modelo del dominio, se identificaron las clases que tendrán durabilidad en el tiempo, obteniéndose como resultado el diagrama de clases persistentes.Diagrama de interacción

Un diagrama de interacción muestra los mensajes que se envían como respuesta a la invocación de un método. La secuencia de estos mensajes se traduce en una serie de sentencias en la definición del método.

Los diagramas de interacción representan la forma en cómo un cliente u objeto se comunican entre sí en petición a un evento determinado. Existen dos tipos de diagramas de interacción: diagrama de colaboración y diagrama de secuencia, este último es el empleado para diseñar cómo será el flujo de mensajes entre los objetos. Muestra la interacción de un conjunto de objetos en una aplicación a través del tiempo. Este tipo de diagrama contiene detalles de implementación del escenario, incluyendo los objetos y clases que se usan para implementar el mismo y los mensajes intercambiados entre los objetos.

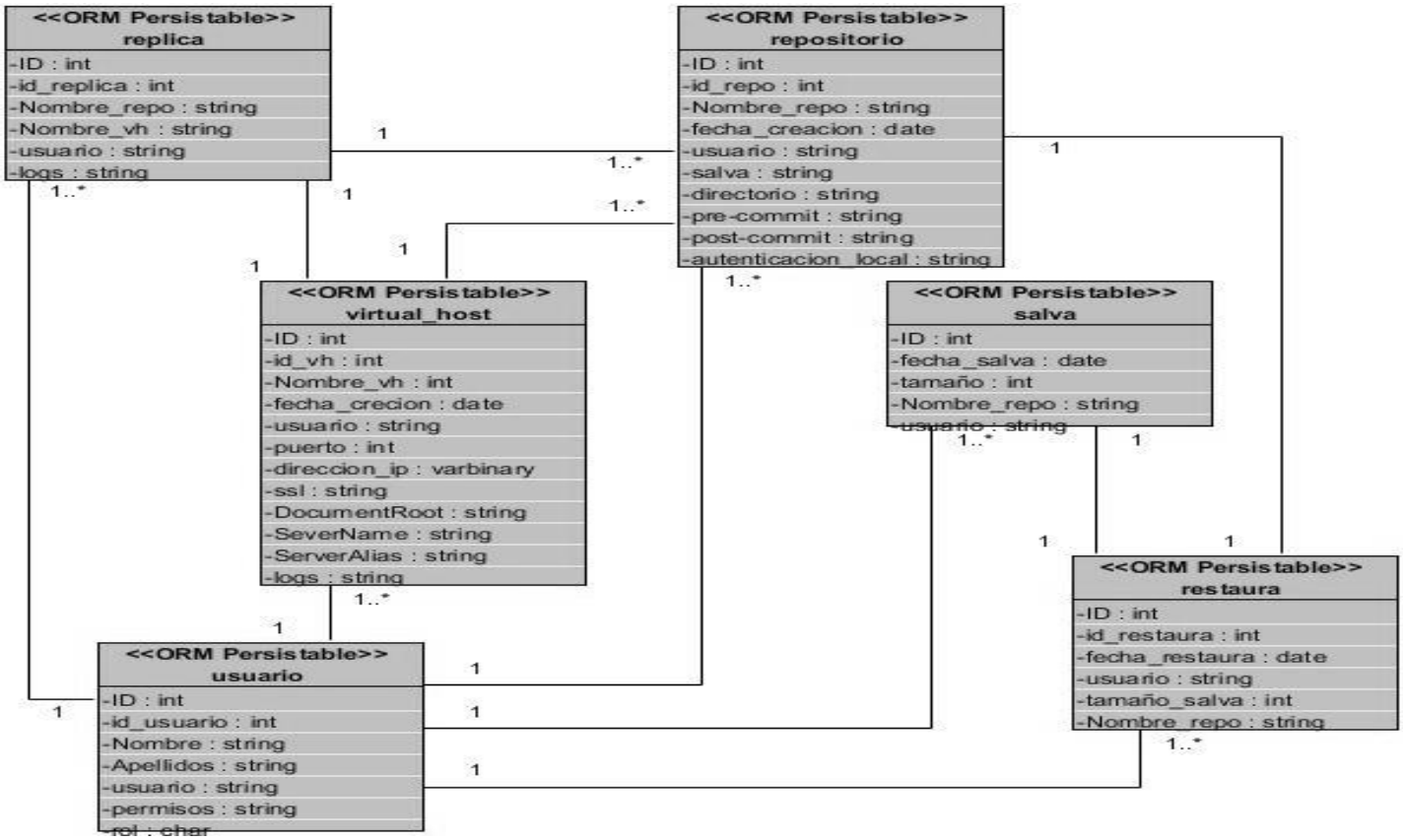


Figura 9: Diagrama de clases persistentes

A continuación, se muestra el diagrama de secuencia del requisito funcional Crear Repositorio.

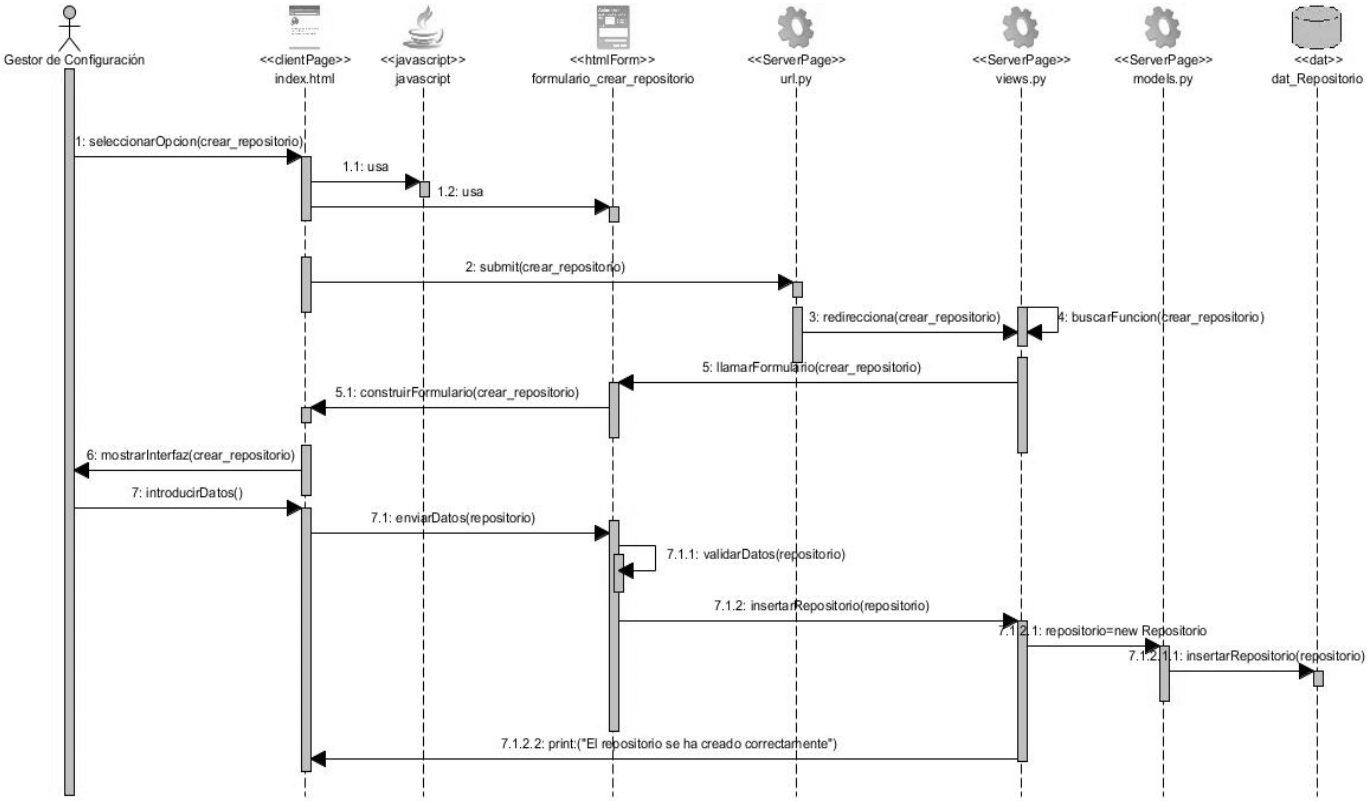


Figura 10: Diagrama de secuencia Crear Repositorio

2.6.3 Diagrama de componentes

Un diagrama de componentes ilustra los fragmentos de software, controladores embebidos, etc. que conformarán un sistema. Un diagrama de componentes tiene un nivel de abstracción más elevado que un diagrama de clase, usualmente un componente se implementa por una o más clases (u objetos) en tiempo de ejecución.

A continuación, la figura muestra el diagrama de componentes:

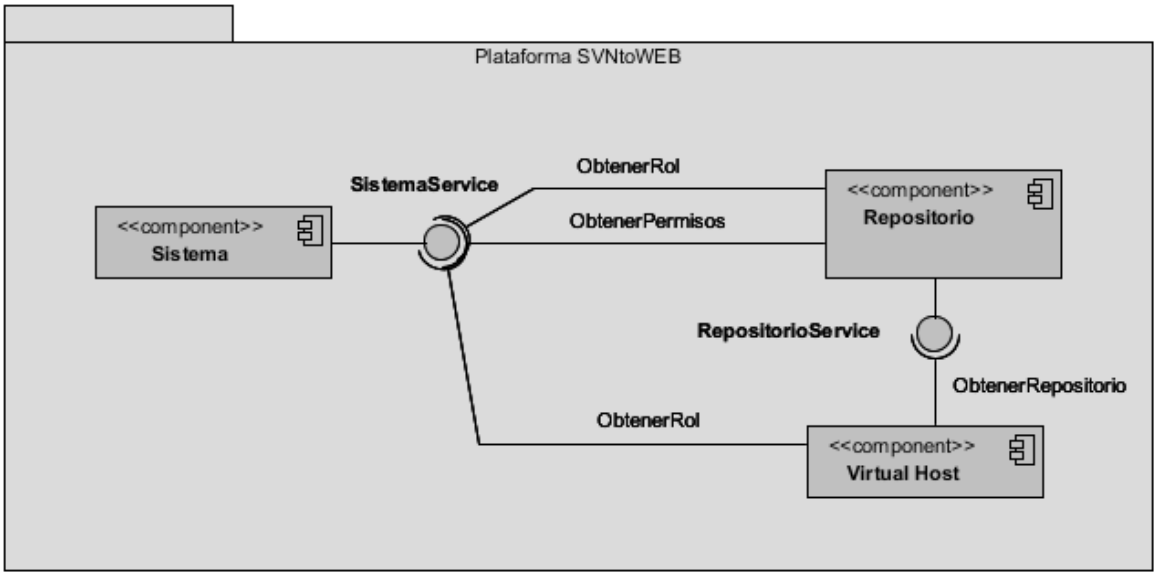


Figura 11: Diagrama de componentes

2.6.4 Patrones de diseño

Un patrón es un par problema/solución con nombre que se puede aplicar en nuevos contextos, con consejos acerca de cómo aplicarlo en nuevas situaciones y discusiones sobre sus compromisos. La mayoría de estos patrones son impuestos por los *frameworks* seleccionados por el equipo de arquitectura, se explican a continuación los más relevantes aplicados en el módulo en cuestión, que se encuentran dentro de los Patrones Generales de Software de Asignación de Responsabilidades (GRAPS), los cuales describen los principios fundamentales de la asignación de responsabilidades.

Patrones GRASP

Los parones generales de software para asignar responsabilidades (GRASP por sus siglas en inglés General Responsability Asignment Software Patterns), describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en forma de patrones (Larman, 1999).

Descripción de los patrones utilizados

Experto: Este patrón es el encargado de asignar una responsabilidad a la clase que tiene la información necesaria para cumplir una actividad específica. (Larman, 1999). Se pone de manifiesto dicho patrón en las clases modelos de los diferentes módulos. En la clase *VirtualHost* se evidencia el patrón experto, contiene toda la información y las funcionalidades necesarias para crear uno de estos.

Creador: Se le asigna a la clase B la responsabilidad de crear una instancia de la clase A, ya sea agregarle objetos, contenerlos, registrar instancias de los objetos, utilizar específicamente los objetos de la clase A, además la clase B tiene los datos de inicialización que serán transmitidos a A cuando sea creado. (Larman, 1999). Se aplica en las clases controladoras, las cuales son las encargadas de crear objetos de diferentes clases modelos.

Controlador: Se asigna la responsabilidad del manejo de un mensaje de los eventos de un sistema a una clase. Un controlador es un objeto de interfaz no destinado al usuario que se encarga de manejar un evento del sistema, además define el método de su operación (Larman, 1999). Este patrón se evidencia en la clase *views.py* y es quien delega las funcionalidades.

Bajo acoplamiento: Es tener las clases lo menos ligadas entre sí, en caso de producirse una modificación en alguna de ellas, haya la mínima repercusión posible en el resto de las clases, reforzando la reutilización y disminución de dependencia entre las clases. Se evidencia en la estructura misma del framework Django (*models*, *view, template*).

Patrones GoF

Decorator (Decorador): Es un patrón de tipo estructura, permite que clases y objetos sean utilizados para componer estructuras de mayor tamaño. Este patrón añade dinámicamente nuevas responsabilidades a un objeto. Django contienen un decorador que permite agregar funcionalidades dinámicamente a las aplicaciones desarrolladas bajo sus principios. De *"base.html"* es de donde heredan las demás plantillas, siendo esta la plantilla contenedora de la estructura y del diseño básico.

2.7 Prototipo de interfaz de usuario funcional

A continuación, se muestra el prototipo de interfaz de usuario funcional del requisito Adicionar VirtualHost que se obtuvo como resultado del diseño del sistema (Ver Figura 11).

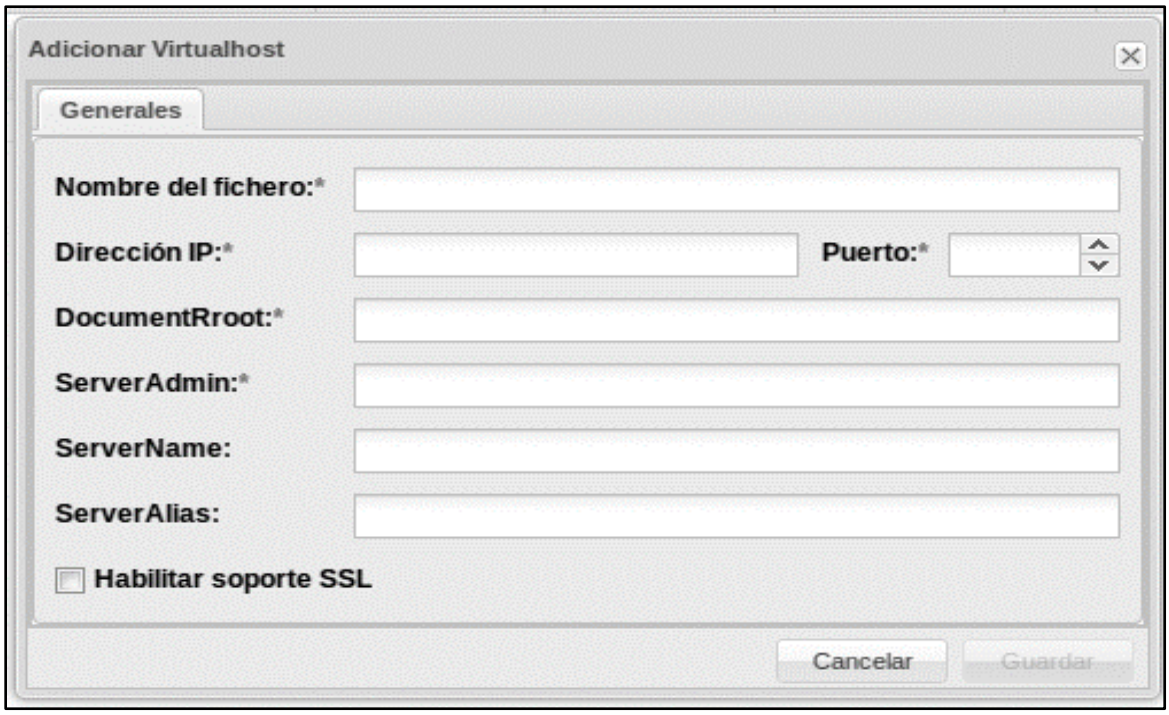


Figura 12: Prototipo de Interfaz de Usuario Funcional RF Adicionar VirtualHost

El resto de los prototipos de interfaz de usuario funcionales se pueden ver en el **Anexo 3**.

2.8 Conclusiones del capítulo

En este capítulo quedan definidos los artefactos, mapa de procesos, el modelo conceptual y los diagramas referentes a los procesos de negocio relacionados a la plataforma web, identificando los requerimientos funcionales y no funcionales del sistema. Se diseñó la arquitectura del sistema a partir de la especificación de las clases correspondientes a las capas de presentación, negocio, acceso a datos y datos. Se definieron los patrones de diseño utilizados en la solución planteada y se diseñaron los diagramas establecidos, permitiendo la implementación de los componentes que conforman la plataforma web.

Capítulo 3: Implementación y evaluación

En este capítulo se define el diagrama de despliegue para representar físicamente sus componentes y como se relacionan. Se presentan los estándares de codificación, citando ejemplos mediante ilustraciones. Además, se plantean las pruebas a realizar y sus resultados para la evaluación de la solución.

3.1 Diagrama de despliegue

Un diagrama de despliegue representa la distribución física del sistema en términos de cómo se distribuirán las funcionalidades entre los nodos, cada nodo representa un recurso de cómputo, siendo estos procesadores o dispositivos hardware que se necesitarán para el despliegue del sistema.

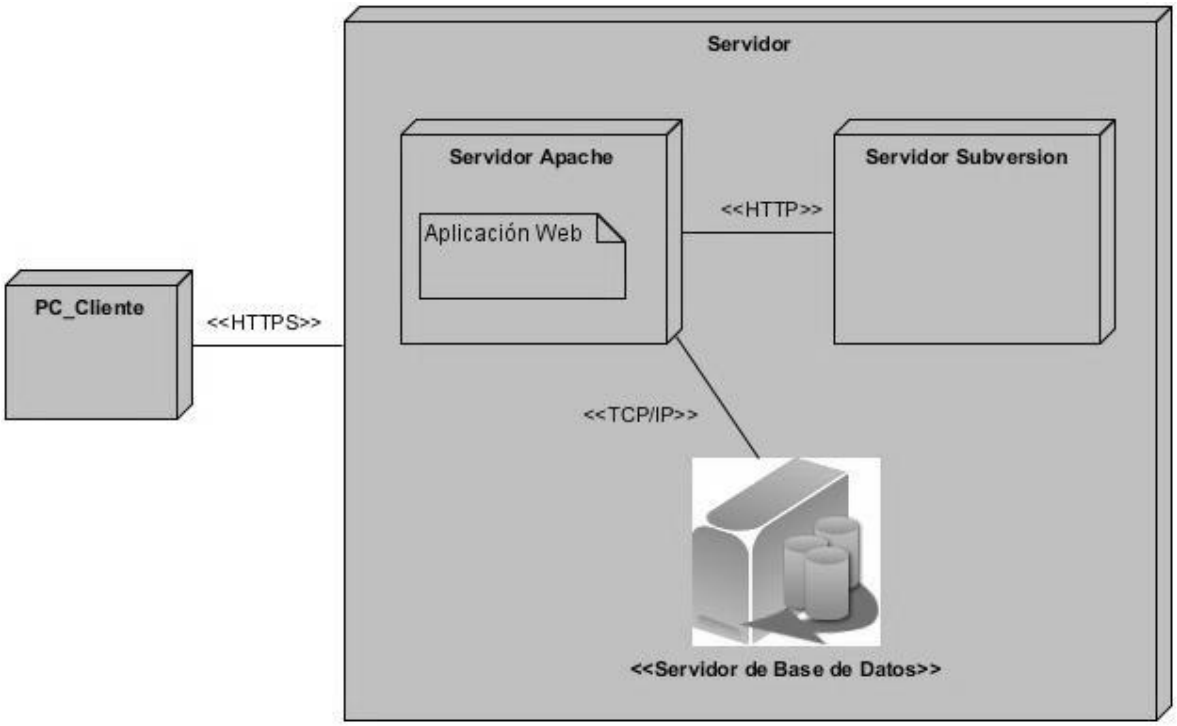


Figura 13: Diagrama de despliegue

PC_Cliente: computador en el cual el cliente ejecutará la aplicación a través de un navegador web.

Servidor: Es el servidor donde se encuentra alojada la plataforma web junto a los servidores Apache y Subversion.

Servidor Apache: Se encarga de atender y responder todas las peticiones realizadas por el usuario desde la PC_Cliente.

Servidor Subversion: En este servidor se encuentra el control de versiones Subversion, encargado de llevar el seguimiento de las versiones.

Servidor de Base de Datos: Este servidor es el encargado del almacenamiento de los datos del sistema. Se comunica con el servidor que ejecuta la plataforma web, posibilitando el acceso mediante el usuario con privilegios para las operaciones determinadas a realizarse en el mismo.

3.2 Implementación

El objetivo principal de este es desarrollar el diseño de la arquitectura propuesta y el sistema como un todo. De forma más específica, los propósitos de la Implementación son: Planificar las integraciones del sistema necesarias en cada iteración, siguiendo para ello un enfoque incremental; implementar clases, componentes y subsistemas encontrados durante el diseño; integrar componentes.

3.2.1 Estándares de codificación

Los estándares de codificación tienen como objetivo establecer estilos de códigos sólidos con vistas de que un proyecto de software se convierta en un producto fácil de comprender y de mantener. En el documento Estándar de Codificación para Python v1.0, elaborado en la Empresa de Tecnologías de la Información para la defensa XETID, establece pautas de codificación atendiendo a las particularidades del lenguaje de programación Python con el fin de utilizar las mismas técnicas, definiciones y nomenclaturas en la construcción del software (González, 2013).

3.2.1.1 Las Vistas

En la siguiente figura se muestra la estructura de carpetas de las interfaces de usuario de la plataforma web realizadas con Ext JS. Se aprecia el Modelo-Vista-Controlador, aunque no todos sus componentes, el Modelo se encuentra en el fichero models.py.

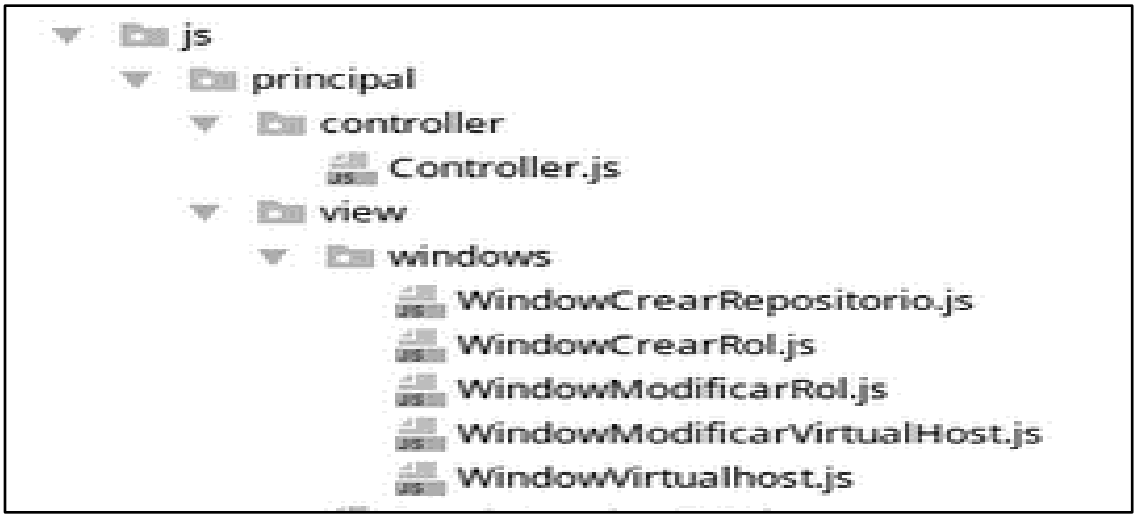


Figura 15: Estructura de carpeta de las interfaces de usuario

```
Ext.define('Principal.principal.view.windows.WindowCrearRepositorio', {
    extend: 'Ext.window.Window',
    title: 'Crear Repositorio',
    alias:'widget.windowcrear_repo',
    closable: true,
    resizable: false,
    width: 370,
    layout: {"type": 'fit'...},
    initComponent: function () {
        var self = this;
        var required='<span style="color: red;">*</span>'
        self.items = [
            {
                xtype: 'form',
                frame: true,
                layout:'anchor',
                defaults: {
                    anchor: '100%',
                    labelWidth:155,
                    margin:'10 0 5 5'
                },
                items: [
                    {"xtype": 'textfield'...},
                    {"xtype": 'textfield'...}
                        ],
                 buttons:[
                        {
                            text:'Cancelar',
                            handler:function(){
                                self.close()
                            }
                        },
                        {
                            text:'Guardar',
                            formBind:true       }]}];

        self.callParent();
    }
})
```

Figura 14: Ejemplo de código de una clase vista

Clases Controladoras

En la siguiente figura se muestra la estructura de carpetas de la clase controladora views.py utilizada por el *framework* Django, encargada de atender la parte lógica de la plataforma web.

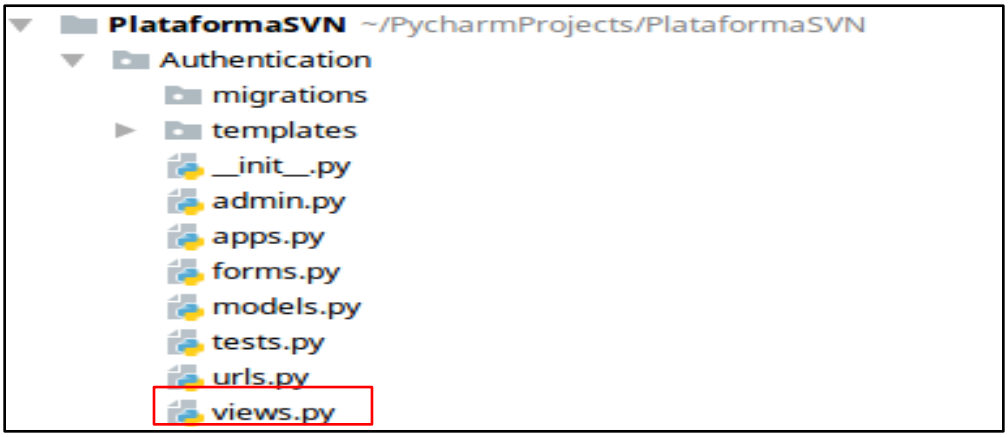


Figura 166: Estructura de carpetas donde se encuentra la clase controladora

```
def guardar_repositorio(request):
    try:
        # Comprobar en la tabla Repositorio si no existe uno con el mismo al que se va crear
        if(not Repositorio.objects.filter(nombre_repo=request.POST['nombre_repo']).exists()):
            #Crear un repositorio SVN en la ruta especificada
            r1=subprocess.call(['mkdir', request.POST['direccion_repo']])
            r2=subprocess.call(['svnadmin', 'create',request.POST['direccion_repo'] + request.POST['nombre_repo']])
            r3=subprocess.call(['chmod', '-R', '777', request.POST['direccion_repo']])
            #Comprobar que no hubo error al crear el repositorio
            if r2==0 and r3==0:
                Repo = Repositorio(
                    nombre_repo=request.POST['nombre_repo'],
                    direccion_repo=request.POST['direccion_repo']
                )
                #Guardar el repositorio y notificarle al usuario
                Repo.save()
                return HttpResponse(json.dumps({'success': True}))
            #Notificarle al usuario el error al crear el repositorio
            else:
                return HttpResponse(json.dumps(
                    {'success': False, 'msg': 'Ha ocurrido un error creando el repositorio.'}))
        # Notificarle al usuario que existe un repositorio con el mismo nombre
        else:
            return HttpResponse(json.dumps({'success': False,
                                            'msg': 'Ya existe un repositorio con ese nombre.'}))
    except Exception as e:
        return HttpResponse(json.dumps(
            {'success': False, 'msg': str(e)}))
```

Figura 17: Ejemplo de código en la clase controladora del RF Adicionar VirtualHost

3.3 Pruebas a la plataforma web

Según el Proceso de Desarrollo de Software (PRODESOFT), el objetivo principal de las pruebas es asegurar que la herramienta cumpla con las especificaciones requeridas y eliminar los posibles defectos que esta pudiera tener.

3.3.1 Descripción de las pruebas realizadas

Se realizaron pruebas de caja negra, de caja blanca, de usabilidad, de carga y estrés en un entorno virtual de prueba.

Pruebas Funcionales:

Método de caja negra

Las pruebas de caja negra, también llamadas pruebas de comportamiento, se enfocan en los requerimientos funcionales del software; es decir, las técnicas de prueba de caja negra le permiten derivar conjuntos de condiciones de entrada que revisarán por completo todos los requerimientos funcionales para un programa. Las pruebas de caja negra no son una alternativa para las técnicas de caja blanca. Las pruebas de caja negra intentan encontrar errores en las categorías siguientes: 1) funciones incorrectas o faltantes, 2) errores de interfaz, 3) errores en las estructuras de datos o en el acceso a bases de datos externas, 4) errores de comportamiento o rendimiento y 5) errores de inicialización y terminación (Pressman, 2010).

Método de caja blanca

La prueba de caja blanca, en ocasiones llamada prueba de caja de vidrio, es una filosofía de diseño de casos de prueba que usa la estructura de control descrita como parte del diseño a nivel de componentes para derivar casos de prueba. Al usar los métodos de prueba de caja blanca, puede derivar casos de prueba que: 1) garanticen que todas las rutas independientes dentro de un módulo se revisaron al menos una vez, 2) revisen todas las decisiones lógicas en sus lados verdadero y falso, 3) ejecuten todos los bucles en sus fronteras y dentro de sus fronteras operativas y 4) revisen estructuras de datos internas para garantizar su validez (Pressman, 2010).

- ✓ **Prueba del camino básico:** El método de ruta básica permite al diseñador de casos de prueba derivar una medida de complejidad lógica de un diseño de procedimiento y usar esta medida como guía para definir un conjunto básico de rutas de ejecución. Los casos de prueba derivados para

revisar el conjunto básico tienen garantía para ejecutar todo enunciado en el programa, al menos una vez durante la prueba (Pressman, 2010).

Pruebas de usabilidad

Prueba orientada a factores humanos, estéticos y consistencia en la interfaz de usuario. “Se determina además la calidad de la experiencia de un usuario en la forma en la que éste interactúa con el sistema, se considera la facilidad de uso y el grado de satisfacción del usuario” (Esquijarosa Bonilla, 2011)

Pruebas de Rendimiento

- ✓ **Prueba de carga:** La intención de la prueba de carga es determinar cómo responderán la aplicación y su entorno del lado servidor a varias condiciones de carga. Se definen un conjunto de condiciones de prueba tales como el número de usuarios concurrentes, el número de transacciones en línea por unidad de tiempo y la carga de datos procesados por el servidor en cada transacción (Pressman, 2010).
- ✓ **Prueba de resistencia (estrés):** La prueba de esfuerzo es una continuación de la prueba de carga, pero en esta instancia las variables: números de usuarios concurrentes, el número de transacciones en línea por unidad de tiempo y la carga de datos procesados por el servidor en cada transacción; se fuerzan a satisfacerse y luego se superan los límites operativos (Pressman, 2010).

3.3.2 Herramientas utilizadas en las pruebas automáticas

La automatización de las pruebas, establecen la utilización de herramientas que prueban el sistema sin la interacción constante de las personas, permiten una disminución de tiempo, esfuerzo y gasto de recursos, y se obtiene una mayor cobertura del software a probar. Todo esto implica lograr una selección factible de acuerdo al ambiente de desarrollo de trabajo y a las pruebas de software que se quieran automatizar.

JMeter

Es una herramienta Java desarrollada dentro del proyecto Jakarta, que permite realizar pruebas de rendimiento y pruebas funcionales sobre aplicaciones web. Permite realizar pruebas web clásicas, pero también realizar test de FTP, JDBC, JNDI, LDAP, SOAP/XML-RPC, y Web Services. Puede ejecutar pruebas de rendimiento distribuidas entre distintos ordenadores. Posibilita activar o desactivar una parte del test, muy útil cuando se está desarrollando un test largo, y se desea deshabilitar ciertas partes iniciales

que sean muy pesadas o largas. Posee la variante de generar un caso de prueba a través de una navegación de usuario. Como herramienta de prueba dispone de varios componentes que facilitan la elaboración de los escenarios de prueba, y permite simular para cada uno de los escenarios miles de usuarios. Se caracteriza por sus funcionalidades para realizarle pruebas de estrés, carga y rendimiento a las aplicaciones de manera independiente, al permitir aislar los subsistemas de la aplicación. Además puede ser configurable porque permite definir la cantidad de usuarios que va a simular y las pruebas que se le aplican al marco de trabajo (Caballero Redondo, 2012).

- ✓ Se comporta como un proxy local para que a través de él transite toda la información en tiempo real.
- ✓ Permite analizar su rendimiento, dadas determinadas condiciones como la navegabilidad en el sistema.
- ✓ Numera los errores ocurridos durante la transferencia de datos del servidor al navegador, muchos de estos errores no son visibles para el cliente, pero pueden provocar dificultades a corto o largo plazo.
- ✓ El muestreo de la información a través de tablas y gráficas facilita la comprensión del flujo de datos y sus características.
- ✓ Además, brinda información sobre el tiempo de servicio al sistema, el por ciento de rendimiento, el tráfico de información entre el cliente y el servidor tanto los correctos como los incorrectos.

3.3.3 Resultados de las pruebas

Caja negra

Las siguientes tablas muestran el Diseño de caso de prueba para el requisito funcional Adicionar Repositorio. Otros diseños se pueden encontrar en el **Anexo 4.**

Descripción del caso de prueba Adicionar Repositorio

Condiciones de ejecución:

- ✓ El usuario debe estar autenticado.
- ✓ Se ubica: **Menú de Opciones/Subversion/Repositorio/Adicionar**

Tabla 5: Diseño de caso prueba de caja negra RF Adicionar Repositorio

Nombre del requisito	Descripción general	Escenarios de pruebas	Flujo del escenario
Adicionar Repositorio	Permite agregar un nuevo Repositorio	EP 1.1: Adicionar Repositorio correctamente.	1. Se muestra la interfaz Repositorio. 2. Se selecciona el botón **Adicionar**. 3. Se muestra la interfaz Crear Repositorio. 4. Se introducen todos los datos. 5. Se presiona el botón **Guardar**. 6. El sistema muestra el mensaje de información: "*El* Repositorio *se ha creado satisfactoriamente*" y vuelve a la interfaz Repositorio.
		EP 1.2: Adicionar Repositorio incorrectamente.	1. Se muestra la interfaz Repositorio. 2. Se selecciona el botón **Adicionar**. 3. Se muestra la interfaz Crear Repositorio. 4. Se introducen todos los datos incorrectos. 5. El sistema marca los campos seleccionados de color rojo con una alerta notificando el error cometido. 6. Al presionar el botón **Guardar** si no se solucionan los errores el sistema no permite realizar la operación y muestra un mensaje de error en los *tooltips* de los campos según corresponda.
		EP 1.3: Cancelar operación.	1. Se muestra la interfaz Repositorio. 2. Se selecciona el botón **Adicionar**. 3. Se muestra la interfaz Crear Repositorio. 4. Se introducen todos los datos.

			5. Se presiona el botón **Cancelar**. 6. Se cancela la acción, se cierra la ventana y se muestra la interfaz Repositorio.

Descripción de variables

Tabla 6: Descripción de variables del RF Adicionar Repositorio

No	Nombre del campo	Clasificación	Puede ser nulo	Descripción
1	Dirección del Repositorio **(DR)**	Campo de texto	no	Se especifica la dirección del repositorio.
2	Nombre del Repositorio **(NR)**	Campo de texto	no	Se especifica el nombre del repositorio.

Juego de datos a probar

Tabla 7: Juego de datos a probar RF Adicionar Repositorio

Id del escenario	Escenario	Variable 1	Variable 2	Respuesta del sistema
		DR	NR	
E.P 1.1	Adicionar Repositorio correctamente.	/home/juanca/Repositorios	RepositorioA	El sistema muestra el mensaje de información: "*El* Repositorio *se ha creado satisfactoriamente*" y vuelve a la interfaz Repositorio.
E.P 1.2	Adicionar Repositorio incorrectamente.	*Vacío*	*Vacío*	NA

		/home/juanca/Repositorios	*Vacío*	El sistema no permite realizar la operación y muestra un mensaje de error en los *tooltips* de los campos según corresponda.
		Vacío	RepositorioA	El sistema no permite realizar la operación y muestra un mensaje de error en los *tooltips* de los campos según corresponda.
		Caracteres Incorrectos	*Caracteres Incorrectos*	El sistema no permite realizar la operación y muestra un mensaje de error en los *tooltips* de los campos según corresponda.
		(/home/juanca/Repositorios) ***I****Vacío****I****Caracteres Incorrectos*	(/home/juanca/Repositorios) ***I****Vacío****I****Caracteres Incorrectos*	El sistema no permite realizar la operación y muestra un mensaje de error en los *tooltips* de los campos según corresponda.
E.P 1.3	Cancelar operación.	(/home/juanca/Repositorios) ***I****Vacío****I****Caracteres Incorrectos*	(/home/juanca/Repositorios) ***I****Vacío****I****Caracteres Incorrectos*	NA

Resultado de las pruebas de Caja Negra

Después de realizar las pruebas funcionales mediante el método de Caja Negra, se comprobaron las funcionalidades del sistema y la correcta validación de sus campos. Fueron detectadas un total de 14 no conformidades durante 3 iteraciones, que consistían en errores funcionales del sistema y errores ortográficos, a las mismas se le dieron seguimiento y fueron resueltas a medida que se avanzó en el proceso de prueba. La figura 18 muestra un gráfico que representa las iteraciones realizadas.

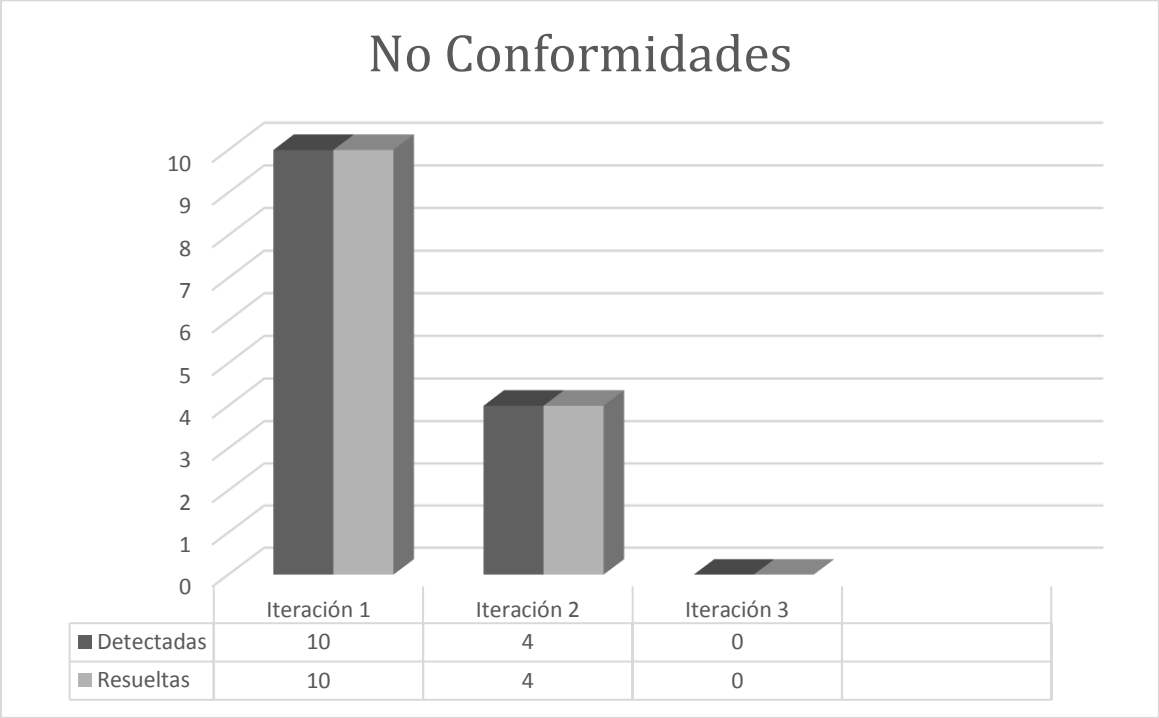


Figura 17: Resultado de las pruebas funcionales

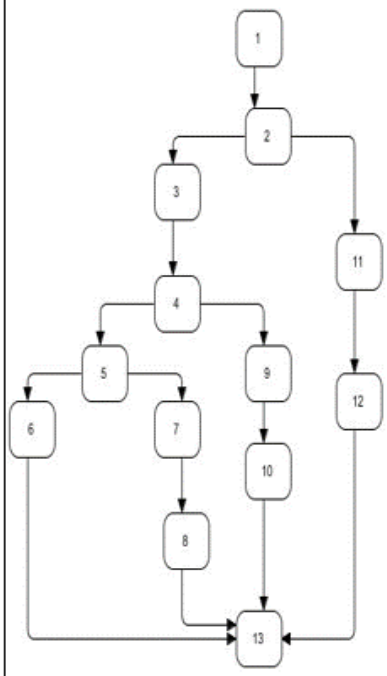
Caja blanca

Prueba de Caja Blanca

Código al que se aplica:

```
14  def post(self, request):
15      form = AuthenticationLoginForm(request.POST)//1
16      if form.is_valid()://2
17          user = form.cleaned_data.get('user')//3
18          password = form.cleaned_data.get('password')
19          acceso = authenticate(username=user, password=password)
20          if acceso is not None://4
21              if acceso.is_active://5
22                  login(request, acceso)//6
23                  respuesta='../base'//6
24
25              else://7
26                  respuesta='../authentication/login'//8
27          else://9
28               respuesta='../authentication/login',{'error':"El user"}//10
29
30      else://11
31          respuesta='Authentication/login.html'//12
32
33      return respuesta//13
```

Representación en grafo de flujo:



Complejidad ciclomática:

V(G) = (CantAristas - CantNodos) + 2 = (15-13) + 2 = 4

Caminos independientes:

1. 1-2-11-12
2. 1-2-3-4-9-10-13
3. 1-2-3-4-5-6-13
4. 1-2-3-4-5-7-8-13

Figura 18: Diseño del caso de prueba caja blanca

Usabilidad

Se analiza el equilibrio en la interfaz de usuario, la facilidad de uso y la calidad en la interacción con el usuario mientras ocurren los procesos de administración de los sistemas.

Para realizar la prueba se aplicó la Lista de chequeo Pruebas de Usabilidad, aprobada en el Centro de Calidad y Estándares de la XETID.

Principios de usabilidad:

1. Reconocimiento apropiado

- La plataforma web es fácil de entender y utilizar.
- Funcionalidades intuitivas.
- Se entienden las acciones a realizar en cada una de las operaciones.
- Tiene una única interfaz de trabajo.
- El sistema posee un *favicon* como elemento significativo.
- Posee logo y se encuentra ubicado correctamente.

2. Aprendibilidad
   - No existe documentación que describa las funcionalidades.
   - No existe la ayuda para los usuarios.
   - No se ofrecen valores por defecto en los campos en caso de que tengan que estar completados y el usuario desconozca como hacerlo.
   - Los mensajes de error son entendibles.
3. Operatividad
   - Las acciones a realizar varían según el rol autenticado.
   - Cuando se realiza ciertas acciones, el sistema muestra un mensaje y una barra de progreso indicando que se está procesando la acción.
   - Se provee una retroalimentación cuando una tarea ha sido completada correctamente.
   - El usuario puede redimensionar las interfaces.
   - La plataforma muestra en la interfaz web niveles básicos de las acciones realizadas.
4. Calidad de las interfaces
   - En los formularios se indican los campos obligatorios.
   - La fuente y la tipografía es homogénea.
   - El diseño no tiene grandes contrastes y es común para toda la plataforma.
5. Protección de errores contra usuario

- Se dan indicaciones para completar campos problemáticos.
- Solo algunas operaciones están validadas.
- Los *tooltip* están correctos.
- El sitio hace fácil corregir los errores.

6. Accesibilidad

- Si se incrementa el tamaño de la letra el esquema del sitio no se quiebra.
- El cursor se desplaza adecuadamente en un formulario al presionar “tabulador”.
- Es posible la navegación sin ratón.

Como resultado se obtuvieron mejoras en la validación de campos y tratamiento de errores. Se incrementó la seguridad en la gestión de roles. Se perfecciona u optimiza la implementación de funcionalidades.

Rendimiento (carga y estrés)

La prueba realizada consistió en definir 2 pruebas de 250 y 350 hilos de concurrencia cada una, las cuales simulan 250 y 350 accesos de usuarios respectivamente, desplegado en un entorno de desarrollo **(Ver Anexo 5).**

Para un mejor entendimiento de los componentes “Reporte resumen” que se verán a continuación, se explica cada parámetro que la compone.

#Muestras: cantidad de hilos utilizados para la URL.

Media: tiempo promedio en milisegundos para un conjunto de resultados.

Min: tiempo mínimo que demora un hilo en acceder a una página.

Max: tiempo máximo que demora un hilo en acceder a una página.

Rendimiento: rendimiento medido en los requerimientos por segundo / minuto / hora.

Kb/sec: rendimiento medido en Kbytes por segundo.

Los valores totales obtenidos por el componente “Reporte resumen” para 250 hilos se muestran en la Tabla 8.

Tabla 8: Resultados de prueba de carga y estrés con el acceso de 250 usuarios

TOTAL	#Muestras	Media	Min	Max	Error	Rendimiento	Kb/sec
	5250	2473	2	25845	3	28.0414/segundos	2248.3

Los valores totales obtenidos por el componente “Reporte resumen” para 350 hilos se muestran en la Tabla 9.

Tabla 9: Resultados de prueba de carga y estrés con el acceso de 350 usuarios

TOTAL	#Muestras	Media	Min	Max	Error	Rendimiento	Kb/sec
	10110	4082	4	49553	7	31.152/segundos	4781.7

Análisis de los resultados

Tanto para la prueba con 250 y 350 usuarios, los tiempos de respuesta promedio obtenidos fueron de 2.473 y 4.082 segundos respectivamente. Bajo estas condiciones es un resultado positivo. Todas las funcionalidades trabajan correctamente y el % de errores es bajo, se deben sobre todo al entorno virtual de Python.

3.4 Evaluación técnica de la solución

Se utilizó el método Opinión de Expertos (Esquijarosa Bonilla, 2011), para realizar la evaluación técnica de la solución integral. Este método permite tomar decisiones para aceptar o rechazar determinada propuesta de acuerdo a criterios definidos y la evaluación dada por los expertos seleccionados (Hernández León, 2009). Para llevar a cabo la evaluación se realizaron los siguientes pasos:

Paso 1: Se determinan los criterios para la selección de los expertos.

La selección de los expertos se basó en los siguientes criterios:

- Graduado de Ingeniería en Informática, Telecomunicaciones o Automática.
- Experiencia en el trabajo con servidores Apache.
- Experiencia en el trabajo con servidores Subversion.
- Arquitecto de datos de la XETID.

- Gestor de configuración de la XETID.
- Administrador de sistemas de la XETID.
- No tienen que estar certificados oficialmente en ninguna de estas tecnologías.

Paso 2: Se elaboran los criterios utilizados en la evaluación y se agrupan de acuerdo a las características de la propuesta de solución integral.

Grupo 1: Criterios de mérito científico.

1. Mérito científico de la solución.

Grupo 2: Criterios de creación.

2. Necesidad de empleo de la solución.
3. Posibilidades de implantación.
4. Ahorro de costos.

Grupo 3: Criterios de flexibilidad.

5. Escalabilidad.
6. Integración.
7. Fiabilidad.
8. Disponibilidad.

Grupo 4: Criterios de impacto.

9. Impacto en el área para la cual está destinada la propuesta de solución.
10. Gestión de carga de trabajo.
11. Rendimiento.

Grupo 5: Criterios de usabilidad.

12. Aprovisionamiento y despliegue.
13. Fácil instalación y administración.

14. Monitorización.

15. Usabilidad de los requisitos implementados.

Paso 3: Se le asigna un valor relativo a cada grupo de criterios de acuerdo al porciento que representa cada grupo del total y los intereses a evaluar.

Grupo 1: 10

Grupo 2: 15

Grupo 3: 20

Grupo 4: 15

Grupo 5: 40

Paso 4: Se seleccionan siete expertos teniendo en cuenta los criterios del primer paso.

Paso 5: Se entrega a los expertos el acceso a la aplicación y una documentación breve sobre la solución, para que se documenten sobre el tema y expresen sus criterios. Los expertos conceden pesos de cero (valor mínimo) a diez (valor máximo) a cada uno de los criterios establecidos y una apreciación cualitativa con una clasificación final del proyecto en alta, media, baja, y fracaso.

Paso 6: Después de recibir los valores del peso relativo de cada criterio se construye la tabla correspondiente, donde $C1\ldots 15$ son los criterios a evaluar y $E1\ldots 7$ la evaluación de los expertos.

Tabla 10: Pesos otorgados por los expertos a los criterios

G	C/E	E1	E2	E3	E4	E5	E6	E7	Ep = ΣE / 7
10	**C1**	6	6	7	5	7	4	6	5,857142
15	**C2**	9	10	9	8	8	7	10	8,714285
	C3	7	6	6	5	7	5	4	5,714285
	C4	4	5	4	3	4	4	4	4,0
20	**C5**	8	9	10	7	8	10	8	8,571428
	C6	7	8	7	6	6	7	6	6,714285
	C7	5	4	6	7	5	6	4	5,285714
	C8	6	5	4	4	4	6	5	4,857142

15	**C9**	8	9	8	8	8	10	8	8,428571
	C10	7	7	8	8	7	9	7	7,571428
	C11	7	6	8	7	7	6	6	6,714285
40	**C12**	6	7	7	6	7	6	5	6,285714
	C13	7	6	6	6	7	8	5	6,428571
	C14	6	6	7	5	7	6	5	6,0
	C15	9	10	8	8	9	10	8	8,857142
Total		**102**	**104**	**105**	**100**	**101**	**104**	**91**	**99,999992**

Paso 7: Se verifica el trabajo de los expertos, utilizando el coeficiente de concordancia de Kendall y el estadígrafo Chi cuadrado (X^2). A continuación, el procedimiento:

- Sea C el número de criterios que van a evaluarse y E el número de expertos que realizan la evaluación.
- Para cada criterio se determina la ΣE que representa la sumatoria del peso dado por cada experto, E_p que es la puntuación promedio de los pesos correspondientes a cada criterio.
- Se determina la desviación de la media, que posteriormente se eleva al cuadrado para obtener la dispersión S por la expresión: $\mathbf{S = \Sigma\ (\Sigma E - \Sigma\Sigma E/C)^2}$.

Tabla 11: Cálculo de la Dispersión (S) para hallar la concordancia entre expertos

Expertos/ Criterios	ΣE	ΣE/C	ΣE - ΣΣE/C	$(\Sigma E - \Sigma\Sigma E/C)^2$
C1	41	2,733333	-8,036304	64,582181
C2	61	4,066666	11,963696	143,130021
C3	40	2,666666	-9,036304	81,654789
C4	28	1,866666	-21,036304	442,526085
C5	60	4,0	10,963696	120,202629
C6	47	3,133333	-2,036304	4,146533

C7	37	2,466666	-12,036304	144,872613
C8	34	4,857142	-15,036304	226,090437
C9	59	3,933333	9,963696	99,275237
C10	53	3,3125	3,963696	15,710885
C11	47	3,133333	-2,036304	4,146533
C12	44	2,933333	-5,036304	25,364357
C13	45	3,0	-4,036304	16,291749
C14	42	2,8	-7,036304	49,509573
C15	62	4,133333	12,963696	168,057413
ΣΣE/C		**49,036304**		
S = Σ (ΣE - ΣΣE/C)^2				**1605,561035**

- Se calcula el coeficiente de concordancia de Kendall, conociendo la dispersión:

$W = S / (E^2 (C^3 - C) / 16)$

- Se calcula el Chi cuadrado real, sabiendo el valor del coeficiente de concordancia de Kendall:

$X^2 = E * (C - 1) * W$

Tabla 12: Cálculo de concordancia de Kendall

S	**E^2**	**$C^3 - C$**	**$E^2 (C^3 - C)$**	**$E^2 (C^3 - C) / 16$**	**$W = S / (E^2 (C^3 - C) / 16)$**	**$X^2 = E * (C - 1) * W$**
1605,561035	49	3360	272160	17010	0,094389	**11,893014**

- Se compara el Chi cuadrado real calculado con el que se obtiene de la Tabla de Distribución Chi Cuadrado **(Ver Anexo 6)**, se toma 1 – α = 0.99 dónde α = 0.01 es el error permisible.

Si se cumple que el X^2 real < $\mathbf{X^2}$ **(α, c-1)** puede decirse que existe concordancia en el trabajo de los expertos.

Según el resultado del cálculo en la tabla anterior: **11,893014 < 30,578** por tanto, existe concordancia entre los expertos.

Paso 8: Posteriormente se identifica el peso (P) relativo de cada criterio y se calcula el Índice de Aceptación (IA) de la propuesta de solución (Hernández León, 2009).

- Se calcula el peso de cada criterio (P), conociendo el número de expertos que realizan la evaluación E y la sumatoria de las puntuaciones de cada criterio (C).
- Conociendo el peso de cada criterio y la cantidad de expertos se puede obtener el valor de P × C_p, donde C_p, es el criterio promedio concebido por los expertos en escala de 5.
- Con el valor anterior se calcula el Índice de Aceptación (IA). **IA = Σ (P x C_p) /5**.

Tabla 13: Calificación de cada criterio e índice de aceptación

Criterios	Cp	P	P x Cp
C1	2,928571	0,058571	0,171529
C2	4,3571425	0,087142	0,379690
C3	2,8571425	0,057142	0,163262
C4	2,0	0,04	0,08
C5	4,285714	0,085714	0,367345
C6	3,357142	0,067142	0,225405
C7	2,642857	0,052857	0,149693
C8	2,428571	0,048571	0,117958
C9	4,214285	0,084285	0,355201
C10	3,785714	0,075714	0,286631
C11	3,357142	0,067142	0,225405

C12	3,142857	0,062857	0,297550
C13	3,214285	0,064285	0,206630
C14	3,0	0,06	0,18
C15	4,428571	0,088571	0,392242
Σ (P x Cp)			**4,188541**
IA			**0,8377082**

Paso 9: Por último, se determina la probabilidad de éxito de la propuesta, ubicando el IA calculado anteriormente en rangos que están predefinidos en la Tabla 13, en dependencia de su ubicación será la probabilidad de éxito que tenga la propuesta.

Fracaso seguro: no cumple con un mínimo de variables claves.

Baja probabilidad de éxito: no cumple con los criterios de creación.

Media probabilidad de éxito: cumple solo con los criterios de flexibilidad.

Alta probabilidad de éxito: cumple con las variables claves de usabilidad e impacto.

Tabla 14: Rangos predefinidos para el IA

0.7 < IA	Alta probabilidad de éxito
0.5 < IA < 0.7	Media probabilidad de éxito
0.3 < IA < 0.5	Baja probabilidad de éxito
IA < 0.3	Fracaso seguro

El IA calculado es **0,8377082** lo que significa que existe alta probabilidad de éxito con la propuesta de solución.

3.5 Conclusiones del capítulo

La confección del diagrama de despliegue estableció una mayor comprensión de la estructura del sistema; la utilización de los estándares de codificación permitió garantizar al código alta calidad, menos errores, legibilidad, estandarización y fácil comprensión. La realización de las pruebas de software ayudó a

determinar los errores para corregirlos, asegurando calidad y rendimiento en el sistema. Evaluar la propuesta de solución utilizando la Opinión de Expertos ayudó a delimitar las limitaciones y el índice de aceptación que se tiene de la propuesta, así como lo acertado de la solución implementada para dar cumplimiento a los objetivos propuestos.

Conclusiones

Luego de realizada la presente investigación se concluye que:

- ✓ Se obtuvo un producto a la medida del cliente, para alcanzar tal resultado, se realizó un estudio de sistemas similares para una mejor comprensión utilizando tecnologías libres, el cual permite administrar servidores web Apache y control de versiones Subversion de manera integrada.
- ✓ La utilización del Proceso de Desarrollo de Software (PRODESOFT) para el diseño y construcción de la solución propuesta, generó los artefactos establecidos en el mismo, quedando definidos los requisitos funcionales y no funcionales.
- ✓ En la fase de implementación de la solución, para una mejor comprensión y entendimiento, se tuvo en cuenta los estándares y buenas prácticas de programación establecidos por la XETID.
- ✓ Las pruebas realizadas al sistema demostraron el cumplimiento de diferentes atributos de calidad, evidenciando que el mismo cumple satisfactoriamente con los requisitos definidos inicialmente.
- ✓ La evaluación técnica a través de la Opinión de Expertos determinó la probabilidad de éxito de la plataforma web, siendo esta probabilidad alta.

Recomendaciones

1. Administrar con la plataforma creada, servidores web y servidores Subversion previamente instalados.
2. Efectuar el despliegue distribuido para más de un servidor.

Bibliografía

Abalde, Carlos. 2010. *Control de Versiones con Subversion.* 2010.

Acens. 2006. *Servidor web Nginx, una clara alternativa a Apache.* 2006.

acunetix. 2012. acunetix. [En línea] 2012. [Citado el: 17 de 3 de 2018.] https://www.acunetix.com/wp-content/uploads/2012/10/wvsmanual.pdf.

Aiteco. 2013. Aiteco. *Origen del Mapa de Procesos – Gestion de Procesos.* [En línea] 2013. [Citado el: 12 de enero de 2018.] https://www.aiteco.com.

Baena, Luis Rodríguez. 2014. *Frameworks CSS: Bootstrap.* 2014.

Ben Collins-Sussman, Brian W. Fitzpatrick, C. Michael Pilato. 2011. *Control de versiones con Subversion.* 2011.

—. 2011. *Control de versiones con Subversion.* 2011.

Bertha Mariel Márquez Avendaño, José Manuel Zulaica Rugarcía. 2004. *Implementación de un reconocedor de voz gratuito a el sistema de ayuda a invidentes Dos-Vox en español .* 2004.

Caballero Redondo, Jose. 2012. *Anális de la aplicación JMeter.* 2012.

Castellanos, Luis. 2014. Sistemas Operativos. *Sistemas Operativos para Servidores Web.* [En línea] 2014. [Citado el: 15 de noviembre de 2017.] htpps://lcsistemasoperativos.wordpress.com.

Cera, Arley Enrique. 2015. *MOTOR DE EVALUACION AQTIEngine.* 2015.

Collado, Manuel. 2010. *Representación XML de elementos software.* 2010.

Cruz, Andrés. 2017. DesarrolloLibre. *Doctrine archivos de mapeo y entidades en Symfony.* [En línea] febrero de 14 de 2017. [Citado el: 12 de diciembre de 2017.] http://www.desarrollolibre.net/blog/tema/292/symfony/doctrine-archivos-de-mapeo-y-entidades-en-symfony#.WnCvTbODNdI.

de León, Lianet. 2009. *Nuevo software Controlador de Versiones para el Polo de Bioinformática.* 2009.

—. 2009. *Nuevo software Controlador de Versiones para el Polo de Bioinformática.* 2009.

Delgado, Nestor. 2014. *Mejora en el rendimiento del servidor Apache en el módulo Web de HMAST.* 2014.

Departamento Nacional de Planeación. 2013. *Guía de elaboración de modelos conceptuales.* Colombia : s.n., 2013.

Donato, Gerardo Morgade. 2009. *ClioBD. Sistema de control de versiones para bases de datos.* 2009.

Duarte, Eugenio. 2013. CAPACITY. *jQuery: Qué es, Orígenes, Ventajas y Desventajas.* [En línea] 16 de marzo de 2013. [Citado el: 12 de diciembre de 2017.] http://blog.capacityacademy.com/2013/03/16/jquery-que-es-origenes-ventajas-desventajas/.

Esquijarosa Bonilla, Dany. 2011. *Plataforma Integral de Virtualización y Almacenamiento.* 2011.

Fernández, Rubén Domínguez. 2014. @rubendomfer. *Compartir archivos fácilmente con: HFS (Http File Server).* [En línea] 14 de agosto de 2014. [Citado el: 21 de noviembre de 2017.] http://www.rubendomfer.com.

Fernández, Yenisleidy. 2012. *Patrón Modelo-Vista-Controlador.* 2012.

Forgiarini, Luis. 2017. Ejemplos de Servidores Web Mas Utilizados. *luisforgiarini.* [En línea] 2 de enero de 2017. [Citado el: 15 de noviembre de 2017.] https://luisforgiariniblog.com.

Gámez, Raúl. 2012. Aodbc in the Cloud. *¿Cual es el mejor software servidor web?* [En línea] 12 de noviembre de 2012. [Citado el: 18 de noviembre de 2017.] http://blog.aodbc.es.

Gang of Four. *Design Patterns: Elements of reusable object oriented software.*

González, Yaisel Hurtado. 2013. *Estándar de codificación para Python v1.0.* La Habana : s.n., 2013.

Grant, Yassef Amed Galarraga. 2010. *Propuesta de entorno de integración continua para el desarrollo de software en el Centro de Informatización Universitaria.* 2010.

Guaita, Álvaro Martínez. 2009. desarrolloweb.com. *Cherokee Web Server, el más rápido de los servidores web.* [En línea] 17 de agosto de 2009. [Citado el: 17 de noviembre de 2017.] https://desarrolloweb.com.

Guerrero, Esneyddis. 2014. *Desarrollo del Módulo gestión de ubicación de productos del subsistema de Inventario de la Plataforma Logística. v.2.0.* 2014.

Hernández León, Rolando. 2009. *Una Introducción a la Gestión de Proyectos.* 2009.

Herrera, Yenisleidy. 2015. *Portafolio de Desarrollo Personal para la centralización de las evidencias de trabajo v2.0.* 2015.

Holovaty, Adrian. 2008. *El libro de Django.* 2008.

Ilies María Giraldo Cisneros, Ramón Felipe Sosa Pérez. 2010. *Entorno de gestión del tiempo para los proyectos de desarrollo de software del Centro de Identificación y Seguridad Digital de la UCI.* 2010.

Irnel Victoria Sosa, Reidel Cabrera Martínez. 2010. *Propuesta de una aplicación para la configuración de seguridad de los Servidores Web y de Base de Datos del Proyecto Sistema de Gestión Fiscal.* 2010.

JetBrains. Python IDE for Professional Developers. [En línea] [Citado el: 12 de enero de 2018.] https://www.jetbrains.com/pycharm/.

Larman, Craig. 1999. *UML y Patrones. Introducción al análisis y diseño orientado a objetos.* Ciudad de mexico : s.n., 1999.

Mateu, Carles. 2011. *Desarrollo de aplicaciones web.* 2011.

Moreno, Lorenzo Ruiz. 2014. *Django Framework.* 2014.

Muñiz, Patricia Lucio. 2012. INGENIERÍA DEL SOFTWARE . [En línea] 21 de abril de 2012. [Citado el: 15 de marzo de 2018.] http://infsoftware-gisse.blogspot.com/2012/04/pruebas-de-software.html.

Nápoles, Yailin. 2015. *Aplicación web para la gestión de repositorios SVN.* 2015.

Narváez, Alexander Romero. 2012. *Framework Symfony PHP.* 2012.

Ortega, Alvaro López. 2003. *Cherokee Web server.* 2003.

Pardo, Lisandro. 2015. Neoteo. *HTTP File Server: Comparte archivos de manera sencilla con un mini-servidor web.* [En línea] 13 de julio de 2015. [Citado el: 18 de noviembre de 2017.] https://www.neoteo.com.

Pérez, Alejandro Mengana. 2014. *Desarrollo de la versión 3.0 del módulo Banco para el Sistema Integral de Gestión Empresarial.* 2014.

Pressman, Roger S. 2010. *Ingeniería de Software 7 edición.* 2010.

Ramírez, Antonio Jiménez. 2013. *Definición de la metodología para la gestión y verificación de proyectos Ágiles, haciendo uso de Microsoft Team Foundation Server, para su posterior uso en una aplicación real.* 2013.

Reyes, Juniel Cosme. 2010. *Propuesta de un entorno para la Gestión de Proyectos y Gestión de la Configuración de Software en Proyectos del MININT.* 2010.

Rossum, Guido Van. 2009. El Tutorial de Python. [En línea] 2009. [Citado el: 12 de enero de 2018.] http://python.org.ar/pyar/Tutorial.

sencha.com. Sencha Ext JS. *Sencha Ext JS.* [En línea] [Citado el: 7 de Enero de 2018.] https://www.sencha.com/products/extjs/#overview.

Sosa, Irnel Victoria. 2010. *Propuesta de una aplicación para la configuración de seguridad de los Servidores Web y de Base de Datos del Proyecto Sistema de Gestión Fiscal.* 2010.

Sparx Systems. 2013. *El Modelo de Proceso de Negocio .* 2013.

Tarragó, Dianela Galardy. 2012. *Propuesta de solución para el proceso de evaluación y control docente en el Sistema de Gestión Académica de Pregrado.* 2012.

Toro, Luigys. 2012. Desde Linux. *LIGHTTPD – un servidor web muy ágil y liviano.* [En línea] 4 de septiembre de 2012. [Citado el: 17 de noviembre de 2017.] https://blog.desdelinux.net.

UCID. 2009. *Proceso de Desarrollo y Gestión de Proyectos de Software (Versión 1.5).* 2009.

Anexos

Anexo 1

Tabla 15: Especificación del requisito funcional Modificar VirtualHost

Conceptos tratados	**Conceptos**	**Atributos**
	VirtualHost	id, nombre_vh, fecha_creacion, IP, puerto, ssl, DocumentRoot, logs, servername, serveralias.
Precondiciones	**Precondiciones**	**Pre-requisito**
	1. El usuario debe estar autenticado. 2. Debe existir al menos un VirtualHost creado anteriormente.	1. Autenticar usuario. 2. Adicionar VirtualHost
Descripción	1. Se ubica el **Menú de Opciones/Apache**. 2. Se despliega el árbol de Apache y se hace clic en la opción VirtualHost. 3. Se muestra la interfaz de VirtualHost. 4. Se selecciona de la lista de VirtualHost uno de estos. 5. Se selecciona el botón Modificar. 6. Se muestra la ventana con los campos para Modificar VirtualHost. 7. Se llenan los campos en dependencia a las modificaciones que se desean realizar 8. Se presiona el botón Guardar si están completos y validados los campos. 9. Se guardan los cambios realizados en el sistema. 10. Concluye el requisito	

Complejidad	Baja
Validaciones	5.1 En caso de no existir ningún VirtualHost, el botón Modificar no se habilita. 7.1 Si el usuario ingresa caracteres incorrectos en determinados campos, el sistema le notifica.
Post-condiciones	No procede.
Post-requisito	No procede.

Tabla 16: Especificación del requisito funcional Eliminar VirtualHost

Conceptos tratados	**Conceptos**	**Atributos**
	VirtualHost	id, nombre_vh, fecha_creacion, IP, puerto, ssl, DocumentRoot, logs, servername, serveralias.
Precondiciones	**Precondiciones**	**Pre-requisito**
	1. El usuario debe estar autenticado. 2. Debe existir al menos un VirtualHost creado anteriormente.	1. Autenticar usuario. 2. Adicionar VirtualHost
Descripción	1. Se ubica el **Menú de Opciones/Apache**. 2. Se despliega el árbol de Apache y se hace clic en la opción VirtualHost. 3. Se muestra la interfaz de VirtualHost. 4. Se selecciona de la lista de VirtualHost uno de estos. 5. Se selecciona el botón Eliminar. 6. Se muestra el mensaje de confirmación. 7. Se presiona el botón Sí. 8. Se guardan los cambios realizados en el sistema.	

	9. Concluye el requisito
Complejidad	Media
Validaciones	5.1 En caso de no existir ningún VirtualHost, el botón Eliminar no se habilita
Post-condiciones	No procede.
Post-requisito	No procede.

Tabla 17: Especificación del requisito funcional Buscar VirtualHost

Conceptos tratados	**Conceptos**	**Atributos**
	VirtualHost	id, nombre_vh, fecha_creacion, IP, puerto, ssl, DocumentRoot, logs, servername, serveralias.
Precondiciones	**Precondiciones**	**Pre-requisito**
	1. El usuario debe estar autenticado. 2. Debe existir al menos un VirtualHost creado anteriormente.	1. Autenticar usuario. 2. Adicionar VirtualHost
Descripción	1. Se ubica el **Menú de Opciones/Apache**. 2. Se despliega el árbol de Apache y se hace clic en la opción VirtualHost. 3. Se muestra la interfaz de VirtualHost. 4. Se ubica la barra de búsqueda. 5. Se escribe en la barra el valor por el que se desea buscar el VirtualHost. 6. Se lista(n) la(s) posible(s) coincidencia(s). 7. Concluye el requisito	
Complejidad	Baja	

Validaciones	5.1 El sistema le hace saber al usuario mediante el listado de VirtualHost si lo encontró o no.
Post-condiciones	No procede.
Post-requisito	No procede.

Anexo 2

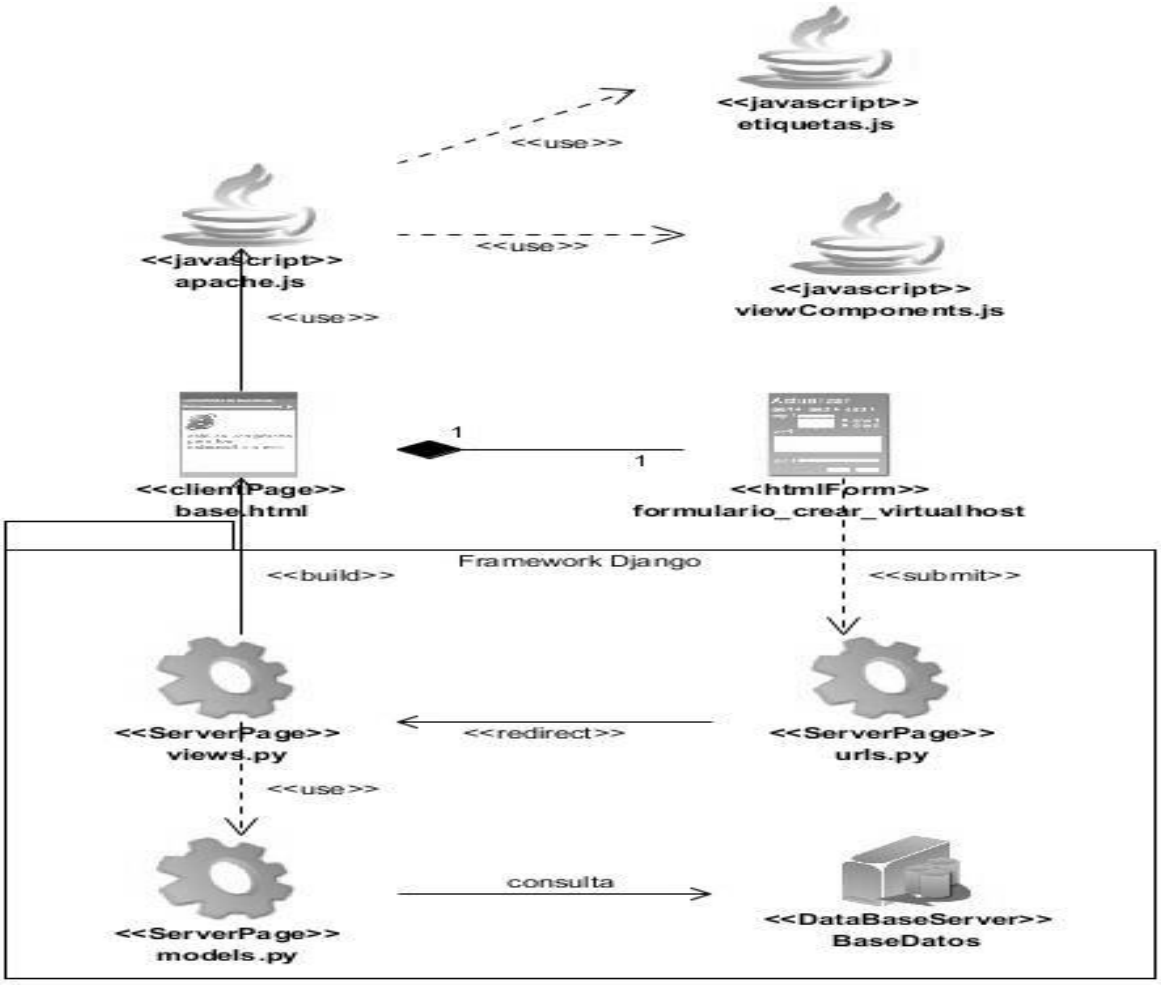


Figura 19: Diagrama de clases del diseño Crear VirtualHost

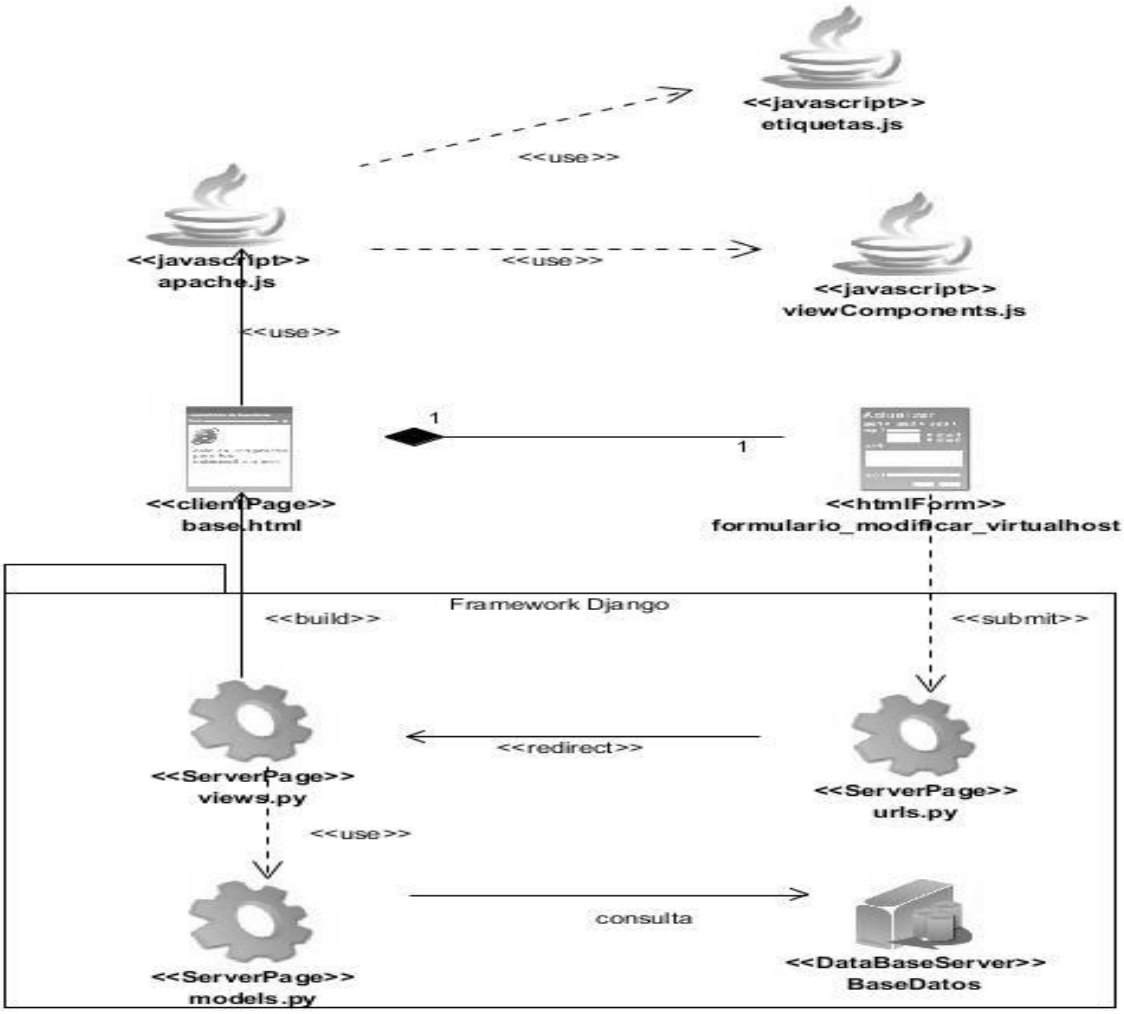


Figura 20: Diagrama de clases del diseño Modificar VirtualHost

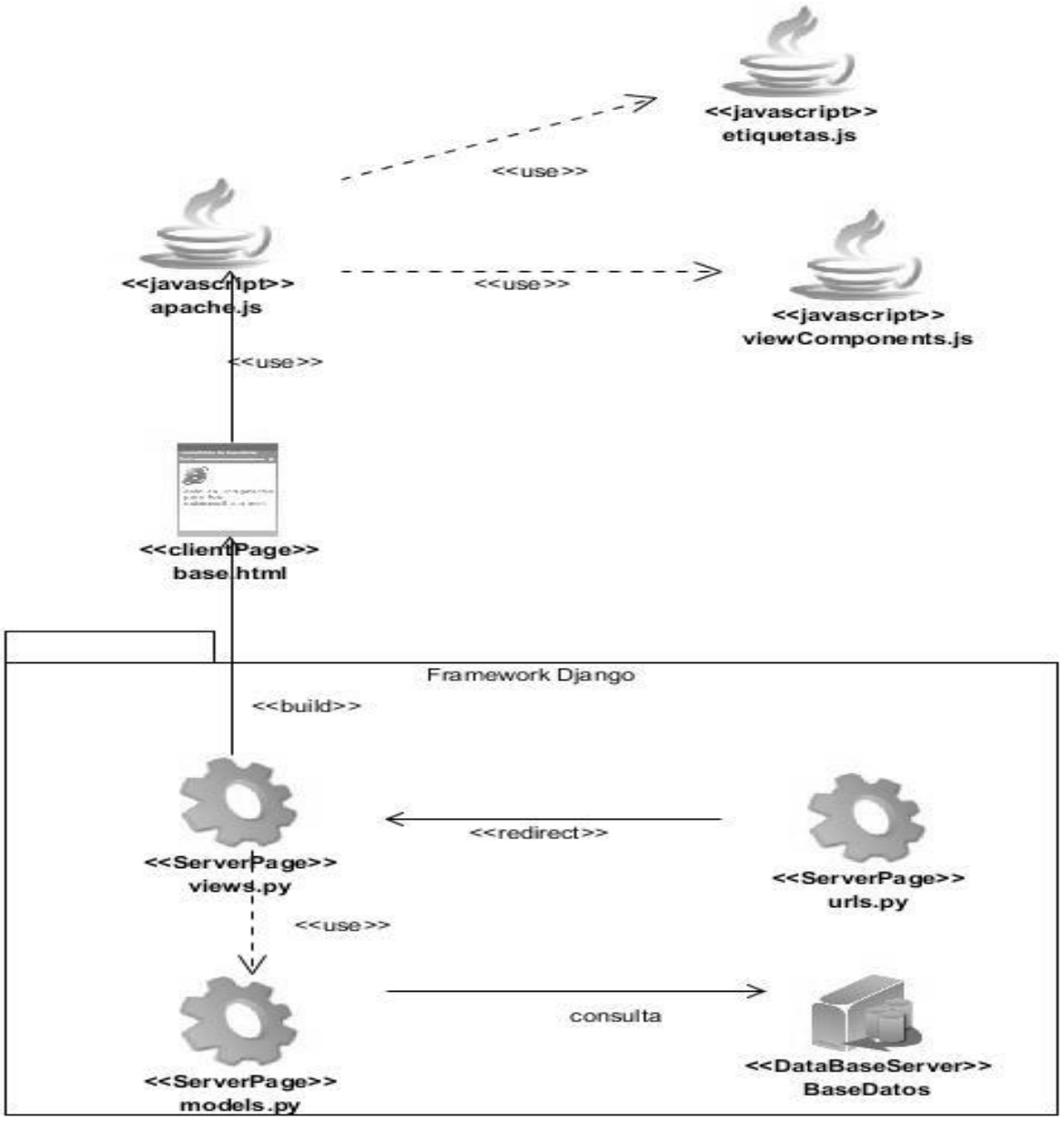


Figura 21: Diagrama de clases del diseño Eliminar VirtualHost

Anexo 3

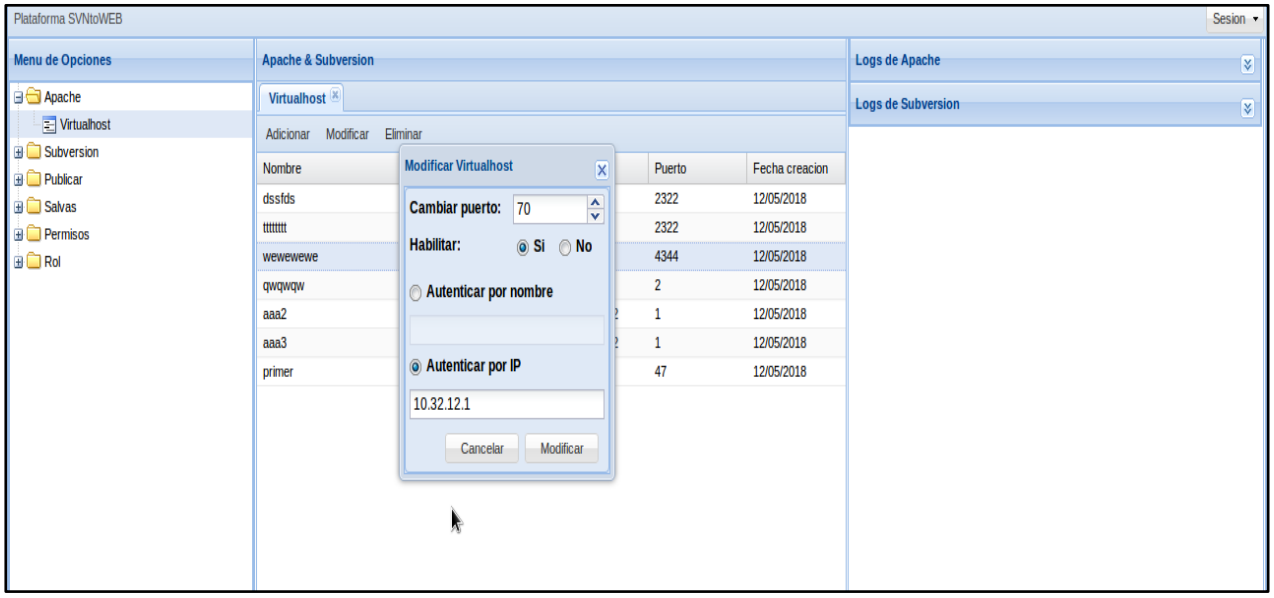


Figura 22: Prototipo de Interfaz de Usuario Funcional Modificar VirtualHost

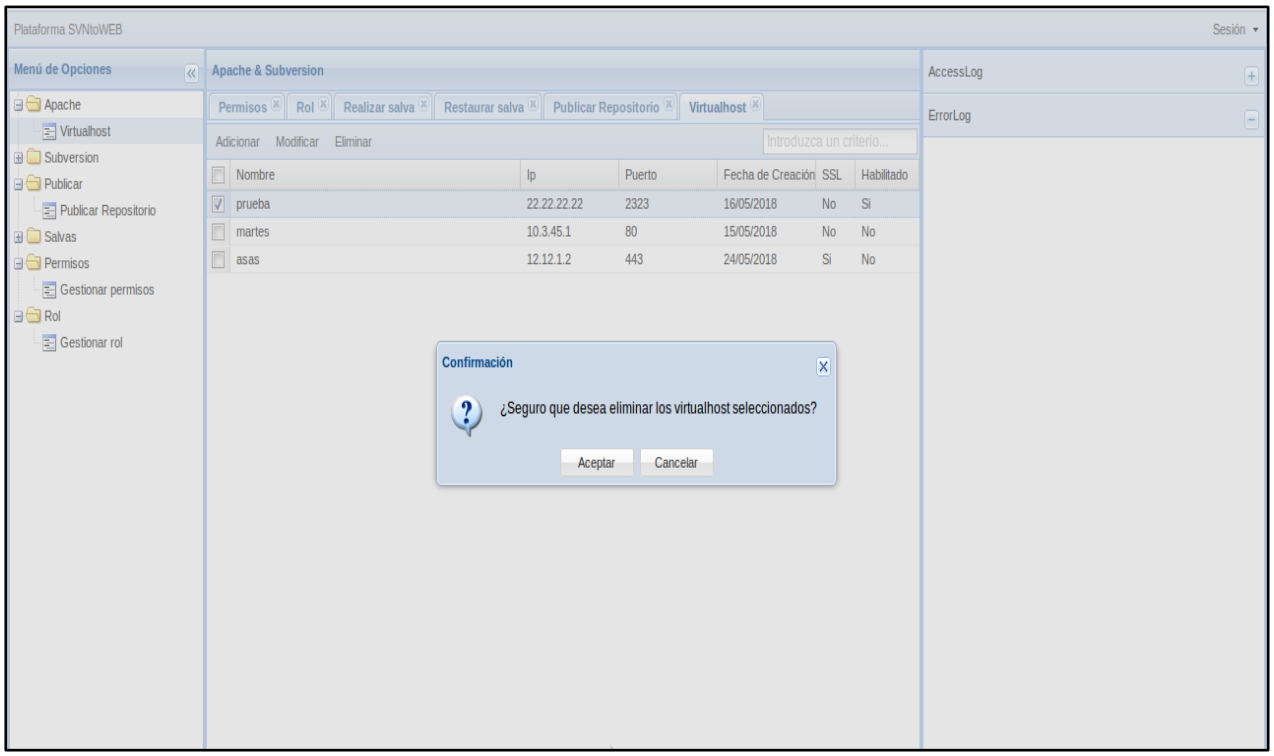


Figura 23: Prototipo de Interfaz de Usuario Funcional Eliminar VirtualHost

Anexo 4

Descripción del caso de prueba Adicionar VirtualHost

Condiciones de ejecución:

- ✓ El usuario debe estar autenticado.
- ✓ Se ubica el **Menú de Opciones/Apache/VirtualHost/Adicionar**.

Tabla 18: Diseño de caso prueba de caja negra RF Adicionar VirtualHost

Nombre del requisito	Descripción general	Escenarios de pruebas	Flujo del escenario
Adicionar VirtualHost	Permite agregar un nuevo VirtualHost.	EP 1.1: Adicionar VirtualHost correctamente.	1. Se muestra la interfaz VirtualHost. 2. Se selecciona el botón **Adicionar**. 3. Se muestra la interfaz Adicionar VirtualHost. 4. Se introducen todos los datos. 5. Se presiona el botón **Guardar**. 6. El sistema muestra el mensaje de información: "*El VirtualHost se ha adicionado satisfactoriamente*" y vuelve a la interfaz VirtualHost.
		EP 1.2: Adicionar VirtualHost incorrectamente.	1. Se muestra la interfaz VirtualHost 2. Se selecciona el botón **Adicionar**.

			3. Se muestra la interfaz Adicionar VirtualHost. 4. Se introducen todos los datos incorrectos. 5. El sistema marca los campos seleccionados de color rojo con una alerta notificando el error cometido. 6. Al presionar el botón **Guardar** si no se solucionan los errores el sistema no permite realizar ninguna operación y muestra un mensaje de error en los tooltips de los campos según corresponda.
		EP 1.3: Cancelar operación.	1. Se muestra la interfaz VirtualHost. 2. Se selecciona el botón **Adicionar**. 3. Se muestra la interfaz Adicionar VirtualHost. 4. Se introducen todos los datos. 5. Se presiona el botón **Cancelar**.

Descripción de variables

Tabla 19: Descripción de variables del RF Adicionar VirtualHost

No	Nombre del campo	Clasificación	Puede ser nulo	Descripción

1	Nombre del fichero **(NF)**	Campo de texto	no	Se especifica el nombre del VirtualHost.
2	Dirección IP **(IP)**	Campo de números	no	Dirección IP del VirtualHost.
3	Puerto **(P)**	Campo de números	no	Puerto por se escucha sirve el VirtualHost.
4	DocumentRoot **(DR)**	Campo de texto	no	Dirección del proyecto que se desea servir.
5	ServerAdmin **(SAd)**	Campo de texto	no	Especifica la dirección de correo electrónico del administrador.
6	ServerName **(SN)**	Campo de texto	sí	Especifica el nombre y el puerto que el servidor utiliza para identificarse
7	ServerAlias **(SAl)**	Campo de texto	sí	Especifica un alias como otra manera de identificar el servidor.
8	ErrorLog **(EL)**	Campo de texto	no	Ruta del fichero con los registros de errores.
9	CustomLog **(CL)**	Campo de texto	no	Especifica primero el archivo donde se anotan las peticiones hechas al servidor y en segundo lugar el tipo de anotación según se haya definido.

Juego de datos aprobar

Tabla 20: Juego de datos a probar RF Adicionar VirtualHost

Id del escenario	Escenario	Variable 1	Variable 2	Variable 3	Variable 4	Variable 5	Variable 6	Variable 7	Variable 8	Variable 9	Respuesta del sistema
		(NF)	(IP)	(P)	(DR)	(SAd)	(SN)	(SAl)	(EL)	(AL)	
EP 1.1	Adicionar VirtualHost correctamente	pepe	10.8.123.231	80	/home/juanca/VH/	jcmatos@uci.cu	prueba.uci.cu	test.uci.cu	/home/juanca/VH/logs/	/home/juanca/VH/logs/	El sistema muestra el mensaje de información: "*El* VirtualHost *se ha creado satisfactoriamente*" y vuelve a la interfaz Repositorio.

EP 1.2	.Adicionar VirtualHost incorrectamente.	*Vacío*	*Vacío*	*Vacío*	*Vacío*	*Vacío*	*Vacío*	*Vacío*	*Vacío*	*Vacío*	NA
		Caracteres Incorrectos	*Caracteres Incorrectos*	*Caracteres Incorrectos*	*Caracteres Incorrectos*	*Caracteres Incorrectos*	*Caracteres Incorrectos*	*Caracteres Incorrectos*	*Caracteres Incorrectos*	*Caracteres Incorrectos*	El sistema no permite realizar la operación y muestra un mensaje de error en los tooltips de los campos según corresponda.
		(pepe)***IVacióI***(Caracteres	(10.8.123.231)***IVacióI***(Caracteres Incorrectos)	(80)***IVacióI***(Caracteres	(/home/juanca/VH/)***IVacióI***(Caracteres	(jcmatos@uci.cu)***IVacióI***(Caracteres Incorrectos)	(prueba.uci.cu)***IVacióI***(Caracteres Incorrectos)	(test.uci.cu)***IVacióI***(Caracteres Incorrectos)	(/home/juanca/VH/logs/)***IVacióI***(Caracteres	(/home/juanca/VH/logs/)***IVacióI***(Caracteres	El sistema no permite realizar la operación y muestra un mensaje de error en los

		Incorrectos)		Incorrectos)	Incorrectos)				Incorrectos)	Incorrectos)	tooltips de los campos según corresponda.
EP 1.3	Cancelar operación	(pepe)*IVaciól* (Caracteres Incorrectos)	(10.8.123.231)*IVaciól*(Caracteres Incorrectos)	(80) *IVaciól*(Caracteres Incorrectos	(/home/juanca/VH/)*IVaciól*(Caracteres Incorrectos)	(jcmatos@uci.cu)*IVaciól*(Caracteres Incorrectos)	(prueba.uci.cu)*IVaciól*(Caracteres Incorrectos)	(test.uci.cu)*IVaciól*(Caracteres Incorrectos)	(/home/juanca/VH/logs/)*IVaciól*(Caracteres Incorrectos)	(/home/juanca/VH/logs/)*IVaciól*(Caracteres Incorrectos)	NA

Anexo 5

Característica del hardware y versión del software utilizado en los escenarios de pruebas.

Servidores Físicos	RAM	CPU Intel Xeon	Interfaces	HDD	SO	RAID
Blade Huawei 9000	160 Gb	32 (2.8GHz)	8 (Gb Ethernet)	2 (600GB)	Proxmox VE (Debian)	1
Servidor Virtual	**RAM**	**CPU Intel Xeon**	**Interfaces**	**HDD**	**SO**	**RAID**
MV KVM	8 GB	6 (2.8GHz)	1 (Gb Ethernet)	1 (120 GB)	Debian 8	-

Hardware complementario:

- **Switch Huawei 7703,** 48 puertos, 1 Gb Ethernet.
- **Cable Fibra Óptica** 1Gb

Versión del software utilizado en los escenarios de pruebas virtuales:

Software	Versión
Proxmox	4.4
Debian	8
KVM	2.1.3
JMeter	2.4

Anexo 6

P	0.99	0.95	0.90	0.85	0.80	0.75	0.30	0.20	0.15	0.10	0.05	0.025	0.01	0.005	0.001
v=1	0.000	0.004	0.016	0.036	0.064	0.102	1.074	1.642	2.072	2.706	3.841	5.024	6.635	7.879	10.828
2	0.020	0.103	0.211	0.325	0.446	0.575	2.408	3.219	3.794	4.605	5.991	7.378	9.210	10.597	13.816
3	0.115	0.352	0.584	0.798	1.005	1.213	3.665	4.642	5.317	6.251	7.815	9.348	11.345	12.838	16.266
4	0.297	0.711	1.064	1.366	1.649	1.923	4.878	5.989	6.745	7.779	9.488	11.143	13.277	14.860	18.467
5	0.554	1.145	1.610	1.994	2.343	2.675	6.064	7.289	8.115	9.236	11.070	12.833	15.086	16.750	20.515
6	0.872	1.635	2.204	2.661	3.070	3.455	7.231	8.558	9.446	10.645	12.592	14.449	16.812	18.548	22.458
7	1.239	2.167	2.833	3.358	3.822	4.255	8.383	9.803	10.748	12.017	14.067	16.013	18.475	20.278	24.322
8	1.646	2.733	3.490	4.078	4.594	5.071	9.524	11.030	12.027	13.362	15.507	17.535	20.090	21.955	26.124
9	2.088	3.325	4.168	4.817	5.380	5.899	10.656	12.242	13.288	14.684	16.919	19.023	21.666	23.589	27.877
10	2.558	3.940	4.865	5.570	6.179	6.737	11.781	13.442	14.534	15.987	18.307	20.483	23.209	25.188	29.588
11	3.053	4.575	5.578	6.336	6.989	7.584	12.899	14.631	15.767	17.275	19.675	21.920	24.725	26.757	31.264
12	3.571	5.226	6.304	7.114	7.807	8.438	14.011	15.812	16.989	18.549	21.026	23.337	26.217	28.300	32.909
13	4.107	5.892	7.042	7.901	8.634	9.299	15.119	16.985	18.202	19.812	22.362	24.736	27.688	29.819	34.528
14	4.660	6.571	7.790	8.696	9.467	10.165	16.222	18.151	19.406	21.064	23.685	26.119	29.141	31.319	36.123
15	5.229	7.261	8.547	9.499	10.307	11.037	17.322	19.311	20.603	22.307	24.996	27.488	**30.578**	32.801	37.697
16	5.812	7.962	9.312	10.309	11.152	11.912	18.418	20.465	21.793	23.542	26.296	28.845	32.000	34.267	39.252
17	6.408	8.672	10.085	11.125	12.002	12.792	19.511	21.615	22.977	24.769	27.587	30.191	33.409	35.718	40.790
18	7.015	9.390	10.865	11.946	12.857	13.675	20.601	22.760	24.155	25.989	28.869	31.526	34.805	37.156	42.312
19	7.633	10.117	11.651	12.773	13.716	14.562	21.689	23.900	25.329	27.204	30.144	32.852	36.191	38.582	43.820
20	8.260	10.851	12.443	13.604	14.578	15.452	22.775	25.038	26.498	28.412	31.410	34.170	37.566	39.997	45.315
21	8.897	11.591	13.240	14.439	15.445	16.344	23.858	26.171	27.662	29.615	32.671	35.479	38.932	41.401	46.797
22	9.542	12.338	14.041	15.279	16.314	17.240	24.939	27.301	28.822	30.813	33.924	36.781	40.289	42.796	48.268
23	10.196	13.091	14.848	16.122	17.187	18.137	26.018	28.429	29.979	32.007	35.172	38.076	41.638	44.181	49.728
24	10.856	13.848	15.659	16.969	18.062	19.037	27.096	29.553	31.132	33.196	36.415	39.364	42.980	45.559	51.179
25	11.524	14.611	16.473	17.818	18.940	19.939	28.172	30.675	32.282	34.382	37.652	40.646	44.314	46.928	52.620
26	12.198	15.379	17.292	18.671	19.820	20.843	29.246	31.795	33.429	35.563	38.885	41.923	45.642	48.290	54.052
27	12.879	16.151	18.114	19.527	20.703	21.749	30.319	32.912	34.574	36.741	40.113	43.195	46.963	49.645	55.476
28	13.565	16.928	18.939	20.386	21.588	22.657	31.391	34.027	35.715	37.916	41.337	44.461	48.278	50.993	56.892
29	14.256	17.708	19.768	21.247	22.475	23.567	32.461	35.139	36.854	39.087	42.557	45.722	49.588	52.336	58.301
30	14.953	18.493	20.599	22.110	23.364	24.478	33.530	36.250	37.990	40.256	43.773	46.979	50.892	53.672	59.703

Figura 24: Tabla de Distribución Chi-cuadrado