



Facultad 2

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

Sistema de administración de servidores PostgreSQL para el Departamento de Desarrollo de Componentes del Centro Telemática V2.0

Autora: Sunaidy Camila Jauregui Fajardo

Tutores: Ing. Antonio Hernández Domínguez

Ing. Osmar Capote Vázquez

La Habana, 4 de junio del 2019

“Año 61 de la Revolución”

*"No pretendamos que las cosas cambien, si siempre hacemos lo mismo.
La crisis, es la mejor bendición que puede sucederle a personas y países, porque la crisis trae
progresos.
La creatividad nace de la angustia como el día nace de la noche oscura.
Es en la crisis que nace la inventiva, los descubrimientos y las grandes estrategias.
Quien supera la crisis se supera a sí mismo sin quedar superado.
Quien atribuye a la crisis sus fracasos y penurias, violenta su propio talento y respeta más a
los problemas que a soluciones. La verdadera crisis, es la crisis de la incompetencia.
El inconveniente de las personas y los países es la pereza para encontrar las salidas y soluciones.
Sin crisis no hay desafíos, sin desafíos la vida es una rutina, una lenta agonía. Sin crisis no hay
méritos.
Es en la crisis donde aflora lo mejor de cada uno, porque sin crisis todo viento es caricia. Hablar
de crisis es promoverla, y callar en la crisis es exaltar el conformismo. En vez de esto, trabajemos
duro. Acabemos de una vez con la única crisis amenazadora, que es la tragedia de no querer luchar
por superarla."*

Albert Einstein.

1879-1955.

Declaración de autoría

Declaro ser autora de la presente tesis que tiene por título: “Sistema de administración de servidores PostgreSQL para el Departamento de Desarrollo de Componentes del Centro Telemática V2.0” y reconozco a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo. Para que así conste firmo la presente a los ____ días del mes de junio del año 2019.

Sunaidy Camila Jauregui Fajardo

Firma del Autora

Ing. Osmar Capote Vázquez

Ing. Antonio Hernández Domínguez

Firma del Tutor

Firma del Tutor

Datos de contacto

Tutor 1:

Antonio Hernández Domínguez

Profesor Asistente de la Facultad 2. Director del Centro de Telemática de la UCI.

Centro de Telemática. Facultad 2. UCI

Email: ahdominguez@uci.cu

Tutor 2:

Osmar Capote Vázquez

Ingeniero en ciencias informáticas.

Centro de Telemática. Facultad 2. UCI

Email: ocapote@uci.cu

Agradecimientos

Primeramente, quisiera agradecerle a la única persona que nunca he valorado y que siempre ha estado presente en cada momento de mi vida y que le debo todo lo que soy, te amo mami. Le agradezco a mi papi por toda la comprensión y el amor dado. A mi abuela Camila por mimarme y aconsejarme siempre que puede. A mis hermanos, pero sobre todo a Fer por ser la persona que me hace querer ser mejor cada día. A mi tía Yamila y mi tío Rafael por siempre exigir la mejor versión de mí. A Bia porque pese a la distancia siempre ha estado presente, te quiero amiga.

También le agradezco al escuadrón mete la pata y a Dionni por darme la mejor bienvenida a la universidad y por hacer de mi primer año inolvidable. A Fofi, Nito, Geider, Geiser, Fabra, Anicel y Ailed por aguantarme y ser tan buenos amigos. A Chan, el Luisma, Pedro, Vega y el Robe por ayudarme a estudiar para cada prueba. A Huguito por cada noche dedicada a que aprendiera inglés. A Eileen y Mickey por acogerme como un miembro más de la familia. A mi perra, a Daniela, a Liliana y a Rachel por hacer de los últimos años de la carrera una locura. A Baby y Jesus por su apoyo incondicional y por transmitirme tranquilidad en momentos de desesperación.

A la persona que ha estado presente desde mis primeros pasos en la UCI, que ha llorado y reído a mi lado, que me ha aconsejado pese a que no le hago caso ni a media palabra que me dice, al hombre de mi vida y dueño de mis pensamientos, el que me levantó cuando pensé que ya todo estaba perdido, al hombre que amo y que no tengo palabras para decirte gracias.

Le agradezco a mis tutores Antonio y Osmar por las atenciones y porque han logrado que hoy sea una profesional independiente y capaz. A Idalis, Yenli y Ramon por toda la ayuda brindada y a Mónica y Baquero por todas las recomendaciones realizadas.

A todas esas personas que de una forma u otra han estado presente en los momentos importantes de la carrera. ¡Muchas Gracias a todos!

Dedicatoria

Dedicado especialmente a mis padres, mis hermanos y mis abuelas, quienes son partícipes de cada momento importante de mi vida...

Resumen

El Departamento de Desarrollo de Componentes del Centro Telemática adscrito a la Facultad 2 de la Universidad de las Ciencias Informáticas (UCI) cuenta con un sistema para administrar y monitorear de forma remota y centralizada los servidores de bases de datos PostgreSQL (AD&MONPSQL), presentes en el mismo. Actualmente se cuenta con servidores instalados en los sistemas operativos CentOS7 y Windows Server 2012. AD&MONPSQL carece de las funcionalidades (instalar, desinstalar, iniciar, detener y reiniciar) necesarias para realizar la administración de PostgreSQL en dichos sistemas operativos, así como, realizar la gestión de las bases de datos geoespaciales (BDG) y la configuración del protocolo capa de sockets seguros (SSL, según siglas en idioma inglés), necesario para garantizar la seguridad en la conexión a los servidores. En la presente investigación se implementó el sistema AD&MONPSQL v2.0 que garantiza la gestión de las BDG y la configuración de los certificados SSL en la administración de los servidores PostgreSQL instalados en los sistemas operativos CentOS7 y Windows Server 2012. Para guiar el desarrollo de la gestión del resultado se utilizó como metodología de desarrollo de software el Proceso Unificado Ágil (AUP) para la UCI, como lenguaje de programación Python, HTML, CCS y JavaScript, como marco de trabajo Django, Bootstrap y jQuery y como entorno de desarrollo integrado (IDE) el PyCharm.

Palabras clave: administrar, bases de datos geoespaciales, PostgreSQL, SSL.

Abstract

The Department of Development of Components of the Telematic Center assigned to the Faculty 2 of the University of Informatics Sciences (UCI) has a system to administer and monitor in a remote and centralized way the servers of PostgreSQL databases (AD & MONPSQL), present in it Currently has servers installed on the operating systems CentOS7 and Windows Server 2012. AD & MONPSQL lacks the functionality (install, uninstall, start, prevent and restart) to perform the administration of PostgreSQL in such systems, as well as, perform the management of the geospatial databases (BDG) and the configuration of the SSL protocol, is necessary to guarantee the security in the connection to the servers. In this research, the AD & MONPSQL v2.0 system was implemented, which guarantees the management of BDG and the configuration of SSL certificates in the administration of PostgreSQL servers in the CentOS7 and Windows Server 2012 operating systems. To guide the development of the management, the result, the tool, the language, the software and the writing, the language, the language, the language, the language, the language, the language, the language and the programming. IDE) the PyCharm.

Keywords: *manage, geospatial database, PostgreSQL, SSL.*

Índice

Introducción.....	1
Capítulo I. Fundamentos teóricos	6
1.1. Conceptos asociados.....	6
1.2. Estudio de los diferentes sistemas informáticos que administran el SGBD PostgreSQL	9
1.2.1. Internacional.....	9
1.2.2. Nacional	10
1.3. Metodología de desarrollo de software	11
1.4. Herramientas y tecnologías informáticas	12
1.5. Conclusiones del capítulo.....	14
Capítulo II. Análisis y diseño del sistema AD&MONPSQL v2.0.....	16
2.1. Características de la propuesta de solución.....	16
2.2. Requisitos del sistema	17
2.2.1. Técnicas de obtención de requisitos del software	17
2.2.2. Requisitos funcionales.....	18
2.2.3. Requisitos no funcionales.....	20
2.3. Validación de requisitos	21
2.4. Especificación de los requisitos funcionales	21
2.5. Especificación de los requisitos no funcionales	28
2.6. Arquitectura del sistema	30
2.7. Patrones de diseño	32
2.8. Modelo de diseño	33
2.8.1. Diagrama de clases de diseño.....	33
2.9. Diagrama de secuencia.....	34
2.10. Conclusiones del capítulo.....	36
Capítulo III. Implementación y Prueba.....	37
3.1. Modelo de implementación.....	37
3.2. Diagrama de despliegue	38
3.3. Pruebas del software.....	38
3.4. Disciplinas de pruebas.....	39
3.5. Niveles de Prueba.....	39
3.6. Método de prueba.....	40

3.7. Tipos de pruebas.....	41
3.8. Diseños de Casos de Prueba.....	42
3.9. Resultados de las pruebas.....	45
3.10. Conclusiones del capítulo.....	51
Conclusiones.....	52
Recomendaciones.....	53
Referencias Bibliográficas.....	54
Bibliografía.....	57

Índice de tablas

Tabla 1: Comparación de los sistemas.	11
Tabla 2 HU N° 20: Crear base de datos geoespacial en el sistema operativo CentOS7.....	21
Tabla 3 HU N° 21: Crear base de datos geoespacial en el sistema operativo Windows Server 2012.	22
Tabla 4 HU N° 22: Editar base de datos geoespacial en el sistema operativo CentOS7.....	23
Tabla 5 HU N° 23: Editar base de datos geoespacial en el sistema operativo Windows Server 2012.....	24
Tabla 6 HU N° 24: Eliminar base de datos geoespacial en el sistema operativo CentOS7.	25
Tabla 7 HU N° 25: Eliminar base de datos geoespacial en el sistema operativo Windows Server 2012.	26
Tabla 8 HU N° 26: Listar de las BD existentes en el sistema operativo CentOS7.....	27
Tabla 9 HU N° 27: Listar de las BD existentes en el sistema operativo Windows Server 2012.....	27
Tabla 10: Especificación RNF Interoperabilidad	28
Tabla 11: Especificación RNF Usabilidad	29
Tabla 12: Sección editar base de datos geoespacial correspondiente al caso de prueba Gestionar BDG.	42
Tabla 13: Descripción las variables correspondientes con la sección editar base de datos geoespacial	43
Tabla 14: Sección adicionar base de datos geoespacial correspondiente al caso de prueba Gestionar BDG.	43
Tabla 15: Descripción las variables correspondientes con la sección adicionar base de datos geoespacial	44
Tabla 16: Sección eliminar base de datos geoespacial correspondiente al caso de prueba Gestionar BDG.	44
Tabla 17: Caso de prueba de caja blanca para el camino básico 1	47
Tabla 18: Caso de prueba de caja blanca para el camino básico 2.....	47
Tabla 19: Resultados de la prueba de caja negra 1ra iteración.....	48
Tabla 20: Resultados de la prueba de caja negra 2da iteración.	48
Tabla 21: Resultados de la prueba de caja negra 3ra iteración.....	49

Índice de figuras

Figura 1: Propuesta de solución	17
Figura 2: MVT variante del MVC.....	31
Figura 3: Funcionamiento de la arquitectura MVT (Condori, 2012).	31
Figura 4: Clase AdministracionPostgreSQL	32
Figura 5: Clase BDG	32
Figura 6: Diagrama de clase de diseño del paquete Gestionar_Base de_Datos_Geospacial.....	34
Figura 7: Diagrama de secuencia del paquete Gestionar_Base_de_Datos_Geospacial (Escenario crear)	35
Figura 8: Diagrama de secuencia del paquete Gestionar_Base_de_Datos_Geospacial (Escenario editar)	35
Figura 9: Diagrama de secuencia del paquete Gestionar_Base_de_Datos_Geospacial (Escenario eliminar).....	36
Figura 10: Diagrama de secuencia del paquete Gestionar_Base_de_Datos_Geospacial (Escenario listar)	36
Figura 11: Ejemplo de uso del estándar CapWords	37
Figura 12: Ejemplo de uso del estándar lower_case_with_underscores	37
Figura 13: Diagrama de despliegue.....	38
Figura 14: Código del método Desinstalar_pgis.	46
Figura 15: Representación del grafo de flujo de camino básico del método Desinstalar_pgis.	46
Figura 16: No conformidades detectadas contra cantidad de iteraciones de prueba.	50

Introducción

Las tecnologías de la información y las comunicaciones han solucionado inconvenientes como la utilización de los volúmenes considerables de información, junto a la necesidad de almacenarla de forma segura y eficiente. Factores que favorecieron el surgimiento de las bases de datos (BD) como fuente de información. Estas garantizan la obtención de la información de forma rápida, además de facilitar el registro y el acceso a las mismas, ahorrando tanto espacio físico como tiempo al consultarlas. Para facilitar la gestión y administración de las BD de forma eficiente se implementaron los sistemas gestores de bases de datos (SGBD) entre los que se destacan Oracle, MySQL, Microsoft SQL Server y PostgreSQL según el sitio oficial DB-Engines.

Cuba, con el objetivo de lograr un desarrollo tecnológico, contribuir al fortalecimiento de la soberanía tecnológica y al desarrollo del software libre, ha desarrollado un programa para la informatización de la sociedad. Programa en el cual se encuentra inmersa la Universidad de las Ciencias Informáticas (UCI); donde a partir del vínculo docencia - investigación - producción como modelo de formación, sirve de soporte a la industria cubana del software. La UCI cuenta con una amplia red de centros de desarrollo de software que realizan proyectos informáticos, con gran impacto en la informatización de la sociedad cubana. Entre ellos está el Centro Telemática (TLM) adscrito a la Facultad 2 de la misma que tiene como objetivo “Desarrollar 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” (UCI, 2002).

El Centro TLM cuenta con el Departamento de Desarrollo de Componentes dedicado a identificar e integrar estándares técnicos a la tecnología de desarrollo del centro, así como demandar nuevas tecnologías que favorezcan a la producción. El departamento desarrolla varias aplicaciones y servicios que poseen servidores de BD, utilizándose como SGBD PostgreSQL. Este cuenta con una aplicación web llamada Sistema de Administración y Monitoreo de PostgreSQL (AD&MONPSQL) en su versión 1.0, la cual fue creada en el departamento y según sus necesidades teniendo como objetivo administrar y monitorear de manera centralizada los servidores de PostgreSQL. A continuación, fueron determinadas las principales fortalezas del sistema que están dadas por (Acosta, Martínez; 2018):

- Configuración del sistema operativo a utilizar (Ubuntu y Debian).
- Instalar, desinstalar y controlar el estado del servicio PostgreSQL.
- Monitoreo del uso de la memoria RAM y de la Unidad Central de Procesamiento (CPU, en idioma inglés) del servidor.

Actualmente el departamento cuenta con servidores de BD instalados en los sistemas operativos CentOS7 y Windows Server 2012. Servidores que el cliente necesita administrar de forma centralizada, pero la aplicación carece de las funcionalidades (instalar, desinstalar, iniciar, reiniciar y detener) necesarias para satisfacer las necesidades del mismo. Lo que trae consigo que el proceso se realiza de forma manual, obstaculizándose el trabajo al administrador.

La seguridad en las comunicaciones entre un cliente y un servidor de BD es un factor fundamental para garantizar la confiabilidad de la información almacenada en las BD (González, Beltrán, Fuentes; 2016). Uno de los mecanismos utilizados para la seguridad en las conexiones, es el protocolo SSL¹, el cual no está concebido para ser configurado desde la actual aplicación. PostgreSQL cuenta con los archivos necesarios para la configuración del protocolo SSL, el cual se realiza en dos de los ficheros de configuración principales (postgresql.conf y pg_hba.conf). Estos ficheros poseen una gran cantidad de instrucciones y la configuración errónea de los mismos, puede afectar la disponibilidad de la información en un determinado servidor. Por lo que el proceso se vuelve engoroso para el administrador y puede conllevar a la ocurrencia de errores humanos.

En el Departamento de Desarrollo de Componentes se necesita trabajar con información y bases de datos geoespaciales (BDG). Pero aún la aplicación AD&MONPSQL en su versión 1.0 carece de las funcionalidades (crear, editar, eliminar y listar) necesarias para administrar las BDG, lo que trae consigo que se obstaculice la administración de las mismas para el administrador.

A partir de las limitaciones de la anterior aplicación se define el siguiente **problema a resolver**: Las limitaciones que presenta AD&MONPSQL en la gestión de las BDG y los certificados SSL, haciendo engoroso el trabajo de administración de los servidores PostgreSQL instalados en los sistemas operativos CentOS7 y Windows Server 2012 de forma centraliza.

A partir del problema científico se define como **objeto de estudio**: El proceso de administración de los servidores PostgreSQL y la gestión de las BDG de forma centralizada. La presente investigación tiene como **objetivo general**: Implementar el sistema AD&MONPSQL v2.0 que garantice la gestión de las BDG y la configuración de los certificados SSL en la administración de los servidores PostgreSQL instalados en los sistemas operativos CentOS7 y Windows Server 2012. Para dar cumplimiento al objetivo general se definen los siguientes **objetivos específicos**:

- Elaborar el marco teórico conceptual de la investigación.
- Realizar el modelado y la especificación de requisitos de la propuesta de solución.

¹ SSL Secure Sockets Layer en español (Capa de sockets seguros).

- Implementar las funcionalidades necesarias para que AD&MONPSQL gestione las BDG y configure los certificados SSL en la administración de los servidores PostgreSQL instalados en los sistemas operativos CentOS7 y Windows Server 2012.
- Realizar pruebas internas y de aceptación al sistema.

Enmarcados en el **campo de acción**: La gestión de las BDG y la configuración de certificados SSL en la administración de los servidores PostgreSQL instalados en los sistemas operativos CentOS7 y Windows Server 2012.

Para accionar del diagnóstico al pronóstico se declaran las siguientes **tareas de investigación**:

- Análisis de los principales conceptos y definiciones asociados a la gestión de las BDG y la configuración de certificados SSL en la administración de los servidores PostgreSQL instalados en los sistemas operativos CentOS7 y Windows Server 2012 para profundizar en el dominio de la investigación.
- Análisis de sistemas que administran el SGBD PostgreSQL de forma centralizada a nivel internacional y nacional para sentar las bases de la investigación.
- Análisis de las herramientas, tecnologías y la metodología de desarrollo de software a utilizar, establecidas por el departamento, para lograr la familiarización con el entorno de trabajo.
- Identificación de los requisitos que posee el sistema AD&MONPSQL 2.0 a partir de las técnicas de captura de requisito, para establecer las funcionalidades que debe cumplir la propuesta de solución.
- Descripción de los requisitos previamente identificados, para explicar con detalles el flujo de cada uno de ellos.
- Desarrollo de los artefactos señalados por la metodología de desarrollo de software, para representar el diseño y la estructura con la que cuenta el sistema AD&MONPSQL 2.0.
- Desarrollar las funcionalidades anteriormente identificadas para que el sistema AD&MONPSQL 2.0 satisfaga las necesidades del cliente.
- Realizar las pruebas unitarias, funcionales y de aceptación para la validación del funcionamiento del sistema.

Teniendo como posible resultado la obtención de la versión 2.0 del sistema AD&MONPSQL que permita gestionar las de BDG y configurar los certificados SSL en la administración de los servidores PostgreSQL instalados en los sistemas operativos CentOS7 y Windows Server 2012 de forma centralizada por medio de una interfaz visual.

Métodos científicos:

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). Para el progreso de la presente investigación se utilizaron los siguientes métodos científicos de la investigación como guía para el desarrollo que condujo al logro del objetivo propuesto.

Métodos teóricos:

Analítico-Sintético: Posibilitó extraer e identificar conceptos, características y otros elementos de la bibliografía consultada para sentar las bases en el estudio del comportamiento y las características del sistema AD&MONPSQL con el fin de determinar la necesidad de actualización del mismo como propuesta de solución.

Modelación: Se utilizó para ofrecer información acerca de la estructura del diseño y las diferentes aristas y relaciones que se dan en la administración de PostgreSQL con el sistema AD&MONPSQL, así como, facilitar el entendimiento y estudio del fenómeno real. Esto permite reflejar la estructura, las relaciones y características de la solución propuesta.

Métodos Empíricos:

Observación: Se utilizó como punto de partida en el diseño de la investigación, ya que se observan todos los procedimientos que se realizan para administrar los servidores PostgreSQL por medio del sistema AD&MONPSQL. Para ello se elabora el plan de observación donde se precisa lo siguiente:

- Tipo de observación: Abierta, es cuando el objeto de la investigación sabe que está siendo observado.
- Indicadores de observación:
 - La gestión de las BDG.
 - La administración de PostgreSQL.
 - Generación de los certificados SSL y la configuración de los mismos en los ficheros de PostgreSQL.
- Condiciones en que se realiza la observación: Se realiza en condiciones naturales.
- Resultados deseados: Adquirir conocimientos y recoger información en el diagnóstico inicial y final de la gestión de las BDG y la configuración de certificados SSL en la administración de los servidores PostgreSQL instalados en los sistemas operativos CentOS7 y Windows Server 2012.

Entrevista no estructurada: Se aplicó al especialista en el tema con el objetivo de obtener criterios de expertos. Se realizan antes, durante y después de la investigación como fuente de recopilación de información y para entender qué problemas existen en la aplicación web AD&MONPSQL y saber cómo solucionarlos. Los indicadores analizados fueron los siguientes: (Anexo 3 Guía para la realización de la entrevista).

- La gestión de las BDG.
- La administración de PostgreSQL.
- Generación de los certificados SSL y la configuración de los mismos en los ficheros de PostgreSQL.

El documento está estructurado por 3 capítulos, las conclusiones generales, las referencias bibliográficas, la bibliografía general utilizada y los anexos. La estructura de los capítulos se define a continuación:

Capítulo 1: Fundamentación teórica: Se definen los conceptos asociados y se analizan las principales herramientas informáticas que permiten administrar al SGBD PostgreSQL. Una vez realizado este análisis se describen brevemente las principales características de AD&MONPSQL, como sistema base de la aplicación a desarrollar. Por último, se fundamentan las tecnologías, lenguajes de programación, metodología y herramientas informática utilizadas en el desarrollo del sistema.

Capítulo 2: Análisis y diseño del sistema: Se presenta la propuesta de solución al problema planteado, quedando definidos los requisitos funcionales y no funcionales del sistema. Así como todos los artefactos que propone la metodología de desarrollo de software. También se detalla la arquitectura del sistema y elementos del diseño del sistema (patrones GRASP).

Capítulo 3: Implementación y pruebas del sistema: Se describen los estándares de codificación seguidos para la implementación de la propuesta de solución. Así como representación de los diagramas de componentes y de despliegue correspondientes con dicha propuesta. Para culminar se representan las pruebas realizadas al sistema, a partir de los indicadores a evaluar para sus resultados.

Capítulo I. Fundamentos teóricos

En el presente capítulo se abordan los conceptos asociados con la investigación, se realiza el análisis de los diferentes sistemas que administren el SGBD PostgreSQL. También se caracteriza la metodología, las tecnologías, lenguajes y herramientas a utilizar para el desarrollo de la solución.

1.1. Conceptos asociados

Bases de datos

Colección de datos, donde los datos están lógicamente relacionados entre sí, tienen una definición y descripción común 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).

En aras de profundizar en la definición de las bases de datos, es importante mencionar el libro: “Bases de datos relacionales y modelado de datos UF1471”, en el cual se explica que las bases de datos relacionales son aquellas en que se representan los datos y las relaciones entre los datos mediante un conjunto de tablas, cada una con un nombre único, donde una fila de una tabla representa una relación entre un conjunto de valores (Jiménez, 2014). A su vez existen las BDG que se conceptualizan a continuación.

Bases de datos geoespaciales

Modelo genérico para el tratamiento de información geoespacial, almacena objetos geográficos, sus atributos, sus relaciones (espaciales o no) y el comportamiento de cada uno de sus elementos. Incluye definiciones, reglas de integridad y comportamiento para una colección de datos integrados que se usan para representar una colección de capas temáticas. (Gutiérrez Nieto C., Castellanos Fajardo L. A., 2014).

Es un conjunto de datos geográficos organizados de tal manera que permiten la realización de análisis y la gestión del territorio dentro de aplicaciones de Sistemas de Información Geográfica (SIG). Además, una BDG se utiliza de soporte para la implantación de servicios geográficos relacionados con las Infraestructuras de Datos Espaciales, y su contenido es la base fundamental en los procesos de producción cartográficos (Geodatabase, 2018)

Sistema Gestor de Bases de Datos

Colección de programas informáticos para la gestión de bases de datos con el objetivo de evitar la manipulación directa por el usuario a una base de datos, así como establecer un marco estándar para que los datos sean manipulados y accedidos desde otros programas (Miguel y Ángel, 2017).

Es un sistema que permite la creación, gestión y administración de bases de datos y de las estructuras necesarias para el almacenamiento y búsqueda de información, ya sea de forma interactiva o a través de un lenguaje de programación. Se encargan de la privacidad, la integridad, la seguridad de los datos y la interacción con el sistema operativo. Estos sistemas gestionan el almacenamiento y la recuperación de la información de una BD de forma eficiente y práctica.

Un SGBD permite:

- Crear y definir una BD. Especifica la estructura, el tipo de datos, las restricciones y las relaciones.
- Gestionar los datos. Se utiliza para consultar, actualizar e insertar datos utilizando el lenguaje de manipulación de datos (DML, en idioma inglés).
- Controlar el acceso a los datos de la BD. Establecer los mecanismos de seguridad de acceso de los usuarios.
- Mantener la integridad y consistencia. Evitar que los datos sean alterados por cambios no permitidos.
- Controlar el acceso compartido de la BD. Supervisar la interacción entre usuarios concurrentes.
- Recuperar y respaldar la información. Reestablecer la información en caso de que ocurra un fallo en la BD.

Servidor de bases de datos

Un servidor de bases de datos es un sistema bajo la arquitectura cliente/servidor que proporciona servicios de gestión, administración y protección de la información (datos) a través de conexiones de red (Iruela, 2016). Es necesario un servidor de bases de datos para aquellas BD con múltiples usuarios donde se puede acceder a ellas desde terminales o equipos con un programa llamado cliente.

Es un tipo de software de servidor que permiten la organización de la información mediante el uso de tablas, índices y registros. A nivel de hardware, un servidor de base de datos es un equipo informático especializado en servir consultas a clientes remotos o locales que solicitan información o realizan modificaciones a los registros y tablas que existen dentro de las bases de datos del sistema (en muchos casos desde un servidor web o de aplicaciones).

PostgreSQL

PostgreSQL es un sistema de gestión de bases de datos objeto-relacional, distribuido bajo licencia de distribución de software Berkeley (BSD, según acrónimo de sus siglas en idioma inglés) y con su código fuente

disponible libremente. Utiliza un modelo cliente/servidor y multiprocesos para garantizar la estabilidad del sistema (*The PostgreSQL Global Development Group*, 2018).

PostgreSQL está considerado como el sistema de gestión de bases de datos de código abierto más avanzado del mundo, debido a que proporciona un gran número de funcionalidades que normalmente sólo se encuentran en sistemas de bases de datos comerciales tales como Oracle. Entre sus características principales se encuentra el soporte de distintos tipos de datos como fecha, monetarios, elementos gráficos, datos sobre redes y cadenas de bits. Además, cuenta con comunidades activas, varias de estas en castellano. Tiene un soporte nativo para los lenguajes más populares, por ejemplo: PHP, C, C++, Perl, Python, entre otros.

PostGIS

PostGIS es la extensión geográfica del sistema de gestión de bases de datos PostgreSQL, que permite almacenar objetos de tipo SIG como parte de las tablas de datos. Agrega soporte a funciones de análisis y procesamiento de objetos geográficos permitiendo que las consultas de ubicación se ejecuten en SQL. Esencialmente, PostGIS es una herramienta poderosa que permite gestionar datos geográficos complejos y explorar visualmente estos datos (Márquez, 2015). Es un módulo que añade soporte de objetos geográficos a la base de datos objeto-relacional PostgreSQL. Convierte una base de datos objeto-relacional en una base de datos espacial, permitiendo la utilización de los datos en los diferentes sistemas que gestionan información de este tipo.

Protocolo SSL

Provee privacidad y confiabilidad entre dos aplicaciones que se comunican. Permite tanto al cliente como al servidor autenticarse e intercambiar un algoritmo de inscripción y llaves criptográficas antes de que el protocolo de aplicación reciba o transmita cualquier bit de datos (Diana Carolina, Juan Felipe, 2017). Algunas de las ventajas que presenta este protocolo es de aplicación independiente, capaz de negociar un algoritmo de cifrado y una clave de sesión. Garantiza la privacidad ya que todos los datos del protocolo de la aplicación se transmiten encriptados.

Protocolo SSH

Secure Shell (SSH) es un protocolo de transporte seguro y de bajo nivel. Proporciona cifrado sólido, autenticación criptográfica de host y protección de la integridad. A diferencia de otros protocolos de comunicación remota, SSH encripta la sesión de conexión, haciendo imposible que alguien pueda obtener contraseñas no encriptadas (Ylonen y Lonvick, 2006).

1.2. Estudio de los diferentes sistemas informáticos que administran el SGBD PostgreSQL

Se realizó el estudio de los sistemas homólogos existentes a nivel internacional y nacional para lograr una mejor comprensión de las características necesarias para la mejora del sistema. A continuación, se expone el análisis desarrollado.

1.2.1. Internacional

Webmin

Webmin representa una herramienta para la configuración mediante la web, para GNU/Linux y otros sistemas Unix. Permite el control de los servidores no solo de forma local, sino también de forma remota (Webmin, 2017). Esta herramienta posee algunas características que enriquecen la investigación, estas son:

- Crear base de datos geoespacial: Una vez instalados PostgreSQL y PostGIS se procede a seleccionar desde la página principal la opción “Crear un nuevo enlace de base de datos” arriba o debajo de la tabla de iconos de BD existentes. Una vez seleccionado ingresa un nombre único para la base de datos en el campo “Nombre de las BD” y se procede a seleccionar la opción “Crear” con la cual se agregará la base de datos y volverá a la página principal sistema.
- Gestión de seguridad mediante SSL: Consiste en realizar una solicitud de certificado a una autoridad de certificación, una vez obtenido el certificado se debe añadir al servidor de configuración del webmin específicamente en “*Manage SSL Certificate*”. Después de ser copiados los certificados deben ser configurados y encriptados. Una vez realizados estos pasos ya estará habilitado el certificado.

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 (EMS Database Management Solutions, 2016). Esta herramienta posee algunas características que enriquecen la investigación, estas son:

- Crear base de datos geoespacial: Una vez instalados PostgreSQL y PostGIS se accede el botón correspondiente de la barra de herramientas “btnCreateDB”, después se procede a configurar todas las opciones necesarias (habilitar PostGIS en la base de datos) utilizando el asistente que guiará todo el proceso de creación de una nueva base de datos.
- Eliminar base de datos geoespacial: Se selecciona la base de datos, después la opción eliminar el elemento del menú principal de las BD y se confirma la caída en la ventana de diálogo correspondiente.

- Gestión de la seguridad mediante SSL: permite incorporar al sistema certificados SSL.

1.2.2. Nacional

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

Módulo de administración para el servicio PostgreSQL en la Herramienta para la Migración y Administración de Servicios Telemáticos (HMAST) 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 bases de datos. Esto es posible debido a que la aplicación se conectaría vía SSH al servidor y accede a los ficheros de configuración y los modifica de acuerdo a las necesidades del gestor de configuración (Fernández Casasayas, 2014). Esta herramienta posee algunas características que enriquecen la investigación, estas son:

- Instalar y desinstalar el servicio PostgreSQL: Se realiza a partir de botones con el nombre de la acción a realizar y una vez seleccionada se visualiza una imagen animada que indica el progreso de la instalación o desinstalación.
- Iniciar, reiniciar y detener el servicio PostgreSQL: Se realiza a partir de iconos una vez seleccionada la opción deseada el sistema ejecuta el comando “service postgresql start” en caso de iniciar el servicio, el comando “service postgresql stop” en caso de detener el servicio y en el caso de reiniciar se ejecuta el comando “service postgresql restart”.
- Habilitar y deshabilitar la opción de conexiones SSL: Consiste en habilitar y deshabilitar las conexiones seguras al servidor a través de una casilla de verificación.

AD&MONPSQL 1.0

Aplicación desarrollada en el año 2017 en la UCI. Permite administrar y monitorear por vía SSH a los diferentes servidores de PostgreSQL alojados en los sistemas operativos Ubuntu y Debian (Acosta, Martínez; 2018). La misma cuenta con una organización interna de 3 capas lógicas:

- Interfaz: En esta capa están implementadas todas las interfaces gráficas con las que interactúa el usuario. Estas interfaces se relacionan directamente con las funcionalidades que se encuentran implementadas en la capa de negocio.
- Negocio: En esta capa están incluidas todas las tareas y funcionalidades que realiza el sistema.
- Bases de datos: En esta capa se encuentran las BD con las que trabaja el sistema.

Dentro de las características principales del sistema se encuentran las siguientes que son las que enriquecen la presente investigación (Acosta, Martínez; 2018):

- Conectarse por SSH a una máquina virtual servidora (se concibe que una máquina virtual servidora es una máquina ya sea virtual o física donde se encuentran instalados uno o varios servidores PostgreSQL).
- Instalar PostgreSQL.
- Controlar el estado del servicio PostgreSQL.
- Desinstalar PostgreSQL.

A continuación, se muestra una tabla con la comparación de los sistemas informáticos anteriormente descritos tomándose como criterio de comparación las necesidades del cliente descritas en la situación problemática.

Tabla 1: Comparación de los sistemas.

Sistemas /Criterio	Gestión de BDG	Generación de certificados SSL	Administración centralizada por la Web
Webmin	X	-	-
EMS SQL Manager para PostgreSQL	X	-	X
HMAST	-	-	X
AD&MONPSQL	-	-	X

Una vez realizada la comparación de los sistemas informáticos se determina realizar la versión 2.0 del sistema AD&MONPSQL por las razones siguiente:

- Para la utilización del Webmin, es necesaria la instalación de este en cada máquina donde se vaya a configurar PostgreSQL.
- EMS SQL Manager for PostgreSQL es necesario pagar una licencia para su utilización.
- HMAST solo está concebido para software libre.

1.3. Metodología de desarrollo de software

AUP para la UCI

Las metodologías de desarrollo de software constituyen el conjunto de procedimientos, técnicas y herramientas que ayuda a los desarrolladores a realizar un nuevo producto. Con el objetivo de guiar el desarrollo de la propuesta de solución se utilizó la metodología AUP para la UCI, la cual fue seleccionada por las siguientes características:

- Para la realización del proyecto es necesaria la utilización de una metodología ágil ya que el equipo de trabajo es pequeño y de vasta experiencia en el tema a desarrollar.
- Define el ciclo de vida determinado para la actividad productiva de la UCI (Inicio, Ejecución, Cierre) y a su vez coincide con la duración del proyecto.
- Consta de una amplia documentación.
- Sus artefactos presentan una descripción del comportamiento del sistema fácil de entender.

La misma plantea 4 escenarios para la obtención de los requisitos del software, los cuales se describen a continuación:

- Escenario 1: Proyectos que modelen el negocio con “caso de uso del negocio” solo pueden modelar el sistema con “caso de uso del sistema”.
- Escenario 2: Proyectos que modelen el negocio con “modelo conceptual” solo pueden modelar el sistema con “caso de uso del sistema”.
- Escenario 3: Proyectos que modelen el negocio con “descripción del proceso de negocio” solo pueden modelar el sistema con “descripción de requisitos por procesos”.
- Escenario 4: Proyectos que no modelen el negocio solo pueden modelar el sistema con “historia de usuario”.

Dentro de los escenarios planteados se seleccionó el escenario 4 dado a que:

- El negocio se encuentra bien definido.
- El cliente está siempre acompañando al equipo de desarrollo.
- El proyecto no es extenso.
- Para lograr la correlación entre la versión a desarrollar y la anterior del sistema AD&MONPSQL.

1.4. Herramientas y tecnologías informáticas

Las tecnologías informáticas utilizadas fueron las seleccionadas por los desarrolladores de la v 1.0 de AD&MONPSQL con el objetivo de garantizar la compatibilidad entre ambas versiones.

Lenguaje informático del lado del servidor:

Python 3.6: Es un lenguaje de programación interpretado, orientado a objetos, multiplataforma, flexible y su implementación está bajo una licencia de código abierto PSFL ². (Python Software Foundation, 2017). Se utilizó para dar cumplimiento a las funcionalidades del sistema.

Lenguaje informático del lado del cliente:

HTML 5: Lenguaje de Marcado de Hipertexto (HTML5, por sus siglas en inglés) es 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. Además, provee básicamente tres características: estructura, estilo y funcionalidad (Gauchat, 2012). Se utilizó para realizar interfaces amigables y mostrar información al usuario a través del navegador web.

JavaScript: Es un lenguaje interpretado. Principalmente es utilizado para crear páginas web dinámicas. Basado en objetos y guiado por eventos, diseñado específicamente para el desarrollo de aplicaciones cliente-servidor dentro del ámbito de Internet (Eguíluz Pérez, 2012). Se utilizó en la validación de los formularios llenados por el usuario en la realización del sistema.

CCS3: Define el diseño de un sitio web en un documento externo y eso mismo permite que modificando ese documento (la hoja de estilo en cascada o CSS) se pueda cambiar el diseño entero de un sitio web, el mismo sitio web puede variar totalmente de diseño cambiando solo la CSS, sin tocar para nada los documentos HTML, JSP³ o ASP⁴ que lo componen (Gauchat, 2012). Se utilizó para estandarizar el diseño del sistema.

Lenguaje de modelado:

Lenguaje Unificado de Modelado (UML)⁵ es un lenguaje para visualizar, especificar, construir y documentar sistemas de software. Se utilizó en la realización del modelo de diseño de la aplicación. Se utilizó para realizar todos los diagramas en la fase de análisis de la presente investigación.

Marcos de trabajos:

Django 1.11.6: Es un marco de trabajo para el desarrollo web escrito en Python, que permite construir y mantener aplicaciones web de alta calidad. Django, provee un alto nivel de abstracción para patrones comunes

² PSFL: Python Software Foundation License (Licencia de fundación de software).

³ Jsp (siglas en inglés de Java Server Pages) es una tecnología que ayuda a los desarrolladores de software a crear páginas web dinámicas basadas en HTML, XML, entre otros tipos de documentos.

⁴ Asp (siglas en inglés de Active Server Pages) es la tecnología desarrollada por Microsoft para la creación de páginas dinámicas del servidor.

⁵ UML (Unified Modeling Language).

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).

jQuery 1.9.1: Es una biblioteca de JavaScript, que permite simplificar la manera de interactuar con los documentos HTML, manipular el árbol Document Object Model (DOM), gestionar eventos, desarrollar animaciones y agregar interacción con la técnica AJAX⁶ a páginas web (Sánchez; Castilla; Escalona; Vargas y Sorí, 2017). Se utilizó para facilitar el trabajo con el lenguaje interpretado JavaScript.

Bootstrap 3.3.1: Es un marco de trabajo 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 de las aplicaciones web (Maldonado y Mauricio, 2017).

Herramienta CASE:

Visual Paradigm 8.0: Visual Paradigm para UML (VP-UML) es una poderosa herramienta CASE⁷ multiplataforma (Windows/Linux/Mac OS X) que soporta el ciclo de vida completo del desarrollo de software. 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 (Visual Paradigm Group, 2018).

IDE de desarrollo:

PyCharm 2018.1.2: Este IDE proporciona la finalización inteligente de códigos, inspecciones de códigos, resaltado de errores sobre la marcha, soluciones rápidas 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 Sistema Control de Versiones (SCV) principal y herramientas de BD incorporadas y capacidades de desarrollo remoto con intérpretes remotos (Jetbrains, 2017).

1.5. Conclusiones del capítulo

Al realizar el estudio de los conceptos asociados al dominio se logró un mejor entendimiento del problema a resolver. Luego de realizado el análisis de múltiples herramientas para la administración del gestor PostgreSQL, se decidió la elaboración de AD&MONPSQL 2.0 teniendo en cuenta que no cumplen con todas las funcionalidades (administración de PostgreSQL, gestión de las BDG y la configuración de los certificados

⁶ AJAX (acrónimo de Asynchronous JavaScript and XML) es una técnica que permite la comunicación asíncrona entre un servidor y un navegador en formato XML mediante programas escritos en JavaScript.

⁷ CASE: Computer Aided Software Engineering en español (Ingeniería de Software Asistida por Computadoras).

SSL) que se requieren para la propuesta de solución. Se describe la metodología: AUP para la UCI, definida por el Centro, para guiar el proceso de desarrollo de software en la elaboración de la propuesta de solución. Además, se detallaron las herramientas informáticas a utilizar en la implementación del sistema con el fin de desarrollar un sistema multiplataforma y de código abierto.

Capítulo II. Análisis y diseño del sistema AD&MONPSQL v2.0

En el presente capítulo se diseñan los artefactos que propone la metodología de desarrollo de software seleccionada en el capítulo anterior. Para la realización de los artefactos fue necesaria la captura de los requisitos del software y a partir de estos se describen los diagramas de clases de diseño y los diagramas de secuencia. También se expone la arquitectura del sistema, los patrones de diseño que fueron utilizados y el modelo de diseño de la aplicación.

2.1. Características de la propuesta de solución

A continuación, se presentan las funcionalidades que contiene la propuesta solución con la que se garantiza la administración centralizada por la web de los servidores PostgreSQL instalados en los sistemas operativos Windows Server 2012 y CentOS7, dentro del Departamento de Componentes del Centro TLM. Para ello el usuario (administrador) accede al sistema desde su computadora y por medio de un navegador web, una vez dentro, el sistema permite realizar varias funciones mediante conexiones seguras SSH a los servidores; las cuales son:

- Instalar o desinstalar el servicio PostgreSQL, así como controlar el estado (iniciar, reiniciar y detener) del mismo.
- Instalar el extensor para la creación de las BDG PostGIS.
- Gestionar (crear, editar y eliminar) las BDG
- Realizar las configuraciones necesarias para generar y habilitar los certificados SSL.
- Visualizar el consumo de recursos (CPU y Memoria) de la máquina virtual servidora.

habilidad del analista y de su preparación para la entrevista (Anexo 3 Guía para la realización de la entrevista).

- **Desarrollo conjunto de aplicaciones (JAD, siglas en idioma inglés):** Se realiza en un conjunto de reuniones que cuentan con recursos especiales (diagramas, multimedia), trabajando directamente sobre los documentos de requisitos por generar. Además de exigir una alta intervención del analista, se precisa que el interesado conozca elementos técnicos como diagramas, generalmente reservados a los analistas (Zapata, 2007).

2.2.2. Requisitos funcionales

Los requisitos funcionales son aquellos directamente relacionados con las funciones o relaciones que el sistema debe proporcionar (Ramos, 2016). Teniendo en cuenta las necesidades del Departamento de Desarrollo de Componentes fueron definidos 33 requisitos funcionales agrupados en 8 paquetes de funcionalidades, los mismos se muestran a continuación:

Paquete de funcionalidad número 1: Adicionar máquina virtual servidora.

RF1 Adicionar máquina virtual servidora con el sistema operativo CentOS7.

RF2 Adicionar máquina virtual servidora con el sistema operativo Windows Server 2012.

Paquete de funcionalidad número 2: Administración del servicio PostgreSQL.

RF3 Instalar PostgreSQL en el sistema operativo CentOS7.

RF4 Instalar PostgreSQL en el sistema operativo Windows Server 2012.

RF5 Iniciar PostgreSQL en el sistema operativo CentOS7.

RF6 Iniciar PostgreSQL en el sistema operativo Windows Server 2012.

RF7 Detener PostgreSQL en el sistema operativo CentOS7.

RF8 Detener PostgreSQL en el sistema operativo Windows Server 2012.

RF9 Reiniciar PostgreSQL en el sistema operativo CentOS7.

RF10 Reiniciar PostgreSQL en el sistema operativo Windows Server 2012.

RF11 Desinstalar PostgreSQL en el sistema operativo CentOS7.

RF12 Desinstalar PostgreSQL en el sistema operativo Windows Server 2012.

Paquete de funcionalidad número 3: Instalación del extensor PostGIS.

RF13 Instalar PostGIS en el sistema operativo CentOS7.

RF14 Instalar PostGIS en el sistema operativo Windows Server 2012.

RF15 Desinstalar PostGIS en el sistema operativo CentOS7.

Paquete de funcionalidad número 4: Configuración de certificados SSL.

RF16 Generar certificados SSL en el sistema operativo CentOS7.

RF17 Generar certificados SSL en el sistema operativo Windows Server 2012.

RF18 Habilitar certificados SSL en el sistema operativo CentOS7.

RF19 Habilitar certificados SSL en el sistema operativo Windows Server 2012.

Paquete de funcionalidad número 5: Gestión de las BDG.

RF20 Crear base de datos geoespacial en el sistema operativo CentOS7.

RF21 Crear base de datos geoespacial en el sistema operativo Windows Server 2012.

RF22 Editar base de datos geoespacial en el sistema operativo CentOS7.

RF23 Editar base de datos geoespacial en el sistema operativo Windows Server 2012.

RF24 Eliminar base de datos geoespacial en el sistema operativo CentOS7.

RF 25 Eliminar base de datos geoespacial en el sistema operativo Windows Server 2012.

RF26 Listar las BD existentes en el sistema operativo CentOS7.

RF27 Listar las BD existentes en el sistema operativo Windows Server 2012.

Paquete de funcionalidad número 6: Actualizar repositorios.

RF28 Actualizar los repositorios en el sistema operativo CentOS7.

RF29 Actualizar el archivo de los repositorios en el sistema operativo CentOS7.

Paquete de funcionalidad número 7: Cantidad de memoria RAM.

RF30 Mostrar la cantidad de memoria RAM que utiliza el servidor en el sistema operativo CentOS7.

RF31 Mostrar la cantidad de memoria RAM que utiliza el servidor en el sistema operativo Windows Server 2012.

Paquete de funcionalidad número 8: Cantidad de CPU.

RF32 Mostrar la cantidad de CPU que utiliza el servidor de bases de datos en el sistema operativo CentOS7.

RF33 Mostrar la cantidad de CPU que utiliza el servidor de bases de datos en el sistema operativo Windows Server 2012.

2.2.3. Requisitos no funcionales

Los requisitos no funcionales son restricciones impuestas al funcionamiento del sistema, tales como las limitaciones de tiempo, presupuesto, proceso de desarrollo, las políticas de la organización y normas que deben adoptarse (Ramos, 2016). Se puede decir que los requisitos no funcionales se refieren a cualidades que imponen restricciones en el diseño y la implementación del sistema. A continuación, se muestran los requisitos no funcionales con los que cuenta el sistema.

Interoperabilidad:

- La máquina en que se ejecute el sistema debe constar con un navegador web (Chrome 56 o superior, Mozilla Firefox. 55 o superior, Internet Explorer 10 o superior).

Usabilidad:

- El sistema debe visualizar todos los mensajes en idioma español.
- Las funcionalidades principales del sistema estarán orientadas a íconos para un mayor reconocimiento por parte del usuario.
- Las máquinas a la que se desee acceder desde el sistema tienen que tener configurado el protocolo SSH.
- Las máquinas virtuales que contengan servidores PostgreSQL instalados en el sistema operativo CentOS7 tienen que tener configurados repositorios que contengan PostgreSQL y PostGIS.
- Las máquinas virtuales que contengan servidores PostgreSQL instalados en el sistema operativo Windows Server 2012 la conexión SSH debe realizarse con el usuario de tipo administrador.

Hardware:

- Se requiere para los servidores las siguientes condiciones mínimas:
 - Procesador: 1.5Ghz.
 - Memoria RAM: 2GB.
- Se requiere para las estaciones de trabajo cliente las siguientes condiciones mínimas:

- Poseer tarjeta de red.
- Memoria RAM: 1GB.

2.3. Validación de requisitos

- **Acta de conformidad del cliente:** En esta técnica se recogen los requisitos funcionales detectados, luego de ser revisada y firmada por el cliente se da fin a este ciclo (Anexo 1 Acta de conformidad de los requisitos por el cliente).

2.4. Especificación de los requisitos funcionales

Para la especificación de los requisitos funcionales se utilizaron las HU, como se hizo referencia en el capítulo anterior. Las mismas se caracterizan por ser tarjetas en las cuales el cliente describe brevemente las características que el sistema debe poseer. A continuación, se muestran las HU que le corresponden al paquete “Gestionar_Base_de_Datos_Geospacial”. El resto de las HU se pueden visualizar en el **¡Error! No se encuentra el origen de la referencia..**

Tabla 2 HU N° 20: Crear base de datos geoespacial en el sistema operativo CentOS7.

Número: 20	Nombre del requisito: Crear base de datos geoespacial en el sistema operativo CentOS7.
Programador: Sunaidy C. Jauregui Fajardo	Iteración Asignada: 1
Prioridad: Media	Tiempo Estimado: 3 días
Riesgo en Desarrollo: <ul style="list-style-type: none"> • Planificación irreal. 	Tiempo Real: 3 días
<p>Descripción: Funcionalidad que permite crear una base de datos geoespacial en el sistema operativo CentOS7. Se debe mostrar un mensaje de confirmación en cualquier caso (ejemplo: “La base de datos se ha creado con éxito” o “El nombre de la base de datos no es válido”).</p> <p>Campos:</p> <ul style="list-style-type: none"> • Nombre: Campo de texto que permite nombrar la base de datos geoespacial (solo puede contener letras). <p>Botones:</p> <ul style="list-style-type: none"> • Cancelar: Opción que permite regresar a la interfaz principal, descartando todos los cambios realizados. 	

- Aceptar: Opción que valida que los datos de la interfaz sean correctos.

Observaciones: El usuario debe estar autenticado en el sistema. Tienen que estar instalados PostgreSQL y PostGIS.

Prototipo de interfaz:

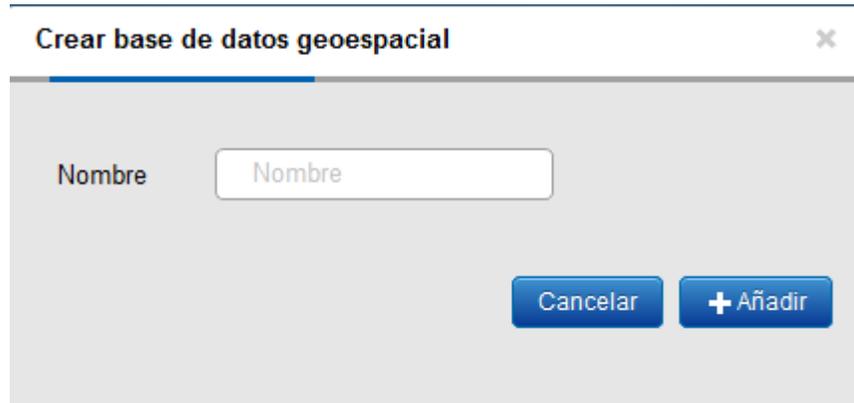


Tabla 3 HU N° 21: Crear base de datos geoespacial en el sistema operativo Windows Server 2012.

Número: 21	Nombre del requisito: Crear base de datos geoespacial en el sistema operativo Windows Server 2012.
Programador: Sunaidy C. Jauregui Fajardo	Iteración Asignada: 1
Prioridad: Media	Tiempo Estimado: 3 días
Riesgo en Desarrollo: <ul style="list-style-type: none"> • Planificación irreal. 	Tiempo Real: 3 días
<p>Descripción: Funcionalidad que permite crear una base de datos geoespacial en el sistema operativo Windows Server 2012. Se debe mostrar un mensaje de confirmación en cualquier caso (ejemplo: “La base de datos se ha creado con éxito” o “El nombre de la base de datos no es válido”).</p> <p>Campos:</p> <ul style="list-style-type: none"> • Nombre: Campo de texto que permite nombrar la base de datos geoespacial (solo puede contener letras). <p>Botones:</p> <ul style="list-style-type: none"> • Cancelar: Opción que permite regresar a la interfaz principal, descartando todos los cambios realizados. • Aceptar: Opción que valida que los datos de la interfaz sean correctos. 	

Observaciones: El usuario debe estar autenticado en el sistema. Tienen que estar instalados PostgreSQL y PostGIS.

Prototipo de interfaz:

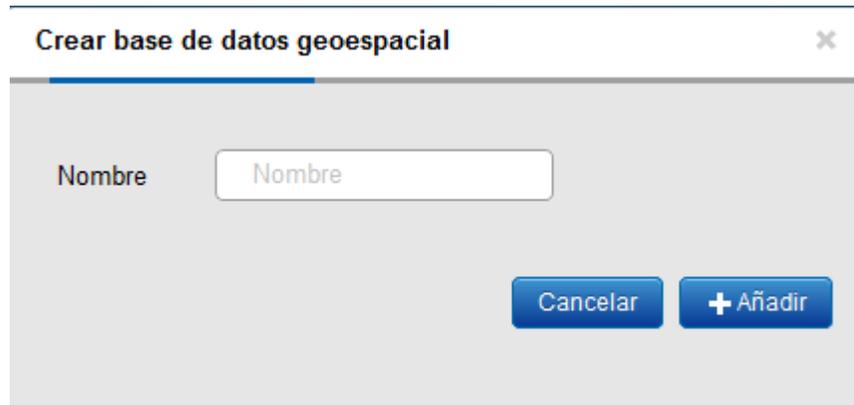


Tabla 4 HU N° 22: Editar base de datos geoespacial en el sistema operativo CentOS7.

Número: 22	Nombre del requisito: Editar base de datos geoespacial en el sistema operativo CentOS7.
Programador: Sunaidy C. Jauregui Fajardo	Iteración Asignada: 1
Prioridad: Media	Tiempo Estimado: 3 días
Riesgo en Desarrollo: <ul style="list-style-type: none"> Planificación irreal. 	Tiempo Real: 3 días
<p>Descripción: Funcionalidad que permite editar una base de datos geoespacial en el sistema operativo CentOS7.</p> <p>Campos:</p> <ul style="list-style-type: none"> Nombre: Campo de texto que permite renombrar la base de datos geoespacial seleccionada (solo puede contener letras). <p>Botones:</p> <ul style="list-style-type: none"> Cancelar: Opción que permite regresar a la interfaz principal, descartando todos los cambios realizados. Renombrar: Opción que valida que los datos de la interfaz sean correctos. 	
Observaciones: El usuario debe estar autenticado en el sistema. Tienen que estar instalados PostgreSQL y PostGIS.	

Prototipo de interfaz:

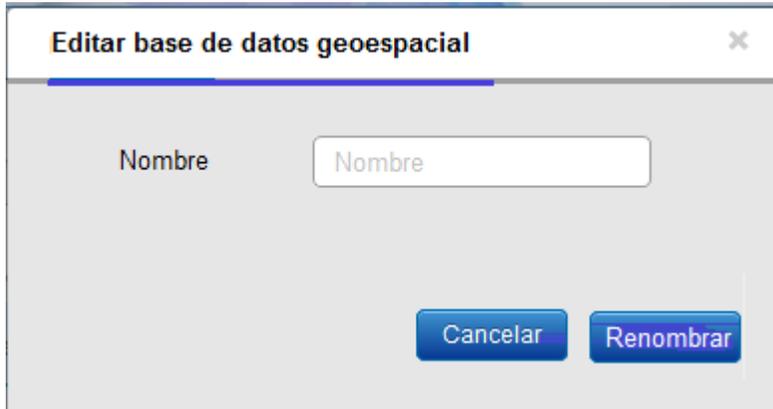


Tabla 5 HU N° 23: Editar base de datos geoespacial en el sistema operativo Windows Server 2012.

Número: 23	Nombre del requisito: Editar base de datos geoespacial en el sistema operativo Windows Server 2012.
Programador: Sunaidy C. Jauregui Fajardo	Iteración Asignada: 1
Prioridad: Media	Tiempo Estimado: 3 días
Riesgo en Desarrollo: <ul style="list-style-type: none"> Planificación irreal. 	Tiempo Real: 3 días
<p>Descripción: Funcionalidad que permite editar una base de datos geoespacial en el sistema operativo Windows Server 2012.</p> <p>Campos:</p> <ul style="list-style-type: none"> Nombre: Campo de texto que permite renombrar la base de datos geoespacial seleccionada (solo puede contener letras). <p>Botones:</p> <ul style="list-style-type: none"> Cancelar: Opción que permite regresar a la interfaz principal, descartando todos los cambios realizados. Renombrar: Opción que valida que los datos de la interfaz sean correctos. 	
<p>Observaciones: El usuario debe estar autenticado en el sistema. Tienen que estar instalados PostgreSQL y PostGIS.</p>	

Prototipo de interfaz:

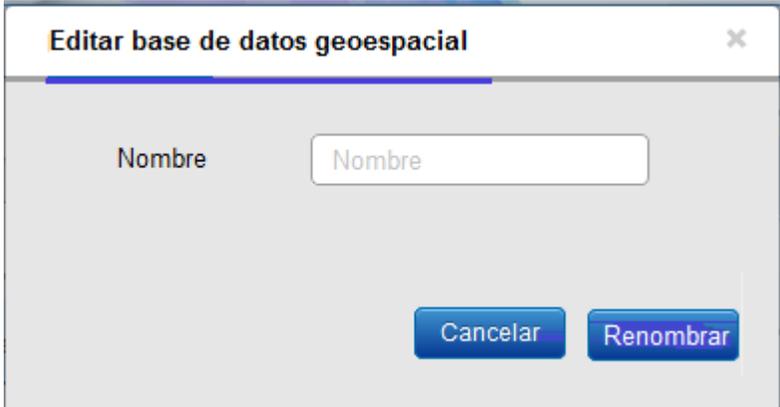


Tabla 6 HU N° 24: Eliminar base de datos geoespacial en el sistema operativo CentOS7.

Número: 24	Nombre del requisito: Eliminar base de datos geoespacial en el sistema operativo CentOS7.
Programador: Sunaidy C. Jauregui Fajardo	Iteración Asignada: 1
Prioridad: Media	Tiempo Estimado: 3 días
Riesgo en Desarrollo: <ul style="list-style-type: none"> Planificación irreal. 	Tiempo Real: 3 días
<p>Descripción: Funcionalidad que permite eliminar una base de datos geoespacial en el sistema operativo CentOS7. Se debe mostrar un mensaje de confirmación (La base de datos ha sido eliminada con éxito o La base de datos no se ha podido eliminar).</p> <p>Botones:</p> <ul style="list-style-type: none"> Eliminar: Opción que elimina la base de datos seleccionada. 	
<p>Observaciones: El usuario debe estar autenticado en el sistema. Tienen que estar instalados PostgreSQL y PostGIS. Debe haber por lo menos una base de datos en el servidor.</p>	
<p>Prototipo de interfaz:</p>	

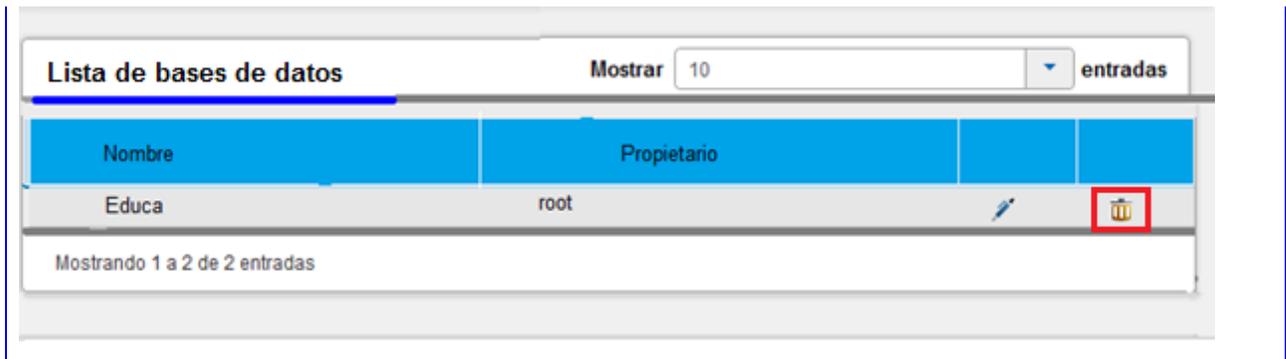


Tabla 7 HU N° 25: Eliminar base de datos geoespacial en el sistema operativo Windows Server 2012.

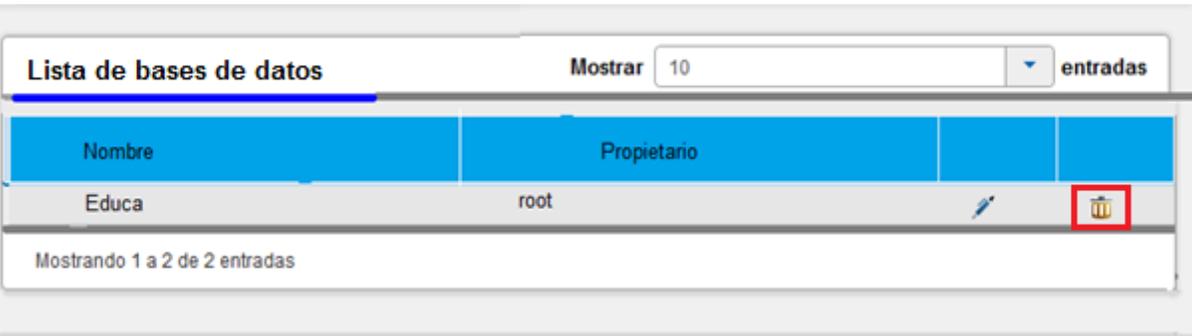
Número: 25	Nombre del requisito: Eliminar base de datos geoespacial en el sistema operativo Windows Server 2012.
Programador: Sunaidy C. Jauregui Fajardo	Iteración Asignada: 1
Prioridad: Media	Tiempo Estimado: 3 días
Riesgo en Desarrollo: <ul style="list-style-type: none"> Planificación irreal. 	Tiempo Real: 3 días
<p>Descripción: Funcionalidad que permite eliminar una base de datos geoespacial en el sistema operativo Windows Server 2012. Se debe mostrar un mensaje de confirmación (La base de datos ha sido eliminada con éxito o La base de datos no se ha podido eliminar).</p> <p>Botones:</p> <ul style="list-style-type: none"> Eliminar: Opción que elimina la base de datos seleccionada. 	
<p>Observaciones: El usuario debe estar autenticado en el sistema. Tienen que estar instalados PostgreSQL y PostGIS. Debe haber por lo menos una base de datos en el servidor.</p>	
<p>Prototipo de interfaz:</p> 	

Tabla 8 HU N° 26: Listar las BD existentes en el sistema operativo CentOS7.

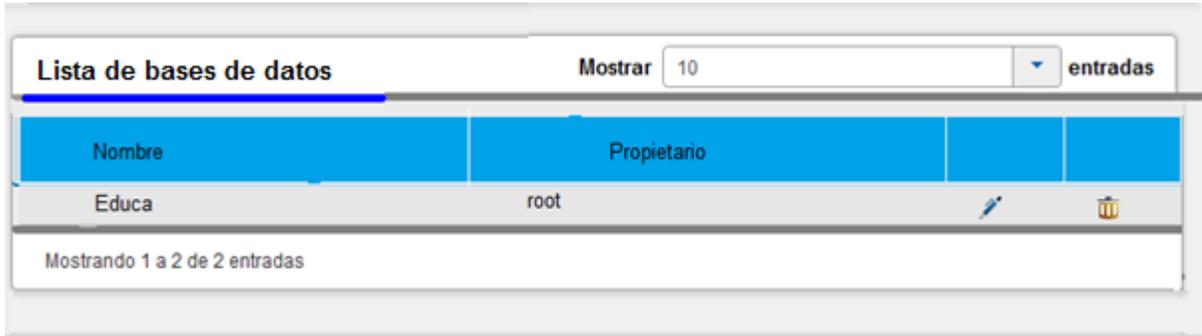
Número: 26	Nombre del requisito: Listar las BD existentes en el sistema operativo CentOS7.
Programador: Sunaidy C. Jauregui Fajardo	Iteración Asignada: 1
Prioridad: Media	Tiempo Estimado: 3 días
Riesgo en Desarrollo: Planificación irreal.	Tiempo Real: 3 días
Descripción: Funcionalidad que permite listar las BD existentes en el sistema operativo CentOS7. En caso de no existir ninguna base de datos se muestra una tabla vacía.	
Observaciones: El usuario debe estar autenticado en el sistema. Tienen que estar instalados PostgreSQL y PostGIS.	
Prototipo de interfaz:	
	

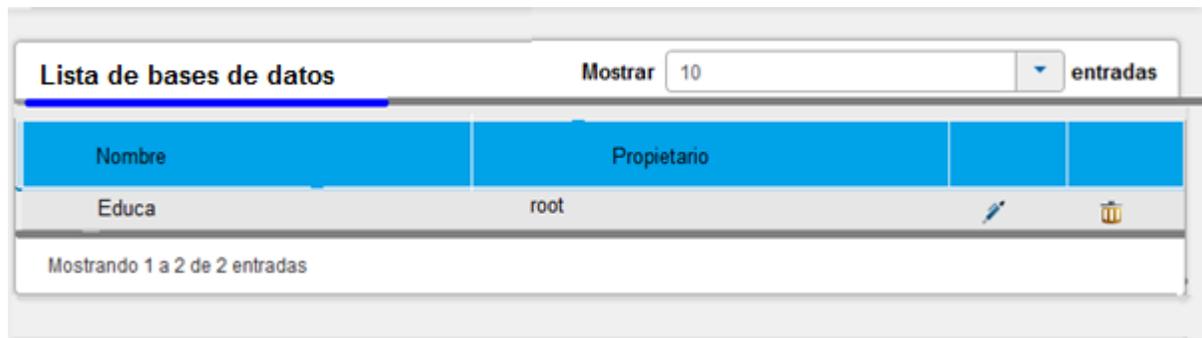
Tabla 9 HU N° 27: Listar las BD existentes en el sistema operativo Windows Server 2012.

Número: 27	Nombre del requisito: Listar las BD existentes en el sistema operativo Windows Server 2012.
Programador: Sunaidy C. Jauregui Fajardo	Iteración Asignada: 1
Prioridad: Media	Tiempo Estimado: 3 días
Riesgo en Desarrollo: Planificación irreal.	Tiempo Real: 3 días

Descripción: Funcionalidad que permite listar las BD existentes en el sistema operativo Windows Server 2012. En caso de no existir ninguna base de datos se muestra una tabla vacía.

Observaciones: El usuario debe estar autenticado en el sistema. Tienen que estar instalados PostgreSQL y PostGIS.

Prototipo de interfaz:



2.5. Especificación de los requisitos no funcionales

La metodología AUP para la UCI propone que los requisitos no funcionales sean especificados y agrupados por atributo de calidad. La propuesta de solución cuenta con un total de 8 requisitos no funcionales agrupados en 2 atributos de calidad mostrados a continuación.

Tabla 10: Especificación RNF Interoperabilidad

Atributo de Calidad	Interoperabilidad
Sub-atributos/Sub-características	Cumplimiento de interoperabilidad: La máquina en que se ejecute el sistema debe constar con un navegador web (Chrome 56 o superior, Mozilla Firefox. 55 o superior, Internet Explorer 10 o superior).
Objetivo	Lograr la ejecución del sistema en cualquier navegador web.
Origen	Administrador
Artefacto	El sistema
Entorno	El sistema está funcionando correctamente.

Estímulo	Respuesta: Flujo de eventos
N° 1. A	
N/A	N/A
Medida de respuesta	
El sistema funciona en cualquier navegador web.	

Tabla 11: Especificación RNF Usabilidad

Atributo de Calidad	Usabilidad
Sub-atributos/Sub-características	<p>Cumplimiento de Usabilidad:</p> <p>Visualizar todos los mensajes en idioma español.</p> <p>Las funcionalidades principales del sistema deben estar orientadas a íconos.</p> <p>Las máquinas a la que se desee acceder desde el sistema tienen que tener configurado el protocolo SSH.</p> <p>Las máquinas virtuales que contengan servidores PostgreSQL instalados en el sistema operativo CentOS7 tienen que tener configurados los repositorios UCI.</p> <p>Las máquinas virtuales que contengan servidores PostgreSQL instalados en el sistema operativo Windows Server 2012 la conexión SSH debe realizarse con el usuario administrador.</p>
Objetivo	Lograr que el sistema sea entendible para las personas de habla hispana.
Origen	Administrador
Artefacto	El sistema
Entorno	El sistema está correctamente.
Estímulo	Respuesta: Flujo de eventos
N° 2. A	
Seleccionar la opción instalar PostgreSQL.	Se muestra un mensaje de notificación (en idioma español) en cualquier caso (correcto o incorrecto).
N° 2. B	
Seleccionar el icono del cesto de basura cuando se desee eliminar una base de datos o una máquina virtual servidora.	Se muestra un mensaje de notificación (en idioma español) en cualquier caso (correcto o incorrecto).

Medida de respuesta

Utilización de las características señaladas para la ejecución del sistema.

2.6. Arquitectura del sistema

El estándar 1471-2000 del Instituto de Ingeniería Eléctrica y Electrónica (IEEE, según sus siglas en inglés) describe que: “la arquitectura del software es la organización fundamental de un sistema formada por sus componentes, las relaciones entre ellos y el contexto en el que se implantarán, y los principios que orientan su diseño y evolución”. Un mismo sistema puede utilizar varias arquitecturas, en este caso por las características del sistema es necesario utilizar las siguientes:

Modelo Cliente-Servidor

El modelo Cliente/Servidor 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. Las aplicaciones Clientes realizan peticiones a una o varias aplicaciones Servidores, que deben encontrarse en ejecución para atender dichas demandas. En este modelo intervienen únicamente dos entidades: El Cliente y El Servidor (Marini, 2012).

- El papel del cliente lo desempeña la aplicación final del usuario, que implementará todas las funciones correspondientes a la lógica de presentación, más algunas de las funciones relacionadas con la lógica del negocio, como pueden ser determinadas validaciones de datos y condiciones de recuperación.
- El papel del servidor lo desempeña el propio SGBD, el cual se ocupará de todas las funciones correspondientes a la lógica de datos, más las restantes funciones correspondientes a la lógica del negocio.

Modelo Vista Plantilla (MTV)

El marco de trabajo Django, utilizado para la realización de la solución, implementa el patrón arquitectónico MTV. La misma es una implementación especial del Modelo-Vista-Controlador. A continuación, se muestra una imagen donde se puede visualizar la variación entre los patrones.

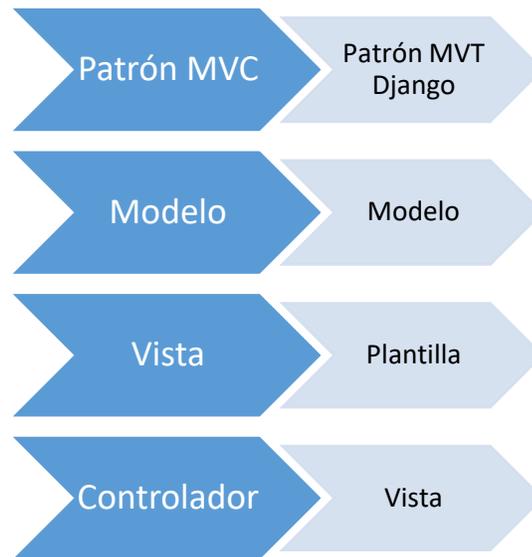


Figura 2: MVT variante del MVC

Esta arquitectura permite una separación entre interfaz, lógica del negocio y de presentación, lo que permite una gran reutilización de componentes, así como facilidad para desarrollar prototipos y realizar pruebas unitarias de los componentes. La siguiente imagen muestra con más detalles como funciona.

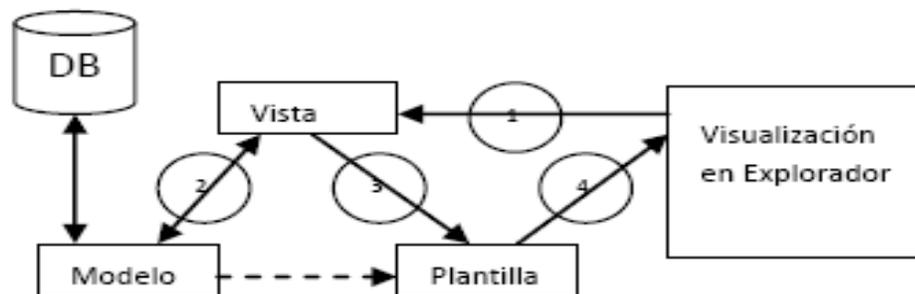


Figura 3: Funcionamiento de la arquitectura MVT (Condori, 2012).

Leyenda:

- 1- El navegador manda una solicitud.
- 2- La vista interactúa con el modelo para obtener los datos.
- 3- La vista llama a la plantilla.
- 4- La plantilla renderiza la respuesta a la solicitud del navegador.

2.7. Patrones de diseño

Son principios generales de soluciones que emplean ciertos estilos que ayudan a la creación de software. Brindan una solución ya probada y documentada a problemas de desarrollo de software que están sujetos a contextos similares (Craig, 1999).

Patrones GRASP⁸:

Los patrones GRASP describen los principios fundamentales para la asignación de responsabilidades a objetos, expresado en forma de patrones (Craig, 1999). A continuación, son descritos los de este tipo que son utilizados para el diseño de la solución propuesta.

Experto: Es un patrón que permite asignar la responsabilidad de realizar una labor a la clase que contiene toda la información necesaria para cumplir la tarea encomendada. En el caso de la presente investigación se tiene la clase “AdministracionPostgreSQL” que es experta en realizar las configuraciones de PostgreSQL ya que contiene toda la información necesaria para realizar esa tarea.

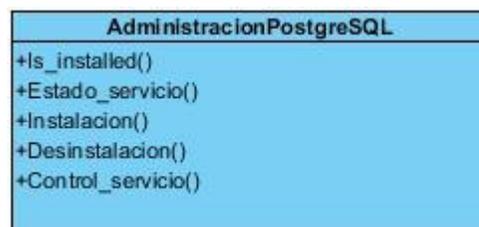


Figura 4: Clase AdministracionPostgreSQL

Creador: Es un patrón que tiene responsabilidades relacionadas con la creación de objetos. El propósito fundamental del mismo, es encontrar un creador que se debe conectar con el objeto producido en cualquier evento. Uno de los ejemplos del patrón en la aplicación se visualiza a continuación, en la clase “BDG” la cual es la encargada de crear objetos de tipo “BDG”.

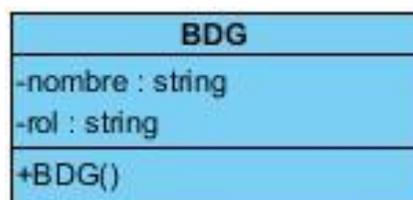


Figura 5: Clase BDG

⁸ Patrones Generales de Software para Asignar Responsabilidades.

Bajo acoplamiento: El acoplamiento es una medida de la fuerza con que una clase está conectada a otras. Acoplamiento bajo significa que una clase no depende de muchas otras. El uso de este patrón permite que las clases no se afecten por cambios de otros componentes, haciendo posible que sean fáciles de entender y de reutilizar. En el caso de la presente investigación se cumple dicho patrón porque las clases tienen solo dos dependencias y esto proporciona que se reduzca el impacto de cambios posteriores en el sistema.

Alta cohesión: La cohesión es una medida de cuán relacionadas y enfocadas están las responsabilidades de una clase. Una alta cohesión caracteriza a las clases con responsabilidades estrechamente relacionadas que no realicen un trabajo enorme. Se hizo necesario la utilización de este patrón en el sistema en cuestión con el fin de controlar la complejidad de cada clase utilizada para mantener un buen comportamiento de las mismas, por esta razón, las que se identificaron con una amplia cantidad de funcionalidades, se dividieron en otras clases siguiendo el propósito de distribuir de forma equitativa el peso de la complejidad, manteniendo, además, la coherencia entre ellas.

2.8. Modelo de diseño

El modelo de diseño expande y detalla los modelos de análisis tomando en cuenta todas las implicaciones y restricciones técnicas. El propósito es especificar una solución que trabaje y pueda ser fácilmente convertida en código fuente durante la fase de implementación (Pressman, 1998). El modelo de diseño se realizó haciendo uso de la herramienta Visual Paradigm donde se definió una estructura de paquetes para la organización del trabajo mostrada a continuación.

2.8.1. Diagrama de clases de diseño

Los diagramas de clases son diagramas de estructura estática que muestran las clases del sistema y sus interrelaciones. Los diagramas de clase son el pilar básico del modelado con UML, siendo utilizados tanto para mostrar lo que el sistema puede hacer (análisis), como para mostrar cómo puede ser construido (diseño) (...) (Craig, 1999). En estos se muestran las clases que componen un sistema y cómo se relacionan entre sí. A continuación, se muestran el diagrama de clases de diseño correspondiente con el paquete “Gestionar_Base_de_Datos_Geospacial”, el resto de los diagramas pueden encontrarse en el **¡Error! No se encuentra el origen de la referencia.**

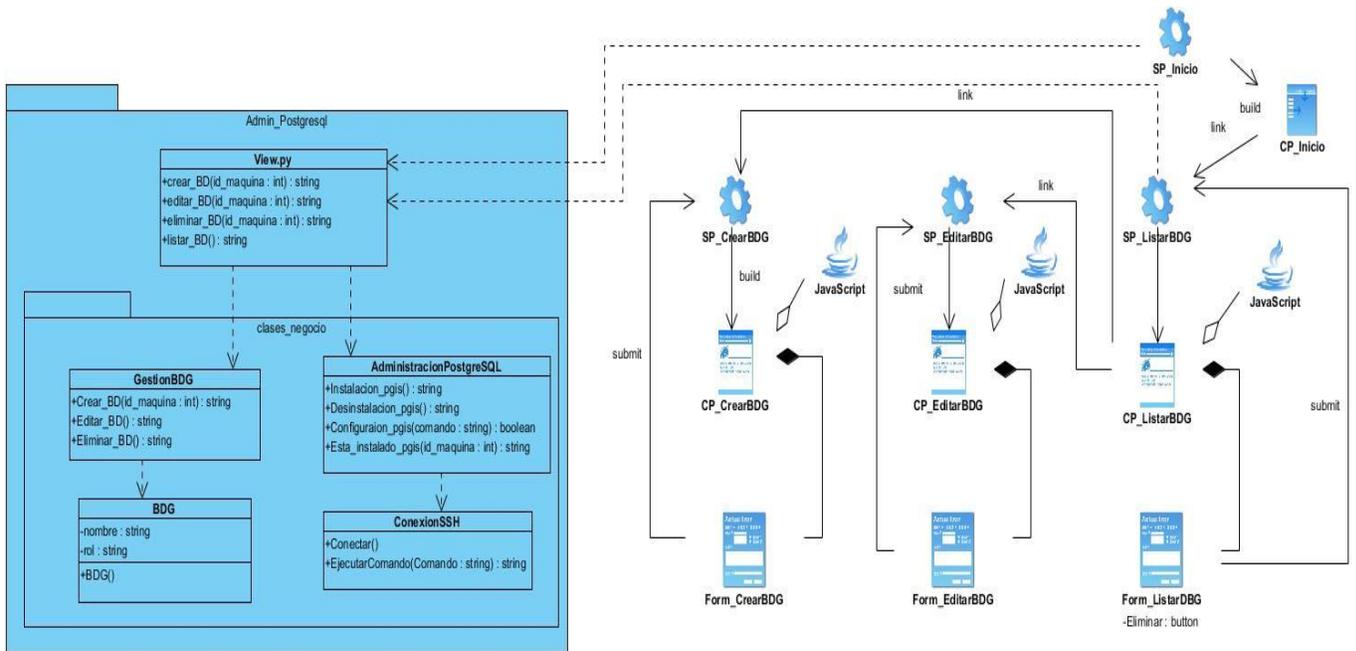


Figura 6: Diagrama de clase de diseño del paquete Gestionar_Base de_Datos_Geospacial

Descripción de las Clases:

Client page (CP): Representan páginas web encargadas de mostrar los formularios (Form) e información al usuario.

Form: Representa una colección de campos de entrada de datos, es parte de una Client Page.

Server page (SP): Representan clases encargadas de generar las Client Page.

View.py: Representa la lógica del proyecto.

AdministracionPostgreSQL: Es la clase encargada de realizar todas las funcionalidades relacionadas con el servicio PostgreSQL.

ConexionSSH: Es la clase por la cual se comunica la aplicación con el protocolo SSH configurado en la máquina virtual servidora.

GestionBDG: Es la clase encargada de gestionar las BDG.

BDG: Es la clase encargada de generar los objetos de tipo “base de datos geoespacial”.

2.9. Diagrama de secuencia

Para una acción determinada en el sistema este diagrama representa la interacción entre diversos objetos a través de las llamadas ejecutadas y de retorno, lo que permite ver la secuencia de llamadas a través del tiempo

(Ramos, 2016). Con dichos diagramas se muestran las interacciones entre las clases mediante la transferencia de mensajes. A continuación, se muestran los diagramas de secuencia correspondientes con el paquete Gestionar_Base_de_Datos_Geospacial (**¡Error! No se encuentra el origen de la referencia.**).

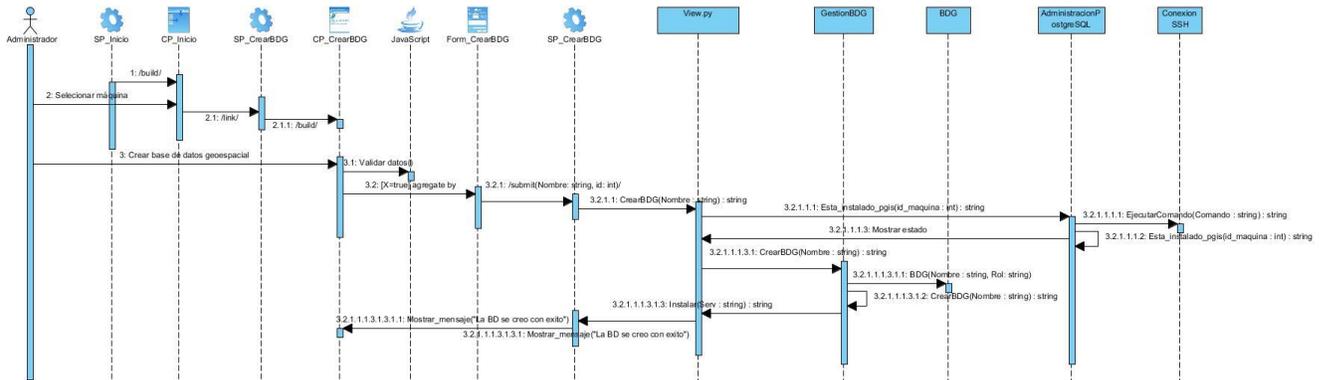


Figura 7: Diagrama de secuencia del paquete Gestionar_Base_de_Datos_Geospacial (Escenario crear)

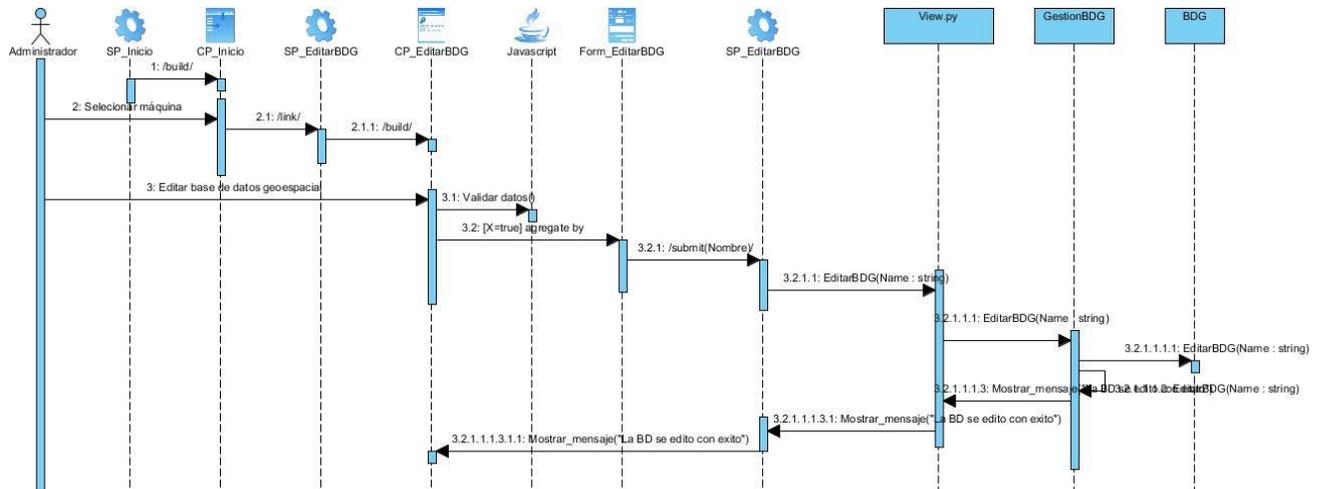


Figura 8: Diagrama de secuencia del paquete Gestionar_Base_de_Datos_Geospacial (Escenario editar)

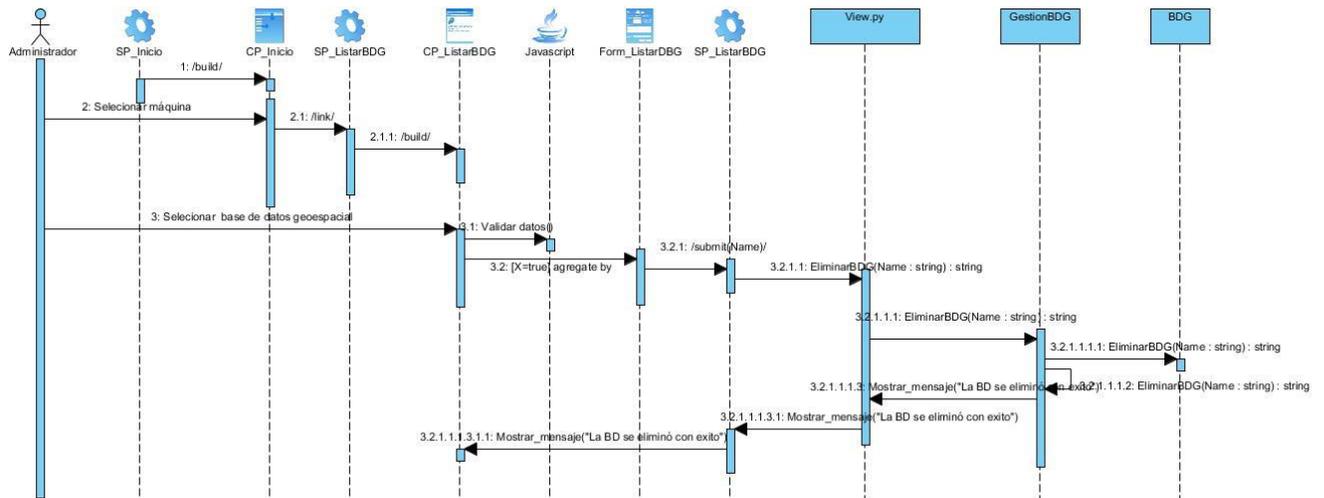


Figura 9: Diagrama de secuencia del paquete Gestionar_Base_de_Datos_Geospatial (Escenario eliminar)

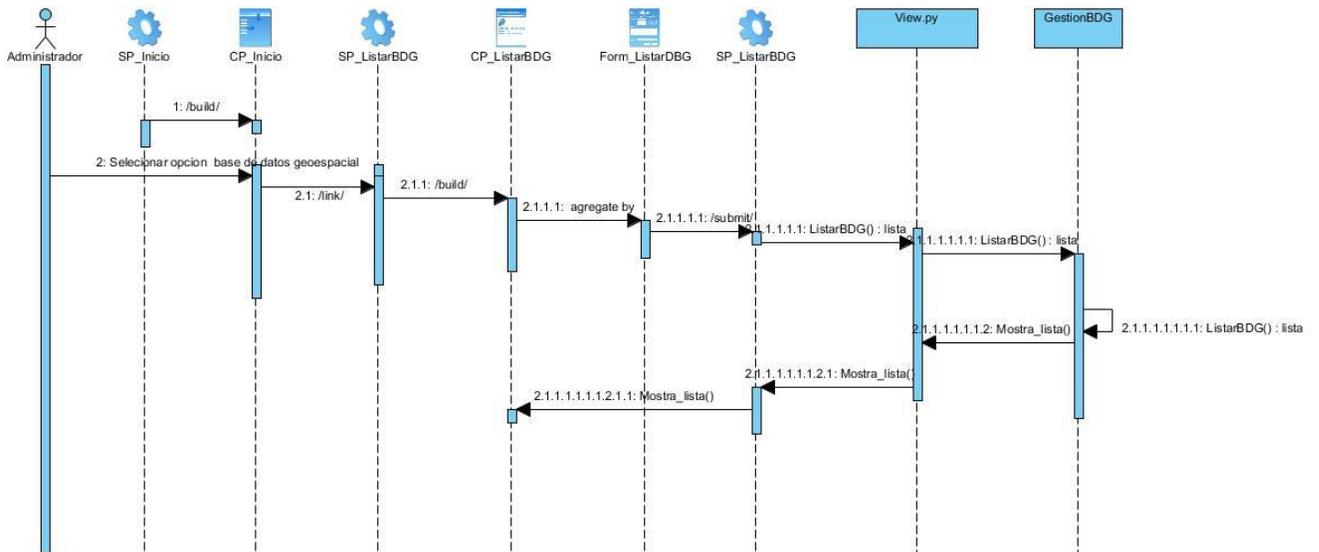


Figura 10: Diagrama de secuencia del paquete Gestionar_Base_de_Datos_Geospatial (Escenario listar)

2.10. Conclusiones del capítulo

La identificación de los requisitos permitió el posterior diseño de la propuesta de solución enfatizando en las características funcionales que este debe cumplir. Se fundamentaron los patrones de diseño empleados, estos permitieron especificar la estructura y comportamiento de las clases, presentando una solución que atiende los niveles de reutilización y mantenimiento. Por consiguiente, una vez precisadas las funcionalidades y la estructura de la solución es posible dar inicio a la fase de implementación.

Capítulo III. Implementación y Prueba

El presente capítulo muestra los artefactos ingenieriles relacionados con la implementación y validación del sistema a desarrollar. Entre los principales elementos se encuentran el modelo de implementación con los respectivos artefactos generados a partir de ellos. Además, se detallan las pruebas realizadas al sistema.

3.1. Modelo de implementación

Tiene como objetivo implementar cada una de las clases significativas del diseño.

Estándares de codificación

Los estándares de codificación se definen por el equipo de desarrollo para lograr la estandarización en la programación del software. Se basan en la estructura y apariencia física de un programa, con el fin de facilitar la lectura, comprensión, mantenimiento del código, reutilización a lo largo del proceso de desarrollo de un software y no en la lógica del programa. Con el fin de facilitar el entendimiento del código y fijar un modelo a seguir, se establecieron los siguientes estándares de codificación:

CapWords: Para nombrar las clases, definiendo que las mismas deben comenzar con letra mayúscula, y en caso de estar compuesta por más de una palabra, todas deben iniciar con mayúscula. A continuación, se muestra una imagen que demuestra la utilización del estándar en el sistema.

```
class BDG: ...
class GestionBDG: ...
class AdministracionPostgreSQL: ...
```

Figura 11: Ejemplo de uso del estándar CapWords

lower_case_with_underscores: Para nombrar los métodos del programa principal, definiendo que las palabras son escritas en minúsculas separadas por guion bajo. A continuación, se muestra una imagen que demuestra la utilización del estándar en el sistema.

```
def crear_bdg(request):
    if request.is_ajax and request.method == 'GET':
        id_maquina = request.GET['id_maquina']
        maq = direcciones_ip_registradas.objects.get(pk=id_maquina)
        return render(request, 'modal_agregar_dbg.html', {'servidor': maq})
```

Figura 12: Ejemplo de uso del estándar lower_case_with_underscores

3.2. Diagrama de despliegue

Un diagrama de despliegue es utilizado para representar la distribución física de un sistema. El mismo muestra la configuración y la comunicación entre las instancias de los componentes y objetos que residen en ellos. El diagrama de despliegue correspondiente con el sistema AD&MONPSQL 2.0 es el siguiente:

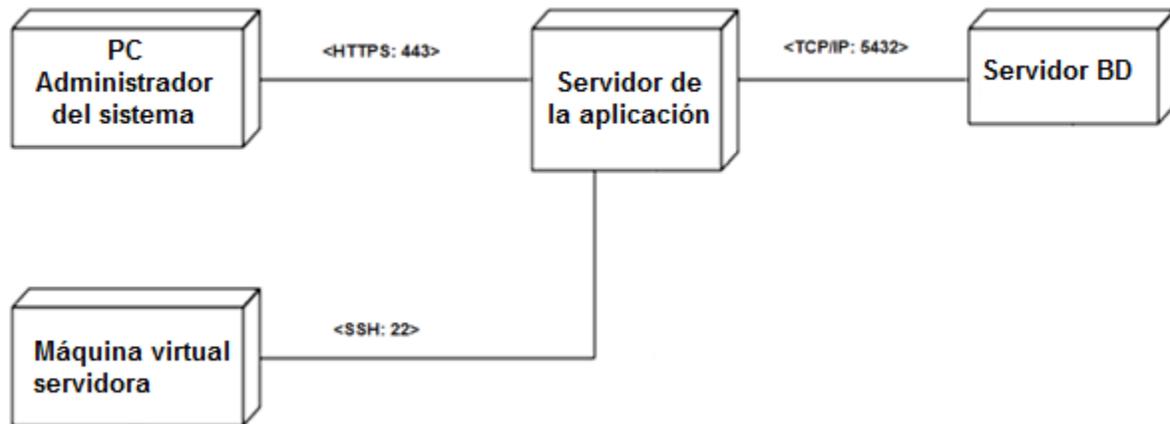


Figura 13: Diagrama de despliegue

Para la distribución física del sistema se utilizó un nodo “PC_Administrador del sistema” que representa la computadora del usuario. La PC_Administrador del sistema se conecta por el protocolo seguro de transferencia de hipertexto (https por sus siglas en inglés) al nodo “Servidor de la aplicación” web. Con el nodo del Servidor de la aplicación se puede acceder a la “Máquina servidora” con el protocolo SSH y con el nodo “Servidor de BD” mediante el protocolo diseñado para facilitar la reutilización de código de base de datos (TCP/IP).

3.3. Pruebas del software

Las pruebas del software constituyen una etapa imprescindible durante el proceso de desarrollo del software. Su objetivo principal es asegurar que el software cumpla con las especificaciones requeridas y eliminar los posibles defectos que este pudiera tener. Es importante considerar que las pruebas de software no garantizan que un sistema esté libre de errores, sino que se detecten la mayor cantidad de defectos posibles para su debida corrección.

Estrategia de prueba

Trazar una estrategia de prueba es primordial para realizar la ejecución de las pruebas de un software. Mediante esta quedan plasmado los niveles de prueba a tratar, los tipos de pruebas que se deben llevar a cabo por cada nivel, los métodos de pruebas a aplicar y las técnicas a utilizar por el método.

Una estrategia de prueba de software proporciona una guía que describe los pasos que deben realizarse como parte de la prueba, cuándo se planean y se llevan a cabo dichos pasos, y cuánto esfuerzo, tiempo y recursos se requieran. Por tanto, cualquier estrategia de prueba debe incorporar la planificación de la prueba, el diseño de casos de prueba, la ejecución de la prueba y la recolección y evaluación de los resultados. Una estrategia de prueba de software debe ser suficientemente flexible para promover un uso personalizado de la prueba. Al mismo tiempo, debe ser suficientemente rígida para alentar la planificación razonable y el seguimiento de la gestión conforme avanza el proyecto (Pressman, 2010).

3.4. Disciplinas de pruebas

La metodología AUP para la UCI plantea varias disciplinas para guiar el desarrollo de las pruebas, a continuación, se describen las disciplinas presentes en la investigación:

Pruebas internas: En esta disciplina se verifica el resultado de la implementación probando cada construcción, incluyendo tanto las construcciones internas como intermedias, así como las versiones finales a ser liberadas. Se deben desarrollar artefactos de prueba como: diseños de casos de prueba, listas de chequeo y de ser posible componentes de prueba ejecutables para automatizar las pruebas (Sánchez, 2015).

Pruebas de Aceptación: Es la prueba final antes del despliegue del sistema. Su objetivo es verificar que el software está listo y que puede ser usado por usuarios finales para ejecutar aquellas funciones y tareas para las cuales el software fue construido (Sánchez, 2015).

3.5. Niveles de Prueba

A continuación, se puntualizan los niveles de pruebas que corresponden a las disciplinas de pruebas descritas anteriormente. Dichos niveles propiciaron la detección de los errores existentes.

- **Unitarias:** Las pruebas de unidad se concentran en la lógica del procesamiento interno y en las estructuras de datos de los límites de un componente. El objetivo de las pruebas unitarias es aislar cada parte del programa y mostrar que las partes individuales son correctas (Jacobson, Grady). Se puede decir que consiste en aislar cada parte del programa (clases, módulos, objetos, paquetes, subsistemas) para comprobar que cada una de esas partes funciona correctamente por separado.
- **Aceptación:** En este nivel se evidencian las pruebas realizadas por el usuario en un entorno muy similar al de producción para demostrar que el sistema cumple las especificaciones funcionales y requisitos del cliente. Son las últimas pruebas realizadas donde el cliente prueba el software y verifica que cumpla con sus expectativas, estas son fundamentales por lo cual deben incluirse obligatoriamente en el plan de pruebas de software (Jacobson, 2014).

- **Sistema:** Tienen como propósito fundamental ejercitar profundamente el sistema desarrollado, con el objetivo de verificar que se hayan integrado correctamente todos los elementos del sistema y que realizan correctamente las funciones descritas. Este tipo de pruebas estudia el producto completo para analizar defectos globales o para estudiar aspectos específicos de su comportamiento, tales como seguridad o rendimiento (Jacobson, 2014).

3.6. Método de prueba

La metodología propone 2 métodos fundamentales que serán usados en el proceso de desarrollo de las pruebas al sistema, los cuales serán puntualizados a continuación.

Pruebas de caja negra:

Las pruebas de cajas negras o funcionales son realizadas a la interfaz del software y para aplicarlas solo se necesita conocer la funcionalidad del mismo. Estas pruebas se centran en los requisitos funcionales, permitiendo derivar conjuntos de condiciones de entradas que ejercitarán por completo dichos requerimientos.

Para confeccionar los casos de prueba existen distintos criterios:

Técnica de la Partición de Equivalencia: Esta técnica divide el campo de entrada en clases de datos que tienden a ejercitar determinadas funciones del software.

Técnica del Análisis de Valores Límites: Esta técnica prueba la habilidad del programa para gestionar datos que se encuentran en los límites aceptables.

Técnica de Grafos de Causa-Efecto: Es una técnica que permite al encargado de la prueba validar complejos conjuntos de acciones y condiciones.

Para la realización de los casos de prueba del sistema propuesto se selecciona el criterio de Partición de Equivalencia pues permite examinar los valores válidos e inválidos de las entradas existentes en el software.

Prueba de caja blanca:

Estas pruebas, también suelen ser llamadas estructurales o de cobertura lógica. En ellas se pretende investigar sobre la estructura interna del código, exceptuando detalles referidos a datos de entrada o salida. Realizan un seguimiento del código fuente según se va ejecutando los casos de prueba, determinándose de manera concreta las instrucciones y/o bloques; que han sido ejecutados. A continuación, se describen las técnicas utilizadas para la realización del método.

- **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).

- **Ruta o trayectoria básica:** Permite al diseñador de casos de prueba derivar una medida de complejidad lógica de un diseño de procedimiento y usar esta medida como guía para definir un conjunto básico de rutas de ejecución. Los casos de prueba derivados para revisar el conjunto básico tienen garantía para ejecutar todo enunciado en el programa, al menos una vez durante la prueba (Pressman, 2010).

3.7. Tipos de pruebas

Existen diferentes tipos de pruebas que se pueden aplicar para verificar que el sistema cumple con todos los requisitos identificados en el proceso de análisis y comprobar su correcto funcionamiento. A continuación, se explican los tipos de pruebas seleccionados:

- **Pruebas unitarias:** Enfocan los esfuerzos de verificación en la unidad más pequeña del diseño de software. Al usar la descripción del diseño de componente como guía, las rutas de control importantes se prueban para descubrir errores dentro de la frontera del módulo. Se enfocan en la lógica de procesamiento interno y de las estructuras de datos dentro de las fronteras de un componente. Este tipo de pruebas puede realizarse en paralelo para múltiples componentes (Pressman, 2010).
- **Pruebas funcionales:** Son un proceso de control de calidad que consiste en asegurar el cumplimiento de un sistema o componente con requerimientos funcionales. El objetivo principal de las pruebas funcionales es analizar el producto terminado y determinar si cumple con todo lo que debería hacer y si lo hace correctamente. Este tipo de pruebas entran dentro de lo que se llaman pruebas de caja negra: aquí no se centra en cómo se generan las respuestas del sistema, solo se analizan los datos de entrada y los resultados obtenidos (Oterino, 2014).
- **Pruebas de aceptación:** Son realizadas con el cliente y define su aceptación del sistema. Son básicamente pruebas funcionales, sobre el sistema completo, y buscan una cobertura de la especificación de requisitos. Estas pruebas no se realizan durante el desarrollo, sino una vez pasadas todas las pruebas por parte del desarrollador. Pueden realizarse durante un período de semanas o meses, y mediante ellas implementación y pruebas del componente propuesto se descubren errores acumulados que con el tiempo puedan degradar el sistema. La mayoría de los constructores de productos de software usan un proceso llamado prueba alfa y prueba beta para descubrir errores que al parecer sólo el usuario final es capaz de encontrar (Pressman, 2010).

Las **pruebas alfa** se realizan en el espacio del desarrollador por un grupo representativo de usuarios finales. El software se usa en un escenario natural con el desarrollador y se realizan en un ambiente controlado.

Las **pruebas beta** se realiza en uno o más espacios del usuario final. A diferencia de la prueba alfa, por lo general el desarrollador no está presente. El cliente registra todos los problemas que se encuentran durante la prueba y los reporta al desarrollador periódicamente. En ocasiones se realiza una variación de la prueba beta, llamada prueba de aceptación del cliente, cuando el software se entrega a un cliente bajo contrato. El cliente realiza una serie de pruebas específicas con la intención de descubrir errores antes de aceptar el software del desarrollador.

3.8. Diseños de Casos de Prueba

Un Diseño de Caso de Prueba (DCP) está compuesto por un conjunto de entradas, respuestas (que emite el sistema de acuerdo a esas entradas) y el flujo central que indica el camino del escenario descrito. Estos son desarrollados para verificar el cumplimiento total o parcial de un requisito. Las entradas representan las variables que se pueden especificar y las mismas contienen: V, I, o N/A. V indica válido, I indica inválido, y N/A que no es necesario proporcionar un valor de la variable en un determinado caso, ya que es irrelevante.

A continuación, se muestran los casos de pruebas generados para el paquete de funcionalidades “Gestionar_Base_de_Datos_Geospacial”. La descripción de los demás casos de pruebas aplicados a los requisitos definidos, se pueden consultar en la carpeta “AD&MONPSQL_Pruebas_de_Software” la cual se encuentra adjunta a la presente investigación.

Tabla 12: Sección editar base de datos geoespacial correspondiente al caso de prueba Gestionar BDG.

Escenario	Descripción	Nombre	Respuesta del sistema	Flujo central
EC 2.1 Editar base de datos geoespacial satisfactoriamente.	Se selecciona la opción gestionar BDG. Se selecciona la opción editar base de datos, se llenan los datos requeridos y se selecciona la opción aceptar.	V	El sistema muestra un mensaje informando que la base de datos fue creada exitosamente	1-Autenticarse en el sistema. 2- Seleccionar la opción maquina servidora. 3-Seleccionar la opción Gestionar BDG. 4-Seleccionar editar base de datos.
EC 2.2 Seleccionar la opción cancelar.	El usuario selecciona la opción cancelar en el formulario de editar base de datos.	NA	El sistema deshace todos los cambios y vuelve a la página anterior	1-Autenticarse en el sistema.

				2- Seleccionar la opción maquina servidora. 3-Seleccionar la opción Gestionar BDG. 4- Seleccionar editar base de datos.
EC 2.3 Campos vacíos.	El usuario selecciona la opción aceptar sin llenar los campos.	V	El sistema muestra un mensaje informando que existen campos vacíos.	1- Autenticarse en el sistema. 2- Seleccionar la opción maquina servidora. 3- Seleccionar la opción Gestionar BDG. 4- Seleccionar editar base de datos.

A continuación, se muestra la descripción las variables correspondientes con el caso de prueba “Editar BDG”.

Tabla 13: Descripción las variables correspondientes con la sección editar base de datos geoespacial

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
I	Nombre	Campo de texto	No	Cadena de caracteres con un máximo de 50

Tabla 14: Sección adicionar base de datos geoespacial correspondiente al caso de prueba Gestionar BDG.

Escenario	Descripción	Nombre	Respuesta del sistema	Flujo central
EC 1.1 Crear base de datos geoespacial satisfactoriamente.	Se selecciona la opción gestionar BDG. Se selecciona la opción adicionar base de datos, se llenan los datos requeridos y se selecciona la opción aceptar.	V	El sistema muestra un mensaje informando que la base de datos fue creada exitosamente	1- Autenticarse en el sistema. 2- Seleccionar la opción maquina servidora. 3- Seleccionar la opción Gestionar BDG. 4- Seleccionar adicionar base de datos.

EC 1.2 Seleccionar la opción cancelar.	El usuario selecciona la opción cancelar en el formulario de crear base de datos.	NA	El sistema deshace todos los cambios y vuelve a la página anterior	1-Autenticarse en el sistema. 2- Seleccionar la opción maquina servidora. 3-Seleccionar la opción Gestionar BDG. 4-Seleccionar adicionar base de datos.
EC 1.3 Campo en blanco.	El usuario selecciona la opción aceptar sin llenar ningún campo.	NA	El sistema muestra un mensaje de informando que existen campos vacíos.	1-Autenticarse en el sistema. 2- Seleccionar la opción maquina servidora. 3-Seleccionar la opción Gestionar BDG. 4- Seleccionar adicionar base de datos.

A continuación, se muestra la descripción las variables correspondientes con el caso de prueba “Adicionar BDG”.

Tabla 15: Descripción las variables correspondientes con la sección adicionar base de datos geoespacial

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
1	Nombre	Campo de texto	No	Cadena de caracteres con un máximo de 50

Tabla 16: Sección eliminar base de datos geoespacial correspondiente al caso de prueba Gestionar BDG.

Escenario	Descripción	Nombre	Respuesta del sistema	Flujo central
-----------	-------------	--------	-----------------------	---------------

<p>EC 3.1 Eliminar base de datos geoespacial satisfactoriamente.</p>	<p>Se selecciona la opción gestionar BDG. Se selecciona la opción eliminar base de datos, selecciona la opción aceptar.</p>	<p>V</p>	<p>El sistema muestra un mensaje notificación para verificar si está seguro de eliminar la base de datos.</p>	<p>1-Autenticarse en el sistema. 2- Seleccionar la opción maquina servidora. 3-Seleccionar la opción Gestionar BDG. 4-Seleccionar en la lista de BD la que se quiere eliminar. 5-Seleccionar eliminar base de datos.</p>
<p>EC 3.2 Seleccionar la opción cancelar.</p>	<p>El usuario selecciona la opción cancelar en el formulario de eliminar base de datos.</p>	<p>NA</p>	<p>El sistema deshace todos los cambios y vuelve a la página anterior</p>	<p>1-Autenticarse en el sistema. 2-Seleccionar la opción maquina servidora. 3-Seleccionar la opción Gestionar BDG. 4- Seleccionar eliminar base de datos.</p>

3.9. Resultados de las pruebas

Pruebas de Caja Blanca (Unitarias):

A continuación, se muestra el caso de prueba correspondiente a la función “desinstalar_pgis”, el cual se localiza en la clase Admin_PostgreSQL. El objetivo de este método es desinstalar PostGIS en la máquina virtual servidora. Se selecciona este método teniendo en cuenta la importancia que representa para el resultado de este trabajo. Para obtener el conjunto de caminos independientes se construye el grafo de flujo asociado y se calcula su complejidad ciclomática. Para poder elaborar el grafo de flujo, primero se deben enumerar las sentencias del código. A continuación, se muestra el código fuente referente al método descrito anteriormente y cómo fue aplicada la Técnica de Camino Básico al mismo:

```

171 def Desinstalar_pgis(self):
172     if self.so == 'centos7':
173         salidaIns, erroresIns = self.conex.EjecutarComando("yum -y remove postgres")
174     if len(errores) > 0:
175         msgIns = "No se ha podido desinstalar Postgis" + "\n"
176         for i in errores:
177             msgIns = msgIns + i + "\n"
178         return False, msgIns
179     return True, "Se ha desinstalado Postgis con éxito"

```

Figura 14: Código del método Desinstalar_pgis.

Posteriormente se procede a la elaboración del grafo de flujo teniendo en cuenta dicha enumeración:

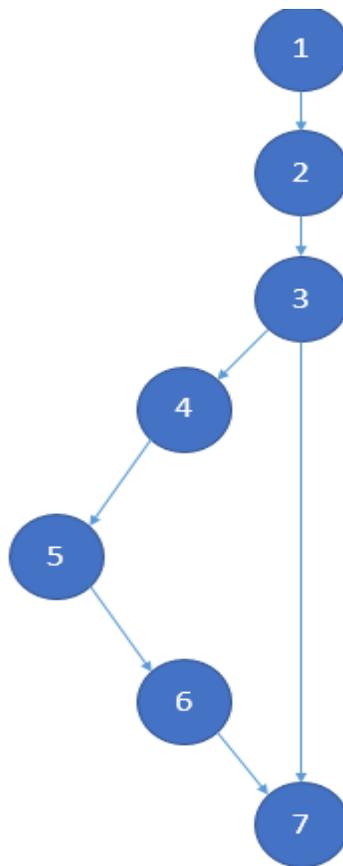


Figura 15: Representación del grafo de flujo de camino básico del método Desinstalar_pgis.

La complejidad ciclomática es la métrica de software con que se define la cantidad de caminos independientes de cada una de las funcionalidades del programa y provee el límite superior para el número de pruebas que se deben realizar para asegurar que se ejecute cada sentencia al menos una vez (Pressman, 2010). La complejidad ciclomática se basa en la teoría gráfica y para que el cálculo sea correcto, todos deben arrojar el mismo resultado:

El número de regiones corresponde a la complejidad ciclomática “V (G)”.

$V(G) = R$ Donde R es la cantidad total de regiones.

$V(G) = 2$

$V(G) = E - N + 2$ Donde E es el número de aristas y N número de nodos.

$V(G) = 7 - 7 + 2$

$V(G) = 2$

$V(G) = P + 1$ Donde P es la cantidad de nodos predicados

$V(G) = 1 + 1 = 2$

El valor V (G) expresa la cantidad de caminos linealmente independientes de la estructura de control del programa, por lo que se definen los siguientes 2 caminos:

- Camino básico 1: 1-2-3-4-5-6-7
- Camino básico 2: 1-2-3-7

Cada camino independiente es un caso de prueba a realizar, lo que se garantiza que durante la prueba se ejecute por lo menos una vez cada sentencia del programa. En el caso anterior se calcularon dos caminos básicos, por tanto, surge la necesidad de efectuar igual número de casos de prueba. A continuación, se muestran los casos de pruebas realizados:

Tabla 17: Caso de prueba de caja blanca para el camino básico 1

Entrada	N/A
Resultados Esperados	No se pudo desinstalar PostGIS en la máquina virtual servidora alojada en el sistema operativo CentOS7.
Condiciones	Se envía un mensaje al usuario (No se ha podido desinstalar PostGIS).

Tabla 18: Caso de prueba de caja blanca para el camino básico 2

Entrada	N/A
Resultados Esperados	Se instala PostGIS en la máquina virtual servidora alojada en el sistema operativo CentOS7.
Condiciones	Se envía un mensaje al usuario de confirmación (Se ha desinstalado PostGIS correctamente).

Una vez realizadas las pruebas unitarias mediante la técnica de camino básico al código seleccionado, se puede afirmar que la prueba concluye de forma satisfactoria, por tanto, se obtuvo el resultado esperado.

Pruebas de Caja Negra:

A través del método de caja negra, y apoyados en el diseño de Casos de Prueba se realizaron 3 iteraciones de pruebas internas pertenecientes al nivel de sistema. Dichas pruebas fueron realizadas con el objetivo de detectar y corregir errores que impidieran el correcto funcionamiento de la solución.

Para evaluar la solución se realizaron 3 iteraciones donde se probó el software íntegramente, finalmente se realizó una prueba final donde se comprobó la resolución de todas las no conformidades detectadas. A continuación, se presentan los resultados arrojados durante las diferentes pruebas aplicadas:

Tabla 19: Resultados de la prueba de caja negra 1ra iteración

Casos de Prueba	No Conformidades			
	Alta	Media	Baja	Total
Adicionar máquina virtual servidora	0	0	1	1
Administración del servicio PostgreSQL	2	4	2	8
Instalación del extensor PostGIS	3	2	5	10
Configuración de certificados SSL	0	2	4	6
Gestión de las BDG	1	1	3	5
Actualizar repositorios	0	2	4	6
Cantidad de memoria RAM	0	2	2	4
Cantidad de CPU	0	1	4	5
Total.	6	14	25	45

Tabla 20: Resultados de la prueba de caja negra 2da iteración.

Casos de Prueba	No Conformidades			
	Alta	Media	Baja	Total

Adicionar máquina virtual servidora.	0	0	0	0
Administración del servicio PostgreSQL	0	2	1	3
Instalación del extensor PostGIS	0	0	1	1
Configuración de certificados SSL	1	0	5	6
Gestión de las BDG	0	3	4	7
Actualizar repositorios	1	0	2	3
Cantidad de memoria RAM	0	0	2	2
Cantidad de CPU	0	0	1	1
Total	2	5	16	23

Tabla 21: Resultados de la prueba de caja negra 3ra iteración

Casos de Prueba	No Conformidades			
	Alta	Media	Baja	Resuelta
Adicionar máquina virtual servidora	0	0		0
Administración del servicio PostgreSQL	0	3	3	6
Instalación del extensor PostGIS	0	0	0	0
Configuración de certificados SSL	0	0	1	1
Gestión de las BDG	1	0	0	1
Actualizar repositorios	0	1	1	2
Cantidad de memoria RAM	1	0	1	2
Cantidad de CPU	0	0	2	2
Total	2	4	8	14

Las pruebas se centraron en el cumplimiento de los paquetes de funcionalidades descritos en el capítulo II del presente documento. Las no conformidades se clasificaron en Alta, Media o Baja en dependencia del impacto que tuvieran, generalmente las altas responden a errores técnicos relacionados directamente con la funcionalidad interna del paquete de funcionalidad y las bajas tienden a ser errores ortográficos y validaciones.

La figura muestra en una gráfica de barras los principales resultados de las pruebas de caja negra realizadas. Como se puede apreciar, se realizaron 3 iteraciones de pruebas, detectando en la primera, segunda y tercera iteración un total de 45, 23 y 14 no conformidades respectivamente. Comparando los resultados entre dichas iteraciones se evidencia una reducción considerable (alrededor de un 68.89%) de los principales errores detectados en el sistema una vez ejecutada la 3^{ra} iteración. Finalmente se realizó una última prueba donde se corrigieron todas las no conformidades. Se recomienda realizar la solicitud a Calidad Centro y Calidad UCI para realizar las correspondientes pruebas de liberación, las cuales no forman parte del alcance de este trabajo de diploma, debido al tiempo que demoran las mismas.

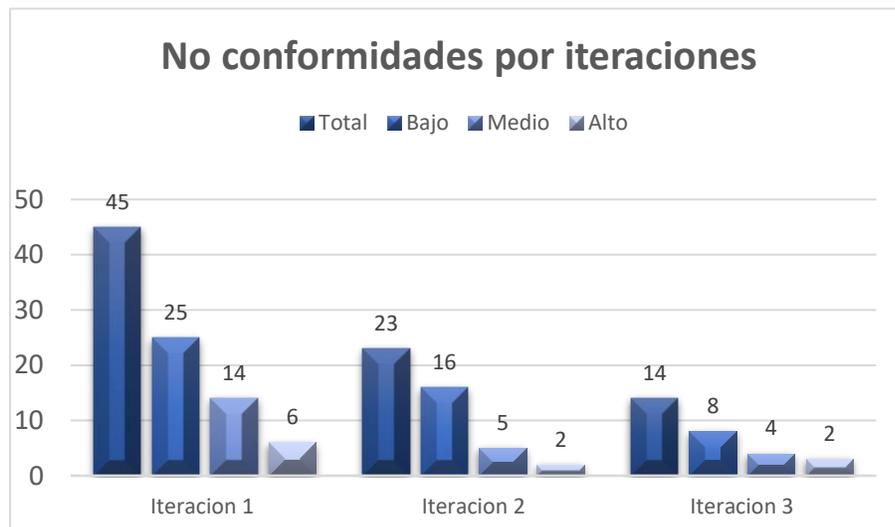


Figura 16: No conformidades detectadas contra cantidad de iteraciones de prueba.

Pruebas de aceptación:

Se empleó la técnica de prueba alfa, por el cliente y se evaluaron las funcionalidades del sistema basándose en la especificación de requisitos del expediente de proyecto. Los interesados verificaron cada requisito funcional y comprobaron si estos correspondían con los requerimientos planteados. (Anexo 2 Acta de aceptación de desarrollo de productos informáticos).

Al ser concluida las pruebas, se liberó el sistema de administración para PostgreSQL, entregándole al equipo de desarrollo la carta de aceptación en la que consta que el sistema está apto para ser utilizado y cumple con las expectativas planteadas.

3.10. Conclusiones del capítulo

Al realizar la implementación del sistema siguiendo los estándares de codificación definidos, se desarrolló un código reutilizable y comprensible por los integrantes del equipo de desarrollo. La elaboración del modelo de despliegue permitió representar la distribución física del sistema. Mediante los casos de prueba funcionales, el sistema fue probado y fueron resueltas todas las no conformidades que surgieron en el proceso. El sistema desde el punto de vista funcional cumple con los requerimientos definidos y especificados en las primeras etapas de desarrollo a partir de las necesidades del cliente.

Conclusiones

Se concluye este trabajo de diploma dando cumplimiento al objetivo propuesto para la realización del mismo. Mediante el presente documento, se ilustra al lector, acerca del proceso de desarrollo del Sistema de administración de servidores PostgreSQL para el Departamento de Desarrollo de Componentes del Centro Telemática V2.0. Además, esta investigación arrojó los siguientes resultados:

- La elaboración del marco teórico conceptual de la investigación permitió la realización de un marco de referencia para conformar una solución bien fundamentada, aportando ideas para el desarrollo del sistema AD&MONPSQL 2.0.
- A partir de la realización del análisis y diseño del sistema AD&MONPSQL 2.0 se obtuvo como resultado los diagramas y artefactos necesarios para guiar el desarrollo del mismo.
- Con la implementación de las funcionalidades definidas se obtuvo el sistema AD&MONPSQL 2.0 que garantiza la gestión de las BDG y la configuración de los certificados SSL en la administración de los servidores PostgreSQL instalados en los sistemas operativos CentOS7 y Windows Server 2012 de forma centralizada.
- El desarrollo de las pruebas permitió detectar y corregir los errores no detectados durante la implementación, posibilitando cumplir con las especificaciones requeridas y la validación del sistema implementado.
- El sistema AD&MONPSQL 2.0 permite administrar los servidores PostgreSQL instalados en los sistemas operativos CentOS7 y Windows Server 2012, garantizando la gestión de las BDG y la generación de certificados SSL de forma centralizada. Con esto se logra dar cumplimiento al objetivo propuesto.

Recomendaciones

Tomando como base la investigación realizada y la experiencia acumulada durante la realización de este trabajo de diploma, se proponen las siguientes recomendaciones:

- Se recomienda la inclusión, como parte de las funcionalidades del sistema, permitir la configuración del protocolo de autenticación de redes de computadoras Kerberos.
- Los resultados obtenidos sugieren continuar la investigación con el propósito de aumentar las funcionalidades del sistema AD&MONPSQL, para obtener mejoras en versiones futuras de la misma y seguir perfeccionando la administración de los servidores PostgreSQL en el Departamento de Componentes del Centro de Telemática de la Facultad 2 de la Universidad de las Ciencias Informáticas.

Referencias Bibliográficas

1. **Acosta Caldevilla, Bárbara Dayanne y Martínez Méndez, Jesús. 2018.** *Sistema para la administración y monitoreo de servidores PostgreSQL*. Tesis de investigación de pregrado
2. **Angulo, Diana Carolina y Henao, Juan Felipe. 2017.** *Análisis de herramientas de interceptación para el control de ataques reales de suplantación con certificados SSL*
3. **Benítez, Ángel y Arias, Miguel Ángel. 2017.** *Curso de Introducción a la Administración de Bases de Datos*. 2ª Edición, Editado por IT Campus Academy, 2017. ISBN 978-1542964890.
4. **Caldera Vergara, Roberto. 2017.** Estudio del framework de desarrollo web Django. [En línea]. <https://ebuah.uah.es/dspace/handle/10017/32018>.
5. **Casasayas Fernández, Lillian. 2014.** *Módulo de administración para el servicio PostgreSQL en la Herramienta para la Migración y Administración de Servicios Telemáticos*. Tesis de investigación de pregrado.
6. **Condori Ayala, José Luis. 2012.** *Python – Django Framework de desarrollo web para perfeccionistas Basado en el Modelo MTV*. p.p. 1997-4044
7. **Craig, Larman. 1999.** *Introducción al análisis y diseño orientado a objetos*. ISBN 970-17-0261-1
8. **DB-Engines. 2018.** DB-Engines Ranking. [En línea] 2018. <https://db-engines.com/en/ranking>
9. **Eguíluz Pérez, Javier. 2012.** *Introducción a javascript*
10. **España Sarasty , Hugo y Fernando. 2016.** *Documentación y análisis de los principales frameworks de arquitectura de software en aplicaciones empresariales*
11. **Foundation GNOME. Portal Web de GNOME.** [En línea] [Citado el: 27 de abril de 2014.] <http://live.gnome.org/Dia>
12. **FOUNDATION, PYTHON SOFTWARE. 2018.** *PEP 8 – Style Guide for Python Code*. [En línea]. <https://www.python.org/dev/peps/pep-0008/>
13. **Gauchat, Juan Diego. 2012.** *El gran libro de HTML5, CSS3 y Javascript*.
14. **Geodatabase. 2018.** Geodatabase (ESRI). [En línea] 2018. <http://www.esri.com/software/arcgis/geodatabase>
15. **Gómez Fuentes, Dra. María del Carmen. 2013.** [En línea] 2013. http://www.cua.uam.mx/pdfs/conoce/libroselec/Notas_del_curso_Bases_de_Datos.pdf
16. **González Paz, Alex , Beltrán Casanova, Msc. David y Fuentes Gari, Dr. C. Ernesto Roberto . 2016.** *Propuesta De Protocolos De Seguridad Para La Red Inalámbrica Local De La Universidad De Cienfuegos*. 2016. p.p. 2218-3620

17. **The PostgreSQL Global Development Group. 2018.** [En línea]. <https://www.PostgreSQL.org/about/>
18. **Group, Visual Paradigm. 2018.** Visual Paradigm. [En línea]. <https://www.visual-paradigm.com/>.
19. **Gutiérrez Nieto, C. M y Castellanos Fajardo, L. A. M.** ¿Qué son bases de datos geoespaciales? Centro de Investigación en Geografía y Geomática
20. **Gutiérrez Nieto, C. M y Castellanos Fajardo, L. A. M.** ¿Qué son bases de datos geoespaciales? Centro de Investigación en Geografía y Geomática
21. **Hernández León, Rolando Alfredo y Coello Gonzá, Sayda. 2012.** *El proceso de investigación científica*. s.l. : Editorial Universitaria del MINISTERIO DE EDUCACIÓN SUPERIOR. 978-959-16-1557-2
22. **Holovaty, Jacob y Kaplan-Moss, Adrian. 2009.** *The Definitive Guide To Django: Web development done right*. Apress
23. **Iruela, Juan. 2016.** *Los gestores de bases de datos más usados*. Revista digital INESEM
24. **Jimenez, M. C. 2014.** *Bases de datos relacionales y modelado de datos*. s.l.: UF1471. IC Editorial
25. **Maldonado, Wilson y Tituaña , Mauricio. 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*. *Aplicativo: Sistema de control de notas, para la unidad educativa mariano Suarez Veintimilla*. <http://repositorio.utn.edu.ec/handle/123456789/6903>
26. **Marini, Emiliano. 2012.** El Modelo Cliente/Servidor. [En línea]. <https://www.linuxito.com/docs/el-modelo-cliente-servidor.pdf>
27. **Marquez, Angel. 2015.** *PostGIS Essentials*. 978-1-78439-529-2.
28. **Moreno, Andrés y Sánchez, Camilo. 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*. 2013.
29. **Oterino, A.M. del C.G. 2014.** *¿Pruebas de integración, funcionales, de carga? ¿Qué diferencias hay?*
30. **Pressman, Roger S. 1998.** *Ingeniería del software. Un enfoque práctico*. Cuarta edición.
31. **Pressman, R.S. 2010.** *Ingeniería de Software, un enfoque práctico*. 2010. Séptima edición
32. **Ramos Cardozzo, Daniel . 2016.** *Desarrollo de Software: Requisitos, Estimaciones y Análisis*. s.l. : Createspace Independent Pub, 2016.
33. **Rumbaugh James, Jacobson Ivar y Booch, Grady. 2014.** *El Proceso Unificado de Desarrollo de Software. Manual de Referencia*. Addison Wesley.
34. **Rumbaugh, James y Jacobson, Ivar. 2014.** *El Proceso Unificado de Desarrollo de Software. Manual de Referencia*. Addison Wesley

35. **Rumbaugh, James , Jacobson, Ivar y Booch, Grady . 2014.** *El Proceso Unificado de Desarrollo de Software. Manual de Referencia.* s.l. : S.l.: Addison Wesley
36. **Sánchez Rodríguez, Tamara. 2015.** *Metodología de desarrollo para la Actividad productiva de la UCI.* Habana
37. **Sánchez, Yoiry, López , Yusbel y Chávez, Escal. 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. [En línea] 2017. <http://revistas.unica.cu/index.php/uciencia/article/view/302/1090>
38. **Sánchez, Tamara Rodríguez. 2015.** *Metodología de desarrollo para la Actividad productiva de la UCI.* La Habana. Cuba
39. **EMS Database Management Solutions. 2016.** EMS SQLManager for PostgreSQL. [En línea] 2016. www.sqlmanager.net/products/PostgreSQL/manager
40. **Webmin. 2017.** Webmin. [En línea] 2017. <https://doxfer.webmin.com/Webmin/Introduction>
41. **Ylonen, Chris y Lonvick, Tatu. 2006.** *The secure shell (SSH) connection protocol*
42. **Zapata, Carlos Mario , Palacio, Carolina y Olay, Natalí. 2007.** *UNC-analista: hacia la captura de un corpus de requisitos a partir de la aplicación del experimento mago de oz.* s.l. : Revista EIA, Junio 2007. ISSN 1794-1237.

Bibliografía

1. **Bamrara, A.** 2015. *Evaluating database security and cyber attacks: A relational approach*. The Journal of Internet Banking and Commerce, vol. 20, no 2
2. **Acosta Caldevilla, Bárbara Dayanne y Martínez Méndez, Jesús.** 2018. *Sistema para la administración y monitoreo de servidores PostgreSQL*. Tesis de investigación de pregrado
3. **Angulo, Diana Carolina y Henao, Juan Felipe.** 2017. *Análisis de herramientas de interceptación para el control de ataques reales de suplantación con certificados SSL*.
4. **Benítez, Ángel y Arias, Miguel Ángel.** 2017. *Curso de Introducción a la Administración de Bases de Datos*. s.l. : 2ª Edición, Ediado por IT Campus Academy, 2017. ISBN 978-1542964890
5. **Caldera Vergara, Roberto.** 2017. Estudio del framework de desarrollo web Django. [En línea]. <https://ebuah.uah.es/dspace/handle/10017/32018>
6. **Casasayas Fernández, Lillian.** 2014. *Módulo de administración para el servicio PostgreSQL en la Herramienta para la Migración y Administración de Servicios Telemáticos*
7. **Condori Ayala, José Luis .** 2012. *Python – Django Framework de desarrollo web para perfeccionistas Basado en el Modelo MTV*. 2012. p.p. 1997-4044
8. **Craig, Larman.** 1999. *Introducción al análisis y diseño orientado a objetos*. 1999. ISBN 970-17-0261-1
9. **DB-Engines.** 2018. DB-Engines. [En línea] 2018. <https://db-engines.com/en/ranking>
10. **Eguíluz Pérez, Javier.** 2012. *Introducción a javascript*
11. **España Sarasty , Hugo y Fernando.** 2016. *Documentación y análisis de los principales frameworks de arquitectura de software en aplicaciones empresariales*
12. **Foundation GNOME.** live.gnome.org. *Portal Web de GNOME*. [En línea] [Citado el: 27 de 04 de 2014.] <http://live.gnome.org/Dia>
13. **FOUNDATION, PYTHON SOFTWARE.** 2018. PEP 8 – Style Guide for Python Code. [En línea] 2018. <https://www.python.org/dev/peps/pep-0008/>
14. **Gauchat, Juan Diego.** 2012. *El gran libro de HTML5, CSS3 y Javascript*. 2012
15. **Geodatabase.** 2018. Geodatabase (ESRI). [En línea] 2018. <http://www.esri.com/software/arcgis/geodatabase>
16. **Gómez Fuentes, Dra. María del Carmen.** 2013. [En línea] 2013. http://www.cua.uam.mx/pdfs/conoce/libroselec/Notas_del_curso_Bases_de_Datos.pdf.

17. **González Paz, Alex , Beltrán Casanova, Msc. David y Fuentes Gari, Dr. C. Ernesto Roberto. 2016.** *Propuesta De Protocolos De Seguridad Para La Red Inalámbrica Local De La Universidad De Cienfuegos.* 2016. p.p. 2218-3620
18. **Groomer , Michael y Murthy, Uday S. 2018.** 2018. *Continuous Auditing of Database Applications: An Embedded Audit Module Approach.* 2018. 978-1-78743-414-1
19. **Group, The PostgreSQL Global Development. 2018.** [En línea]. <https://www.PostgreSQL.org/about/>.
20. **Group, Visual Paradigm. 2018.** Visual Paradigm. [En línea] 2018. <https://www.visual-paradigm.com/>.
21. **Gutiérrez Nieto, C. M y Castellanos Fajardo, L. A. M.** ¿Qué son bases de datos geoespaciales?
Centro de Investigación en Geografía y Geomática
22. **Gutiérrez Nieto, C. M y Castellanos Fajardo, L. A. M.** ¿Qué son bases de datos geoespaciales?
Centro de Investigación en Geografía y Geomática
23. **Hernández León, Rolando Alfredo y Coello Gonzá, Sayda. 2012.** *El proceso de investigación científica.* s.l. : Editorial Universitaria del MINISTERIO DE EDUCACIÓN SUPERIOR. 978-959-16-1557-2
24. **Holovaty , Jacob y Kaplan-Moss, Adrian. 2009.** *The Definitive Guide To Django.* Ed. por APRESS,Editorial.
25. **Holovaty. 2009.** *The Definitive Guide To Django.* 2009
26. **Iruela, Juan. 2016.** *Los gestores de bases de datos más usados.* Revista digital INESEM
27. **Jimenez, M. C. 2014.** *Bases de datos relacionales y modelado de datos.* s.l. : UF1471. IC Editorial, 2014
28. **Maldonado, Wilson y Tituaña , Mauricio. 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.* s.l. : Aplicativo: Sistema de control de notas, para la unidad educativa mariano Suarez Veintimilla.
<http://repositorio.utn.edu.ec/handle/123456789/6903>
29. **Marini, Emiliano . 2012.** El Modelo Cliente/Servidor. [En línea]. <https://www.linuxito.com/docs/el-modelo-cliente-servidor.pdf>
30. **Marquez, Angel. 2015.** *PostGIS Essentials.* 2015. 978-1-78439-529-2
31. **Martinez Rubi, Oscar , Ivanovab, Milena y Horhamm, Mike. 2015.** *Massive point cloud data management: Design, implementation and execution of a point cloud benchmark.* [En línea] 2015.
<https://doi.org/10.1016/j.cag.2015.01.007>

32. **Moreno, Andrés y Sánchez, Camilo. 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*
33. **Oterino, A.M. del C.G. 2014.** *¿Pruebas de integración, funcionales, de carga? ¿Qué diferencias hay?*
34. **Pressman , Roger S. 1998.** *Ingeniería del software. Un enfoque práctico.* 1998. Cuarta edición.
35. **Pressman, R.S. 2010.** *Ingeniería de Software, un enfoque práctico.* 2010. Séptima Edición
36. **Ramos Cardozzo, Daniel . 2016.** *Desarrollo de Software: Requisitos, Estimaciones y Análisis.* Createspace Independent Pub
37. **Rumbaugh James, Jacobson Ivar y Booch, Grady.** *El Proceso Unificado de Desarrollo de Software. Manual de Referencia.* s.l. : Addison Wesley
38. **Rumbaugh, James y Jacobson, Ivar . 2014.** *El Proceso Unificado de Desarrollo de Software. Manual de Referencia.* s.l. : Addison Wesley
39. **Rumbaugh, James , Jacobson, Ivar y Booch, Grady . 2014.** *El Proceso Unificado de Desarrollo de Software. Manual de Referencia.* s.l. : S.l.: Addison Wesley
40. **Sánchez Rodríguez, Tamara. 2015.** *Metodología de desarrollo para la Actividad productiva de la UCI.* Habana
41. **Sánchez, Yoiry, López , Yusbel y Chávez, Escal. 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. [En línea] 2017. <http://revistas.unica.cu/index.php/uciencia/article/view/302/1090>
42. **Sánchez, Tamara Rodríguez. 2015.** *Metodología de desarrollo para la Actividad productiva de la UCI.* La Habana. Cuba
43. **SOLUTIONS, EMS DATABASE MANAGEMENT. 2016.** EMS SQLManager for PostgreSQL. [En línea] 2016. www.sqlmanager.net/products/PostgreSQL/manager
44. **Webmin. 2017.** Webmin. [En línea] 2017. <https://doxfer.webmin.com/Webmin/Introduction>.
45. **Ylonen , C y Lonvick, T. 2006.** *The secure shell (SSH) connection protocol*
46. **Zapata, Carlos Mario , Palacio, Carolina y Olay, Natalí . Junio 2007.** *UNC-ANALISTA: HACIA LA CAPTURA DE UN CORPUS DE REQUISITOS A PARTIR DE LA APLICACIÓN DEL EXPERIMENTO MAGO DE OZ.* s.l. : Revista EIA, Junio 2007. ISSN 1794-1237

Anexo 1 Acta de conformidad de los requisitos por el cliente

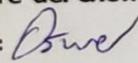
	CENTRO DE TELEMÁTICA DEPARTAMENTO DESARROLLO DE COMPONENTES	 CENTRO DE TELEMÁTICA
---	--	---

La Habana, 16 de enero de 2019
"Año 61 de la Revolución"

Mediante el presente documento se certifica que los 33 requerimientos del Sistema: Administración de servidores PostgreSQL, quedan aprobados y validados por el cliente.

Observaciones:

Recomendaciones:

Nombre del cliente: Osmar Capote Vázquez.
Firma: 



Universidad de las Ciencias Informáticas
Carretera San Antonio Km 2 ½ Torrens. Boyero,
La Habana, Cuba.
Teléfono: +53 7 835 8151 y +53 7 835 8152

Anexo 2 Acta de aceptación de desarrollo de productos informáticos

	CENTRO DE TELEMÁTICA DEPARTAMENTO DESARROLLO DE COMPONENTES	CENTRO DE TELEMÁTICA								
La Habana, 27 de mayo de 2019 "Año 61 de la Revolución"										
<h3>ACTA DE ACEPTACIÓN DE DESARROLLO DE PRODUCTOS INFORMÁTICOS</h3>										
<p>En cumplimiento con la fase de desarrollo y en función de la ejecución del proyecto: Sistema de administración de servidores PostgreSQL para el departamento de Componentes del Centro Telemática V2.0, se hace entrega de los productos que se relacionan a continuación:</p>										
<p>• <i>Sistema de administración de servidores PostgreSQL para el departamento de Componentes del Centro Telemática V2.0</i></p>										
<table border="0" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%; border-top: 1px solid black; border-bottom: 1px solid black;">Entrega</td> <td style="width: 50%; border-top: 1px solid black; border-bottom: 1px solid black;">Recibe</td> </tr> <tr> <td style="border-bottom: 1px solid black;">Nombre y apellidos: Sunaidy Camila Jauregui Fajardo</td> <td style="border-bottom: 1px solid black;">Nombre y apellidos: Ing. Osmar Capote Vázquez</td> </tr> <tr> <td style="border-bottom: 1px solid black;">Cargo: Tesista</td> <td style="border-bottom: 1px solid black;">Cargo: Jefe de Departamento</td> </tr> <tr> <td style="border-bottom: 1px solid black;">Firma: </td> <td style="border-bottom: 1px solid black;">Firma: </td> </tr> </table>			Entrega	Recibe	Nombre y apellidos: Sunaidy Camila Jauregui Fajardo	Nombre y apellidos: Ing. Osmar Capote Vázquez	Cargo: Tesista	Cargo: Jefe de Departamento	Firma:	Firma:
Entrega	Recibe									
Nombre y apellidos: Sunaidy Camila Jauregui Fajardo	Nombre y apellidos: Ing. Osmar Capote Vázquez									
Cargo: Tesista	Cargo: Jefe de Departamento									
Firma:	Firma:									
Universidad de las Ciencias Informáticas Carretera San Antonio Km 2 ½ Torrens. Boyero. La Habana, Cuba. Teléfono: +53 7 835 8151 y +53 7 835 8152										

Anexo 3 Guía para la realización de la entrevista

Guía de preguntas:

ok ¿Cómo se realiza actualmente la administración de PostgreSQL en el departamento?

ok ¿Cuáles son las necesidades actuales del departamento?

ok ¿Qué usuarios tienen acceso al sistema?

ok ¿Cómo se realiza la configuración de los certificados SSL?

ok ¿Cómo se realiza la gestión de las bases de datos geoespaciales?