Universidad de las Ciencias Informáticas
Departamento de Componentes, TLM, Facultad 2

Sistema para la administración y monitoreo de servidores PostgreSQL

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

Autores: Barbara Dayanne Acosta Caldevilla
Jesús Martínez Méndez
Tutor: Ing. Osmar Capote Vázquez

La Habana, 2018

"Si algo es lo suficientemente importante, incluso si las probabilidades están en tu contra, de todas formas deberías hacerlo."

Elon Musk

Dedicatoria

Barbara

A mi familia y a mi Jesu.
A todos ustedes les dedico este logro por ser los protagonistas de este sueño hecho realidad, no tengo como pagarles todo lo que han hecho por mí. Por darme fuerza para continuar cada día, pues todo esfuerzo tiene en esta vida una recompensa.

Jesús

El resultado de este Trabajo de Diploma se lo dedico a las personas más importantes de mi vida: mi familia.

Agradecimientos

Barbara

Quiero agradecerles a todas las personas que me ayudaron a realizar este sueño. A mi mamá Dayami y a su esposo Eloy, a mi papá Osniel y a su esposa Marielis. Por ayudarme y apoyarme en las decisiones que he tomado en la vida, por aconsejarme, por ser mis guías. A mis tres hermanos Osnie, Aimeé y Ale y a José Pedro, por quererme sin condiciones, por contar conmigo como la hermana mayor. A mis abuelos Teresita (mi Tayta) y Papá Minito, por darme el cariño y la educación, por apoyarme, por aplaudirme mis triunfos y ayudarme a levantar en mis fracasos, por ser pilares importantes, gracias. A mi abuela Esperanza por recibirme siempre feliz en su casa, por quererme sin condiciones gracias y a mi abuelo Pipo porque siempre me ayudó en lo que pudo, me dio consejos y apoyó en este largo proceso. A mi bis abuela Minerva (Mi conchita) porque desde que comencé en esta universidad le pidió a Dios que le diera vida para ayudarme económicamente, por malcriarme tanto, por todo eso y mucho más te dedico este logro. A mis tíos Mayra y Jesús, que me acogieron con amor y cariño como una hija más, por ayudarme en este tiempo y por apoyarme siempre. A mis tíos Daymí y Angelín porque desde siempre me ha cuidado como un hermano mayor, porque me apoyan incondicionalmente sin esperar nada a cambio. A mi tío Carlito y su esposa Yamirka por ser como un padre para mí, por quererme tanto, por ayudarme siempre que lo he necesitado, por acogerme en su casa siempre que lo necesité, por todos tus gestos cariñosos, mil gracias de verdad. A mi primito Favian, mi gordo lindo. A mis primas Yurisbel y Anabel y sus hermosas familias. A mis primos Leyanis, Leyenis y Yankii. A mi familia cumanayaguense. A todos, familia, muchas gracias de corazón.

Muchas son las personas que en el transcurso de estos años me han apoyado, quiero agradecerles a todos mis compañeros de grupo, en especial a Jesús que lo mencionaré de último no porque sea el menos sino el más importante, a Nexty, por ser un amigo comprensible y paciente conmigo porque de verdad yo, soy yo, a Brenda y Devorat, a Claudia mi compañera de dominó y de otras tareas, Ilianis por tocar lo que no debía y hacerme pasar momentos de muchas risas, a Wendy por su ayuda cuando la necesitaba y siempre estar inventando. A Emilio porque con su carisma muchas veces nos alegró el alma, a Lisardo, a Ramón, por los momentos felices y a Reybel cafeínaaa!!! gracias por los cafés a toda hora, por las conversaciones de relajación para después continuar tesiando, por los consejos. A todos, muchas gracias porque con ustedes compartí cosas y momentos que nunca olvidaré, estos años nunca se olvidan. A los profesores que me impartieron las asignaturas en todos estos años, a todos muchas gracias. Gracias a mi tutor, Osmar Capote Vázquez, por darme su ayuda cuando la necesité y la posibilidad de realizar este trabajo. A los compañeros de la UCI, Darmys gracias, tu no estuviste en el intermedio, pero si en el principio y el final de este proceso tan difícil y creo que mereces un reconocimiento especial, aunque a

veces tenía ganas de tirarte por el balcón, muchas veces me ayudaste y apoyaste, a Oswald por estar siempre arriba del invento y por tenernos siempre en cuenta, a Lenia, Aliuska, Leydis y Yoandi y los demás muchachos de Sandino, porque a veces nos veíamos cuando nos íbamos o veníamos de la casa, pero siempre supe que estaban ahí por si los necesitaba. A mis compañeras de apartamento Brenda, Devorat, Eileen, Jessica y Erlin gracias por soportarme estos años, muchas veces tuvimos diferencias, pero siempre lo superamos, creo que eso sucede hasta en las mejores familias. Con ustedes compartí muchos momentos inolvidables a todas muchas gracias.

A mi compañero de tesis, tengo tantas cosas que agradecerte que este momento no alcanza. Jesús tú fuiste mi compañero de aula, eres mi amigo, mi compañero de vida, mi ejemplo, mi guía, mi ídolo, TE AMO. Muchas gracias por tener tanta paciencia, por ser tan comprensivo, amable, cariñoso, por alegrarme cuando estaba triste, por darme ánimos cuando no los tenía, por explicarme las cosas una y otra y otra y otra vez cuando no entendía o no estaba para entender, por siempre tratarme con delicadeza a pesar de que yo no lo hiciera. Gracias.

Jesús

A la Revolución por darme la oportunidad de estudiar y formarme como ingeniero en la UCI. A mis padres por el apoyo incondicional y la confianza brindada durante mis años de estudio. A mi hermana y mis sobrinos por preocuparse por mí en estos años de estudio. A mi hermano y su familia por el cariño y la preocupación. A mi abuela por la preocupación y el apoyo. A mi tía por ser como una madre y estar pendiente de mí en todo momento. A mis primos por ser como hermanos y preocuparse por mí durante la carrera. A mi familia de Manicaragua por estar pendientes de mí y apoyarme. A mi cuñado y su familia, a Ramón, Marci, Mildrey, Arddiel, Leydianis y su nena por la preocupación y el apoyo incondicional. A mi familia pinareña por acogerme como un hijo. A Néstor y Migue por ser excelentes amigos. A mis amigos de la UCI, mis compañeros de grupo, a mis compañeros de apartamento gracias por la amistad sincera y por los momentos que compartimos juntos. A los profes de todos estos años gracias por su dedicación. A mi tutor por guiarme en este proceso y por transmitirme sus experiencias. Por último a mi compañera tesis. Por estar siempre a mi lado, cuidarme y quererme tanto, apoyarme en todo momento. Por los buenos momentos en los que convivimos. Eres una gran persona y me encanta tenerte a mi lado. TE AMO.

A TODOS AQUELLOS QUE DE UNA FORMA U OTRA SIEMPRE HAN PUESTO SU GRANITO DE ARENA EN MI FORMACIÓN COMO PERSONA, SER HUMANO Y PROFESIONAL. GRACIAS DESDE LO PROFUNDO DE MI CORAZÓN.

Declaración de autoría

Declaramos ser autores de la presente tesis titulada: Sistema para la administración y monitoreo de servidores PostgreSQL y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales sobre esta, con carácter exclusivo.

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

Barbara Dayanne Acosta Caldevilla
Autor

Jesús Martínez Méndez
Autor

Ing. Osmar Capote Vázquez
Tutor

Resumen

El presente trabajo tiene por objetivo la descripción de una propuesta que mejore el proceso de administración y monitoreo de los servidores PostgreSQL que se utilizan dentro del Departamento de Componentes del Centro Telemática de la Facultad 2. Actualmente en dicho Centro de Datos las configuraciones a servidores PostgreSQL necesitan un cúmulo de tareas que resultan complejas, debido a que no se cuenta con una herramienta informática gráfica que los administre y monitoree de forma remota y centralizada. Teniendo en cuenta esta necesidad, se ha modelado e implementado un sistema web que permitirá centralizar el proceso que se lleva a cabo para administrar y monitorear los servidores PostgreSQL dentro del departamento.

Para guiar el desarrollo de la gestión del mismo se utilizó Programación Extrema como metodología de desarrollo de software, la cual proporcionó los artefactos necesarios para obtener el sistema deseado. Como producto final, se obtuvo una herramienta capaz de administrar y monitorear de forma centralizada los servidores PostgreSQL que se utilizan en el Departamento de Componentes.

Palabras clave: PostgreSQL, servidores, sistema, web, administrar, monitorear.

Abstract

The present work has as objective the description of a proposal that improves the process of administration and monitoring of the PostgreSQL servers that are used within the Components Department of the Telematics Center of the Faculty 2. Currently in said Data Center the configurations to PostgreSQL servers they need an accumulation of tasks that are complex, because they do not have a graphical tool that manages and monitors them remotely and centrally. However, there is a need for this process to be carried out through a system created by the department that allows managing PostgreSQL servers. Taking into account this need, a system has been modeled and implemented for the web that will allow centralizing the process that is carried out to administer and monitor the PostgreSQL servers within the department. To guide the development of its management, Extreme Programming was used as a methodology, which provided the necessary artifacts to obtain the desired system. As a final product, a tool capable of centrally managing and monitoring the PostgreSQL servers used in the Components Department was obtained.

Keywords: PostgreSQL, servers, system, web, manage, monitor..

Índice general

Índice de figuras

Índice de tablas

Lista de códigos fuentes

Introducción

Durante los últimos años se ha registrado un incremento considerable a nivel global de la influencia de la informática en los diferentes procesos de la sociedad. Instituciones gubernamentales, entidades empresariales, sectores educativos, sectores de salud, la telefonía móvil o sencillamente el ocio, por solo mencionar algunos, son campos donde el desarrollo constante de nuevas aplicaciones y propuestas informáticas va en índice creciente. Pues se van modificando procesos y conceptos que hasta entonces se concebían de otra forma. Las Tecnologías de Información y Comunicación (TIC) juegan un papel decisivo, lo que posibilita alcanzar una visión integral y globalizante, pues están presentes en todas las actividades y procesos que se realizan en las organizaciones.

Con el acelerado avance de la sociedad moderna en lo que va del siglo XXI y el desarrollo empresarial alcanzado en esta etapa, surge la necesidad de almacenar los datos. Desde que los datos dieron el salto de lo analógico a lo digital su crecimiento ha aumentado considerablemente (Parnell, 2015). Interactuar con los datos, clasificarlos en pequeños grupos que describan sus características principales, basándose en la similitud o diferencia entre ellos es una de las principales funciones de las bases de datos. Las cuales necesitan de servidores que las gestionen.

Los servidores de base de datos, surgen por la necesidad que tienen las empresas para manejar grandes y complejos volúmenes de datos. Dentro de los servidores de base de datos utilizados actualmente por su código abierto, simplicidad y seguridad se encuentra PostgreSQL. Muchos sistemas en los que la consistencia y persistencia de las bases de datos es fundamental, utilizan como opción a dichos servidores, delegando en los mismos la responsabilidad de la gestión y almacenamiento de la información.

Para nuestro país, es inminente un desarrollo tecnológico y en contribución al fortalecimiento de la soberanía tecnológica cubana se adoptó PostgreSQL como Sistema Gestor de Bases de Datos (SGBD). Es un SGBD objeto-relacional de código abierto. Se ejecuta en la mayoría de los sistemas operativos, incluidos Linux, UNIX y Windows (The PostgreSQL Global Development Group, 2017).

La Universidad de las Ciencias Informáticas (UCI) se desempeña como principal impulsor en la aplicación de las tecnologías libres en el país. Tiene la misión de formar profesionales comprometidos con su Patria, altamente calificados en la rama de la informática, y producir aplicaciones y servicios informáticos, a partir del vínculo docencia - investigación - producción como modelo de formación, sirviendo de soporte a la industria cubana del software. Cuenta con centros de desarrollo de software que realizan proyectos, con gran impacto en la informatización de la sociedad cubana.

El Centro de Telemática (TLM) adscrito a la Facultad 2, desarrolla sistemas y servicios informáticos

integrales para las Telecomunicaciones y la Seguridad Informática, altamente comprometido con la Revolución, capaz de integrar los procesos docentes, productivos e investigativos con alto nivel, contando con un personal especializado en dichas áreas (Ciencias Informáticas, 2018). De acuerdo a su estructura interna está organizado por dos departamentos productivos. El departamento de Componentes, cuenta con proyectos como el Centro de Datos.

Actualmente en dicho Centro de Datos las configuraciones a servidores PostgreSQL necesitan un cúmulo de tareas que resultan complejas. Los administradores deben conectarse al servicio a través de la consola de comandos (terminal). Además, deben conocer varios comandos, ficheros, rutas de ficheros y parámetros disponibles en el servicio; lo que la mayoría de las veces ralentiza el trabajo. Debido a que no se cuenta con una herramienta gráfica que administre y monitoree de forma remota y centralizada los servidores PostgreSQL.

Para dar solución a la problemática descrita anteriormente se presenta como **problema a resolver**: ¿Cómo administrar y monitorear de manera centralizada el SGBD PostgreSQL?

Por ello el **objeto de estudio** lo constituye la administración y monitoreo del SGBD PostgreSQL.

Para dar solución al problema planteado en la presente investigación se define como **objetivo general**: Desarrollar un sistema que permita administrar y monitorear de manera centralizada el SGBD PostgreSQL.

Desglosándose en los siguientes **objetivos específicos**:

- Elaborar el marco teórico de la investigación.
- Realizar un estudio del SGBD PostgreSQL.
- Definir las funcionalidades que debe cumplir el sistema.
- Implementar un sistema informático para la administración y el monitoreo de los servidores PostgreSQL que cumpla con los requerimientos definidos.
- Aplicar pruebas al sistema resultante para garantizar la calidad y fiabilidad del producto.

El **campo de acción** de esta investigación lo representan los sistemas informáticos para la administración y el monitoreo del SGBD PostgreSQL.

Preguntas científicas:

- ¿Cuáles son las posiciones teóricas en cuanto a los sistemas para la administración y monitoreo de los servidores PostgreSQL?
- ¿Cuál es el estado real de desarrollo alcanzado en la informatización de las áreas involucradas en el flujo y manejo de información de los servicios de PostgreSQL?
- ¿Cómo diseñar un sistema para la administración y monitoreo que se ajuste a las características de los servidores PostgreSQL?

Para lograr el objetivo planteado, se proponen las siguientes **tareas de investigación**:

- Revisión de los sistemas homólogos existentes para conocer aspectos regulares en el diseño del sistema enfocados a los servidores PostgreSQL.
- Selección de las herramientas informáticas y metodología de desarrollo de software para realizar la implementación del sistema.

- Realización de un estudio de los archivos de configuración del SGBD PostgreSQL, pg_hba.conf y postgresql.conf para conocer su estructura.
- Identificación de las principales funcionalidades del sistema para la posterior implementación del mismo.
- Implementación del sistema para la administración y el monitoreo del SGBD PostgreSQL.
- Realización de pruebas para la validación del funcionamiento del sistema.

Método científico de investigación

El método científico de investigación es la forma de abordar la realidad, de estudiar la naturaleza, la sociedad y el pensamiento, con el propósito de descubrir su esencia y sus relaciones. Se clasifican en teóricos y empíricos, los cuales están dialécticamente relacionados (Hernández León y Coello González, 2012).

Métodos teóricos

Permiten estudiar las características del objeto de investigación que no son observables directamente, facilitan la construcción de modelos y crean las condiciones para ir más allá de las características fenomenológicas y superficiales de la realidad (ibíd.).

Analítico-Sintético

Análisis: Permite la descomposición mental del objeto o fenómeno en sus múltiples relaciones o componentes para facilitar su estudio.

Síntesis: Establece mentalmente la unión entre las partes previamente analizadas, permite descubrir sus características generales y las relaciones esenciales entre ellas (ibíd.).

Aplicado para analizar la información obtenida en el proceso de investigación en el centro de desarrollo TLM de la facultad 2, posibilitando identificar aspectos que puedan ser empleados en el desarrollo de la presente investigación. Además, para el análisis de la información existente dentro y fuera del país, acerca de los servidores de PostgreSQL. Identificando así, conceptos, definiciones, avances y otros elementos concluyentes de utilidad para la investigación en cuestión.

Inducción-Deducción

Inducción: Es una forma de razonamiento a través del cual se pasa de un conocimiento de cosas particulares a un conocimiento más general que va a reflejar lo que hay de común en esos fenómenos individuales.

Deducción: Es un procedimiento que permite, a partir de conocimientos generales, inferir casos particulares por un razonamiento lógico (ibíd.).

Se utiliza para analizar el comportamiento y las características que poseen las herramientas para la administración de servidores PostgreSQL. Reunir elementos para llegar a una conclusión del porque es necesario implementar un nuevo sistema que cumpla con los requerimientos de la aplicación que se desea implementar.

Métodos empíricos

Describen y explican las características fenomenológicas del objeto, representan un nivel de la investigación cuyo contenido procede de la experiencia y es sometido a cierta elaboración racional (ibíd.).

Observación

Se utilizó, como método básico para recopilar información pues fue el primer método científico empleado

(Hernández León y Coello González, 2012). En esta investigación se utiliza para adquirir conocimientos y recopilar información de sistemas realizados con anterioridad para la administración y el monitoreo de SGBD PostgreSQL.

Entrevista

La entrevista con el cliente y los trabajadores (que se van a relacionar con la aplicación web) fue una de las fundamentales fuentes de recopilación de información, para entender qué problemas existen y saber cómo solucionarlos. Además de aplicar esta técnica también con trabajadores de otros centros productivos que están inmersos en el tema de administración de servidores de PostgreSQL.

En esta investigación se realiza el **modelado** mediante diagramas. Esto permite reflejar la estructura, las relaciones y características de la solución propuesta. En este caso, con el uso de Notación para el Modelado de Procesos de Negocio (BPMN, por sus siglas en inglés), se facilita también el diseño de clases necesario para la implementación del sistema.

El contenido de este documento está estructurado como se muestra a continuación:

Capítulo 1 "Fundamentación Teórica"

En este capítulo se fundamentan los elementos teóricos necesarios que sustentan la investigación. Se analizan las tendencias actuales en cuanto a los SGBD PostgreSQL. Se realiza un estudio del estado del arte de sistemas que administran y monitorean servidores PostgreSQL mediante el análisis de herramientas similares existentes a nivel nacional e internacional. Se selecciona el entorno de desarrollo y los lenguajes y frameworks para la construcción de la solución, así como la metodología de desarrollo de software a utilizar.

Capítulo 2 "Análisis y diseño del sistema"

En este capítulo se documenta el proceso de desarrollo del sistema de acuerdo a lo establecido por la metodología utilizada. Se presenta la propuesta de solución del sistema para administrar y monitorear servidores PostgreSQL. Se delimitan las funcionalidades del sistema a través de requisitos funcionales. Se especifican las Historias de Usuario (HU), así como todo lo referente a la arquitectura del sistema, además se presentan los patrones de diseño utilizados y las tarjetas Clase-Responsabilidad-Colaborador (CRC, por sus siglas en inglés).

Capítulo 3 "Implementación y pruebas"

En este capítulo se describe la estructura del código implementado mostrando las tareas de ingeniería, así como el estándar de codificación utilizado, y se presentan los diferentes tipos de pruebas realizadas al sistema y los resultados arrojados por estas. Para finalizar se presentan las conclusiones, las recomendaciones, las referencias bibliográficas, la bibliografía consultada, un glosario de términos y los anexos.

CAPÍTULO 1

Fundamentación Teórica

1.1. Introducción

En el presente capítulo se estará proporcionando una breve descripción de los SGBD PostgreSQL. Se presentará, un estudio de los principales sistemas informáticos existentes para la administración y monitoreo del SGBD PostgreSQL, recorriendo el ámbito internacional y nacional. Además, se realizará un estudio de las principales tecnologías informáticas y metodologías para el desarrollo de software en función de un análisis de las tendencias actuales.

1.2. Bases de datos

Colección de datos, donde los datos están lógicamente relacionados entre sí, tienen una definición y descripción comunes y están estructurados de una forma particular. Una base de datos es también un modelo del mundo real y, como tal, debe poder servir para toda una gama de usos y aplicaciones (Gómez Fuentes, 2013).

1.3. Sistema Gestor de Bases de Datos

Un SGBD consiste en una colección de datos interrelacionados y un conjunto de programas para acceder a dichos datos. La colección de datos, normalmente denominada base de datos, contiene información relevante para una empresa. El objetivo principal de un SGBD es proporcionar una forma de almacenar y recuperar la información de una base de datos de manera que sea tanto práctica como eficiente.

Los sistemas de bases de datos se diseñan para gestionar grandes cantidades de información. La gestión de los datos implica tanto la definición de estructuras para almacenar la información como la provisión de mecanismos para la manipulación de la información. Además, los sistemas de bases de datos deben proporcionar la fiabilidad de la información almacenada, a pesar de las caídas del sistema o los intentos de

acceso sin autorización. Si los datos van a ser compartidos entre diversos usuarios, el sistema debe evitar posibles resultados anómalos (Silberschatz; Korth y Sudarshan, 2002).

1.4. El SGBD PostgreSQL

Es un SGBD objeto-relacional de código abierto. Cuenta con más de 15 años de desarrollo activo y una arquitectura probada que se ha ganado una sólida reputación de fiabilidad e integridad de datos. Se ejecuta en la mayoría de los sistemas operativos, incluidos Linux, UNIX y Windows. Admite conjuntos de caracteres internacionales, codificaciones de caracteres *multibyte, Unicode,* y es compatible con el formateo y la distinción entre mayúsculas y minúsculas. Es altamente escalable tanto en la gran cantidad de datos que puede administrar como en la cantidad de usuarios simultáneos que puede manejar. Hay sistemas PostgreSQL activos en entornos de producción que administran más de 4 terabytes de datos (The PostgreSQL Global Development Group, 2017).

1.4.1. Arquitectura de PostgreSQL

PostgreSQL está basado en una arquitectura cliente-servidor. El programa servidor se llama postgres y entre los muchos programas cliente se tiene, por ejemplo, pgaccess (un cliente gráfico) y psql (un cliente en modo texto).

Un proceso servidor postgres puede atender exclusivamente a un solo cliente; es decir, hacen falta tantos procesos servidor postgres como clientes haya. El proceso postmaster es el encargado de ejecutar un nuevo servidor para cada cliente que solicite una conexión.

Se llama sitio al equipo anfitrión (host) que almacena un conjunto de bases de datos PostgreSQL. En un sitio se ejecuta solamente un proceso postmaster y múltiples procesos postgres. Los clientes pueden ejecutarse en el mismo sitio o en equipos remotos conectados por TCP/IP (Dataprix, 2018).

1.4.2. Archivo pg_hba.conf

La autenticación de los clientes está controlada por un archivo de configuración, que tradicionalmente se denomina pg_hba.conf y se almacena en el directorio de datos del clúster de la base de datos. Un archivo pg_hba.conf predeterminado se instala cuando se inicializa el directorio de datos. Sin embargo, es posible colocar el archivo de configuración de autenticación en otro lugar.

El formato general del archivo pg_hba.conf es un conjunto de registros, uno por línea. Las líneas en blanco se ignoran, al igual que cualquier texto después del carácter de comentario #. Un registro se compone de una serie de campos que están separados por espacios y / o pestañas. Los campos pueden contener espacios en blanco si el valor del campo tiene comillas dobles. Entrecomillar una de las palabras clave en los campos *database*, *user* o *address* (por ejemplo, *all* o *replication*) hace que la palabra pierda su significado especial, y solo coincida con una base de datos, usuario o servidor con ese nombre.

Cada registro especifica un tipo de conexión, un rango de dirección IP del cliente (si el tipo de conexión no es local), un nombre de base de datos, un nombre de usuario y el método de autenticación que se utilizará para las conexiones que coinciden con estos parámetros. El primer registro que coincida con un tipo de conexión, dirección de cliente, base de datos solicitada y nombre de usuario, se utiliza para realizar la autenticación. Si se elige un registro y la autenticación falla, no se consideran los registros posteriores. Si no coincide ningún registro, se deniega el acceso.

Un registro puede tener uno de los siguientes siete formatos (PostgreSQL, 2016):

- *local databas user auth-method [auth-options]*
- *host database user address auth-method [auth-options]*
- *host database user address auth-method [auth-options]*
- *hostssl database user address auth-method [auth-options]*
- *host database user IP-address IP-mask auth-method [auth-options]*
- *hostssl database user IP-address IP-mask auth-method [auth-options]*
- *hostnossl database user IP-address IP-mask auth-method [auth-options]*

1.4.3. Archivo postgresql.conf

El archivo postgresql.conf normalmente se guarda en el directorio de datos y se instala una copia predeterminada cuando se inicializa el directorio del clúster de la base de datos. Hay muchos parámetros de configuración en este fichero que afectan el comportamiento del sistema de base de datos. Todos los nombres de los parámetros no distinguen entre mayúsculas y minúsculas. Cada parámetro toma un valor que puede ser de los siguientes tipos: booleano, cadena, entero, punto flotante o enumerado (*enum*).

Se especifica un parámetro por línea. El signo igual entre nombre y valor es opcional. El espacio en blanco es insignificante (excepto dentro de un valor de parámetro que esté entrecomillado) y las líneas en blanco se ignoran. Los caracteres de número (#) indican que el resto de la línea es un comentario. Los valores de parámetros que no son identificadores simples o números deben ser entrecomillados por una sola comilla. Para insertar una comilla simple en un valor de parámetro, el valor debe estar entre dos comillas o comillas invertidas.

Un ejemplo de cómo se vería este archivo es:

 # This is a comment
log_connections = yes
log_destination = 'syslog'
search_path = '"$ user", public'
shared_buffers = 128MB

1.5. Estudio de sistemas informáticos para la administración y monitoreo del SGBD PostgreSQL.

Es necesario realizar un estudio de los sistemas homólogos existentes a nivel internacional y nacional para lograr una mejor comprensión de las características del sistema a desarrollar, identificando similitudes al propuesto en la presente investigación. A continuación, se expone el análisis desarrollado de los sistemas homólogos.

1.5.1. Internacional

Webmin

Webmin representa una herramienta para la configuración mediante la web, para GNU/Linux y otros sistemas Unix. Posibilita modificar y controlar muchas aplicaciones de código abierto, como el servidor web Apache, PHP Hypertext Preprocessor (PHP), MySQL, PostgreSQL, Domain Name System (DNS), Secure SHell (SSH), Lightweight Directory Access Protocol (LDAP), Dynamic Host Configuration Protocol (DHCP), File Transfer Protocol (FTP), entre otros. Permite el control de todos los servidores no solo de forma local, sino también de forma remota (Webmin, 2017).

Esta herramienta posee tres características dentro de muchas que enriquecen la investigación. Estas son:

- Instala y desinstala postgresql.
- Crea reglas de acceso.
- Controla el servicio postgresql.

Figura 1.1. Administración de PostgreSQL en Webmin

ManageEngine Applications Manager

ManageEngine Applications Manager ayuda a monitorear el rendimiento y la disponibilidad de aplicaciones tales como aplicaciones web, servidores de aplicaciones, servidores web, bases de datos, servicios de

red, sistemas, etc. Ayuda también a identificar y analizar fallas y problemas de rendimiento en sus aplicaciones antes de afectar a los usuarios finales (ManageEngine, 2017a).

El cliente web de Applications Manager ofrece amplias capacidades de supervisión de PostgreSQL que permiten a los Administradores de Bases de Datos (DBA, por sus siglas en inglés) monitorear proactivamente la disponibilidad y el rendimiento de los servidores de base de datos PostgreSQL críticos para la empresa. La monitorización se realiza de forma continua, las 24 horas del día, los 7 días de la semana, y garantiza que cualquier desviación en el funcionamiento normal de la base de datos PostgreSQL se ponga inmediatamente en conocimiento del DBA.

El cliente web de Application Manager le permite visualizar los datos de supervisión de PostgreSQL desde su nivel de base de datos junto con los niveles de su aplicación. Proporciona métricas integrales e informes exhaustivos que le permiten identificar, analizar y resolver rápidamente los cuellos de botella que afectan el tiempo de respuesta de PostgreSQL.

Algunas de las métricas proporcionadas por Applications Manager para las bases de datos PostgreSQL incluyen (ManageEngine, 2017b):

- Estadísticas de conexión.
- Uso de memoria.
- Detalles de transacciones.
- Estadísticas de consultas.
- Detalles del uso de disco.
- Unidad Central de Procesamiento (CPU, por sus siglas en inglés), memoria y utilización del disco.

A continuación, se muestra una captura de la aplicación.

Figura 1.2. ManageEngine Applications Manager mostrando el estado de PostgreSQL

EMS SQL Manager for PostgreSQL

EMS SQL Manager para PostgreSQL es una herramienta de alto rendimiento para la administración y desarrollo de bases de datos PostgreSQL. Funciona con varias de las versiones de PostgreSQL y soporta las características de este, incluyendo restricciones de exclusión, cláusula '*when*' de *triggers*, funciones de retorno de tabla y otras. SQL Manager para PostgreSQL ofrece muchas herramientas de base de datos poderosas como *Visual Database Designer* para crear bases de datos PostgreSQL en pocos clics, *Visual Query Builder* para construir complicadas consultas PostgreSQL, buen editor Binary Large OBject (BLOB, por sus siglas en inglés) y muchas más funciones útiles para la administración de PostgreSQL. SQL Manager para PostgreSQL tiene una interfaz gráfica de usuario con sistema de asistente bien descrito. Principales características (EMS Database Management Solutions, 2016):

- Soporte completo de PostgreSQL hasta la versión 9.5.
- Gestión de bases de datos y navegación.
- Explorador de bases de datos mejorado para facilitar el manejo de todos los objetos PostgreSQL.
- Gestión de la seguridad.
- Asistentes que realizan tareas de mantenimiento de PostgreSQL.
- Conexión a través del puerto local mediante el túnel SSH.
- Acceso al servidor PostgreSQL a través del protocolo Protocolo de Transferencia de Hipertexto (HTTP, por sus siglas en inglés).
- Diseñador de informes con asistente de construcción de informes.

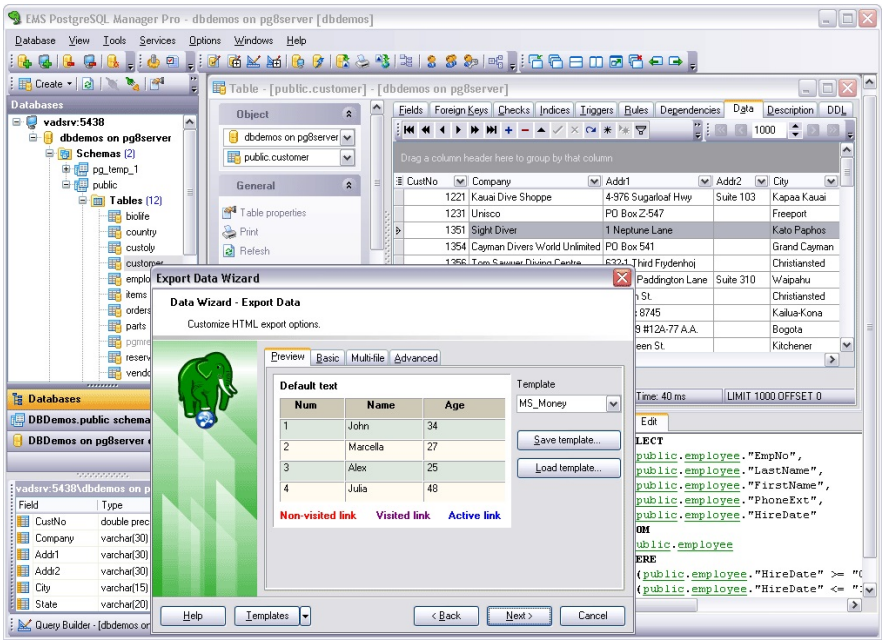

Figura 1.3. Captura del EMS SQL Manager for PostgreSQL

1.5.2. Nacional

Módulo de administración para el servicio PostgreSQL en la Herramienta para la Migración y Administración de Servicios Telemáticos

Dicho módulo fue desarrollado en el año 2014 en la UCI. Permite administrar, a través de una interfaz de usuario, los parámetros de configuración del servidor de base de datos. Esto es posible debido a que la aplicación se conecta vía SSH al servidor y accede a los ficheros de configuración y los modifica de acuerdo a las necesidades del gestor de configuración. Entre las principales funcionalidades que contiene la aplicación se encuentran instalar y desinstalar el servicio, gestionar los ficheros de configuración y el control de acceso. También posibilita garantizar la integridad de los datos, así como administrar el uso de los recursos que se le asignan al servidor, los registros del sistema y las estadísticas de ejecución. Además, realizar ajustes de consultas y revisiones periódicas a las tablas con modificaciones considerables.

Entre las funcionalidades de esta herramienta se encuentran (Fernández Casasayas, 2014):

- Instalar, desinstalar, iniciar, reiniciar, detener y mostrar el servicio PostgreSQL.
- Mostrar y modificar ruta del directorio de almacenamiento de datos.
- Mostrar, eliminar y modificar direcciones de escucha.
- Mostrar y modificar puerto.
- Mostrar y modificar máximo de conexiones permitidas al PostgreSQL.
- Habilitar y deshabilitar la opción de conexiones SSL.
- Habilitar y deshabilitar el cifrado de contraseña.
- Mostrar y modificar cantidad de memoria que el servidor de base de datos utiliza para buffers.
- Crear, listar, modificar y eliminar reglas de control de acceso al servidor PostgreSQL.

Luego de analizar los sistemas expuestos anteriormente se arriba a la conclusión de que es factible realizar el sistema propuesto para administrar y monitorear servidores PostgreSQL porque en el caso de la herramienta Webmin, es necesaria la instalación de esta en cada máquina donde se vaya a configurar PostgreSQL y posee limitadas funcionalidades de administración para PostgreSQL. EMS SQL Manager para PostgreSQL gestiona los archivos de configuración y las reglas de control de acceso al servidor, pero es necesario pagar una licencia para su utilización al igual que para ManageEngine Applications Manager, además, este último solo monitorea PostgreSQL. El Módulo de administración para el servicio PostgreSQL en la Herramienta para la Migración y Administración de Servicios Telemáticos no se encuentra disponible, aunque se cuenta con la documentación. Además, no es posible la integración del sistema de administración y monitoreo de PostgreSQL con dicha aplicación por no estar basada en las mismas tecnologías.

1.6. Metodologías de desarrollo de software

Las metodologías de desarrollo de software son el conjunto de procedimientos, técnicas, herramientas y un soporte documental, que ayuda a los desarrolladores a realizar un nuevo software. La metodología define

quién debe hacer qué, cuándo y cómo debe hacerlo para obtener los distintos productos parciales y finales. Se clasifican en dos tipos: tradicionales y ágiles. Aunque el proceso de desarrollo de software incluye otros aspectos importantes y determinantes, es precisamente la metodología utilizada, el elemento rector a lo largo del mismo. Por tal motivo es que se debe analizar, de acuerdo a las características del software a desarrollar y del equipo de desarrollo, cuál es la más óptima. Por las ventajas que ofrecen las metodologías ágiles, se propone su uso para el desarrollo del trabajo de diploma, específicamente la metodología Programación Extrema (XP, por sus siglas en inglés).

1.6.1. Programación Extrema

La metodología XP, creada por Kent Beck, es una metodología liviana para desarrollar software. Se caracteriza porque planifica, analiza y diseña en un mismo momento y durante el desarrollo. De esta forma se decide si se va por buen camino, y se evita el retroceso en etapas adelantadas, es decir, se trabaja a partir de prueba y error. El equipo está formado por entre dos y doce personas que trabajan en parejas. Es un método simple que desarrolla sólo lo que requiere, permite la retroalimentación constante, la toma de decisiones difíciles y remediar los errores tan pronto como se detectan; existe comunicación continua entre los clientes y el grupo de trabajo (Medina Velandia y López López, 2015).

La presente investigación es un proyecto de corto alcance, de poca duración, se tiene un equipo de desarrollo integrado por pocas personas, y el cliente es parte de la investigación. Por tales motivos el equipo de desarrollo propone el uso de una metodología de desarrollo de software ágil. Dentro de este tipo de metodología de desarrollo de software se escoge utilizar XP:

- Se considera como la adopción de las mejores metodologías de desarrollo de software.
- Los cambios de requisitos sobre la marcha son un aspecto natural.
- Plantea la programación en parejas.
- Apoya la integración del equipo de programación con el cliente o usuario.
- Consta de una alta gama de bibliografía.
- Sus artefactos presentan una descripción del comportamiento del sistema fácil de entender.

1.7. Herramientas y lenguajes informáticos

El desarrollo de un sistema informático no es tarea fácil para un equipo de desarrollo, es por ello que el sistema que se desarrollará estará basado en una estrategia marcaria implementada en la UCI. Dentro de la misma se escoge XILEMA, que se enfoca a software desarrollado para la rama de teleinformática y redes y seguridad informática. Además, establece un conjunto de pautas a tener en cuenta para el desarrollo de productos de software en la UCI.

1.7.1. Python 3.6

Python es un lenguaje de programación creado por Guido van Rossum a principios de los años 90 cuyo nombre está inspirado en el grupo de cómicos ingleses "Monty Python". Es un lenguaje similar a Perl, pero con una sintaxis muy limpia y que favorece un código legible (González Duque, 2011). Cuenta con estructuras de datos eficientes y de alto nivel y un enfoque simple pero efectivo a la programación orientada a objetos. Es un lenguaje que se caracteriza por ser fuertemente tipado, de tipado dinámico, multiplataforma, simple, libre, de fuente abierta y portable.

1.7.2. Lenguaje de Marcado de Hipertexto (HTML)5

Lenguaje de Marcado de Hipertexto (HTML5, por sus siglas en inglés) provee básicamente tres características: estructura, estilo y funcionalidad. Es considerado el producto de la combinación de HTML, Hojas de estilo en cascada (CSS, por sus siglas en inglés) y JavaScript. Estas tecnologías son altamente dependientes y actúan como una sola unidad organizada bajo la especificación de HTML5 (Gauchat, 2012). HTML5 se caracteriza por ser un lenguaje extensible, que se le pueden añadir características, etiquetas y funciones adicionales para el diseño de páginas web, generando un producto vistoso, rápido y sencillo.

1.7.3. Hojas de estilo en cascada (CSS)3

La web demanda diseño y funcionalidad, no solo organización estructural o definición de secciones. En este nuevo paradigma, HTML se presenta junto con CSS y JavaScript como un único instrumento integrado. CSS es un lenguaje que trabaja junto con HTML para proveer estilos visuales a los elementos del documento, como tamaño, color, fondo, bordes, entre otros (ibíd.). El propósito del desarrollo de CSS es separar la estructura y el contenido de la presentación estética en un documento.

1.7.4. JavaScript

JavaScript es un lenguaje interpretado usado para múltiples propósitos, pero solo considerado como un complemento hasta ahora (ibíd.). Principalmente es utilizado para crear páginas web dinámicas. Una página web dinámica es aquella que incorpora efectos como texto que aparece y desaparece, animaciones, acciones que se activan al pulsar botones y ventanas con mensajes de aviso al usuario. Técnicamente, JavaScript es un lenguaje de programación interpretado, por lo que no es necesario compilar los programas para ejecutarlos. En otras palabras, los programas escritos con JavaScript se pueden probar directamente en cualquier navegador sin necesidad de procesos intermedios (Eguíluz Pérez, 2012).

1.7.5. Django 1.11.6

Django es un framework para desarrollo web escrito en Python, que permite construir y mantener aplicaciones web de alta calidad con el mínimo esfuerzo posible. Django permite a los desarrolladores concentrarse

en el código de la aplicación propiamente dicha, manejando tareas repetitivas y complejas propias de la programación web. De esta forma, provee un alto nivel de abstracción para patrones comunes en el desarrollo web, incrementando la calidad de las soluciones, aumentando la productividad y disminuyendo los errores en el código (Holovaty y Kaplan-Moss, 2009).

1.7.6. jQuery 1.9.1

jQuery es una biblioteca de JavaScript, creada inicialmente por John Resig, que permite simplificar la manera de interactuar con los documentos HTML, manipular el árbol Document Object Model (DOM), manejar eventos, desarrollar animaciones y agregar interacción con la técnica Asynchronous JavaScript And XML (AJAX) a páginas web. jQuery consiste en un único fichero JavaScript que contiene las funcionalidades comunes de DOM, eventos, efectos y AJAX. La característica principal de la biblioteca es que permite cambiar el contenido de una página web sin necesidad de recargarla, mediante la manipulación del árbol DOM y peticiones AJAX (Sánchez; Castilla; Escalona; Vargas y Sorí, 2017).

1.7.7. Bootstrap 3.3.1

Bootstrap es un framework que simplifica el proceso de creación de diseños web combinando CSS y JavaScript. Ha sido desarrollado por Twitter. La mayor ventaja es que se pueden crear interfaces que se adapten a los distintos navegadores (*responsive design*) apoyándose en framework potente con numerosos componentes webs que ahorrarán mucho esfuerzo y tiempo.

Bootstrap, es un framework que facilita el desarrollo de la interfaz web y está basado en los estándares de HTML, CSS y JavaScript, llevan incorporado varias plantillas prediseñadas, formulario, botones, menús y otros componentes que facilitan y agilizan el desarrollo frontal de las aplicaciones web (Maldonado y Mauricio, 2017).

1.7.8. Visual Paradigm 8.0

Está diseñado para un amplio rango de usuarios, incluyendo ingenieros de software, analistas de sistema, analistas de negocio, arquitectos de sistema y quienes estén interesados en la construcción de sistemas de software confiables mediante el uso de la Orientación a Objetos. Este software facilita una rápida construcción de aplicaciones de calidad y a un menor coste. Visual Paradigm para UML soporta un conjunto de lenguajes (Java, C ++, PHP, Ada y Python), tanto en generación de código como ingeniería inversa. Entre sus características principales se pueden citar (Visual Paradigm, 2018)

- Facilita la comunicación de todo el equipo de desarrollo mediante el uso de un lenguaje estándar común.
- Posibilita el desarrollo de la ingeniería directa e inversa.
- Durante todo el ciclo de desarrollo el modelo y el código permanecen sincronizados, permitiendo la generación de código a partir de diagramas y viceversa.

- Se encuentra disponible en múltiples versiones y plataformas.

1.7.9. Notación para el Modelado de Procesos de Negocio (BPMN)

BPMN es un estándar de modelado de procesos de negocio, en donde se presentan gráficamente las diferentes etapas del proceso del mismo. BPMN tiene un enfoque en procesos de negocio, UML se enfoca al diseño de software y por lo tanto ambas notaciones son totalmente compatibles. El modelado de negocios, y más específicamente el modelado de procesos de negocio, es la forma idónea para la comunicación entre los usuarios (Ventura y Estefanía, 2013).

1.7.10. PyCharm 4.5.4

PyCharm proporciona la finalización inteligente de códigos, inspecciones de códigos, resaltado de errores sobre la marcha y soluciones rápidas, junto con refactorizaciones automáticas de códigos y capacidades de navegación avanzadas.

La colección de herramientas de PyCharm incluye un depurador y corrector de prueba integrado; una terminal incorporada; integración con un Version Control System (VCS) principal y herramientas de base de datos incorporadas; capacidades de desarrollo remoto con intérpretes remotos y una terminal SSH integrado (JetBRAINS, 2017).

1.7.11. SQLite 3

SQLite es una biblioteca en proceso que implementa un motor de base de datos SQL transaccional independiente, sin servidor y sin configuración. El código para SQLite es de dominio público y, por lo tanto, es gratuito para cualquier uso, comercial o privado. SQLite es un motor de base de datos SQL incorporado. A diferencia de la mayoría de las otras bases de datos SQL, SQLite no tiene un proceso de servidor por separado. SQLite lee y escribe directamente en archivos en el disco. Una base de datos SQL completa con múltiples tablas, índices, disparadores y vistas, está contenida en un solo archivo. El formato de archivo de la base de datos es multiplataforma: puede copiar libremente una base de datos entre sistemas de 32 bits y de 64 bits (SQLite, 2017).

1.8. Conclusiones del capítulo

En este capítulo se han resumido diferentes aspectos teóricos relacionados con la administración y monitoreo del SGBD PostgreSQL posibilitando la preparación del equipo de desarrollo en estos temas. Se realizó un estudio de las principales herramientas que administran y monitorean el SGBD, permitiendo obtener los aspectos positivos que contribuyeron a la concepción del sistema a desarrollar. Es utilizada como guía para el proceso de desarrollo de software la metodología XP. También, se definen las herramientas informáticas a

utilizar en la implementación del sistema, con el fin de desarrollar un sistema multiplataforma, simple, libre y de fuente abierta.

CAPÍTULO 2

Análisis y diseño del sistema

2.1. Introducción

Este capítulo contiene las características asociadas al dominio de la aplicación; la propuesta de solución del sistema a desarrollar, el rol definido para administrar y proveer la seguridad a la aplicación. Las funcionalidades se describen mediante las HU. Se realiza el plan de iteraciones donde se muestran las HU a realizar en cada iteración según su prioridad en el negocio. Se lleva a cabo la realización del plan de entrega, donde se indican las HU que se implementarán para cada entrega de la aplicación propuesta, así como las fechas en las que se realizarán las mismas.

2.2. Proceso de administración de servidores PostgreSQL

El proceso para la administración de un servidor PostgreSQL actualmente comienza con la instalación del servidor, luego el administrador configura, de cada clúster, las reglas de control de acceso en el archivo pg_hba.conf y otras configuraciones como direcciones de escucha, puerto y máxima cantidad de conexiones concurrentes, en el archivo postgresql.conf. Luego reinicia el servicio mediante la consola, y posteriormente se accede al servidor, por una de las aplicaciones clientes que existen. A continuación, se muestra el proceso que se lleva a cabo actualmente para administrar un servidor PostgreSQL.

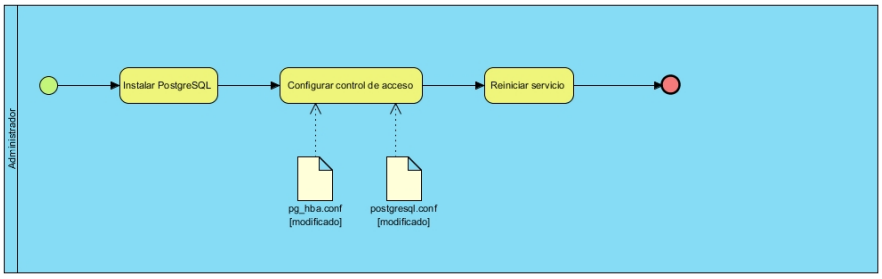

Figura 2.1. Proceso del negocio actual

2.3. Características de la propuesta de solución

Con la implementación de la solución planteada se pretende facilitar la administración y el monitoreo de servidores PostgreSQL dentro del Departamento de Componentes del Centro TLM. El usuario (administrador) accederá al sistema mediante un navegador web, una vez dentro, el sistema le permitirá realizar varias funciones mediante conexiones seguras SSH a las máquinas servidoras. Entre dichas funciones se encuentran instalar o desinstalar el servidor PostgreSQL. Permitirá gestionar clústers y controlar sus estados, así como el estado del servicio postgresql. Además, con el sistema se podrá gestionar las reglas de control de acceso que se encuentran en el archivo pg_hba.conf. También algunas de las configuraciones del archivo postgresql.conf. Va a permitir obtener el uso que está haciendo el servidor, de CPU y Memoria de Acceso Aleatorio (RAM, por sus siglas en inglés). Lo mostrará en una gráfica.

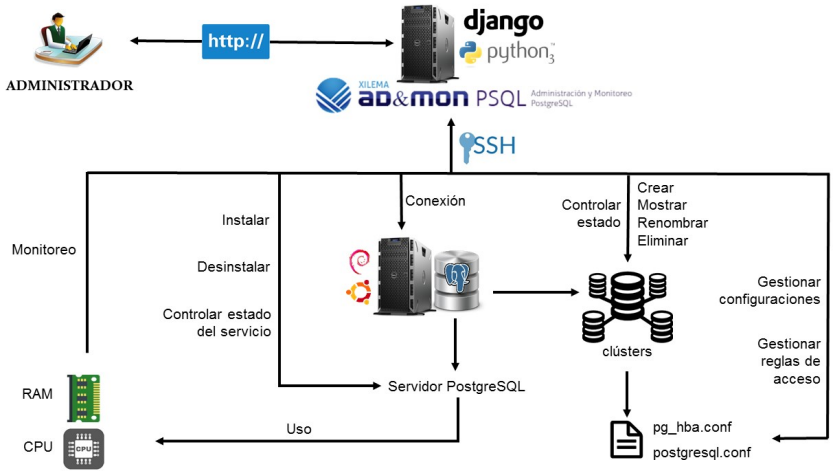

Figura 2.2. Propuesta de solución del sistema

Uno de los puntos fundamentales que se debe tener en cuenta al desarrollar un sistema es la especificación de la audiencia a la que está dirigido el mismo. El rol y las funciones se detallan a continuación.

2.4. Rol y funcionalidades

2.4.1. Administrador del sistema

Es el único rol existente en el sistema, y tiene el control total del mismo.

2.4.2. Funcionalidades del sistema

1. Autenticar un usuario en el sistema.
2. Gestionar máquinas servidoras.

    Agregar una máquina servidora.

Eliminar una máquina servidora.

Editar una máquina servidora.

Listar las máquinas servidoras.

3. Conectarse por SSH a una máquina servidora.
4. Instalar PostgreSQL.
5. Controlar el estado del servicio postgresql.

    Iniciar el servicio postgresql.

    Detener el servicio postgresql.

    Reiniciar el servicio postgresql.

    Mostrar el estado del servicio postgresql.
6. Desinstalar PostgreSQL.
7. Gestionar clústers.

    Crear clúster.

    Eliminar clúster.

    Renombrar clúster.

    Listar clúster.
8. Gestionar el estado de un clúster.

    Iniciar el clúster.

    Detener el clúster.

    Reiniciar el clúster.

    Recargar el clúster.
9. Gestionar reglas de control de acceso de un clúster.

    Mostrar reglas de control de acceso de un clúster.

    Crear reglas de control de acceso de un clúster.

    Modificar reglas de control de acceso de un clúster.

    Eliminar reglas de control de acceso de un clúster.
10. Mostrar y modificar el puerto de escucha de un clúster.
11. Mostrar y modificar el número máximo de conexiones permitidas de un clúster.
12. Mostrar y modificar la cantidad de conexiones reservadas para los superusuarios de un clúster.
13. Mostrar y modificar las direcciones por las cuales el servidor escucha.
14. Mostrar y modificar el tiempo máximo de autenticación de los clientes de un clúster.
15. Mostrar y modificar la opción de contraseñas cifradas para la autenticación de un clúster.
16. Mostrar y modificar la cantidad de memoria que el clúster utiliza para buffers de memoria compartida.
17. Mostrar y modificar el número máximo de buffers temporales usados por cada sesión de la Base de Datos.
18. Mostrar y modificar la cantidad de memoria de trabajo.
19. Mostrar y modificar la cantidad de memoria de trabajo para las operaciones de mantenimiento.

20. Monitorear el uso de los recursos a través de gráficos.

2.5. Fase de exploración

En esta fase, los clientes plantean a grandes rasgos las HU que son de interés para la primera entrega del producto. Al mismo tiempo el equipo de desarrollo se familiariza con las herramientas, tecnologías y prácticas que se utilizarán en el proyecto. Se prueba la tecnología y se exploran las posibilidades de la arquitectura del sistema construyendo un prototipo. La fase de exploración toma de pocas semanas a pocos meses, dependiendo del tamaño y familiaridad que tengan los programadores con la tecnología (Beck, 2000).

2.5.1. Historias de usuario

Las HU son la técnica utilizada en XP para especificar los requisitos del software. Se trata de tarjetas en las cuales el cliente describe brevemente las características que el sistema debe poseer. El tratamiento de las HU es muy dinámico y flexible, en cualquier momento las HU pueden reemplazarse por otras más específicas o generales, añadirse nuevas o ser modificadas. Cada HU es lo suficientemente comprensible y delimitada para que los programadores puedan implementarla en unas semanas. Para efectos de planificación, las HU pueden ser de una a tres semanas de tiempo de programación (para no superar el tamaño de una iteración) (Jeffries; Anderson y Hendrickson, 2001).

En las HU se considera:

La prioridad en el negocio:

- Muy Alta: Cuando son consideradas por los clientes esenciales para el funcionamiento del negocio.
- Alta: Cuando son consideradas necesarias por los clientes, para el funcionamiento del negocio, e intervienen directamente en el desarrollo del negocio.
- Media: Cuando el cliente cree que son necesarias, pero estas no intervienen en gran medida en el desarrollo del negocio.
- Baja: Cuando constituyen procesos que se deben tener en cuenta, pero su ausencia no perjudica el flujo principal del negocio.

El riesgo en desarrollo:

- Alto: Cuando en la implementación de las HU pueden surgir errores que lleven a la inoperatividad del código.
- Medio: Cuando en la implementación de las HU pueden existir errores que retrasen la entrega del producto.
- Bajo: Cuando pueden aparecer errores que serán tratados con relativa facilidad sin que traigan perjuicios para el desarrollo del proyecto.

Un punto de estimación equivale a una semana de programación, una semana de programación corresponde a 40 horas en desarrollo, 8 horas durante 5 días de la semana del calendario normal.

A continuación, una descripción de algunas HU definidas, las restantes se encuentran en el Apéndice A.

Tabla 2.1. Historia de usuario # 1

Historia de usuario	
Número: 1	**Nombre:** Autenticar un usuario en el sistema.
Usuario: Administrador	
Prioridad en negocio: Muy Alta	**Riesgo en desarrollo:** Bajo
Puntos estimados: 0.2	**Iteración asignada:** 1
Programador responsable: Jesús Martínez Méndez y Barbara Dayanne Acosta Caldevilla	
Descripción: Permite a un usuario entrar al sistema introduciendo usuario y contraseña. Debe notificar al usuario si existen errores en los datos entrados.	
Observaciones: El usuario debe tener una cuenta creada en el sistema.	

Tabla 2.2. Historia de usuario # 2

Historia de usuario	
Número: 2	**Nombre:** Gestionar máquinas servidoras.
Usuario: Administrador	
Prioridad en negocio: Muy Alta	**Riesgo en desarrollo:** Medio
Puntos estimados: 0.2	**Iteración asignada:** 1
Programador responsable: Jesús Martínez Méndez y Barbara Dayanne Acosta Caldevilla	
Descripción: Permite a un usuario agregar, eliminar, editar y listar las máquinas servidoras en el sistema. Debe notificar al usuario si la gestión tuvo éxito o si ocurrieron errores. Al agregar una máquina servidora deben especificarse el Protocolo de Internet (IP, por sus siglas en inglés) de la máquina, el puerto por el cual el servidor SSH escucha las conexiones, un identificador para la máquina, el sistema operativo de la máquina y un usuario y contraseña para la conexión SSH.	
Observaciones: El usuario debe estar autenticado en el sistema. La contraseña para ser almacenada debe estar encriptada o solicitarla al usuario en cada inicio de sesión. El usuario para realizar la conexión SSH debe tener permisos de superusuario.	

Tabla 2.3. Historia de usuario # 3

Historia de usuario	
Número: 3	**Nombre:** Conectarse por SSH a una máquina servidora.
Usuario: Administrador	
Prioridad en negocio: Muy Alta	**Riesgo en desarrollo:** Alto
Puntos estimados: 0.2	**Iteración asignada:** 1
Programador responsable: Jesús Martínez Méndez y Barbara Dayanne Acosta Caldevilla	
Descripción: Permite conectarse a una máquina servidora registrada en el sistema. Debe notificar al usuario si falla la conexión.	

Continúa en la próxima página

Tabla 2.3. Continuación de la página anterior

Observaciones: El usuario debe estar autenticado en el sistema. La máquina servidora debe tener configurado un servidor SSH.

Tabla 2.4. Historia de usuario # 4

Historia de usuario	
Número: 4	**Nombre:** Instalar PostgreSQL.
Usuario: Administrador	
Prioridad en negocio: Muy Alta	**Riesgo en desarrollo:** Alto
Puntos estimados: 0.2	**Iteración asignada:** 1
Programador responsable: Jesús Martínez Méndez y Barbara Dayanne Acosta Caldevilla	
Descripción: Instala la versión del servidor PostgreSQL disponible en el repositorio. Debe notificar al usuario si la instalación tuvo éxito o si ocurrieron errores.	
Observaciones: El usuario debe estar autenticado en el sistema. La máquina servidora debe tener configurado un servidor SSH. En caso que PostgreSQL esté instalado, solo se muestra la versión. La máquina servidora debe tener configurados los repositorios para la instalación de PostgreSQL.	

Tabla 2.5. Historia de usuario # 5

Historia de usuario	
Número: 5	**Nombre:** Controlar el estado del servicio postgresql.
Usuario: Administrador	
Prioridad en negocio: Muy Alta	**Riesgo en desarrollo:** Medio
Puntos estimados: 2	**Iteración asignada:** 1
Programador responsable: Jesús Martínez Méndez y Barbara Dayanne Acosta Caldevilla	
Descripción: Controla el estado del servicio postgresql, ya sea iniciar, detener, reiniciar o mostrar el estado. Debe notificar al usuario si se cambió o no el estado.	
Observaciones: El usuario debe estar autenticado en el sistema. Es necesario tener instalado el servidor PostgreSQL.	

Tabla 2.6. Historia de usuario # 6

Historia de usuario	
Número: 6	**Nombre:** Desinstalar PostgreSQL.
Usuario: Administrador	
Prioridad en negocio: Muy Alta	**Riesgo en desarrollo:** Medio
Puntos estimados: 0.2	**Iteración asignada:** 1
Programador responsable: Jesús Martínez Méndez y Barbara Dayanne Acosta Caldevilla	

Continúa en la próxima página

Tabla 2.6. Continuación de la página anterior

Descripción: Desinstala el servidor PostgreSQL de una máquina. Debe notificar al usuario si la desinstalación tuvo éxito o si ocurrieron errores.
Observaciones: El usuario debe estar autenticado en el sistema. Es necesario tener instalado el servidor PostgreSQL.

Tabla 2.7. Historia de usuario # 7

Historia de usuario	
Número: 7	**Nombre:** Gestionar clústers.
Usuario: Administrador	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Alto
Puntos estimados: 1	**Iteración asignada:** 2
Programador responsable: Jesús Martínez Méndez y Barbara Dayanne Acosta Caldevilla	
Descripción: Gestiona los clústers del servidor PostgreSQL, permitiendo crear, eliminar, renombrar y listar clústers. Debe notificar al usuario si la gestión tuvo o no éxito, en caso que no, se deben mostrar los errores.	
Observaciones: El usuario debe estar autenticado en el sistema. Es necesario tener instalado el servidor PostgreSQL.	

Tabla 2.8. Historia de usuario # 8

Historia de usuario	
Número: 8	**Nombre:** Gestionar el estado de un clúster.
Usuario: Administrador	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Alto
Puntos estimados: 2	**Iteración asignada:** 2
Programador responsable: Jesús Martínez Méndez y Barbara Dayanne Acosta Caldevilla	
Descripción: Gestiona el estado de un clúster, permitiendo iniciar, detener, reiniciar y recargar el clúster. Debe notificar al usuario si la gestión tuvo o no éxito, en caso que no, se deben mostrar los errores.	
Observaciones: El usuario debe estar autenticado en el sistema. Es necesario tener instalado el servidor PostgreSQL y estar creado el clúster.	

2.6. Fase de planificación

La planificación se puede realizar basándose en el tiempo o el alcance. La velocidad del proyecto es utilizada para establecer cuántas historias se pueden implementar antes de una fecha determinada o cuánto tiempo tomará implementar un conjunto de historias. Al planificar por tiempo, se multiplica el número de iteraciones por la velocidad del proyecto, determinándose cuántos puntos se pueden completar (Beck, 2000).

2.6.1. Estimación de esfuerzos

Las estimaciones de esfuerzo asociado a la implementación de las HU la establecen los programadores utilizando como medida el punto de estimación. Un punto de estimación equivale a una semana de programación. Las HU generalmente valen de 1 a 3 puntos. Por otra parte, el equipo de desarrollo mantiene un registro de la "velocidad" de desarrollo, establecida en puntos por iteración, basándose principalmente en la suma de puntos correspondientes a las HU que fueron terminadas en la última iteración (Beck, 2000).

Tabla 2.9. Estimación de esfuerzo por historia de usuario

Iteración	Historias de usuario		Puntos estimados (semanas)
1	1	Autenticar un usuario en el sistema.	0.2
	2	Gestionar máquinas servidoras.	0.2
	3	Conectarse por SSH a una máquina servidora.	0.2
	4	Instalar PostgreSQL.	0.2
	5	Controlar el estado del servicio postgresql.	2
	6	Desinstalar PostgreSQL.	0.2
2	7	Gestionar clústers.	1
	8	Gestionar el estado de un clúster.	2
3	9	Gestionar reglas de control de acceso de un clúster.	2
	10	Mostrar y modificar el puerto de escucha de un clúster.	0.2
	11	Mostrar y modificar el número máximo de conexiones permitidas de un clúster.	0.2
	12	Mostrar y modificar la cantidad de conexiones reservadas para los superusuarios de un clúster.	0.2
	13	Mostrar y modificar las direcciones por las cuales el servidor escucha.	0.2
	14	Mostrar y modificar el tiempo máximo de autenticación de los clientes de un clúster.	0.2
4	15	Mostrar y modificar la opción de contraseñas cifradas para la autenticación de un clúster.	0.2
	16	Mostrar y modificar la cantidad de memoria que el clúster utiliza para buffers de memoria compartida.	0.2
	17	Mostrar y modificar el número máximo de buffers temporales usados por cada sesión de la Base de Datos.	0.2
	18	Mostrar y modificar la cantidad de memoria de trabajo.	0.2
	19	Mostrar y modificar la cantidad de memoria de trabajo para las operaciones de mantenimiento.	0.2

Continúa en la próxima página

Tabla 2.9. Continuación de la página anterior

	20	Monitorear el uso de los recursos a través de gráficos.	2
Total			**12.0**

2.7. Fase de iteraciones

Esta fase incluye varias iteraciones sobre el sistema antes de ser entregado. El Plan de Entrega está compuesto por iteraciones de no más de tres semanas. En la primera iteración se puede intentar establecer una arquitectura del sistema que pueda ser utilizada durante el resto del proyecto. Esto se logra escogiendo las historias que fuercen la creación de esta arquitectura, sin embargo, esto no siempre es posible ya que es el cliente quien decide qué historias se implementarán en cada iteración (para maximizar el valor de negocio). Al final de la última iteración el sistema estará listo para entrar en producción (ibíd.).

2.7.1. Plan de iteraciones

Los elementos que deben tomarse en cuenta durante el desarrollo del Plan de la Iteración son: historias de usuario no abordadas, velocidad del proyecto, pruebas de aceptación no superadas en la iteración anterior y tareas no terminadas en la iteración anterior. Todo el trabajo de la iteración es expresado en tareas de programación, cada una de ellas es asignada a un programador como responsable, pero llevadas a cabo por parejas de programadores (ibíd.).

Iteración 1

La presente iteración tendrá como objetivo darle cumplimiento a las HU de la 1 a la 6, HU que están relacionadas con la primera etapa del proceso de administración y monitoreo de servidores PostgreSQL, creando con ellas el funcionamiento base del sistema. Además, se obtendrá una primera versión del producto en la que el cliente podrá probar algunas de las funcionalidades solicitadas como: autenticar un usuario en el sistema, gestionar máquinas servidoras, conectarse por SSH a una máquina servidora, instalar PostgreSQL, controlar el estado del servicio postgresql, desinstalar PostgreSQL.

Iteración 2

En esta iteración se implementarán las HU 7 y 8, estas están relacionadas con la segunda etapa del proceso de administración y monitoreo de servidores PostgreSQL, creando con ellas un funcionamiento medio del sistema. Además, se obtendrá una segunda versión del producto en la que el cliente podrá probar las funcionalidades: gestionar clústers y gestionar el estado de un clúster.

Iteración 3

En dicha iteración se implementarán las HU de la 9 a la 14, dichas HU están relacionadas con la tercera etapa del proceso de administración y monitoreo de servidores PostgreSQL, creando con ellas un funcionamiento más abarcador del sistema. Además, se obtendrá la tercera versión del producto en la que el cliente podrá probar algunas de las funcionalidades solicitadas como: gestionar reglas de control de acceso de un clúster, mostrar y modificar el puerto de escucha de un clúster, mostrar y modificar el número máximo de conexiones permitidas de un clúster, mostrar y modificar la cantidad de conexiones reservadas para los superusuarios de un clúster, mostrar y modificar las direcciones por las cuales el servidor escucha, mostrar y modificar el tiempo máximo de autenticación de los clientes de un clúster.

Iteración 4

Esta es la última de las iteraciones en la cual se implementarán las HU de la 15 a la 20, creando con ellas un funcionamiento completo y terminal del sistema. En esta el cliente podrá probar algunas de las funcionalidades solicitadas como: mostrar y modificar la opción de contraseñas cifradas para la autenticación de un clúster, mostrar y modificar la cantidad de memoria que el clúster utiliza para buffers de memoria compartida, mostrar y modificar el número máximo de buffers temporales usados por cada sesión de la Base de Datos, mostrar y modificar la cantidad de memoria de trabajo, mostrar y modificar la cantidad de memoria de trabajo para las operaciones de mantenimiento y monitorear el uso de los recursos a través de gráficos.

2.7.2. Plan de duración de las iteraciones

A continuación, se presenta el plan de duración de las iteraciones. Este plan tiene como finalidad mostrar la duración de cada iteración, así como el orden en que serán implementadas las historias de usuarios en cada una, como se muestra en la siguiente tabla.

Tabla 2.10. Plan de duración de las iteraciones

Iteración	Historias de usuario		Duración (semanas)
1	1	Autenticar un usuario en el sistema.	3.0
	2	Gestionar máquinas servidoras.	
	3	Conectarse por SSH a una máquina servidora.	
	4	Instalar PostgreSQL.	
	5	Controlar el estado del servicio postgresql.	
	6	Desinstalar PostgreSQL.	
2	7	Gestionar clústers.	3.0
	8	Gestionar el estado de un clúster.	
3	9	Gestionar reglas de control de acceso de un clúster.	3.0
	10	Mostrar y modificar el puerto de escucha de un clúster.	

Continúa en la próxima página

Tabla 2.10. Continuación de la página anterior

	11	Mostrar y modificar el número máximo de conexiones permitidas de un clúster.	
	12	Mostrar y modificar la cantidad de conexiones reservadas para los superusuarios de un clúster.	
	13	Mostrar y modificar las direcciones por las cuales el servidor escucha.	
	14	Mostrar y modificar el tiempo máximo de autenticación de los clientes de un clúster.	
4	15	Mostrar y modificar la opción de contraseñas cifradas para la autenticación de un clúster.	3.0
	16	Mostrar y modificar la cantidad de memoria que el clúster utiliza para buffers de memoria compartida.	
	17	Mostrar y modificar el número máximo de buffers temporales usados por cada sesión de la Base de Datos.	
	18	Mostrar y modificar la cantidad de memoria de trabajo.	
	19	Mostrar y modificar la cantidad de memoria de trabajo para las operaciones de mantenimiento.	
	20	Monitorear el uso de los recursos a través de gráficos.	
Total			**12.0**

2.7.3. Plan de entrega

El cliente establece la prioridad de cada HU, y correspondientemente, los programadores realizan una estimación del esfuerzo necesario de cada una de ellas. Se toman acuerdos sobre el contenido de la primera entrega y se determina un cronograma en conjunto con el cliente. Una entrega debería obtenerse en no más de tres meses (Beck, 2000).

Mediante el mismo el equipo de desarrollo mantiene al cliente informado y actualizado del progreso, funcionamiento y calidad del sistema. Gracias al plan de entrega que se les confecciona a las iteraciones del sistema, tanto el cliente como el equipo desarrollador se encuentran dirigidos hacia el avance y perfeccionamiento del producto. A continuación, se refleja la evidencia de lo anteriormente planteado, donde se plasma la fecha en la que se va a realizar cada una de las entregas de las iteraciones del sistema por el equipo de desarrollo al cliente.

2.8. Diseño del sistema

Se debe diseñar la solución más simple que pueda funcionar y ser implementada en un momento determinado del proyecto. La complejidad innecesaria y el código extra debe ser removido inmediatamente.

Tabla 2.11. Plan de Entregas

Iteración	Fecha de entrega
Iteración 1	21 de marzo de 2018
Iteración 2	11 de abril de 2018
Iteración 3	2 de mayo de 2018
Iteración 4	23 de mayo de 2018

Kent Beck dice que en cualquier momento el diseño adecuado para el software es aquel que: supera con éxito todas las pruebas, no tiene lógica duplicada, refleja claramente la intención de implementación de los programadores y tiene el menor número posible de clases y métodos (Beck, 2000).

2.8.1. Patrones de arquitectura

Modelo cliente-servidor

Actualmente existen muchos sistemas de información cuya arquitectura se basa en la denominada dos capas, las cuales solo cuentan con los siguientes niveles o capas:

- Nivel de aplicación.
- Nivel de la base de datos.

Es un modelo de aplicación distribuida en el que las tareas se reparten entre los proveedores de recursos o servicios, llamados servidores, y los demandantes, llamados clientes. Un cliente realiza peticiones a otro programa, el servidor, es quien le da respuesta. Esta idea también se puede aplicar a programas que se ejecutan sobre una sola computadora, aunque es más ventajosa en un sistema operativo multiusuario distribuido a través de una red de computadoras.

Características

En esta arquitectura el remitente se conoce como cliente, el cual es quien realiza lo siguiente:

- Inicia solicitudes o peticiones.
- Debe esperar y recibir respuestas del servidor.
- Puede conectarse a varios servidores.
- Por lo general interactúa con el usuario final a través de una interfaz gráfica.

En cuanto al receptor de la solicitud enviado por el cliente se conoce como servidor, el mismo se caracteriza por (España y Fernando, 2016):

- Desempeñan un papel pasivo en la comunicación, deben esperar a que lleguen las solicitudes.
- Al recibir una solicitud, la misma es procesada y posteriormente envían la respuesta.
- Generalmente pueden aceptar las conexiones de varios clientes a la vez.
- No es frecuente, la interacción directa con el usuario final.

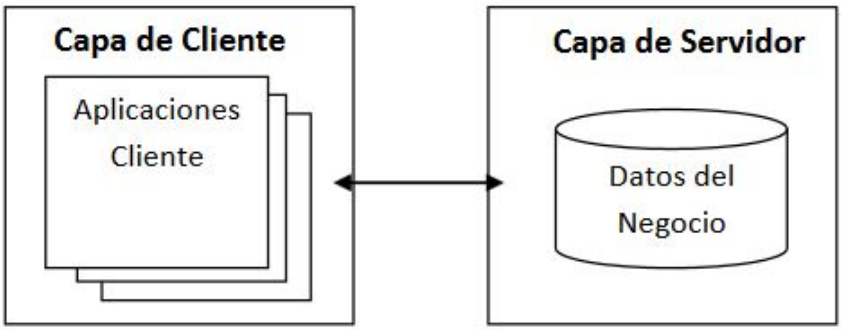

Figura 2.3. Arquitectura Cliente-Servidor

Model Template View (MTV)

En cuanto a la arquitectura de Django, cabe comentar que fue diseñado para que exista un acoplamiento débil entre las distintas partes de la aplicación y una clara separación de las mismas. Por lo que, como se puede deducir, es posible realizar cambios en una parte específica de la aplicación sin afectar a las demás piezas de la misma. Django, por lo tanto, tiene una implementación especial del Modelo Vista Controlador.

En las aplicaciones Django, debido a que el Controlador es manejado por el mismo framework, y la parte más significativa se la llevan los modelos (*Models*), vistas (*Views*) y plantillas (*Templates*), es conocido también como un framework MTV (*Models Views Templates*). Se pasa a describir posteriormente este significado (Caldera Vergara, 2017).

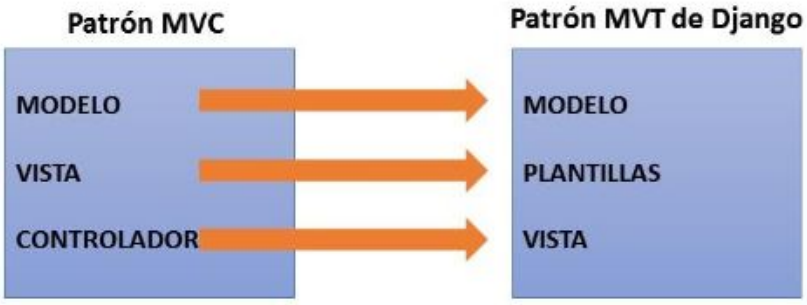

Figura 2.4. Diagrama correspondencia MVC MTV

En Django, se conoce al Controlador como “vista” y a la vista como “plantilla”, ya que en su implementación la vista describe el dato que se muestra al usuario de la aplicación. Por lo que se centran más, en qué dato ve y no cómo lo ve. Por tanto, lo que decide entonces qué dato ve el usuario es la vista, y cómo lo ve se lo delega a la plantilla. La vista, es la capa de la lógica de negocios, donde se encuentra la inteligencia que accede al modelo y la delega a la plantilla adecuada. Por último, el modelo, no cambia con respecto a la explicación del patrón Modelo Vista Controlador convencional, sigue siendo la capa de acceso a la base de datos, que contiene, cómo acceder a ellos, cuál es su comportamiento, como validarlos y las relaciones

entre los datos. Se pasa a mostrar un esquema de la colaboración existente entre los distintos componentes del patrón (Caldera Vergara, 2017):

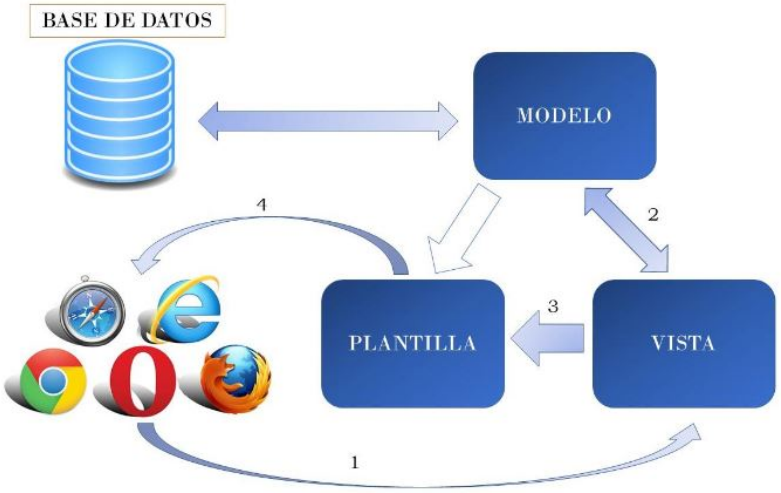

Figura 2.5. Diagrama colaboración capas MTV

- El cliente desde su navegador realiza una solicitud de un servicio web de la aplicación, por lo que se pone en contacto con la capa de la vista.
- La capa de la vista necesita tener contacto directo con la del modelo ya que es de ella de donde cogerá los datos que solicita el usuario. Por lo que entra en acción la capa del modelo. Ésta última capa, gracias a las consultas a la base de datos devuelve los objetos a la capa de la vista, que ella ya se encargará de filtrar y devolver explícitamente los que el usuario ha solicitado en la petición web.
- El siguiente paso es que la capa de la vista envíe a la capa de la plantilla los datos requeridos por el usuario, tras previo procesamiento de los mismos. En este paso, ha actuado indirectamente el modelo, ya que es quien proporcionó los datos a la vista.
- En último lugar, la capa de la plantilla es la encargada de cómo se deben mostrar los datos al usuario. Por lo que, con los datos recibidos de la vista, los incluye en su respectiva plantilla para que los navegadores puedan procesarlo y el usuario visualice correctamente la información.

2.8.2. Patrones de diseño

Patrones GRASP

Es posible comunicar los principios detallados y el razonamiento que se quiere para entender el diseño de objetos básico, y aprender a aplicarlos en un enfoque sistemático que elimina la magia y la ambigüedad.

Los Patrones Generales de Software para Asignación de Responsabilidades (GRASP, por sus siglas en inglés) constituyen un apoyo para la enseñanza que ayuda a uno a entender el diseño de objetos esencial, y aplica el razonamiento para el diseño de una forma sistemática, racional y explicable. Este enfoque para la

comprensión y utilización de los principios de diseño se basa en los patrones de asignación de responsabilidades (Larman, 2003).

- **Creador:** Guía la asignación de responsabilidades relacionadas con la creación de objetos. La intención básica de este patrón es encontrar un creador que necesite conectarse al objeto creado en alguna situación (ibíd.). Este patrón se evidencia entre las clases AdministracionPostgreSQL y ConexionSSH, ya que la primera crea instancias de la segunda.
- **Alta Cohesión:** Grady Booch establece que existe alta cohesión funcional cuando los elementos de un componente (como una clase) "trabajan todos juntos para proporcionar algún comportamiento bien delimitado (ibíd.)." Es decir que las clases del sistema tienen asignadas solo las responsabilidades que les corresponde y mantienen una estrecha relación con el resto de las clases. Este patrón se evidencia entre la clase AdministracionPostgreSQL y la clase FicheroPgHba. Cuando se desean obtener las reglas del fichero pg_hba.conf la clase encargada de dicha tarea es FicheroPgHba, pero la que posee la información es AdministracionPostgreSQL.
- **Bajo Acoplamiento:** Es una medida de la fuerza con que un elemento está conectado a, tiene conocimiento de, confía en, otros elementos. Un elemento con bajo acoplamiento no depende de demasiados otros elementos (ibíd.). Es decir que determina el nivel de dependencia de un elemento con respecto a otros. Este patrón es utilizado por el framework Django en su interior ya que utiliza las URL como forma de separar clases y funcionalidades, esto permite que si deja de funcionar alguna de las vistas no afecta a las demás.
- **Controlador:** Normalmente, un controlador debería delegar en otros objetos el trabajo que se necesita hacer, coordina o controla la actividad. No realiza mucho trabajo por sí mismo. Un importante corolario del patrón Controlador es que los objetos interfaz y la capa de presentación no debían ser responsables de llevar a cabo los eventos del sistema. En otras palabras, las operaciones del sistema se deberían manejar en la lógica de la aplicación o capas de dominio en lugar de en la capa de interfaz del sistema (ibíd.). Se evidencia el uso de este patrón en el archivo views.py que es el encargado de definir clases y funcionalidades que controlan las vistas que se muestran en la aplicación.

Los patrones GRASP han posibilitado seguir una estructura de diseño única mediante la reutilización de los estándares que propone. Además, de formalizar un vocabulario común entre los integrantes del equipo, así como facilitar la implementación a partir de los conocimientos ya existentes y no realizar búsquedas para solucionar los problemas que se presenten a medida que se diseña el sistema.

2.8.3. Tarjetas CRC

Las tarjetas CRC son fichas, una por cada clase, en las que se escriben brevemente las responsabilidades de la clase, y una lista de los objetos con los que colabora para llevar a cabo esas responsabilidades. Se desarrollan normalmente en una sesión de trabajo en grupo pequeño. Las tarjetas CRC son una técnica para registrar los resultados de la asignación de responsabilidades y asignaciones (ibíd.).

Tabla 2.12. Tarjeta CRC # 1

Tarjeta CRC	
Clase: ConexionSSH	
Responsabilidad	**Colaboración**
• Conectarse a un servidor SSH. • Desconectarse del servidor SSH. • Ejecutar comandos en el servidor SSH. • Obtener archivos del servidor SSH. • Subir archivos al servidor SSH.	Librería Paramiko

Tabla 2.13. Tarjeta CRC # 2

Tarjeta CRC	
Clase: FicheroPgHba	
Responsabilidad	**Colaboración**
• Actualizar reglas del fichero pg_hba.conf. • Obtiene las reglas del fichero pg_hba.conf.	Fichero

Tabla 2.14. Tarjeta CRC # 3

Tarjeta CRC	
Clase: AdministracionPostgreSQL	
Responsabilidad	**Colaboración**

Continúa en la próxima página

Tabla 2.14. Continuación de la página anterior

• Instalar PostgreSQL. • Desinstalar PostgreSQL. • Listar versiones instaladas de PostgreSQL. • Listar clústers. • Obtener el estado del servicio postgresql. • Obtener archivo postgresql.conf. • Obtener archivo pg_hba.conf. • Subir archivo postgresql.conf. • Subir archivo pg_hba.conf. • Obtener reglas del fichero pg_hba.conf. • Controlar el servicio postgresql. • Controlar el estado de los clústers. • Eliminar clústers. • Renombrar clústers. • Crear clústers.	FicheroPg_Hba ConexionSSH FicheroPostgreslqconf

Tabla 2.15. Tarjeta CRC # 4

Tarjeta CRC	
Clase: FicheroPostgreslqconf	
Responsabilidad	**Colaboración**
• Actualiza las configuraciones del fichero postgresql.conf. • Obtiene las configuraciones del fichero postgresql.conf.	Fichero

Tabla 2.16. Tarjeta CRC # 5

Tarjeta CRC	
Clase: Fichero	
Responsabilidad	**Colaboración**
• Lee las líneas de un fichero. • Actualiza las líneas de un fichero.	

2.9. Conclusiones del capítulo

En el capítulo se expuso la propuesta de solución, que permitió visualizar las principales funcionalidades del sistema. Se describieron las HU, se identificaron las iteraciones y las HU a realizar en cada una de ellas, así como la planificación del esfuerzo dedicado a la realización de cada una de estas en el orden en que se les dará cumplimiento según las necesidades del cliente.

Además, se presentaron los patrones de diseño asociados a la arquitectura propuesta, los cuales darán mayor independencia a las clases y facilitarán la implementación. Se realizaron las tarjetas CRC para obtener una representación de las principales clases y funcionalidades del sistema.

CAPÍTULO 3

Implementación y pruebas

3.1. Introducción

En el presente capítulo se describen los diferentes mecanismos utilizados para llevar a cabo el desarrollo y validación del sistema propuesto. En la etapa de implementación se trazan las pautas a seguir en cuanto al código, siguiendo varios estándares que propone el lenguaje Python para un mejor entendimiento y organización del mismo. Además, se describen las tareas de ingeniería que es el artefacto que genera la metodología de desarrollo de software ágil XP para detallar las HU descritas en el capítulo anterior. Luego se realizan las pruebas, con el objetivo de comprobar si el sistema cumple sus requisitos. Dentro de ella pueden desarrollarse varios tipos de pruebas en función de los objetivos de las mismas, XP como metodología define realizar pruebas unitarias y funcionales.

3.2. Implementación

La presencia del cliente en las diferentes fases de XP como parte del equipo de trabajo es indispensable. A la hora de codificar o implementar una HU es aún más necesaria su presencia pues son ellos los que crean las mismas, negocian los tiempos en los que serán implementadas, especifican detalladamente lo que esta hará y verifican que la historia implementada cumple la funcionalidad especificada cuando se realicen las pruebas. La codificación debe atenerse a estándares y los programadores lo han de seguir de tal manera que el código en el sistema se vea como si hubiera estado escrito por una sola persona. Ello mantiene el código consistente facilitando su comprensión.

3.2.1. Estándares de codificación

Los estándares son una buena práctica para el desarrollo de software los cuales no sólo se deben utilizar con la metodología XP sino también al aplicar otra metodología. Al aplicar estándares se buscó facilitar la comprensión en el código para el equipo de desarrollo (Castillo Asencio, 2016).

XP promueve la programación basada en estándares, de manera que sea fácilmente entendible por todo el equipo, y que facilite la recodificación. XP enfatiza la comunicación de los programadores a través del código, con lo cual es indispensable que se sigan ciertos estándares de programación (del equipo, de la organización u otros estándares reconocidos para los lenguajes de programación utilizados). Los estándares de programación mantienen el código legible para los miembros del equipo, facilitando los cambios (Canós y Letelier, 2012).

Para facilitar el entendimiento del código y fijar un modelo a seguir, se establecieron estándares de codificación. A continuación, se muestran algunos de estos estándares para el lenguaje Python (Python Software Foundation, 2018).

- Los comentarios se realizan utilizando el símbolo de número # seguido de un espacio.
- Casi sin excepción, los nombres de clases deben utilizar la convención "CapWords" (palabras que comienzan con mayúsculas).
- lower_case_with_underscores: cuando las palabras son escritas en minúsculas separadas por guión bajo.
- Usa 4 (cuatro) espacios por indentación.
- Las importaciones deben estar en líneas separadas, por ejemplo:
  Sí:
  import os
  import sys

  No:
  import sys, os

  Sin embargo, es correcto decir:
  from subprocess import Popen, PIPE

Código fuente 3.1. Ejemplo de uso de los estándares anteriores en el código de la aplicación.

```
1 import os
2 import json
3 from admin_postgresql.mis_clases.ConexionSSH import *
4 from admin_postgresql.mis_clases.GestionFicherosConfig import FicheroConfig
5 from admin_postgresql.mis_clases.pgconfig import *
6 from admin_postgresql.mis_clases.pghba import *
7
8 class FicheroPgHba(FicheroConfig):
9
10     def __init__(self, ruta=None):
11         super().__init__(ruta)
12         self.opcion_actual = -1
13         self.regla = Regla()
```

```
14
15      # recibe una lÃnea del fichero pg_hba, la analiza y retorna la regla asociada a
            esa linea
16      def AnalizarLinea(self, linea):
17          self.opcion_actual = -1
18          self.regla = Regla()
19          self.__Tipo(linea)
20          return self.regla
```

3.2.2. Tareas de ingeniería

Las tareas de ingeniería son el artefacto que, en la implementación y las pruebas, específicamente en la implementación, se encargan de albergar los detalles para la realización de cada una de las HU. Una HU puede tener más de una tarea de ingeniería lo cual depende de la complejidad de dicha HU. Cuando el equipo de desarrollo conforma las tareas de ingeniería deja bien claro en sus campos a qué HU pertenece y los programadores responsables de implementarla. Estos a su vez establecen fecha de inicio y de fin en correspondencia con los datos que aporta la HU en cuestión. Los puntos estimados indican el tiempo real que utilizan los programadores en realizar la funcionalidad. También se especifica el tipo de tarea que se va a realizar y el nombre de la misma, así como la descripción de la tarea; en éste campo como su nombre lo indica se detalla todo lo que se va a realizar y en lo que se van a apoyar para darle solución a la funcionalidad correspondiente.

A continuación, una descripción de algunas de las tareas de ingeniería definidas, las restantes se encuentran en el Apéndice B.

Tabla 3.1. Tarea de ingeniería # 1

Tarea	
Número de tarea: 1	**Número de Historia de usuario:** 1
Nombre de la tarea: Autenticar un usuario en el sistema.	
Tipo de tarea: Autenticación	**Puntos estimados:** 0.2
Fecha de inicio: 1 de marzo de 2018	**Fecha de fin:** 1 de marzo de 2018
Programador responsable: Jesús Martínez Méndez y Barbara Dayanne Acosta Caldevilla	
Descripción: Va a permitir autenticar un usuario en el sistema, teniendo que entrar su nombre de usuario y contraseña. Se verifica en la base de datos del sistema, si el usuario está activo y si los campos son correctos. En caso contrario se muestra un mensaje de error. Si los datos son correctos se muestra la página de inicio.	

Tabla 3.2. Tarea de ingeniería # 2

Tarea	
Número de tarea: 2	**Número de Historia de usuario:** 2
Nombre de la tarea: Gestionar máquinas servidoras.	
Tipo de tarea: Gestión	**Puntos estimados:** 0.2
Fecha de inicio: 2 de marzo de 2018	**Fecha de fin:** 2 de marzo de 2018
Programador responsable: Jesús Martínez Méndez y Barbara Dayanne Acosta Caldevilla	
Descripción: Va a permitir al administrador agregar, eliminar, editar y listar las máquinas servidoras. • Para agregar una máquina servidora, se le muestra un formulario con los siguientes campos: nombre de la máquina (Identificador), dirección IP, puerto del servidor SSH, sistema operativo de la máquina, nombre de usuario y contraseña. En caso de que algún campo esté incorrecto, se mostrará un mensaje. • Al listar las máquinas servidoras se deben mostrar de cada una los siguientes campos: nombre del servidor, IP, puerto SSH, sistema operativo, editar y eliminar. • Al eliminar una máquina servidora se le pide confirmación al usuario. Si el usuario confirma se elimina la máquina y se vuelve a mostrar el listado de máquinas servidoras. • Al editar una máquina servidora se le muestra un formulario con los siguientes campos: nombre del servidor, dirección IP, puerto SSH, usuario y contraseña. Los valores de estos campos son los correspondientes a la máquina servidora. En caso de que algún campo esté incorrecto, se mostrará un mensaje. Luego de la edición de la máquina, se vuelve a mostrar el listado de máquinas servidoras.	

Tabla 3.3. Tarea de ingeniería # 3

Tarea	
Número de tarea: 3	**Número de Historia de usuario:** 3
Nombre de la tarea: Conectarse por SSH a una máquina servidora.	
Tipo de tarea: Conexión	**Puntos estimados:** 0.2
Fecha de inicio: 5 de marzo de 2018	**Fecha de fin:** 5 de marzo de 2018
Programador responsable: Jesús Martínez Méndez y Barbara Dayanne Acosta Caldevilla	
Descripción: Va a permitir al administrador conectarse por SSH a una de las máquinas servidoras registradas. Si la conexión tuvo éxito el sistema le muestra al administrador la página de instalación. Si la conexión falla se muestra un mensaje de error.	

Tabla 3.4. Tarea de ingeniería # 4

Tarea	
Número de tarea: 4	**Número de Historia de usuario:** 4

Continúa en la próxima página

Tabla 3.4. Continuación de la página anterior

Nombre de la tarea: Instalar PostgreSQL.	
Tipo de tarea: Instalación	**Puntos estimados:** 0.2
Fecha de inicio: 6 de marzo de 2018	**Fecha de fin:** 6 de marzo de 2018
Programador responsable: Jesús Martínez Méndez y Barbara Dayanne Acosta Caldevilla	
Descripción: Va a permitir al administrador instalar PostgreSQL en una máquina servidora. Si existe alguna versión de PostgreSQL instalada, se le muestra la opción de desinstalar. Se notificará al concluir la instalación. En caso de existir algún error se muestra un mensaje.	

Tabla 3.5. Tarea de ingeniería # 5

Tarea	
Número de tarea: 5	**Número de Historia de usuario:** 5
Nombre de la tarea: Iniciar el servicio postgresql.	
Tipo de tarea: Control	**Puntos estimados:** 0.6
Fecha de inicio: 7 de marzo de 2018	**Fecha de fin:** 9 de marzo de 2018
Programador responsable: Jesús Martínez Méndez y Barbara Dayanne Acosta Caldevilla	
Descripción: Va a permitir al administrador iniciar el servicio postgresql. Se muestra un mensaje de notificación.	

Tabla 3.6. Tarea de ingeniería # 6

Tarea	
Número de tarea: 6	**Número de Historia de usuario:** 5
Nombre de la tarea: Detener el servicio postgresql.	
Tipo de tarea: Control	**Puntos estimados:** 0.6
Fecha de inicio: 12 de marzo de 2018	**Fecha de fin:** 14 de marzo de 2018
Programador responsable: Jesús Martínez Méndez y Barbara Dayanne Acosta Caldevilla	
Descripción: Va a permitir al administrador detener el servicio postgresql. Se muestra un mensaje de notificación.	

Tabla 3.7. Tarea de ingeniería # 7

Tarea	
Número de tarea: 7	**Número de Historia de usuario:** 5
Nombre de la tarea: Reiniciar el estado del servicio postgresql.	
Tipo de tarea: Control	**Puntos estimados:** 0.4
Fecha de inicio: 15 de marzo de 2018	**Fecha de fin:** 16 de marzo de 2018
Programador responsable: Jesús Martínez Méndez y Barbara Dayanne Acosta Caldevilla	

Continúa en la próxima página

Tabla 3.7. Continuación de la página anterior

Descripción: Va a permitir al administrador reiniciar el servicio postgresql. Se muestra un mensaje de notificación.

Tabla 3.8. Tarea de ingeniería # 8

Tarea	
Número de tarea: 8	**Número de Historia de usuario:** 5
Nombre de la tarea: Mostrar el estado el servicio postgresql.	
Tipo de tarea: Control	**Puntos estimados:** 0.4
Fecha de inicio: 19 de marzo de 2018	**Fecha de fin:** 20 de marzo de 2018
Programador responsable: Jesús Martínez Méndez y Barbara Dayanne Acosta Caldevilla	
Descripción: Va a permitir al administrador ver el estado del servicio postgresql.	

Tabla 3.9. Tarea de ingeniería # 9

Tarea	
Número de tarea: 9	**Número de Historia de usuario:** 6
Nombre de la tarea: Desinstalar PostgreSQL.	
Tipo de tarea: Desinstalación	**Puntos estimados:** 0.2
Fecha de inicio: 21 de marzo de 2018	**Fecha de fin:** 21 de marzo de 2018
Programador responsable: Jesús Martínez Méndez y Barbara Dayanne Acosta Caldevilla	
Descripción: Va a permitir al administrador desinstalar PostgreSQL en una máquina servidora. Se notificará al concluir la desinstalación. En caso de no estar instalado el servidor PostgreSQL se le muestra la opción de instalar. En caso de existir algún error se muestra un mensaje.	

Tabla 3.10. Tarea de ingeniería # 10

Tarea	
Número de tarea: 10	**Número de Historia de usuario:** 7
Nombre de la tarea: Crear un clúster.	
Tipo de tarea: Gestión	**Puntos estimados:** 0.2
Fecha de inicio: 22 de marzo de 2018	**Fecha de fin:** 22 de marzo de 2018
Programador responsable: Jesús Martínez Méndez y Barbara Dayanne Acosta Caldevilla	
Descripción: Va a permitir al administrador crear un clúster. Se le muestra un formulario con el campo nombre del clúster, luego, se verifica que el nombre sea correcto y se le muestra un mensaje de notificación al crearlo. En caso de existir algún error se muestra un mensaje.	

Tabla 3.11. Tarea de ingeniería # 11

Tarea	
Número de tarea: 11	**Número de Historia de usuario:** 7
Nombre de la tarea: Eliminar un clúster.	
Tipo de tarea: Gestión	**Puntos estimados:** 0.2
Fecha de inicio: 23 de marzo de 2018	**Fecha de fin:** 23 de marzo de 2018
Programador responsable: Jesús Martínez Méndez y Barbara Dayanne Acosta Caldevilla	
Descripción: Va a permitir al administrador, de los clústers listados, eliminar uno de ellos. Se muestra un mensaje de notificación al eliminarlo. En caso de existir algún error se muestra un mensaje.	

Tabla 3.12. Tarea de ingeniería # 12

Tarea	
Número de tarea: 12	**Número de Historia de usuario:** 7
Nombre de la tarea: Renombrar un clúster.	
Tipo de tarea: Gestión	**Puntos estimados:** 0.2
Fecha de inicio: 26 de marzo de 2018	**Fecha de fin:** 26 de marzo de 2018
Programador responsable: Jesús Martínez Méndez y Barbara Dayanne Acosta Caldevilla	
Descripción: Va a permitir al administrador, de los clústers listados, renombrar uno, es necesario que entre el nuevo nombre del clúster. Se muestra un mensaje de notificación al renombrar. En caso de existir algún error se muestra un mensaje.	

Tabla 3.13. Tarea de ingeniería # 13

Tarea	
Número de tarea: 13	**Número de Historia de usuario:** 7
Nombre de la tarea: Listar clústers.	
Tipo de tarea: Gestión	**Puntos estimados:** 0.4
Fecha de inicio: 27 de marzo de 2018	**Fecha de fin:** 28 de marzo de 2018
Programador responsable: Jesús Martínez Méndez y Barbara Dayanne Acosta Caldevilla	
Descripción: Va a permitir al administrador ver todos los clústers existentes en el servidor PostgreSQL. Al listar los clústers se deben mostrar de cada uno los siguientes campos: versión, nombre del clúster, estado, propietario, directorio de datos y las siguientes acciones: cambiar el estado (iniciar, detener, reiniciar y recargar), editar y eliminar.	

Tabla 3.14. Tarea de ingeniería # 14

Tarea	
Número de tarea: 14	**Número de Historia de usuario:** 8

Continúa en la próxima página

Tabla 3.14. Continuación de la página anterior

Nombre de la tarea: Iniciar un clúster.	
Tipo de tarea: Gestión	**Puntos estimados:** 0.4
Fecha de inicio: 29 de marzo de 2018	**Fecha de fin:** 30 de marzo de 2018
Programador responsable: Jesús Martínez Méndez y Barbara Dayanne Acosta Caldevilla	
Descripción: Va a permitir al administrador iniciar un clúster. Se muestra un mensaje de notificación al iniciarlo. En caso de existir algún error se muestra un mensaje.	

Tabla 3.15. Tarea de ingeniería # 15

Tarea	
Número de tarea: 15	**Número de Historia de usuario:** 8
Nombre de la tarea: Detener un clúster.	
Tipo de tarea: Gestión	**Puntos estimados:** 0.4
Fecha de inicio: 2 de abril de 2018	**Fecha de fin:** 3 de abril de 2018
Programador responsable: Jesús Martínez Méndez y Barbara Dayanne Acosta Caldevilla	
Descripción: Va a permitir al administrador detener un clúster. Se muestra un mensaje de notificación al detenerlo. En caso de existir algún error se muestra un mensaje.	

Tabla 3.16. Tarea de ingeniería # 16

Tarea	
Número de tarea: 16	**Número de Historia de usuario:** 8
Nombre de la tarea: Reiniciar un clúster.	
Tipo de tarea: Gestión	**Puntos estimados:** 0.6
Fecha de inicio: 4 de abril de 2018	**Fecha de fin:** 6 de abril de 2018
Programador responsable: Jesús Martínez Méndez y Barbara Dayanne Acosta Caldevilla	
Descripción: Va a permitir al administrador reiniciar un clúster. Se muestra un mensaje de notificación al reiniciarlo. En caso de existir algún error se muestra un mensaje.	

Tabla 3.17. Tarea de ingeniería # 17

Tarea	
Número de tarea: 17	**Número de Historia de usuario:** 8
Nombre de la tarea: Recargar el clúster.	
Tipo de tarea: Gestión	**Puntos estimados:** 0.6
Fecha de inicio: 9 de abril de 2018	**Fecha de fin:** 11 de abril de 2018
Programador responsable: Jesús Martínez Méndez y Barbara Dayanne Acosta Caldevilla	
Descripción: Va a permitir al administrador recargar un clúster. Se muestra un mensaje de notificación al recargarlo. En caso de existir algún error se muestra un mensaje.	

Tabla 3.18. Tarea de ingeniería # 18

Tarea	
Número de tarea: 18	**Número de Historia de usuario:** 9
Nombre de la tarea: Mostrar las reglas de control de acceso de un clúster.	
Tipo de tarea: Gestión	**Puntos estimados:** 0.4
Fecha de inicio: 12 de abril de 2018	**Fecha de fin:** 13 de abril de 2018
Programador responsable: Jesús Martínez Méndez y Barbara Dayanne Acosta Caldevilla	
Descripción: Va a permitir al administrador mostrar las reglas de control de acceso de un clúster. En caso de existir algún error se muestra un mensaje.	

Tabla 3.19. Tarea de ingeniería # 19

Tarea	
Número de tarea: 19	**Número de Historia de usuario:** 9
Nombre de la tarea: Modificar una regla de control de acceso de un clúster.	
Tipo de tarea: Gestión	**Puntos estimados:** 0.4
Fecha de inicio: 16 de abril de 2018	**Fecha de fin:** 17 de abril de 2018
Programador responsable: Jesús Martínez Méndez y Barbara Dayanne Acosta Caldevilla	
Descripción: Va a permitir al administrador modificar una regla de control de acceso de un clúster, cambiando los campos que necesita y en caso de existir campos incorrectos se le notifica. Se muestra un mensaje de notificación al modificar la regla. En caso de existir algún error al modificarla se muestra un mensaje.	

Tabla 3.20. Tarea de ingeniería # 20

Tarea	
Número de tarea: 20	**Número de Historia de usuario:** 9
Nombre de la tarea: Crear una regla de control de acceso de un clúster.	
Tipo de tarea: Gestión	**Puntos estimados:** 0.6
Fecha de inicio: 18 de abril de 2018	**Fecha de fin:** 20 de abril de 2018
Programador responsable: Jesús Martínez Méndez y Barbara Dayanne Acosta Caldevilla	
Descripción: Va a permitir al administrador crear una regla de control de acceso de un clúster, llenando los campos que se solicitan y en caso de existir campos incorrectos se le notifica. Se muestra un mensaje de notificación al crear la regla. En caso de existir algún error al crearla se muestra un mensaje.	

Tabla 3.21. Tarea de ingeniería # 21

Tarea	
Número de tarea: 21	**Número de Historia de usuario:** 9

Continúa en la próxima página

Tabla 3.21. Continuación de la página anterior

Nombre de la tarea: Eliminar una regla de control de acceso de un clúster.	
Tipo de tarea: Gestión	**Puntos estimados:** 0.6
Fecha de inicio: 23 de abril de 2018	**Fecha de fin:** 25 de abril de 2018
Programador responsable: Jesús Martínez Méndez y Barbara Dayanne Acosta Caldevilla	
Descripción: Va a permitir al administrador eliminar una regla de control de acceso de un clúster. Se muestra un mensaje de notificación al eliminarla. En caso de existir algún error al eliminarla se muestra un mensaje.	

3.3. Pruebas del software

La metodología XP se centra en la ejecución de pruebas a lo largo del proyecto, con el fin de asegurar la realización de lo planificado al inicio de cada iteración. En este proceso participa el equipo de desarrollo junto con el cliente con sus aportes (Castillo Asencio, 2016).

XP divide las pruebas del sistema en dos grupos: pruebas unitarias, encargadas de verificar el código y diseñada por los programadores, y pruebas funcionales destinadas a evaluar si al final de una iteración se consiguió la funcionalidad requerida diseñadas por el cliente final (Beck, 2000).

3.3.1. Pruebas unitarias

XP sugiere que las pruebas sean escritas antes de empezar con la codificación, por lo cual se crearon las pruebas al inicio del desarrollo. La planificación para la ejecución de estas pruebas se definió desarrollarlas y ejecutarlas durante el tiempo de desarrollo estimado por cada HU (Castillo Asencio, 2016).

Las pruebas unitarias son una de las piedras angulares de XP. Todos los módulos deben de pasar las pruebas unitarias antes de ser liberados o publicados. Por otra parte, las pruebas deben ser definidas antes de realizar el código ("*Test-driven programming*"). Que todo código liberado pase correctamente las pruebas unitarias es lo que habilita que funcione la propiedad colectiva del código. En este sentido, el sistema y el conjunto de pruebas debe ser guardado junto con el código, para que pueda ser utilizado por otros desarrolladores, en caso de tener que corregir, cambiar o recodificar parte del mismo.

Detección y corrección de errores: Cuando se encuentra un error ("*bug*"), éste debe ser corregido inmediatamente, y se deben tener precauciones para que errores similares no vuelvan a ocurrir. Asimismo, se generan nuevas pruebas para verificar que el error haya sido resuelto (Moreno Sánchez, 2013).

Las pruebas unitarias realizadas a través del módulo que ofrece Python llamado Doctest, se le fueron realizadas a la clase ConexionSSH a la funcionalidad Conectar y a la clase AdministracionPostgreSQL a las funcionalidades Listar_versiones_instaladas, Estado_servicio, Control_servicio, Control_clúster, Crear_clúster, Eliminar_clúster y Renombrar_clúster. A continuación, se muestra el resultado de las pruebas unitarias que se realizan al sistema, utilizando el módulo Doctest.

```
7 items passed all tests:
  12 tests in __main__.AdministracionPostgreSQL.Control_cluster
   6 tests in __main__.AdministracionPostgreSQL.Control_servicio
   2 tests in __main__.AdministracionPostgreSQL.Crear_cluster
   2 tests in __main__.AdministracionPostgreSQL.Eliminar_cluster
   2 tests in __main__.AdministracionPostgreSQL.Estado_servicio
   2 tests in __main__.AdministracionPostgreSQL.Listar_versiones_instaladas
   2 tests in __main__.AdministracionPostgreSQL.Renombrar_cluster
28 tests in 35 items.
28 passed and 0 failed.
Test passed.
```

Figura 3.1. Resultados de la ejecución de las pruebas unitarias utilizando el módulo Doctest

3.3.2. Pruebas de aceptación

Las pruebas de aceptación son creadas en base a las HU, en cada ciclo de la iteración del desarrollo. El cliente debe especificar uno o diversos escenarios para comprobar que una HU ha sido correctamente implementada. Las pruebas de aceptación son consideradas como "pruebas de caja negra" ("*Black box system tests*"). Los clientes son responsables de verificar que los resultados de éstas pruebas sean correctos. Asimismo, en caso de que fallen varias pruebas, deben indicar el orden de prioridad de resolución. Una HU no se puede considerar terminada hasta tanto pase correctamente todas las pruebas de aceptación. Dado que la responsabilidad es grupal, es recomendable publicar los resultados de las pruebas de aceptación, de manera que todo el equipo esté al tanto de esta información (ibíd.).

Estas pruebas fueron diseñadas por el cliente, pero con el apoyo de los programadores para poder guiar a los clientes en un correcto diseño de las pruebas y que al final se valide la funcionalidad de la mejor manera (Castillo Asencio, 2016).

Una HU puede tener todas las pruebas de aceptación que desee para asegurar su funcionamiento. El objetivo específico de esta prueba es garantizar que los requerimientos han sido cumplidos y que el sistema ha sido aceptable (Beck, 2000).

A continuación, una descripción de algunos de los casos de prueba de aceptación definidos, los restantes se encuentran en el Apéndice C.

Tabla 3.22. Prueba de aceptación # 1

Caso de prueba de aceptación	
Código: HU1_P1	**Historia de usuario:** 1
Nombre: Autenticar un usuario en el sistema.	
Descripción: Prueba para la funcionalidad autenticar un usuario en el sistema.	
Condiciones de ejecución: • El usuario debe estar previamente registrado. • El usuario y contraseña deben ser válidos.	

Continúa en la próxima página

Tabla 3.22. Continuación de la página anterior

Pasos de ejecución: Se intenta autenticar un usuario en el sistema con los datos válidos.
Resultados esperados: El usuario se autentica correctamente en el sistema.

Tabla 3.23. Prueba de aceptación # 2

Caso de prueba de aceptación	
Código: HU1_P2	**Historia de usuario:** 1
Nombre: Autenticar un usuario en el sistema.	
Descripción: Prueba para la funcionalidad autenticar un usuario en el sistema.	
Condiciones de ejecución:	
Pasos de ejecución: Se intenta autenticar un usuario en el sistema con los datos no válidos.	
Resultados esperados: El usuario no se autentica y el sistema lanza una notificación informando que los datos no son válidos.	

Tabla 3.24. Prueba de aceptación # 3

Caso de prueba de aceptación	
Código: HU2_P1	**Historia de usuario:** 2
Nombre: Gestionar máquinas servidoras.	
Descripción: Prueba para la funcionalidad gestionar máquinas servidoras.	
Condiciones de ejecución: • El usuario debe estar autenticado. • Al agregar una máquina servidora los datos introducidos deben ser correctos. • Para eliminar una máquina servidora, esta debe estar registrada. • Para editar una máquina servidora debe estar registrada y los datos editados deben ser correctos.	
Pasos de ejecución: • Se accede al listado de máquina servidora. • Se prueba agregar una máquina servidora. • Se prueba eliminar una máquina servidora. • Se prueba editar una máquina servidora.	
Resultados esperados: Se listan las máquinas servidoras registradas. El sistema realiza las acciones correctamente al agregar, eliminar y editar una máquina servidora; en cada caso notifica al usuario.	

Tabla 3.25. Prueba de aceptación # 4

Caso de prueba de aceptación	
Código: HU3_P1	**Historia de usuario:** 3
Nombre: Conectarse por SSH a una máquina servidora.	
Descripción: Prueba para la funcionalidad conectarse por SSH a una máquina servidora.	
Condiciones de ejecución: • El usuario debe estar autenticado. • La máquina servidora a la cual se va a realizar la conexión debe estar registrada en el sistema y se debe poder acceder a esta por SSH con los datos registrados.	
Pasos de ejecución: Se prueba realizar la conexión a una máquina servidora.	
Resultados esperados: La conexión se realiza correctamente, el sistema muestra la página de instalación.	

Tabla 3.26. Prueba de aceptación # 5

Caso de prueba de aceptación	
Código: HU4_P1	**Historia de usuario:** 4
Nombre: Instalar PostgreSQL.	
Descripción: Prueba para la funcionalidad instalar PostgreSQL.	
Condiciones de ejecución: • El usuario debe estar autenticado. • El sistema debe estar conectado a la máquina servidora. • La máquina servidora debe estar configurada para acceder a un repositorio en el que esté disponible PostgreSQL, y el repositorio debe estar activo. • No puede estar instalado PostgreSQL en la máquina servidora.	
Pasos de ejecución: Se selecciona la opción instalar PostgreSQL.	
Resultados esperados: El sistema realiza la instalación del servidor PostgreSQL, notifica al usuario y muestra la página de instalación con los paquetes de PostgreSQL instalados.	

Tabla 3.27. Prueba de aceptación # 6

Caso de prueba de aceptación	
Código: HU5.1_P1	**Historia de usuario:** 5
Nombre: Controlar el estado del servicio postgresql.	
Descripción: Prueba para detener el servicio postgresql, de la funcionalidad controlar el estado del servicio postgresql.	

Continúa en la próxima página

Tabla 3.27. Continuación de la página anterior

Condiciones de ejecución: • El usuario debe estar autenticado. • El sistema debe estar conectado a la máquina servidora. • La máquina servidora debe tener instalado el servidor PostgreSQL.
Pasos de ejecución: Se prueba detener el servicio postgresql.
Resultados esperados: Se muestra correctamente el estado del servicio postgresql (detenido). El sistema realiza correctamente la acción de detener el servicio postgresql y notifica que se detuvo.

Tabla 3.28. Prueba de aceptación # 7

Caso de prueba de aceptación	
Código: HU5.2_P1	**Historia de usuario:** 5
Nombre: Controlar el estado del servicio postgresql.	
Descripción: Prueba para iniciar el servicio postgresql, de la funcionalidad controlar el estado del servicio postgresql.	
Condiciones de ejecución: • El usuario debe estar autenticado. • El sistema debe estar conectado a la máquina servidora. • La máquina servidora debe tener instalado el servidor PostgreSQL. • El servidor PostgreSQL no debe tener configuraciones no válidas para que el servicio postgresql se inicie correctamente.	
Pasos de ejecución: Se prueba iniciar el servicio postgresql.	
Resultados esperados: Se muestra correctamente el estado del servicio postgresql (iniciado). El sistema realiza correctamente la acción de iniciar el servicio postgresql y notifica que se inició correctamente.	

Tabla 3.29. Prueba de aceptación # 8

Caso de prueba de aceptación	
Código: HU5.3_P1	**Historia de usuario:** 5
Nombre: Controlar el estado del servicio postgresql.	
Descripción: Prueba para reiniciar el servicio postgresql, de la funcionalidad controlar el estado del servicio postgresql.	

Continúa en la próxima página

Tabla 3.29. Continuación de la página anterior

Condiciones de ejecución: • El usuario debe estar autenticado. • El sistema debe estar conectado a la máquina servidora. • La máquina servidora debe tener instalado el servidor PostgreSQL. • El servidor PostgreSQL no debe tener configuraciones no válidas para que el servicio postgresql se reinicie correctamente.
Pasos de ejecución: Se prueba reiniciar el servicio postgresql.
Resultados esperados: Se muestra correctamente el estado del servicio postgresql (iniciado). El sistema realiza correctamente la acción de reiniciar el servicio postgresql y notifica que se reinició.

Tabla 3.30. Prueba de aceptación # 9

Caso de prueba de aceptación	
Código: HU6_P1	**Historia de usuario:** 6
Nombre: Desinstalar PostgreSQL.	
Descripción: Prueba para la funcionalidad desinstalar PostgreSQL.	
Condiciones de ejecución: • El usuario debe estar autenticado. • El sistema debe estar conectado a la máquina servidora. • La máquina servidora debe tener instalado el servidor PostgreSQL.	
Pasos de ejecución: Se selecciona la opción desinstalar PostgreSQL.	
Resultados esperados: El sistema realiza la desinstalación del servidor PostgreSQL, notifica al usuario y muestra la página de instalación con la opción de instalar PostgreSQL.	

Tabla 3.31. Prueba de aceptación # 10

Caso de prueba de aceptación	
Código: HU7.1_P1	**Historia de usuario:** 7
Nombre: Gestionar clústers.	
Descripción: Prueba para ver el listado de clústers, de la funcionalidad gestionar clústers.	
Condiciones de ejecución: • El usuario debe estar autenticado. • El sistema debe estar conectado a la máquina servidora. • La máquina servidora debe tener instalado el servidor PostgreSQL.	

Continúa en la próxima página

Tabla 3.31. Continuación de la página anterior

Pasos de ejecución: Se accede al listado de clústers.
Resultados esperados: Se listan los clústers existentes correctamente.

Tabla 3.32. Prueba de aceptación # 11

Caso de prueba de aceptación	
Código: HU7.2_P1	**Historia de usuario:** 7
Nombre: Gestionar clústers.	
Descripción: Prueba para agregar un clúster, de la funcionalidad gestionar clústers.	
Condiciones de ejecución: • El usuario debe estar autenticado. • El sistema debe estar conectado a la máquina servidora. • La máquina servidora debe tener instalado el servidor PostgreSQL. • Al agregar un clúster el nombre introducido debe ser correcto y distinto de los demás registrados.	
Pasos de ejecución: Se prueba agregar un clúster.	
Resultados esperados: El sistema realiza correctamente de acción de agregar un clúster y notifica que se agregó.	

Tabla 3.33. Prueba de aceptación # 12

Caso de prueba de aceptación	
Código: HU7.3_P1	**Historia de usuario:** 7
Nombre: Gestionar clústers.	
Descripción: Prueba para eliminar un clúster, de la funcionalidad gestionar clústers.	
Condiciones de ejecución: • El usuario debe estar autenticado. • El sistema debe estar conectado a la máquina servidora. • La máquina servidora debe tener instalado el servidor PostgreSQL. • Para eliminar un clúster, este debe estar registrado.	
Pasos de ejecución: Se prueba eliminar un clúster.	
Resultados esperados: El sistema realiza correctamente de acción de eliminar un clúster y notifica que se eliminó.	

Tabla 3.34. Prueba de aceptación # 13

Caso de prueba de aceptación	
Código: HU7.4_P1	**Historia de usuario:** 7
Nombre: Gestionar clústers.	
Descripción: Prueba para renombrar un clúster, de la funcionalidad gestionar clústers.	
Condiciones de ejecución: • El usuario debe estar autenticado. • El sistema debe estar conectado a la máquina servidora. • La máquina servidora debe tener instalado el servidor PostgreSQL. • Para renombrar un clúster, este debe estar registrado y el nuevo nombre debe ser correcto.	
Pasos de ejecución: Se prueba renombrar un clúster.	
Resultados esperados: El sistema realiza correctamente de acción de renombrar un clúster y notifica que se renombró.	

Tabla 3.35. Prueba de aceptación # 14

Caso de prueba de aceptación	
Código: HU8.1_P1	**Historia de usuario:** 8
Nombre: Gestionar el estado de un clúster.	
Descripción: Prueba para detener el clúster, de la funcionalidad gestionar el estado de un clúster.	
Condiciones de ejecución: • El usuario debe estar autenticado. • El sistema debe estar conectado a la máquina servidora. • La máquina servidora debe tener instalado el servidor PostgreSQL. • El servidor PostgreSQL debe tener al menos un clúster creado. • El clúster debe estar iniciado.	
Pasos de ejecución: Se prueba detener el clúster.	
Resultados esperados: Se muestra correctamente el estado del clúster (detenido). El sistema realiza correctamente la acción de detener el clúster y notifica que se detuvo.	

Tabla 3.36. Prueba de aceptación # 15

Caso de prueba de aceptación	
Código: HU8.2_P1	**Historia de usuario:** 8
Nombre: Gestionar el estado de un clúster.	

Continúa en la próxima página

Tabla 3.36. Continuación de la página anterior

Descripción: Prueba para iniciar el clúster, de la funcionalidad gestionar el estado de un clúster.
Condiciones de ejecución: • El usuario debe estar autenticado. • El sistema debe estar conectado a la máquina servidora. • La máquina servidora debe tener instalado el servidor PostgreSQL. • El servidor PostgreSQL debe tener al menos un clúster creado. • El clúster debe estar detenido.
Pasos de ejecución: Se prueba iniciar el clúster.
Resultados esperados: Se muestra correctamente el estado del clúster (iniciado). El sistema realiza correctamente la acción de iniciar el clúster y notifica que se inició.

Tabla 3.37. Prueba de aceptación # 16

Caso de prueba de aceptación	
Código: HU8.3_P1	**Historia de usuario:** 8
Nombre: Gestionar el estado de un clúster.	
Descripción: Prueba para reiniciar el clúster, de la funcionalidad gestionar el estado de un clúster.	
Condiciones de ejecución: • El usuario debe estar autenticado. • El sistema debe estar conectado a la máquina servidora. • La máquina servidora debe tener instalado el servidor PostgreSQL. • El servidor PostgreSQL debe tener al menos un clúster creado.	
Pasos de ejecución: Se prueba reiniciar el clúster.	
Resultados esperados: Se muestra correctamente el estado del clúster (iniciado). El sistema realiza correctamente la acción de reiniciar el clúster y notifica que se reinició.	

Tabla 3.38. Prueba de aceptación # 17

Caso de prueba de aceptación	
Código: HU8.4_P1	**Historia de usuario:** 8
Nombre: Gestionar el estado de un clúster.	
Descripción: Prueba para recargar el clúster, de la funcionalidad gestionar el estado de un clúster.	

Continúa en la próxima página

Tabla 3.38. Continuación de la página anterior

Condiciones de ejecución: • El usuario debe estar autenticado. • El sistema debe estar conectado a la máquina servidora. • La máquina servidora debe tener instalado el servidor PostgreSQL. • El servidor PostgreSQL debe tener al menos un clúster creado.
Pasos de ejecución: Se prueba recargar el clúster.
Resultados esperados: Se muestra correctamente el estado del clúster. El sistema realiza correctamente la acción de recargar el clúster y notifica que se recargó.

Resultados de las pruebas de aceptación

En la siguiente gráfica, se muestran los resultados obtenidos luego de realizar las pruebas de aceptación al sistema. Las pruebas fueron realizadas por iteraciones, presentando algunas no conformidades y se realizaron cambios durante el proceso de prueba quedando todas las no conformidades resueltas.

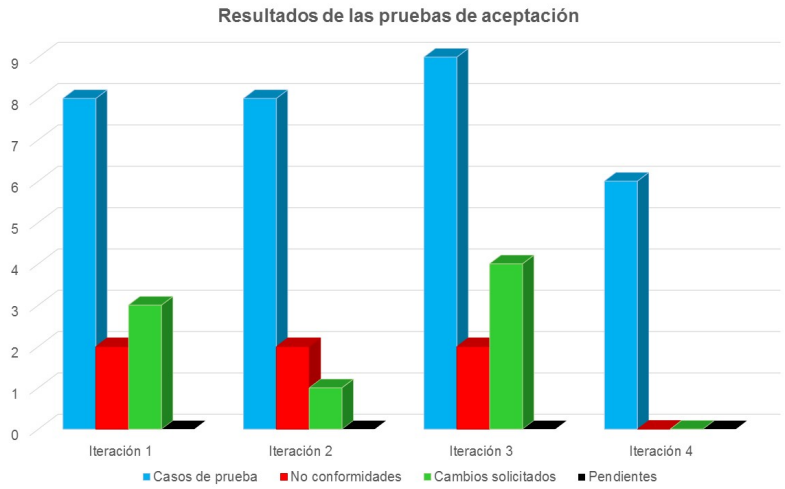

Figura 3.2. Resultados de las pruebas de aceptación

3.4. Conclusiones del capítulo

Al concluir el presente capítulo se obtuvo el producto terminado. Realizar la implementación siguiendo los estándares de codificación definidos permitió desarrollar un código reutilizable y comprensible por todos los integrantes del equipo de desarrollo. Se implementaron las tareas de la ingeniería definidas para cada HU. Finalmente, mediante los casos de prueba funcionales, la aplicación fue probada y seguidamente fueron resueltas todas las no conformidades.

Conclusiones

A partir del reconocimiento de la problemática existente dentro del Departamento de Componentes y la realización del estudio de las herramientas informáticas homólogas existentes, se demostró la viabilidad del sistema propuesto ya que ninguna cumplía con las necesidades específicas del área. El estudio realizado sobre las diferentes metodologías de software, definió a XP como la más acertada para guiar el proceso de diseño e implementación del sistema que se desarrolló. La selección de las herramientas informáticas y lenguajes permitió establecer las bases para la implementación correcta del sistema. Se realizó una descripción del proceso que se realiza dentro del departamento y a partir del entendimiento del mismo se propuso una solución capaz de satisfacer las necesidades del cliente. Se puntualizaron las pautas a seguir en la implementación del sistema posibilitando una mayor agilidad, claridad y organización en cuanto al código. Así como la realización de las pruebas que define la metodología XP que permitieron validar y certificar el correcto funcionamiento del sistema. El producto obtenido permite realizar los procesos de administración y monitoreo del servidor PostgreSQL de manera centralizada a través de una interfaz web.

Por todo lo anteriormente expuesto, se concluye que los objetivos propuestos para el presente trabajo han sido cumplidos satisfactoriamente. El sistema desarrollado contribuirá de manera significativa para la administración y monitoreo de servidores PostgreSQL en el departamento de Componentes del centro TLM de la Facultad 2.

Recomendaciones

Los resultados obtenidos en este trabajo sugieren una serie de recomendaciones para seguir perfeccionando la administración y monitoreo de servidores PostgreSQL en el departamento de Componentes del centro TLM de la Facultad 2. Por lo que los autores recomiendan:

- Realizar la internacionalización del sistema AD&MONPSQL, para una mayor afiliación de usuarios de lenguas extranjeras.
- Continuar la investigación con el propósito de aumentar las funcionalidades del sistema AD&MONPSQL, obteniendo mejoras en futuras versiones de la misma.

Acrónimos

AJAX Asynchronous JavaScript And XML. 14

BLOB Binary Large OBject. 10
BPMN Notación para el Modelado de Procesos de Negocio. 4, 15

CPU Unidad Central de Procesamiento. 9, 18, 75, 83
CRC Clase-Responsabilidad-Colaborador. 4, 33, 35
CSS Hojas de estilo en cascada. 13, 14

DBA Administradores de Bases de Datos. 9
DHCP Dynamic Host Configuration Protocol. 8
DNS Domain Name System. 8
DOM Document Object Model. 14

FTP File Transfer Protocol. 8

GRASP Patrones Generales de Software para Asignación de Responsabilidades. 32, 33

HTML5 Lenguaje de Marcado de Hipertexto. 13
HTTP Protocolo de Transferencia de Hipertexto. 10, 18
HU Historias de Usuario. 4, 17, 20, 21, 25, 27, 29, 35, 36, 38, 45, 46, 54

IP Protocolo de Internet. 21, 39

LDAP Lightweight Directory Access Protocol. 8

PHP PHP Hypertext Preprocessor. 8

RAM Memoria de Acceso Aleatorio. 18, 75, 83

SGBD Sistema Gestor de Bases de Datos. 1–6, 15
SSH Secure SHell. 8, 10, 11, 15, 18, 19, 21, 22, 25, 27, 28, 33, 39, 47, 48

TIC Tecnologías de Información y Comunicación. 1
TLM Telemática. 1, 3, 18, 55, 56

UCI Universidad de las Ciencias Informáticas. 1, 11, 12

VCS Version Control System. 15

XP Programación Extrema. 12, 15, 20, 36, 37, 45, 55

Referencias bibliográficas

BECK, Kent, 2000. *Extreme programming explained: embrace change*. addison-wesley professional (vid. págs. 20, 23-25, 27, 28, 44, 45).

CALDERA VERGARA, Roberto, 2017. Estudio del framework de desarrollo web Django [online] [visitado 2018-02-22]. Disponible desde: https://ebuah.uah.es/dspace/handle/10017/32018 (vid. págs. 29, 30).

CANÓS, José H y LETELIER, Mª Carmen Penadés Patricio, 2012. Metodologías ágiles en el desarrollo de software [online] [visitado 2017-05-09]. Disponible desde: https://scholar.google.es/scholar?hl=es&as_sdt=0%2C5&q=Can%C3%B3s%2C+J.H.%2C+Letelier%2C+P.+%26+Penad%C3%A9s%2C+M.+C.+%282012%29.+%E2%80%9CMetodolog%C3%ADas+%C3%A1giles+en+el+desarrollo+de+software%E2%80%9D.+Espa%C3%B1a%3A+Universidad+Polit%C3%A9cnica+de+Valencia.&btnG= (vid. pág. 36).

CASTILLO ASENCIO, Pedro Luis, 2016. Desarrollo e implementación de un sistema web para generar valor en una pyme aplicando una metodología ágil. Caso de estudio: Manufibras Perez SRL [online] [visitado 2018-04-01]. Disponible desde: http://cybertesis.unmsm.edu.pe/bitstream/handle/cybertesis/4668/Castillo_ap.pdf?sequence=1&isAllowed=y (vid. págs. 35, 44, 45).

CIENCIAS INFORMÁTICAS, Universidad de las, 2018. Universidad de las Ciencias Informáticas [online] [visitado 2018-05-25]. Disponible desde: www.uci.cu (vid. pág. 2).

DATAPRIX, 2018. Arquitectura de PostgreSQL [online] [visitado 2018-05-25]. Disponible desde: http://www.dataprix.com/72-arquitectura-postgresql (vid. pág. 6).

EGUÍLUZ PÉREZ, Javier, 2012. *Introducción a javascript* [online] [visitado 2017-06-19]. Disponible desde: http://www.librosweb.es/javascript (vid. pág. 13).

EMS DATABASE MANAGEMENT SOLUTIONS, 2016. EMS SQL Manager for PostgreSQL [online] [visitado 2016-10-19]. Disponible desde: www.sqlmanager.net/products/postgresql/manager (vid. pág. 10).

ESPAÑA, Sarasty y FERNANDO, Hugo, 2016. *Documentación y análisis de los principales frameworks de arquitectura de software en aplicaciones empresariales* [online] [visitado 2018-02-19]. Disponible desde: http://sedici.unlp.edu.ar/bitstream/handle/10915/52183/Documento_completo..pdf?sequence=3. Tesis doctoral. Facultad de Informática (vid. pág. 28).

FERNÁNDEZ CASASAYAS, Llilian, 2014. Módulo de administración para el servicio PostgreSQL en la Herramienta para la Migración y Administración de Servicios Telemáticos [online] [visitado 2017-05-05] (vid. pág. 11).

GAUCHAT, Juan Diego, 2012. *El gran libro de HTML5, CSS3 y Javascript*. Marcombo. Url: www.minkbooks.com (vid. pág. 13).

GÓMEZ FUENTES, Dra. María del Carmen, 2013. Bases de Datos. *Gestión* [online], págs. 201 [visitado 2017-11-27]. Disponible desde: http://cua.uam.mx/pdfs/conoce/libroselec/Notas_del_curso_Bases_de_Datos.pdf (vid. pág. 5).

GONZÁLEZ DUQUE, Raúl, 2011. *Python para todos.* [online] [visitado 2017-06-14]. Disponible desde: http://mundogeek.net/tutorial-python/ (vid. pág. 13).

HERNÁNDEZ LEÓN, Rolando Alfredo y COELLO GONZÁLEZ, Sayda, 2012. *El proceso de investigación científica*. 2.ª ed. Ed. por MINISTERIO DE EDUCACIÓN SUPERIOR, Editorial Universitaria del. ISBN 978-959-16-1557-2 (vid. págs. 3, 4).

HOLOVATY, Adrian y KAPLAN-MOSS, Jacob, 2009. *The Definitive Guide To Django* [online]. Ed. por APRESS, Editorial [visitado 2017-06-19]. Disponible desde: http://index-of.co.uk/Programming-Library/The%20Definitive%20Guide%20To%20Django%20-%20Holovaty,%20Kaplan-Moss%20-%20Apress%20(2009).pdf (vid. pág. 14).

JEFFRIES, Ron; ANDERSON, Ann y HENDRICKSON, Chet, 2001. *Extreme programming installed*. Addison-Wesley Professional (vid. pág. 20).

JETBRAINS, 2017. PyCharm Features [online] [visitado 2017-06-27]. Disponible desde: https://www.jetbrains.com/pycharm/features/ (vid. pág. 15).

LARMAN, Craig, 2003. *UML y Patrones*. 2da. ISBN 8420534382 (vid. pág. 31).

MALDONADO, Tituaña y MAURICIO, Wilson, 2017. *Estudio de la integración de los framework bootstrap y primefaces para el desarrollo de aplicaciones web adaptativas con java server faces Aplicativo: Sistema de control de notas, para la unidad educativa mariano Suarez Veintimilla* [online] [visitado 2018-02-08]. Disponible desde: http://repositorio.utn.edu.ec/bitstream/123456789/6903/2/ARTICULO.pdf. B.S. thesis (vid. pág. 14).

MANAGEENGINE, 2017a. General FAQs [online] [visitado 2017-02-20]. Disponible desde: https://www.manageengine.com/products/applications_manager/applications-management-genfaq.html (vid. pág. 9).

MANAGEENGINE, 2017b. PostgreSQL Monitoring [online] [visitado 2017-02-20]. Disponible desde: https://www.manageengine.com/products/applications_manager/postgresql-monitoring.html (vid. pág. 9).

MEDINA VELANDIA, Lucy Nohemy y LÓPEZ LÓPEZ, Wílmer Mesías, 2015. ESCOGER UNA METODOLOGÍA PARA DESARROLLAR SOFTWARE, DIFÍCIL DECISIÓN. *Educación en Ingeniería* [online] [visitado 2018-02-20]. ISSN 1900-8260. Disponible desde: http://www.educacioneningenieria.org (vid. pág. 12).

MORENO SÁNCHEZ, Camilo Andrés, 2013. *Manejo de sistemas de información para la organización de procesos de la gerencia de protección y aseguramiento de ingresos de la compañía Claro Colombia SA* [online] [visitado 2018-04-03]. Disponible desde: http://metadirectorio.org/handle/10983/712. B.S. thesis (vid. págs. 44, 45).

PARNELL, Brid-Aine, 2015. De terabytes a zettabytes: el crecimiento de los datos mundiales. *Think Progress* [online] [visitado 2016-11-05]. Disponible desde: www.think-progress.com/es/blog/consideration/de-terabytes-a-zettabytes-el-crecimiento-de-los-datos-mundiales/ (vid. pág. 1).

POSTGRESQL, El equipo de desarrollo de, 2016. *Manual del usuario de PostgreSQL*. Url: https://www.postgresql.org/docs/manuals/ (vid. pág. 7).

PYTHON SOFTWARE FOUNDATION, 2018. PEP 8 – Style Guide for Python Code [online] [visitado 2018-04-02]. Disponible desde: https://www.python.org/dev/peps/pep-0008/ (vid. pág. 36).

SÁNCHEZ, Yoiry López; CASTILLA, Yusbel Chávez; ESCALONA, Liliana Vilahomat; VARGAS, Jarvin Antón y SORÍ, Julio César, 2017. SISTEMA WEB PARA LA GESTIÓN DEL CONTROL DE ALMACÉN EN LA MINI-INDUSTRIA EL MAMBÍ DEL MUNICIPIO DE FLORENCIA EN LA PROVINCIA DE CIEGO DE ÁVILA. *Universidad&Ciencia*. Vol. 6, n.º 3, págs. 36-51. Url: http://revistas.unica.cu/index.php/uciencia/article/view/302/1090 (vid. pág. 14).

SILBERSCHATZ, Abraham; KORTH, Henry F. y SUDARSHAN, S., 2002. *Fundamentos de Bases de Datos* [online]. 4.ª ed. McGraw-Hill [visitado 2017-11-27]. ISBN 8448136543. Disponible desde: https://s3.amazonaws.com/academia.edu.documents/37358813/Fundamentos_de_Bases_de_Datos.pdf?AWSAccessKeyId=AKIAIWOWYYGZ2Y53UL3A&Expires=1511758406&Signature=tgnivdkUO9tByEmoDtX9nxkfuag%3D&response-content-disposition=inline%3B%20filename%3DFundamentos_de_Bases_de_Datos.pdf (vid. pág. 6).

SQLITE, 2017. SQLite [online] [visitado 2017-06-26]. Disponible desde: https://www.sqlite.org/about.html (vid. pág. 15).

THE POSTGRESQL GLOBAL DEVELOPMENT GROUP, 2017. About PostgreSQL [online] [visitado 2017-11-20]. Disponible desde: https://www.postgresql.org/about/ (vid. págs. 1, 6).

VENTURA, Toapanta y ESTEFANÍA, Carolina, 2013. Modelamiento, diseño de procesos bajo tecnología BPMN y propuesta de mejora en la empresa COCEBET S.A. [online] [visitado 2017-06-23]. Disponible desde: http://bibdigital.epn.edu.ec/handle/15000/7067 (vid. pág. 15).

VISUAL PARADIGM, 2018. Visual Paradigm [online] [visitado 2018-02-08]. Disponible desde: https://www.visual-paradigm.com/ (vid. pág. 14).

WEBMIN, 2017. Webmin [online] [visitado 2017-02-22]. Disponible desde: http://www.webmin.com/intro.html (vid. pág. 8).

Bibliografía

BECK, Kent, 2000. *Extreme programming explained: embrace change*. addison-wesley professional (vid. págs. 20, 23-25, 27, 28, 44, 45).

CALDERA VERGARA, Roberto, 2017. Estudio del framework de desarrollo web Django [online] [visitado 2018-02-22]. Disponible desde: https://ebuah.uah.es/dspace/handle/10017/32018 (vid. págs. 29, 30).

CANÓS, José H y LETELIER, Mª Carmen Penadés Patricio, 2012. Metodologías ágiles en el desarrollo de software [online] [visitado 2017-05-09]. Disponible desde: https://scholar.google.es/scholar?hl=es&as_sdt=0%2C5&q=Can%C3%B3s%2C+J.H.%2C+Letelier%2C+P.+%26+Penad%C3%A9s%2C+M.+C.+%282012%29.+%E2%80%9CMetodolog%C3%ADas+%C3%A1giles+en+el+desarrollo+de+software%E2%80%9D.+Espa%C3%B1a%3A+Universidad+Polit%C3%A9cnica+de+Valencia.&btnG= (vid. pág. 36).

CASTILLO ASENCIO, Pedro Luis, 2016. Desarrollo e implementación de un sistema web para generar valor en una pyme aplicando una metodología ágil. Caso de estudio: Manufibras Perez SRL [online] [visitado 2018-04-01]. Disponible desde: http://cybertesis.unmsm.edu.pe/bitstream/handle/cybertesis/4668/Castillo_ap.pdf?sequence=1&isAllowed=y (vid. págs. 35, 44, 45).

CIENCIAS INFORMÁTICAS, Universidad de las, 2018. Universidad de las Ciencias Informáticas [online] [visitado 2018-05-25]. Disponible desde: www.uci.cu (vid. pág. 2).

DATAPRIX, 2018. Arquitectura de PostgreSQL [online] [visitado 2018-05-25]. Disponible desde: http://www.dataprix.com/72-arquitectura-postgresql (vid. pág. 6).

DATE, Chris J., 2001. *Introducción a los Sistemas de Bases de Datos* [online]. 7.ª ed. Pearson Educación [visitado 2016-11-03]. Disponible desde: https://books.google.es/books?hl=es&lr=&id=Vhum351T-K8C&oi=fnd&pg=PR17&dq=Bases+de+datos+concepto&ots=fzH4TU97af&sig=W2wRAlfNeYgVT6_mQXJpXgy3AYY#v=onepage&q=Bases%20de%20datos%20concepto&f=false.

DUARTE, Abdías Gómez; DÍAZ, Anyi Leandra Portela y OLIVOS, Oscar Javier, 2015. SMART-TEST: una aplicación de las TIC al proceso de evaluación de estudiantes universitarios. *Revista Matices tecnológicos*. Vol. 5. Url: http://publicaciones.unisangil.edu.co/index.php/revista-matices-tecnologicos/article/view/208/195.

EGUÍLUZ PÉREZ, Javier, 2012. *Introducción a javascript* [online] [visitado 2017-06-19]. Disponible desde: http://www.librosweb.es/javascript (vid. pág. 13).

EMS DATABASE MANAGEMENT SOLUTIONS, 2016. EMS SQL Manager for PostgreSQL [online] [visitado 2016-10-19]. Disponible desde: www.sqlmanager.net/products/postgresql/manager (vid. pág. 10).

ESPAÑA, Sarasty y FERNANDO, Hugo, 2016. *Documentación y análisis de los principales frameworks de arquitectura de software en aplicaciones empresariales* [online] [visitado 2018-02-19]. Disponible desde: http://sedici.unlp.edu.ar/bitstream/handle/10915/52183/Documento_completo..pdf?sequence=3. Tesis doctoral. Facultad de Informática (vid. pág. 28).

FERNÁNDEZ CASASAYAS, Llilian, 2014. Módulo de administración para el servicio PostgreSQL en la Herramienta para la Migración y Administración de Servicios Telemáticos [online] [visitado 2017-05-05] (vid. pág. 11).

GAUCHAT, Juan Diego, 2012. *El gran libro de HTML5, CSS3 y Javascript*. Marcombo. Url: www.minkbooks.com (vid. pág. 13).

GÓMEZ FUENTES, Dra. María del Carmen, 2013. Bases de Datos. *Gestión* [online], págs. 201 [visitado 2017-11-27]. Disponible desde: http://cua.uam.mx/pdfs/conoce/libroselec/Notas_del_curso_Bases_de_Datos.pdf (vid. pág. 5).

GONZÁLEZ DUQUE, Raúl, 2011. *Python para todos.* [online] [visitado 2017-06-14]. Disponible desde: http://mundogeek.net/tutorial-python/ (vid. pág. 13).

HERNÁNDEZ LEÓN, Rolando Alfredo y COELLO GONZÁLEZ, Sayda, 2012. *El proceso de investigación científica*. 2.ª ed. Ed. por MINISTERIO DE EDUCACIÓN SUPERIOR, Editorial Universitaria del. ISBN 978-959-16-1557-2 (vid. págs. 3, 4).

HOLOVATY, Adrian y KAPLAN-MOSS, Jacob, 2009. *The Definitive Guide To Django* [online]. Ed. por APRESS, Editorial [visitado 2017-06-19]. Disponible desde: http://index-of.co.uk/Programming-Library/The%20Definitive%20Guide%20To%20Django%20-%20Holovaty,%20Kaplan-Moss%20-%20Apress%20(2009).pdf (vid. pág. 14).

JEFFRIES, Ron; ANDERSON, Ann y HENDRICKSON, Chet, 2001. *Extreme programming installed*. Addison-Wesley Professional (vid. pág. 20).

JETBRAINS, 2017. PyCharm Features [online] [visitado 2017-06-27]. Disponible desde: https://www.jetbrains.com/pycharm/features/ (vid. pág. 15).

JOSKOWICZ, José, 2008. Reglas y Prácticas en eXtreme Programming. *Universidad de Vigo* [online], págs. 22 [visitado 2017-05-30]. Disponible desde: http://iie.fing.edu.uy/~josej/docs/XP%20-%20Jose%20Joskowicz.pdf.

LARMAN, Craig, 2003. *UML y Patrones*. 2da. ISBN 8420534382 (vid. pág. 31).

MALDONADO, Tituaña y MAURICIO, Wilson, 2017. *Estudio de la integración de los framework bootstrap y primefaces para el desarrollo de aplicaciones web adaptativas con java server faces Aplicativo: Sistema de control de notas, para la unidad educativa mariano Suarez Veintimilla* [online] [visitado 2018-02-08]. Disponible desde: http://repositorio.utn.edu.ec/bitstream/123456789/6903/2/ARTICULO.pdf. B.S. thesis (vid. pág. 14).

MANAGEENGINE, 2017a. General FAQs [online] [visitado 2017-02-20]. Disponible desde: https://www.manageengine.com/products/applications_manager/applications-management-genfaq.html (vid. pág. 9).

MANAGEENGINE, 2017b. PostgreSQL Monitoring [online] [visitado 2017-02-20]. Disponible desde: https://www.manageengine.com/products/applications_manager/postgresql-monitoring.html (vid. pág. 9).

MEDINA VELANDIA, Lucy Nohemy y LÓPEZ LÓPEZ, Wílmer Mesías, 2015. ESCOGER UNA METODOLOGÍA PARA DESARROLLAR SOFTWARE, DIFÍCIL DECISIÓN. *Educación en Ingeniería* [online] [visitado 2018-02-20]. ISSN 1900-8260. Disponible desde: http://www.educacioneningenieria.org (vid. pág. 12).

MORENO SÁNCHEZ, Camilo Andrés, 2013. *Manejo de sistemas de información para la organización de procesos de la gerencia de protección y aseguramiento de ingresos de la compañía Claro Colombia SA* [online] [visitado 2018-04-03]. Disponible desde: http://metadirectorio.org/handle/10983/712. B.S. thesis (vid. págs. 44, 45).

PARNELL, Brid-Aine, 2015. De terabytes a zettabytes: el crecimiento de los datos mundiales. *Think Progress* [online] [visitado 2016-11-05]. Disponible desde: www.think-progress.com/es/blog/consideration/de-terabytes-a-zettabytes-el-crecimiento-de-los-datos-mundiales/ (vid. pág. 1).

POSTGRESQL, El equipo de desarrollo de, 2016. *Manual del usuario de PostgreSQL*. Url: https://www.postgresql.org/docs/manuals/ (vid. pág. 7).

PYTHON SOFTWARE FOUNDATION, 2018. PEP 8 – Style Guide for Python Code [online] [visitado 2018-04-02]. Disponible desde: https://www.python.org/dev/peps/pep-0008/ (vid. pág. 36).

SÁNCHEZ, Yoiry López; CASTILLA, Yusbel Chávez; ESCALONA, Liliana Vilahomat; VARGAS, Jarvin Antón y SORÍ, Julio César, 2017. SISTEMA WEB PARA LA GESTIÓN DEL CONTROL DE ALMACÉN EN LA MINI-INDUSTRIA EL MAMBÍ DEL MUNICIPIO DE FLORENCIA EN LA PROVINCIA DE CIEGO DE ÁVILA. *Universidad&Ciencia*. Vol. 6, n.º 3, págs. 36-51. Url: http://revistas.unica.cu/index.php/uciencia/article/view/302/1090 (vid. pág. 14).

SILBERSCHATZ, Abraham; KORTH, Henry F. y SUDARSHAN, S., 2002. *Fundamentos de Bases de Datos* [online]. 4.ª ed. McGraw-Hill [visitado 2017-11-27]. ISBN 8448136543. Disponible desde: https://s3.amazonaws.com/academia.edu.documents/37358813/Fundamentos_de_Bases_de_

Datos.pdf?AWSAccessKeyId=AKIAIWOWYYGZ2Y53UL3A&Expires=1511758406&Signature=tgnivdkUO9tByEmoDtX9nxkfuag%3D&response-content-disposition=inline%3B%20filename%3DFundamentos_de_Bases_de_Datos.pdf (vid. pág. 6).

SQLITE, 2017. SQLite [online] [visitado 2017-06-26]. Disponible desde: https://www.sqlite.org/about.html (vid. pág. 15).

THE POSTGRESQL GLOBAL DEVELOPMENT GROUP, 2017. About PostgreSQL [online] [visitado 2017-11-20]. Disponible desde: https://www.postgresql.org/about/ (vid. págs. 1, 6).

VAN ROSSUM, Guido et al., 2007. Python Programming Language. En: *Python Programming Language. USENIX Annual Technical Conference*. Vol. 41, pág. 36.

VENTURA, Toapanta y ESTEFANÍA, Carolina, 2013. Modelamiento, diseño de procesos bajo tecnología BPMN y propuesta de mejora en la empresa COCEBET S.A. [online] [visitado 2017-06-23]. Disponible desde: http://bibdigital.epn.edu.ec/handle/15000/7067 (vid. pág. 15).

VISUAL PARADIGM, 2018. Visual Paradigm [online] [visitado 2018-02-08]. Disponible desde: https://www.visual-paradigm.com/ (vid. pág. 14).

WEBMIN, 2017. Webmin [online] [visitado 2017-02-22]. Disponible desde: http://www.webmin.com/intro.html (vid. pág. 8).

Generado con LaTeX: 4 de junio de 2018: 2:04pm