Universidad de las Ciencias Informáticas

Facultad 3

Componente para búsquedas dinámicas mediante estructuras JSON en PostgreSQL para los proyectos de CEGEL

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

Autora:

Milena Bárbara Mena González

Tutores:

Ing. Yosvany Gómez Perdomo

Ing. Julio Luis Rivera Abreu

La Habana, junio de 2022

DECLARACIÓN DEAUTORÍA

Declaro ser autora de la presente tesis y reconozco a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmo la presentea los__días delmes de_____delaño_.

_____

Milena Bárbara Mena González

(Autora)

_____

Ing. Yosvany Gómez PerdomoIng. Julio L. Rivera Abreu

(Tutor)(Tutor)

AGRADECIMIENTOS

Agradezco a mis padres por su apoyo incondicional, y por darme tanta fuerza cuando ya no podía más, y por hacerme entender que no puedo temer, aunque la tierra sea removida y se traspasen los montes al corazón del mar porque Jehová de los ejercito es mi refugio.

A mi hermana Misleydis y a su esposo Carlos los cuales se han convertido en mis segundos padres, gracias a ustedes también hoy soy lo que soy.

A una persona muy especial sin la cual no me hubiese podido graduar porque me ayudó con mi peor pesadilla, la tesis, como si fuera la deél mismo: al mejor amigo de todos los tiempos: mi cocuyo, gracias por todo. Nunca voy a olvidar tu esfuerzo para conmigo.

A mis tutores por su apoyo y consideración para la realización de este trabajo.

A mis grandes amigos Karla, Ailyn, Juan Pablo y Julio Luis que han estado conmigo en las buenas y en las malas y con las cuales he pasado excelentes e inolvidables momentos. Los quiero mucho.

A mis compañeros de aula que siemprevoy a recordar porque hicieron que mi paso por la universidad fuera más alegre he inolvidable.

A todos los profesores que han contribuido a mi formación como profesional.

DEDICATORIA

A mis padres, a mi hermana y mi cuñado Carlos por ser mi inspiración y mi ejemplo a seguira lo largo de estos años. Los quiero mucho.

A mis amigos por estar ahí cuando los necesito y en es especial a ti Julio.

A todos muchas gracias por ayudarme a llegar a este día.

Milena Bárbara Mena

RESUMEN

El Centro de Gobierno Electrónicode la Universidad de las Ciencias Informáticas surge con el objetivo de informatizar procesos en las instituciones gubernamentales en Cuba y con ello la posibilidad de agilizar los canales de información en las distintas organizaciones. En este centro, se ejecutan varios proyectos que desarrollan disímilessoluciones informáticas para las diferentes entidades del país. Actualmente, enestas soluciones,el proceso de adecuación de los reportes a las nuevas especificaciones se torna lento debido a las diversastecnologías utilizadas para su conformación. Además, si a esto se le adiciona las constantes identificaciones de nuevas necesidades de información,provoca un trabajo exhaustivo por parte del equipo de desarrollo.Por ello, el presente trabajo de diploma se traza como objetivo principal el desarrollo de un componentepara búsquedas dinámicas mediante estructuras *JSON* en PostgreSQL, de forma tal que contribuya a un mayor rendimiento y estandarización de los reportes para los proyectos del centro. Para guiar el desarrollo de la propuesta de solución se empleó como metodología de desarrollo de software el Proceso Unificado Ágil en su variación para la UCI.Además, se utilizaron las herramientas, lenguajes y tecnologías definidas por el equipo de arquitectura del Centro de Gobierno Electrónico.Al mismo tiempo, para garantizar la calidad del sistema, se realizaron las validaciones correspondientes y se aplicaron las pruebas pertinentes, obteniéndose como resultado un componente para la configuración y servicios de las búsquedas dinámicas para los proyectos de CEGEL.

Palabras claves:búsquedas dinámicas, componente, estandarización, reportes.

ÍNDICE

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

INTRODUCCIÓN

Las Tecnologías de la Información y las Comunicaciones (TIC) han revolucionado al mundo actual, de manera tal que cumplen un importante papel enel desarrollo de la sociedad convirtiéndose en uno de los motores principales de la actividad humana Son diversas las ventajas que ofrecen las TIC que van desde la mejora del rendimiento de los procesos en los que se utilizan, hasta la obtención de los resultados con mayor rapidez y disponibilidad. Por tanto, las TIC están en constante proceso de cambio para mejorar sus servicios y calidad de funcionamiento, por lo que se puede considerar que son dinámicas y se adaptan a las necesidades del presente.

En la actualidad las entidades u organizaciones manejan grandes volúmenes de datos como resultado de los procesos de negocios llevados a cabo en el entorno en que se desenvuelven. Estos requieren de aplicaciones y tecnologías que permitan organizar y extraer la información de bases de datos, siendo estas de utilidad para la planeación estratégica, el control y la toma de decisiones de especialistas y directivos.

En Cuba se realizan innumerables esfuerzos con el fin de aumentar el uso masivo de las tecnologías, llevándolas a todas las esferas de la sociedad cubana, con el objetivo de elevar el nivel de vida de su población. Por otro lado, para lograr el control de los procesos de los distintos negocios a informatizar en Cuba, las empresas han puesto su empeño en automatizar en gran medida la gestión de la información, reduciendo los gastos de producción y la ejecución manual de estas operaciones.

Entre las principales funcionalidades de los sistemas informáticos desarrollados en Cuba, se encuentra la de proveer toda la información que se genera durante los distintos procesos de gestión en las empresas, relacionados con el control y supervisión de los mismos, así como los pertenecientes a niveles superiores dentro o fuera de la empresa; siendo el control de calidad, la supervisión y el mantenimiento algunos de los aspectos principales. Para lograr un control eficiente en la gestión de estas tareas, se hace necesario buscar mecanismos para la generación de reportes. Los mismos deben ser capaces de consolidar la información adquirida y mostrarla en formatos entendibles por el personal que espera recibir la información.

INTRODUCCIÓN

Uno de los mecanismos más utilizados en la actualidad son las búsquedas dinámicas, las cuales poseen ciertas características que facilitan tener resultados más exactos, configurables y variables en el tiempo; permiten hacer combinaciones de tal forma que uno o más términos sean buscados en diferentes índices o partes del registro simultáneamente. También permiten añadir o excluir un término específico de la búsqueda o recuperar uno u otro de los términos ingresados. Por tanto, tiene como principal función satisfacer lo mejor posible el deseo de quien busca una determinada información; sin importar cuan variable pueda ser el objeto de búsqueda sobre un concepto; siempre y cuando esté contenido o relacionado con él (Informatica, 2019).

El Centro de Gobierno Electrónico (CEGEL), perteneciente a la Universidad de las Ciencias Informáticas (UCI), tiene como misión satisfacer necesidades de clientes gubernamentales mediante el desarrollo de productos, servicios y soluciones integrales de alta confiabilidad, calidad, competitividad, fidelidad y eficiencia, a partir de un personal altamente calificado(LIGP, 2017). Posee varios productos y soluciones de software que tienen como objetivo la gestión de las entidades o instituciones de los clientes, donde la información que se produce crece exponencialmente y es una prioridad el análisis de los datos guardados por estos sistemas. Entre ellos se destacan: Sistema de Gestión Fiscal, ONEI, SIGAP, SIGIP, SICOM, entre otros.

El noventa y ocho por ciento (98%)de los proyectos se encuentran desarrollados sobre base de datos relacionales, pero son disímiles las tecnologías y formas de implementación en la confección y recuperación de la información de cada sistema desarrollado. Esto trae como consecuencia una mayor dificultad a la hora de realizar las peticiones de nuevas solicitudes de información y exige la experticia por parte del equipo de desarrollo debido a la necesidad de conocer las diferentes formas de codificación en la que se pueden expresar los métodos de recuperación de datos que se definen en cada tecnología utilizada.

De manera general el diseño de la base de datos está más enfocado y condicionado al desarrollo de sistemas registrales que al desarrollo de sistemas de reportes y el acceso rápido a grandes volúmenes de información. En consecuencia, se evidencia que el manejo de temas tan sensibles como: minimización del espacio de almacenamiento, disminución de los tiempos de respuesta y del consumo de recursos, no han constituido prioridades durante el proceso de diseño de la base de datos.

Estos sistemas cuentan con módulos de reportes que realizan búsquedas sintácticas[1]devolviendo solo la información en la cual aparecen elementos constituyentes a la consulta realizada, tal y como ha sido especificada. Además, si se añade que, a medidaque crece la explotación de los sistemas,los reportes emitidos periódicamente y la información solicitada por la dirección cambian con una alta frecuencia. Esto provoca que en muchas ocasiones las respuestas arrojadas no sean las que los usuarios necesitan e implicasolicitudes de cambio constantes para satisfacer las nuevas necesidades de información(CEGEL, 2020).

Otro aspecto importante es que los módulos de reportes no son configurables porque no cuentan con funcionalidades que permitan al administrador gestionar los criterios de búsqueda(CEGEL, 2020). Además, para poder agregar o eliminar un criterio se necesita acudir a los especialistas encargados del mantenimiento del sistema; quienes deben instalar todas las herramientas necesarias para la modificación del módulo y trabajar directamente en el código fuente de la aplicación para realizar dichos cambios. Una vez modificado el código fuente es necesaria la actualización del sistema en el lugar donde se encuentre alojado(CEGEL, 2020). Al realizar todas estas gestiones existe un consumo considerable de tiempo para poderlas llevar a cabo, dificultando así la realización inmediata de los cambios en cuanto a las nuevas necesidades de información.

Dada la situación planteada anteriormente surge el siguiente **problema a resolver:**¿Cómo contribuir a las búsquedas en los proyectos de CEGEL de forma tal que favorezca a un mayor rendimiento y estandarización de los reportes en dichos proyectos?

Tomando como**objeto de estudio:** Búsquedas dinámicas en base de datos relacionales.

Con el fin de corregir el problema planteado se define como**objetivo general:** Desarrollar un componente para búsquedas dinámicas mediante estructuras JSON en PostgreSQL de forma tal que contribuya a un mayor rendimiento y estandarización de los reportes para los proyectos de CEGEL.

Objetivos específicos:

- Identificar los referentes teóricos en los que se sustenta la propuesta de solución sobre las búsquedas dinámicas en base de datos relacionales.

[1] La búsqueda que se realiza sin tener en cuenta el significado de la frase

- Realizar el levantamiento de los requisitos, el análisis y diseñodel componente para las búsquedas dinámicas como aproximación a la implementación.
- Implementar el componente de búsquedas dinámicas para contribuir a un mayor rendimiento y estandarización de los reportes para los proyectos del CEGEL.
- Validar el correcto funcionamiento del componente a través de pruebas de software para garantizar la calidad de los mismos.

Enmarcándose en el**campo de acción:** Búsqueda dinámicas mediante estructuras JSON en PostgreSQL.

Teniendo como referencia el problema a resolver y el objetivo general se plantea la siguiente **idea a defender**: si se desarrolla un componente de búsquedas dinámicas mediante estructuras JSON en PostgreSQL se contribuirá a un mayor rendimiento y estandarización de los reportes para los proyectos del CEGEL**.**

Para dar solución a las tareas antes propuestas se definen los métodos científicos utilizados en la investigación, clasificados en teóricos y empíricos:

Métodos teóricos:

Los métodos teóricospermiten descubrir en el objeto de investigación las relaciones esenciales y las cualidades fundamentales, no detectables de manera sensoperceptual[2]. Por ello se apoya en los procesos de análisis, síntesis, inducción y deducción(Martinez Perez , et al., 2016).

Se emplearon como métodos teóricos:

El método **Analítico–Sintético** se utilizó parasintetizar los elementos más importantes que se relacionaron con el componente para búsquedas dinámicas a partir de tendencias y documentos relacionados con el tema, expresando de manera resumida la posición del investigador.

El análisis **Histórico–Lógico** se ha realizado a través de un estudio crítico sobre las búsquedas dinámicas verificando y comparando los sistemas que poseen este tipo de búsquedas utilizándolos como punto de referencia.

[2]La sensación que se experimenta a partir de los estímulos que se reciben mediante los sentidos(el gusto,el tacto,el olfato,la audición y la vista).

El método **Hipotético–Deductivo** analizando y definiendo la idea a defender, siendo examinada o probada con el desarrollo de la investigación, arribando a conclusiones particulares.

La **Modelación** realizandodiagramas necesarios en el proceso de desarrollo de software, haciendo una representación abstracta de la solución que facilite así el desarrollo de la misma.

Métodos empíricos:

Los métodos empíricos revelan, describen y explican las características y relaciones esenciales del objeto basando su contenido en la experiencia(Martinez Perez , et al., 2016).

Se emplearon como métodos empíricos:

La **Entrevista** se utilizó a través de un intercambio verbal con el cliente para adquirir la mayor cantidad de información posible, entender el proceso de negocio, así como las imperfecciones existentes que permitieron definir el problema a resolver y establecer el objeto de estudio.

La **Observación** de los procesos para la generación y obtención de los reportes en los proyectos del centro CEGEL, así como los mecanismos para la recuperación de datos con el fin de lograr un mayor entendimiento, comprensión en las formas de búsqueda de información.

La **Medición** es elmétodoque se desarrolla con el objetivo de obtener información numérica acerca de una propiedad o cualidad del objeto,proceso o fenómeno,donde se comparan magnitudes medibles conocidas.Es la asignación de valores numéricos a determinadas propiedades de objeto (Martinez Perez , et al., 2016).

Estructuración de la investigación por capítulos

El presente trabajo de diploma está estructurado en 3 capítulos de la siguiente forma:

Capítulo 1:Fundamentación teórica del componente para búsquedas dinámicas

En este capítulo se abordarán los conceptos necesarios para entender el objetivo general y sistematizar el marco teórico de la investigación. De igual forma, se realizará un estudio del arte mediante el análisis deherramientascon similar propósito al de la propuesta de solución, tanto en el ámbito nacional como

internacional.Además, se describe el entorno de desarrollo a partir de la fundamentación del uso de la metodología, tecnologías y herramientas propuestas para el desarrollo de la propuesta de solución.

Capítulo 2: Análisis, diseño e implementación del componente para búsquedas dinámicas

En este capítulo se llevará a cabo la caracterización de la propuesta de solución. Se hace un análisis desde la óptica de la ingeniería de software, donde se especifican los requisitos funcionales y no funcionales a partir de la metodología seleccionada, y se realiza el diseño ingenieril dondese describen los patrones de diseño definidos como buenas prácticas durante el ciclo de desarrollo del software, la realización del modelado de diagramas, los elementos fundamentales del diseño y de la arquitectura que se deben tener en cuenta para llegar a la conclusión de cómo será el componente para búsquedas dinámicas.Finalmente, quedarán definidos los estándares de codificación a utilizar durante la etapa de implementación, para contribuir al futuro mantenimiento y/o actualización del sistema.

Capítulo 3:Validación del componente para búsquedas dinámicas

En este capítulo se evaluará el nivel de calidad y fiabilidad de los resultados obtenidos luego del desarrollo del componente para búsquedas dinámicasmediante estructuras JSON en PostgreSQL para los proyectos deCEGEL. Dicha evaluación se llevará a cabo a partir de la validación del diseño a través de las métricas: Tamaño Operacional de las Clases y Relación entre Clases. Por último, se utiliza la disciplina de pruebas internasque define la metodología que guía el proceso de desarrollo de la solución propuesta, con el fin de verificar y revelar la calidad del producto antes de su entrega al cliente a partir de la realización de pruebas funcionales por los métodos de caja blanca y caja negra.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA DEL COMPONENTE PARA BÚSQUEDAS DINÁMICAS

Introducción

En el presente capítulo se plantean los conceptos fundamentales para soportar los principales referentes teóricos en los que se enmarca el desarrollo de la propuesta de solución. También se argumenta en la selección de la metodología, patrones de diseño, las herramientas y lenguajes de programación por parte del equipo de arquitectura del proyecto, que son usados en el desarrollo del componente para búsquedas dinámicas.

1.1. Conceptos asociados a la investigación

Con el propósito de lograr una mejor comprensión en las temáticas a abordar en el presente trabajo, se muestra un conjunto de conceptos asociados al tema de investigación.

Búsqueda dinámica

Las búsquedas dinámicas son una nueva función que permite al usuario realizar búsqueda en determinado sistema sin tener que navegar en otra aplicación. Además, son utilizadas con mayor frecuencia en la web y a través de buscadores. También son asociadas al término de búsquedas avanzadas es un tipo de búsqueda que posee características adicionales soportadas por el motor de búsqueda. Suele estar presente opcionalmente en ciertos buscadores, programas, herramientas y servicios online (Alegsa, 2016).

Un motor de búsqueda o buscador es un sistema informático que busca archivos almacenados en un servidor web. Las búsquedas se efectúan a través de palabras claves o con árboles jerárquicos y como operan de forma automática los motores de búsquedas contienen más información que los directorios web (Alegsa, 2016).

El primer buscador en realizarse fue Wandex, que ejecutaba la búsqueda en forma de índice, otro de los primeros buscadores fue Aliweb el cual todavía está en funcionamiento y que, a diferencia desu predecesor, este permitía la búsqueda por palabras en cualquier página web. Pero el camino de los buscadores ha sido largo hasta llegar a Google, cambiando de forma radical el funcionamiento de los motores de búsqueda (Alegsa, 2016).

Estandarización

Se denomina estandarización al acto y el resultado de estandarizar:(ajustar a un estándar).La estandarización implica concertar determinados sistemas para que resulte coincidente o concordante con un modelo, un patrón o una referencia.Por tanto, es el proceso de adaptar características en un producto, servicio o procedimiento, con el objetivo de que estos se asemejen a un tipo, modelo o norma en común (Pérez, et al., 2018).

En la presente investigación la estandarización se hará necesaria para establecer una única codificación que sea capaz de resolver de forma dinámica cada petición realizada sin tener en cuenta los conceptos o los filtros que sean precisados. Esto permitirá establecer un servicio que podrá ser utilizado por cada uno de los proyectos sin tener en cuenta la información contenida o el objeto de búsqueda.

Reporte

Un reporte se presenta como resultado en torno a un tema específico con el fin de exponer información objetiva, de manera clara y ordenada. En el ámbito de la informática,los reportes son informes que organizan y exhiben la información contenida en las bases de datos. Su función es aplicar un formato determinado a los datos para mostrarlos por medio de un diseño atractivo y que sea fácil de interpretar por los usuarios. Además, tienen diversos niveles de complejidad desde una lista o enumeraciones hasta gráficos mucho más desarrollados (Pérez, et al., 2015).

Almacén operacional de datos

Un almacén operacional de datos ODS (por su siglas en inglés *OperationalData Store*) es un tipo de base de datos utilizada frecuentemente como área lógica intermedia para un almacén de datos.

Mientras permanecen en un ODS los datos pueden ser borrados, cotejados para evitar redundancias y verificados para asegurar el cumplimiento con la normativa aplicable a la empresa correspondiente. ElODS puede usarse para integrar información muy dispar proveniente de múltiples fuentes a fin de que las operaciones de negocio, el análisis y el reporte puedan llevarse a cabo sin detener las operaciones empresariales. Este es el lugar en el que se guarda la mayor parte de la información empleada en las

operaciones en un momento dado antes de su traslado a un almacén de datos para su almacenamiento o archivo a largo plazo (Rouse, 2019).

Base de datos relacional

Una base de datos es un conjunto de datos pertenecientes a un mismo contexto y almacenados sistemáticamente para su posterior uso.Existen diferentes bases de datos entre las que se encuentran las bases de datos dinámicas, estática,jerárquicas,de red, deductivas y relacional (Oracle, 2019).

Una base de datos relacional es un tipo de base de datos que almacena y proporciona acceso a puntos de datos relacionados entre sí. Las bases de datos relacionales se basan en el modelo relacional: una forma intuitiva y directa de representar datos en tablas. En una base de datos relacional, cada fila de la tabla es un registro con un identificador único llamado clave. Las columnas de la tabla contienen los atributos de los datos y cada registro tiene normalmente un valor para cada atributo, lo que permite establecer fácilmente las relaciones entre los puntos de datos (Oracle, 2019).

Notación de Objetos de JavaScript

JSON (por sus siglas en inglés: *JavaScript ObjectNotation*):es un formato estándar abierto que consta de pares clave-valor. El uso principal de JSON es transportar datos entre un servidor y una aplicación web. A diferencia de otros formatos, JSON es texto legible por humanos (PostgreSQL_JSON, 2019).
Algunas de las características que posee JSON son:

- Las claves deben estar entre comillas.
- El uso de las comillas para considerarse válido exige que sean del tipo "" dobles.
- No permite el uso de indefinido como un valor asociado a una clave.
- No permite como valor el uso de notaciones numéricas del tipo 0023 es decir con ceros al inicio (EDteam, 2019).

Una de las ventajas de JSON sobre otro formato de intercambio de datos es que resulta mucho más sencillo escribir un analizador sintáctico (*parse*) para él(PostgreSQL, 2018).

Luego de revisar la bibliografía asociada al objeto de estudio de la presente investigación y sistematizar sus referentes teóricos, los conceptos antes descritos son los asumidos por la autora. Para el caso de las búsquedas,se hace referencia a la funciónque permite al usuario realizarlas en el sistema sin tener que depender de otra aplicación. Estas búsquedas se realizarán sobre bases de datos relacionales que almacenarán y proporcionarán acceso a puntos de datos relacionados entre sí, integrando la información mediante un modelo basado en conceptosa fin de que las operaciones de negocio, el análisis y el reporte puedan llevarse a cabo sin detener las operaciones empresariales. A su vez, el reporte se refiere a los informes queorganizan y exhiben dicha información contenida en las bases de datos. Mientras que la estandarización se evidenciará al establecer una única codificación, capaz de resolver de forma dinámica cada petición realizada. Esto puede efectuarse sin tener en cuenta los conceptos o los filtros que sean precisados, al hacer uso del formato estándar abierto JSON,el cual permite establecer un servicio que podrá ser utilizado por cada uno de los proyectos del CEGEL. Esta descripción se representa en el mapa conceptual de la Figura X.

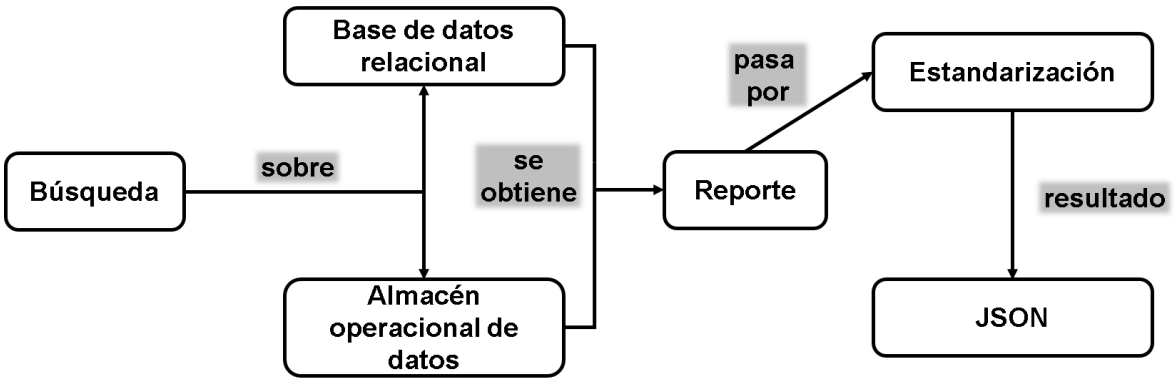


Figura 1: mapa conceptual asociado al dominio de la investigación

Fuente: elaboración propia

1.2. Sistemas homólogos

Un sistema informático es homólogo a otro cuando es equivalente, análogo, parecido, comparable, semejante, es decir que poseen características referidas a su función o clase. Con el objetivo de definir sus características y la viabilidad de su uso en la solución de la problemática existente se realiza un estudio de los sistemas similares tanto nacionales como internacionales.

En la búsqueda realizada no se encontró un sistema que solo realizara las funcionalidades especificadas, sino componentes asociados a estos que poseen características similares. La mayoría de los componentes son buscadores incluidos como parte de las funcionalidades de los sistemas. A continuación, se hace un análisis de estos componentes.

Biblioteca del Centro de Información Científico Técnica UCI: el Centro de Información Científico Técnica ofrece servicios especializados para la gestión de la información que apoyan los procesos claves de la UCI mediante el uso de las TIC. La biblioteca del Centro de Información Científico Técnica, posee una colección de libros actualizados en formato duro, así como una amplia biblioteca digital disponible para toda la comunidad universitaria (Universidad de lasCiencias Informáticas , 2020).Este sistema cuenta con un componente eficiente de búsquedasde documentos,pues su objeto está bien definido y permite realizarlas con mayor facilidad, pero las limita al no permitir la búsqueda a través de varios conceptos y por tanto no posee filtros para ello. Además, este componente no permite ningún tipo de operador (*and/or*) para realizar las mismas, por lo que reduce su alcance.

Repositorio institucional de la UCI: la comunidad universitaria de la UCI dispone de otros recursos de información tales como: un Repositorio Institucional con la producción científica de los autores de la universidad, siendo un lugar de intercambio de información que potencia o promueve la creación, la difusión y el uso del conocimiento generado por la comunidad académica de la universidad (Universidad de las Ciencias Informáticas, 2020). Este repositorio cuenta con un componente de búsquedas de documentos científicos a partir de indicadores asociados a los mismos. Sin embargo, no permite efectuar búsquedas mediante lacorrelación de varios conceptos de información que pueden existir en la base de datos y por tanto no establece filtros que correspondan al objeto de búsqueda definido. De igual manera, no cuentacon operadores (*and/or*) para realizar dichas búsquedas, lo cual trae consigo que sea más engorroso ejecutar el proceso.

Suite Gestión de ProyectosGESPRO: es una suite orientada a la web que permite la planificación, seguimiento y control de productos en forma de proyectos. Cuenta con herramientas para el apoyo a la toma de decisiones a nivel de proyecto, nivel de entidad ejecutora y nivel gerencial. Se presenta en un modelo de negocios basado en servicios que combinan el uso de una solución informática para la dirección integrada de proyectos y un sistema de formación especializada en gestión de proyectos

(Universidad de las Ciencias Informáticas, 2020). Esta suite cuenta con un componente que gestiona las búsquedas por indicadores con sus tipos de datos, pero no permite correlacionar varios conceptos de información. Además, no admite ningún tipo de operadores (*and/or*) para realizar dichas búsquedas.

Repositorio Institucional de Documentos Universidad de Zaragoza: la publicación en un repositorio digital abierto procura una mejor visibilidad de los fondos bibliográficos y de los resultados de trabajos de investigación realizados por la institución. En este último caso mayor visibilidad para sus autores supone a aumentar el impacto de las publicaciones, mejora el posicionamiento de la institución, añade valor a los documentos (a través de citaciones normalizadas y direcciones permanentes) y le asegura el acceso a largo plazo. El repositorio ZAGUAN de documentos digitales de la Universidad de Zaragoza es la herramienta que permite todo ello. Recopila, gestiona, difunde y preserva la producción científica, docente e institucional de la comunidad universitaria (Zaguan Repositorio Insitucionar de Documentos, 2014).

Este repositorio cuenta con una base de datos documental la cual permite realizar búsquedas dinámicas mediante indicadores. También permite utilizar el operador *and*en todas las búsquedas que se deseen realizar, pero se pierden varias formas de análisis de la información que se pueden establecer. Además, al igual que los componentes analizados anteriormente, este no permite la correlación de varios conceptos de información que pueden existir en la base de datos por lo tanto no se pueden establecer filtros que no correspondan al objeto de búsqueda definido.

Tabla 1:comparación entre sistemas homólogos.

Componente de búsqueda	Tipo de sistema	Búsqueda por indicadores	Tipo de operadores en la búsqueda	Búsquedas por conceptos	Filtro por conceptos
Biblioteca del Centro de Información Científico Técnica UCI	Documental	Si	No posee	No	No
Repositorio Institucional de la UCI	Documental	Si	No posee	No	No
Gestión de Proyectos (GESPRO)	Gestión	Si	No posee	No	No
Repositorio institucional de Documentos Universidad de Zaragoza	Documental	Si	Operador and	No	No

Fuente: elaboración propia

Una vez concluido el análisis de los buscadores incluidos dentro de dichossistemas se tiene que:

- Los componentes para dichas búsquedas se encuentran presentes en tipos de sistemas cuyas bases de datos son documentales.
- Estos sistemas poseen un objeto de búsqueda definido, el cual no es variable en el tiempo.
- Permiten establecer búsquedas con mayor facilidad.

- Definen indicadores con sus tipos de datos para realizar dicha búsqueda.

Por otra parte:

- Son pocos los sistemas de gestión que hacen uso de las búsquedas dinámicas.
- Solo se utiliza el operador de unificación *and* para concatenar la búsqueda. Este es el más sencillo de utilizar por el usuario final, pero se pierden varias formas de análisis de la información que se pueden establecer.
- Como solo poseen un concepto como objeto de búsqueda no permiten correlacionar la información sobre los diferentes conceptos que existen en la base de datos.
- Al no poder establecermás de un concepto o grupo de información, no pueden brindar una mayor semántica del objeto de búsqueda de la información.
- No pueden establecer filtros que no correspondan al objeto de búsqueda definido.

Por todo lo antes expuesto, la autora señala que los componentes analizados no cumplen con el objetivo de la presente investigación, pues ninguno reúne todos los requisitos de ser un componente para búsquedas dinámicas que contribuya al rendimiento y la estandarización de reportes. Además no son aplicables a los proyectos del centro CEGEL.Al tener en cuenta que estas características son fundamentales para dar solución al problema de la investigación, se evidencia la necesidad de desarrollar una nueva solución.

1.3. Metodología de desarrollo desoftware

Podría llegar a ser difícil establecer una forma estándar para definir el concepto de metodología de desarrollo de software, no obstante, existen diversas definiciones utilizadas en la actualidad que son de gran importancia. Por ejemplo, el concepto planteado por Pressman que surgiere que “se puede definir metodología de desarrollo de software como un conjunto de procedimientos, técnicas, herramientas y un soporte documental para el desarrollo de productos de software” (Pressman, 2010).

Por tanto, la metodología de desarrollo de software es un proceso en el cual un determinado proyecto de software es culminado o desarrollado mediante procesos o etapas bien delimitadas. También se puede referir como un marco de trabajo que es usado para estructurar, planear y controlar el proceso de desarrollo.

Como metodología de desarrollo de software se utilizó el Proceso Unificado Ágil (AUP por las siglas en inglés de *Agile UnifiedProcess*) variación para la UCI, apoyándose en los lineamientos de la infraestructura productiva de la universidad. Esta explica un modo más simple y fácil de entender la forma de desarrollar aplicaciones de software de negocio usando técnicas ágiles y conceptos que aún se mantienen válidos en la metodología Proceso Unificado Ágil (RUP por sus siglas en inglés RationalUnifiedProcess).

Metodología AUP variación para la UCI

A partir de la utilización de varias metodologías en los proyectos productivos de los centros de desarrollos de la universidad, que no erradicaban los problemas existentes, se decide escoger una metodología para ser adaptada a lo que ya la universidadhabía estado proponiendo como ciclo de vida de los proyectos, sin alejarse de lo que hasta el momento se había trabajado e introduciendo la menor cantidad de cambios posibles. De esta forma surge la metodología AUP variación para la UCI, con el objetivo de estandarizar los procesos que se llevan a cabo en los centros de producción de la UCI.(Metodologia de desarrollo para la Actividad, 2020)

Esta metodología cuenta con las fases de Inicio, Ejecución y Cierre. Además, define las disciplinas para la fase de Ejecución: Modelado de negocio, Requisitos, Análisis y diseño, Implementación, Pruebas internas, Pruebas de liberación y Pruebas de aceptación. Para la disciplina de Requisitos define 4 escenarios:

1. Proyectos que modelen el negocio con CUN (Caso de Uso del Negocio) solo pueden modelar el sistema con CUS (Casos de Uso del Sistema).

2. Proyectos que modelen el negocio con MC (Modelo Conceptual) solo pueden modelar el sistema con CUS (Casos de Uso del Sistema).

3. Proyectos que modelen el negocio con DPN (Descripción de Proceso de Negocio) solo pueden modelar el sistema con DRP (Descripción de Requisitos por Proceso).

4. Proyectos que no modelen negocio solo pueden modelar el sistema con HU (Historias de usuario)(Rodríguez Sánchez, 2015).

La presente investigación se enfocará en el Escenario No 4: el cual plantea que los Proyectos que no modelen negocio solo pueden modelar el sistema con Historia de Usuario (HU)(Metodologia de desarrollo para la Actividad, 2020).

El equipo de desarrollo se guiará por el Escenario No 4, ya que este solo se aplica a los proyectos que hayan evaluado el negocio el cual se va a informatizar y como resultado se obtenga un negocio muy bien definido. Por otro lado, el cliente debe estar siempre acompañando al equipo de desarrollo para acordar los detalles de los requisitos y así poder implementarlos, probarlos y validarlos. Además, se recomienda en proyectos no muy extensos, ya que una HU no debe poseer demasiada información.Dichas características se ajustan a las que posee la presente investigación.

1.4. Arquitectura de software

Al diseñar una arquitectura de software se crean y representan componentes que interactúan entre sí, con responsabilidades específicas y se organizan de forma tal que se logren los requerimientos establecidos. Se puede partir con patrones de soluciones probados que se conocen con el nombre de estilos arquitectónicos, patrones arquitectónicos y patrones de diseño (Rodríguezy otros, 2016).

Los propios autores señalan que la arquitectura de software es la organización fundamental de un sistema enmarcada en sus componentes, las relaciones entre ellos, el ambiente, los principios que orientan su diseño y evolución.

Estilos arquitectónicos

Un estilo arquitectónico es un conjunto coordinado de restricciones arquitectónicas que regula las

funciones/características de los elementos arquitectónicos y las relaciones permitidas entre estos dentro de cualquier arquitectura que se adapte a ese estilo. (Rodríguez y otros, 2016)

Los estilos se clasifican de la siguiente forma:

Estilo de Flujo de datos

- Tuberías yfiltros

Estilos Centrados en datos

- Arquitecturas de pizarra orepositorio

Estilos de Llamada y Retorno

- Modelo-Vista-Controlador (MVC)
- Arquitecturas encapas
- Arquitecturas orientadas aobjetos
- Arquitecturas basadas encomponentes

Estilos de Código móvil

- Arquitectura de máquinasvirtuales

Estilos heterogéneos

- Sistemas de control deprocesos
- Arquitecturas basadas enatributos

Estilos *Peer-to-Peer*

- Arquitecturas basadas eneventos
- Arquitecturas orientadas aservicios
- Arquitecturas basadas enrecursos

Para la realización de la presente investigación se asume el estilo de Llamada y Retorno y dentro de este, arquitectura en capas.

La programación por capas es un estilo de programación en la que el objetivo primordial es la separación de la lógica de negocios de la lógica de diseño, un ejemplo básico de esto es separar la capa de datos de la capa de presentación al usuario.

La arquitectura basada en capas se enfoca en la distribución de roles y responsabilidades de forma jerárquica proveyendo una forma muy efectiva de separación de responsabilidades. El rol indica el modo y tipo de interacción con otras capas, y la responsabilidad indica la funcionalidad que está siendo desarrollada(Pelaez, 2019).

1.5. HerramientasCASE

Las herramientas de Ingeniería de Software Asistidas por Computadoras CASE (por sus siglas en inglés: *ComputerAided Software Engineering*) son diversas aplicaciones informáticas o programas informáticos destinadas a aumentar la productividad en el desarrollo de software reduciendo el costo de las mismas en términos de tiempo y de dinero. A continuación, se enuncian los beneficios de dichas herramientas: (Valderrama, 2016)

- Permiten la aplicación práctica de metodologías estructuradas, las cuales al ser realizadas con una herramienta consiguen agilizar eltrabajo.
- Facilitan la realización de prototipos y el desarrollo conjunto deaplicaciones.
- Simplifican el mantenimiento de losprogramas.
- Mejoran y estandarizar ladocumentación.
- Aumentan la portabilidad de lasaplicaciones.
- Facilitan la reutilización de componentessoftware.
- Permiten un desarrollo y un refinamiento visual de las aplicaciones, mediante la utilización de gráficos.

1.5.1. Visual Paradigmfor UML

Visual Paradigm es una herramienta de diseño y administración poderosa, así como multiplataforma. La misma cuenta con los medios necesarios para extender sus funcionalidades, pues da soporte a las extensiones de aplicación. Provee de forma libre una interfaz de programación, que permite a los desarrolladores implementar y reutilizar clases e interfaces y desarrollar funciones agregadas que son útiles para el desarrollo de software. Facilita una excelente interoperabilidad con otras herramientas CASE y la mayoría de los Entorno de Desarrollo Integrado IDE (por sus siglas en inglés *de IntegratedDevelopmentEnvironment)*líderes que sobresalen en todo su proceso de desarrollo(Visual Paradigm, 2020).

Las ventajas que proporciona *Visual Paradigmfor UML* son(Visual Paradigm, 2020):

- Dibujo: Facilita el modelado de UML, ya que proporciona herramientas específicas para ello. Esto también permite la estandarización de la documentación, ya que la misma se ajusta al estándar soportado por laherramienta.
- Corrección sintáctica:controla que el modelado con UML seacorrecto.
- Coherencia entre diagramas: al disponer de un repositorio común, es posible visualizar el mismo elemento en varios diagramas, evitandoduplicidades.
- Integración con otras aplicaciones: permite integrarse con otras aplicaciones, como herramientas ofimáticas, lo cual aumenta laproductividad.
- Trabajo multiusuario:permite el trabajo en grupo, proporcionando herramientas de compartición detrabajo.
- Reutilización: facilita la reutilización, ya que es una herramienta centralizada donde se encuentran los modelos utilizados para otrosproyectos.
- Generación de código: permite generar código de forma automática, reduciendo los tiempos de desarrollo y evitando errores en la codificación delsoftware.
- Generación de informes: permite generar diversos informes a partir de la información introducida en la herramienta.

Para la realización del componente, se decide utilizar la herramienta Visual Paradigmfor UML en su versión 8.0, ya que posee funcionalidades que favorecen las diferentes etapas por la que transita el

producto de software durante su desarrollo. Además, por la vasta experiencia del equipo de desarrollo en su empleo, lo que posibilita reducir el tiempo en el proceso de desarrollo del software ya que no hay necesidad de capacitar a los desarrolladores. También se puede argumentar que, la UCI posee una licencia académica para su uso en el entorno productivo.

1.6. Lenguaje de programación

Un lenguaje de programación es un lenguaje formal diseñado para realizar procesos que pueden ser llevados a cabo por máquinas como las computadoras. Pueden usarse para crear programas que controlen el comportamiento físico y lógico de una máquina, para expresar algoritmos con precisión, o como modo de comunicación humana. Está formado por un conjunto de símbolos y reglas sintácticas y semánticas que definen su estructura y el significado de sus elementos y expresiones.Al proceso por el cual se escribe, se prueba, se depura, se compila (de ser necesario) y se mantiene el código fuente de un programa informático se le llama programación(Conogasi.org, 2018).

Un lenguaje de programación proporciona los elementos de lenguaje necesarios que son los requeridos para traducir los pasos de un pseudocódigo en formato comprensible para la máquina. En otras palabras, el lenguaje de programación proporciona el puente para hacer la transición de pseudocódigo legible por humano a instrucciones de código legible por máquina (Conogasi.org, 2018).

Para el desarrollo de la propuesta de solución como resultado de la investigación, se decide utilizar como lenguaje de programación Java, como lenguaje de programación ligero interpretado JavaScript y el Lenguaje de consulta estructurada SQL como lenguaje de dominio específico, debido a la experiencia adquirida por el equipo de desarrollode sistemas de escritorio el cual realizó un amplio análisis y además por todas las características descritas en el siguiente epígrafe.

1.6.1. Java

Java es un tipo de lenguaje de programación y una plataforma informática. Se constituye como unlenguaje orientado a objetos, su intención es permitir que los desarrolladores de aplicaciones escriban el programa una sola vez y lo ejecuten en cualquier dispositivo(Rock, 2019).

Las principales características del lenguaje Java son(Rock, 2019):

- Es simple:Javaofrece la funcionalidad de un lenguaje potente, derivado de C y C++.
- Orientado a objetos: el enfoque orientado a objetoses uno de los estilos de programación más populares. Permite diseñar el software de forma que los distintos tipos de datos que se usen estén unidos a sus operaciones.
- Es distribuido**:** Java proporciona una gran biblioteca estándar y herramientas para que los programas puedan ser distribuidos.
- Independiente a la plataforma**:**significa que programas escritos en el lenguaje Java pueden ejecutarse en cualquier tipo de sistema operativo, lo que lo hace portable.
- Recolector de basura:cuando no hay referencias localizadas a un objeto, el recolector de basura de Java borra dicho objeto, liberando así la memoria que ocupaba. Esto previene posibles fugas de memoria.
- Es seguro y sólido: proporcionando una plataforma segura para desarrollar y ejecutar aplicaciones que, administra automáticamente la memoria, provee canales de comunicación segura protegiendo la privacidad de los datos y, al tener una sintaxis rigurosa evita que se quiebre el código.
- Es multihilo:Java logra llevar a cabo varias tareas simultáneamente dentro del mismo programa. Esto permite mejorar el rendimiento y la velocidad de ejecución.

1.6.2. JavaScript

JavaScript (JS) es un lenguaje ligero e interpretado, orientado a objetos con funciones de primera clase, más conocido como el lenguaje de script para páginas web, pero también usado en muchos entornos sin navegador, tales como Node.js o Apache CouchDB. Es un lenguaje script multi-paradigma[3] basado en prototipos, dinámico, soporta estilos de programación funcional, orientada a objetos e imperativa (W-ictea, 2020).

Las principales características de JavaScript son(W-ictea, 2020):

- Imperativo y estructurado: es compatible con gran parte de la estructura de programación de C

[3]Lenguaje de programación que soporta más de un paradigma de programación, o sea, permite crear programas usando más de un estilo de programación.

(por ejemplo, *sentencias if, bucles for, sentencias switch*). Con una salvedad, en parte: en C, el ámbito de las variables alcanza al bloque en el cual fueron definidas; sin embargo, JavaScript no es compatible con esto, puesto que el ámbito de las variables es el que presenta la función en la cual fueron declaradas.

- Dinámico: está formado casi en su totalidad por objetos. Los objetos en JavaScript son*arrays* (arreglos) asociativos, mejorados con la inclusión de prototipos.
- JavaScript incluye la función *eval* que permite evaluar expresiones obtenidas como cadenas de texto o como valor numérico en tiempo de ejecución. Por ello se recomienda que *eval*sea utilizado con precaución y que se opte por utilizar la función *JSON.parse()* en la medida de lo posible, pues resulta mucho más segura.
- Funcional: a las funciones se les suele llamar ciudadanos de primera clase; son objetos en sí mismos. Como tal, poseen propiedades y métodos. Una función anidada es una función definida dentro de otra. Esta es creada cada vez que la función externa es invocada.

1.6.3. Lenguaje de Consulta Estructurada

El Lenguaje de consulta estructurada SQL (por sus siglas en inglés *StructuredQueryLanguage*) es un lenguaje de programación estándar e interactivo para la obtención de información desde una base de datos y para actualizarla. Muchos productos de bases de datos soportan SQL con extensiones propietarias al lenguaje estándar. Las consultas toman la forma de un lenguaje de comandos que permite seleccionar, insertar, actualizar, averiguar la ubicación de los datos, y más. También hay una interfaz de programación(Rouse, 2015).

Las principales características de SQL son(Rouse, 2015):

- Lenguaje de definición de datos: el DDL[4] de SQL proporciona comandos para la definición de esquemas de relación,borrado de relaciones y modificaciones de los esquemas de relación.

- Lenguaje interactivo de manipulación de datos: el DML[5] de SQL incluye lenguaje de consultas basado tanto en álgebra relacional como en cálculo relacional de tuplas.

[4] Lenguaje de definición de datos

[5] Lenguaje de manipulación de datos

- Integridad: el DDL de SQL incluye comandos para especificar las restricciones de integridad que debe cumplir los datos almacenados en la base de datos.
- Definición de vistas: el DDL incluye comando para definir las vistas.
- Control de transacción: SQL tiene comandos para especificar el comienzo y el final de una transacción.
- SQL incorporado y dinámico: se puede incorporar instrucciones de SQL en leguajes de programación como: C++,C, Java,PHP, Cobol, Pascal.
- Autorización: el DDL incluye comandos para especificar los derechos de acceso a las relaciones y las vistas.

1.7. Entorno de desarrollointegrado

Entorno de desarrollo integrado IDE, es un programa informático compuesto por un conjunto de herramientas de programación. Puede dedicarse en exclusiva a un solo lenguaje de programación o bien puede utilizarse para varios. Ha sido empaquetado como un programa de aplicación, es decir, que consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica(Aprendiendoarduino, 2017).

Existen varios IDE de múltiples lenguajes tales como Eclipse, Oracle JDeveloper, Codenvy, Microsoft Visual Studio y el NetBeans. Para el desarrollo del componente para búsquedas dinámicas se define por parte del equipo de desarrollo utilizar NetBeans en su versión 8.2 como IDE, para la selección los desarrolladores tuvieron en cuenta características que se mencionan en el próximo subepígrafe.

1.7.1. Netbeans 8.2

Es un entorno de desarrollo integrado de código abierto dedicado a proporcionar productos de desarrollo de software sólidos que atienden las necesidades de los desarrolladores, usuarios y las empresas que confían en NetBeans como base para sus productos. Para permitir desarrollar estos productos de forma rápida, eficiente y fácil al aprovechar las fortalezas de la plataforma Java y otros estándares relevantes de la industria. El IDE de NetBeans proporciona soporte para varios lenguajes (PHP, JavaFX, C / C++, *JavaScript*, entre otros), así como varios marcos de trabajo(Oracle, 2018).

Las principales características de Netbeans son (NetBeans.org, 2020):

- Hecho principalmente para el lenguaje de programación Java. Existe además un número importante de módulos para extenderlo.
- Es un producto libre y gratuito sin restricciones de uso.
- Permite que las aplicaciones seandesarrolladas a partir de un conjunto de componentes de software llamados módulos. Un módulo es un archivo Java que contiene clases de java escritas para interactuar con las API de NetBeans.
- Las aplicaciones construidas a partir de módulos pueden ser extendidas agregándole otros nuevos.
- Debido a que los módulos pueden ser desarrollados independientemente, las aplicaciones basadas en la plataforma NetBeans pueden ser extendidas fácilmente por otros desarrolladores de software(Domínguez, 2018)

Para el desarrollo del componente se decide la utilización de NetBeans en su versión 8.2, al tener en cuanta todo lo antes descrito y que, además, proporciona soporte para el cliente de control de versiones seleccionado por parte del equipo de arquitectura del proyecto, lo cual permite realizar tareas de control de versiones directamente desde el proyecto dentro del IDE.

1.8. Sistema de Gestor de Base de datos

Un Sistema Gestor de Base de Datos (SGBD) es un sistema que permite la creación, gestión y administración de bases de datos, así como la elección y manejo de las estructuras necesarias para el almacenamiento y búsqueda de la información del modo más eficiente posible (Empresariales, 2016). En el caso de esta investigación se decide utilizar por parte del equipo de desarrollo PostgreSQL11.0.

1.8.1. PostgreSQL 11.0

Es un poderoso sistema de base de datos relacional de objetos de código abierto. Tiene una arquitectura comprobada que le ha valido una sólida reputación de fiabilidad, integridad de datos y corrección. Sus principales características son: llaves ajenas o llaves foráneas, disparadores, vistas, integridad transaccional, acceso concurrente multiversión (no se bloquean las tablas, ni siquiera las filas, cuando un proceso escribe), capacidad de albergar programas en el servidor en varios lenguajes. Herencia de tablas,

tipos de datos y operaciones geométricas. A continuación, se muestran algunas de sus ventajas(PostgreSQL, 2018):

- Es altamente escalable en cuanto a cantidad de datos y usuarios.
- Es multiplataforma.
- Ofrece una documentación bien organizada, pública y libre.
- Presenta conectores de datos foráneos que permiten añadir y consultar fuentes de datos externas desde PostgreSQL.
- Posee soporte al formato JSON e introduce varias mejoras a este formato con la utilización del estándar JSONB.

Para el desarrollo del componente se decide la utilización de PostgreSQL en su versión 11.0, ya que es la seleccionada por el equipo de arquitectura del proyecto al que pertenece la propuesta de solución.

1.9. Control de versionesGit

Es un sistema de control de versiones gratuito, de código abierto, y se distribuye, o sea, todos los desarrolladores tienen el historial completo de su repositorio de códigos a nivel local. Esto hace que el clon inicial del repositorio sea más lento, pero las operaciones subsiguientes, como cometer, culpar, difuminar, fusionar y registrar mucho más rápido. Incluye las funcionalidades de crear ramas y *merges*[6], además de reescribir historiales de repositorios, lo cual ha dado como resultado muchas herramientas innovadoras y eficaces. Básicamente, funciona del siguiente modo (git, 2020):

1. Crea un "repositorio" (proyecto) con una herramienta de alojamiento de Git.
2. Copia (o clona) el repositorio a una máquina local.
3. Añade un archivo al repositorio local y confirma (*commit*) los cambios.
4. Envía (*push*) los cambios a la rama principal.
5. Realiza cambios en un archivo con una herramienta de alojamiento de Git y se confirman.
6. Extrae (*pull*) los cambios a una máquina local.
7. Crea una rama (*branch*, versión), se realiza algún cambio y se confirma.
8. Fusiona (*merge*) a una rama con la rama principal(Murua, y otros, 2019).

[6] Hacer fusiones entre diferentes ramas

Para el desarrollo del componente de configuración de búsquedas dinámicas se decide utilizar Git en su versión 2.17, ya que es la seleccionada por el equipo de arquitectura del proyecto al que pertenece la propuesta de solución.

1.10. Conclusionesparciales

- La sistematización del marco teórico de la investigación y la definición de los principales conceptos asociados al dominio de la misma y las relaciones entre estos, permite alcanzar una mayor comprensión de la propuesta de solución.
- El análisis de los sistemas homólogos, permite identificar las tendencias en cuanto al desarrollo de herramientas informáticas para búsquedas dinámicas, e identificar las deficiencias que impiden sean utilizadas para los proyectos de CEGEL.
- La caracterización de la metodología, herramientas y lenguajes de desarrollo permite definir el entorno de desarrollo para la propuesta de solución en correspondencia con las seleccionadas por el equipo de arquitectura del proyecto al que pertenece la misma.

CAPÍTULO 2: ANÁLISIS, DISEÑO E IMPLEMENTACIÓN DEL COMPONENTE PARA BÚSQUEDAS DINÁMICAS

Introducción

El presente capítulo describe la propuesta de solución como resultado de la presente investigación. Tomando como guía la metodología seleccionada, se obtienen en la disciplina de Requisitos los requisitos funcionales y no funcionales como una primera aproximación de las características que debe cumplir la propuesta de solución. Por otra parte, se analiza la arquitectura base sobre la cual se implementa el sistema, detallando cada uno de sus componentes. Se obtienen los artefactos correspondientes a la disciplina de Análisis y diseño, aplicando los patrones de diseño definidos como buenas prácticas durante el ciclo de desarrollo del software, así como el patrón arquitectónico que da soporte a la propuesta de solución. Finalmente, quedan definidos los estándares de codificación a utilizar como buenas prácticas durante la implementación de dicha propuesta.

2.1. Descripción de la propuesta de solución

La propuesta de solución de este trabajo consiste en el análisis, diseño, implementación y prueba del componente de configuración del modelo de búsquedas dinámicas contribuyendo a su funcionalidad y facilidad de uso.

Dicha solución corresponde a una aplicación de escritorio con las funcionalidades necesarias para la gestión de las conexiones de acceso a los nodos de base de datos correspondiente a cada proyecto. Además, contiene la implantación del modelo y la gestión de los valores para lograr el correcto funcionamiento del servicio de búsqueda dinámica. En este sentido el componente tiene una estructura de 10 nomencladores, de los cuales 5 deben ser gestionados desde el propio componente:

- Tipo de conectores
- Tipo de datos
- Tipo de operadores
- Tipo de operadores por atributo

- Tipo de operadores por dato

Para cada uno, el sistema debe permitir captar los datos necesarios, los cuales posteriormente pueden ser modificados en caso que el usuario lo crea pertinente. Teniendo en cuenta esto, se listará toda la nomenclatura de todos los datos o simplemente se podrá aplicar algún criterio de búsqueda para un filtrado más eficiente. En caso de que algún nomenclador no se desee utilizar en un período de tiempo, el mismo puede ser eliminado. En este sentido solo se podrá desactivar el mismo para evitar incongruencias en la búsqueda de información. Una vez captado cada uno de estos datos en el componente, los mismos pueden utilizarse indistintamente por el servicio de búsqueda dinámica.

Por último, la propuesta comprende el desarrollo de un servicio que permitirá las búsquedas dinámicas sobre los datos contenidos en el modelo previamente cargado y correspondiente al proyecto en cuestión. Además, este servicio define una estructura JSON, tanto para recibir los elementos necesarios para simplificar el objeto de búsqueda como la respuesta con la información obtenida.

2.2. Requisitos de software

Un requisito es la condición o capacidad que un usuario necesita para poder resolver un problema o lograr un objetivo. También se define como la condición o capacidad que debe exhibir o poseer un sistema para satisfacer un contrato, estándar, especificación, u otra documentación formalmente impuesta (Sánchez, 2018).

La tarea principal de la disciplina Requisitos es desarrollar un modelo del sistema que se va a construir. Esta disciplina comprende la administración y gestión de los requisitos funcionales y no funcionales del producto (Rodríguez Sánchez, 2015). El requisito de software consiste en una definición detallada formal de una función del sistema(Sommerville, 2011).

2.2.1. Requisitos funcionales

Los requisitos funcionales (RF) de un sistema, son aquellos que describen cualquier actividad que este deba realizar, en otras palabras, el comportamiento o función particular de un sistema cuando se cumplen ciertas condiciones. Por lo general, estos deben incluir funciones desempeñadas por pantallas específicas, descripciones de los flujos de trabajo a ser desempeñados por el sistema y otros requerimientos de negocio (Pressman, 2010).

El equipo de desarrollo luego de haber realizado un profundo análisis identificó 45 requisitos funcionales, los cuales se describen a continuación en la siguiente tabla.

Tabla 2: requisitos funcionales.

No.	Nombre	Descripción
RF/1	Adicionar conexión	El sistema permite al usuario adicionar una nueva conexión de base de datos.
RF/2	Modificar conexión	El sistema permite al usuario modificaruna conexión seleccionada.
RF/3	Eliminar conexión	El sistema permite al usuario eliminaruna conexión seleccionada.
RF/4	Listar conexiones	El sistema permite al usuario listarlasconexiones.
RF/5	Adicionar modelo	El sistema permite al usuario adicionarun nuevo modelo.
RF/6	Eliminar modelo	El sistema permite al usuario eliminar modelos.
RF/7	Listar modelos	El sistema permite al usuario listar modelos.
RF/8	Adicionar concepto	El sistema permite al usuarioadicionar nuevos conceptos de búsqueda.
RF/9	Modificar concepto	El sistema permite al usuariomodificar el concepto seleccionado.
RF/10	Eliminar concepto	El sistema permite al usuarioeliminar conceptos seleccionados.
RF/11	Listar conceptos	El sistema permite al usuariolistar conceptos.

RF/12	Adicionar atributo	El sistema permite al usuarioadicionar nuevos atributos para la búsqueda.
RF/13	Modificar atributo	El sistema permite al usuariomodificarel atributo seleccionado.
RF/14	Eliminar atributo	El sistema permite al usuarioeliminar atributos seleccionados.
RF/15	Listar atributos	El sistema permite al usuario listar atributos para la búsqueda.
RF/16	Adicionar operador	El sistema permite al usuario adicionar nuevosoperadores para la búsqueda.
RF/17	Modificar operador	El sistema permite al usuario modificar el operador seleccionado.
RF/18	Eliminar operador	El sistema permite al usuario eliminar operadores seleccionados.
RF/19	Listar operadores	El sistema permite al usuario listar los operadores existentes.
RF/20	Adicionar conector	El sistema permite al usuario adicionar nuevos conectores para la búsqueda.
RF/21	Modificar conector	El sistema permite al usuario modificar el conector seleccionado.
RF/22	Eliminar conector	El sistema permite al usuario eliminar el conector seleccionado.
RF/23	Listar conector	El sistema permite al usuario listar los conectores existentes.
RF/24	Adicionartipos de datos	El sistema permite al usuario

		adicionar nuevos tipos de datos
RF/25	Modificar tipos de datos	El sistema permite al usuario modificar los tipos de datos seleccionados.
RF/26	Eliminartipos de datos	El sistema permite al usuario eliminar los tipos de datos seleccionados.
RF/27	Listartipos de datos	El sistema permite al usuario listar los tipos de datos.
RF/28	Adicionar indicadores dado un atributo	El sistema permite al usuario adicionar los indicadores a los atributos que los definan.
RF/29	Modificar indicadores dado un atributo	El sistema permite al usuario modificar los indicadores a los atributos que los definan.
RF/30	Eliminar indicadores dado un atributo	El sistema permite al usuario eliminar los indicadores a los atributos que los definan.
RF/31	Listar indicadores dado un atributo	El sistema permite al usuario listar los indicadores de los atributos.
RF/32	Adicionar operadores dado un atributo	El sistema permite al usuario adicionar los operadores a los atributos según el tipo de dato que contenga.
RF/33	Modificar operadores dado un atributo	El sistema permite al usuario modificar los operadores a los atributos según el tipo de dato que contenga.
RF/34	Eliminar operadores dado un atributo	El sistema permite al usuario eliminar los operadores según los atributos.
RF/35	Listaroperadores dado un	El sistema permite al usuario listar los

	atributo	operadores según los atributos.
RF/36	Adicionartipos de datosdado un operador	El sistema permite al usuario adicionar nuevos tipos de datos según los operadores.
RF/37	Modificartipos de datosdado un operador	El sistema permite al usuario Modificar los tipos de datos según los operadores.
RF/38	Eliminartipos de datos dado un operador	El sistema permite al usuario eliminar los tipos de datos seleccionados según los operadores.
RF/39	Listartipos de datos dado un operador	El sistema permite al usuario listar los tipos de datos según los operadores.
RF/40	Asignar operador por atributo	El sistema permite asignar cada operador teniendo en cuenta el atributo al que se le otorga.
RF/41	Asignar operador por tipo de dato	El sistema permite asignar cada operador teniendo en cuenta el tipo de dato que posee.
RF/42	Crear vista materializada	El sistema permite crear vista materializada.
RF/43	Crear filtros dinámicos	El sistema permite crear nuevos filtros dinámicos que le posibilitan al usuario la búsqueda.
RF/44	Eliminar filtros dinámicos	El sistema permite eliminar los filtros que no son necesarios para la realización de la búsqueda.
RF/45	Crear servicios	El sistema permitirá crear nuevos servicios.

Fuente:elaboración propia

2.2.2. Requisitos no funcionales

Los requisitos no funcionales (RnF) representan características generales y restricciones de la aplicación o sistema que se esté desarrollando. Suelen presentar dificultades en su definición dado que su conformidad o no conformidad podría ser sujeto de libre interpretación, por lo cual es recomendable acompañar su definición con criterios de aceptación que se puedan medir (Pressman, 2010).

El equipo de arquitectura del proyecto definió los requisitos no funcionales del componente para búsquedas dinámicas mediante estructuras JSON en PostgreSQLpara los proyectos de CEGEL. Seguidamente se presenta un resumen de los mismos teniendo en cuenta que forman parte de la solución del presente trabajo de diploma.

Usabilidad

RnF 1. Requisito de usabilidad 1

En el sistema se deben visualizar todos los mensajes en idioma español. La tipografía debe ser uniforme, de un tamaño adecuado.

RnF 2. Requisito de usabilidad 2

El sistema debe facilitar la interacción con el usuario, posibilitando la utilización de combinaciones de teclas, podrán utilizarse los campos de selección en la interfaz en los casos que sea posible y se agruparán los vínculos y botones por grupos funcionales siempre que se cumpla con las pautas de diseño de las interfaces.

RnF 3. Requisito de usabilidad 3

El sistema debe ofrecer una interfaz amigable, fácil de operar. Igualmente tiene que mantener la línea de diseño establecida la cual mantiene la uniformidad y representatividad de la solución. Las interfaces deben poseer un diseño sencillo, con pocas entradas, permitiendo un balance adecuado entre funcionalidad y simplicidad de tal manera que no se haga difícil para los usuarios la utilización del mismo.

Confiabilidad

RnF4. Requisito de confiabilidad 1

El sistema debe permitir la visualización de la información necesaria para advertir cualquier excepción en el mismo. La información detallada será almacenada en registros de los diferentes componentes de la aplicación.

Eficiencia

RnF 5. Requisito de eficiencia 1

Los procesos del sistema que se implementan con transacciones donde se modifica la base de datos, deben tener tiempos de respuesta no mayores a los 3 segundos. En el caso de la obtención de información a través de transacciones que involucran consultas a la base de datos, el tiempo está dado por el volumen de la misma, por lo cual se implementará en todo momento buscando simplificar, al máximo, los datos que se consulten, de manera que la cifra no exceda los 10 segundos.

Restricciones del diseño y la implementación

RnF 6. Requisito de restricción de diseño e implementación 1

Se utilizará la herramienta CASE Visual Paradigm teniendo en cuenta sus ventajas para modelar los diferentes artefactos que se obtienen en los flujos de trabajo y sus diferentes fases. Las restricciones propias del diseño radican en las pautas que se establecerán, así como las diferentes relaciones que se formen durante el modelado del sistema.

2.3. Historias de usuario

Las historias de usuario, son pequeñas descripciones de los requerimientos de un cliente.Al redactar las historias de usuario se debe tener en cuenta describir el Rol, la funcionalidad y el resultado esperado en una frase corta. Su función principal es identificar problemas percibidos, proponer soluciones y estimar el esfuerzo que requieren implementar las ideas propuestas (tenstep, 2016). En la presente solución se definieron 45 historias de usuario de las cuales se presenta, por su alta complejidad, la correspondiente al

Adicionar Conexión.El resto se pueden consultar en el documento creado por la autora de la presente investigación:

Tabla 3: historia de usuario Adicionar conexión.

Número: RF/1	**Nombre del requisito:** Adicionar conexión.	
Programador: Milena Bárbara Mena González		**Iteración Asignada: 1**
Prioridad: alta		**Tiempo Estimado:** 24
Riesgo en Desarrollo N/A		**Tiempo Real:** 22 horas

Descripción: Funcionalidad que permite adicionar una nueva conexión a partir de la selección del gestor de base de dato por parte del usuario.

Campos:

- **Nombre de la Conexión:** campo de texto o campo numérico donde el usuario establece el nombre de la nueva conexión. Es obligatorio.
- **Host:** campo numérico donde se registra el Ip de la base de dato a la que el usuario decidió conectarse. Es obligatorio.
- **Puerto:** campo numérico donde se registra el puerto de la base de dato a la que el usuario decidió conectarse. Es obligatorio.
- **Datos iniciales:** campo de texto donde se registra el nombre de la base de datos. Es obligatorio
- **Nombre del usuario:** campo de texto donde se registra el nombre del usuario que desea accedera la base de datos . Es obligatorio

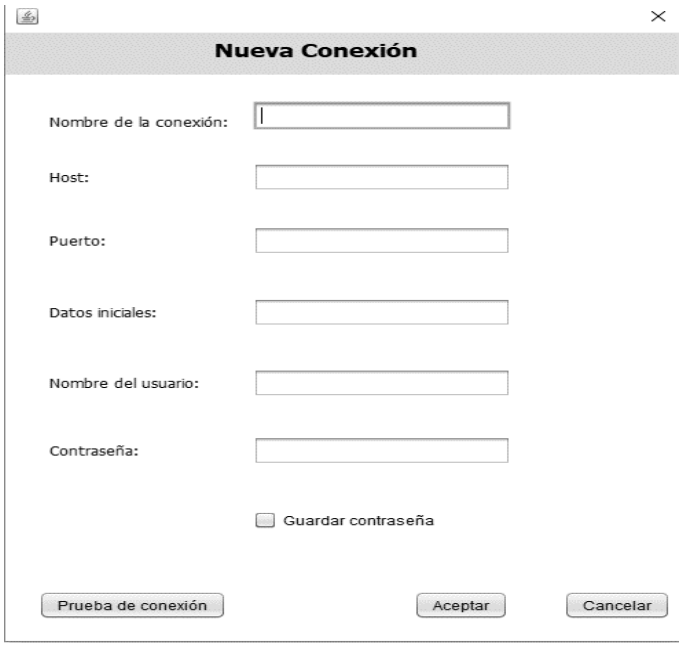
- **Contraseña:** campo de texto o campo numérico donde el usuario establece la contraseña para acceder a la base de dato.
- **Guardar contraseña:** campo de selección que le permite al usuario guardar o no la contraseña. No es obligatorio

Botones:

- **Cancelar:** permite regresar a la interfaz principal, sin tener en cuenta los cambios realizados.
- **Aceptar:** permite validar y registrar los datos de la interfaz.
- **Probar conexión:** permite realizar una conexión de prueba con la base de datos.

Observaciones: al adicionar nueva conexión se necesita llenar los campos obligatorios.

Prototipo de interfaz:Anexo HU_1



Fuente: elaboración propia

2.4. Arquitectura del sistema

A partir del análisis realizado en el capítulo anterior, se define una arquitectura en capas utilizando el patrón Modelo-Vista-Presentador.A continuación, se presenta una descripción detallada de la misma enmarcándose en los estilos arquitectónicos y los patrones que se utilizan.

La arquitectura en capas reporta, entre otros beneficios, un cierto aislamiento ya que se pueden realizar actualizaciones en el interior de las capas sin que esto afecte el resto del sistema. De esta manera, en caso de ocurrir un error o de que sea necesario realizar algún cambio obligatorio solo se realiza el cambio en la capa correspondiente y el resto del sistema se mantiene intacto.

Para el desarrollo del componente se define una arquitectura en capas basada en el patrón arquitectónico Modelo-Vista-Presentador (MVP).

Este patrón tiene como objetivo separar la interfaz de la lógica de las aplicaciones. Está compuesto básicamente por 3 componentes:

- La Vista está compuesta por las ventanas y controles que conforman la interfaz gráfica de usuario. Se encarga de mostrar la información al usuario y de interactuar con él para forjar ciertas operaciones.

- El Modelo se ocupa de toda la lógica de negocio e ignora la información mostrada al usuario y realiza toda la lógica de las aplicaciones usando las entidades del dominio.

- El Presentador escucha los eventos que se producen en la Vista y ejecuta las acciones necesarias a través del modelo y a su vez lo actualiza. Además, puede tener acceso a las vistas a quien también actualiza a través de las interfaces que la Vista debe implementar.

Se propone una estructura de paquetes que deja bien definido en qué lugar se encuentran las clases correspondientes a la vista, al modelo y al presentador.

De igual forma la arquitectura está dividida en tres niveles físicos por la distribución o localización de las capas lógicas involucradas en la solución, esto quiere decir que: en el capa de Presentación estan presente la capa de Usuario y la capa de Interfaz de usuario, en otro nivel se ubicatodo lo referente al

Negocio con las capas de Entidad de Negocio y de Componente de Negocio, y en otro nivel físico la capa de Acceso a Datos que interactúa con un servidor donde se encuentra el SGBD y la base de datos. La Seguridad se encuetra implícita a través de todas las capas de la arquitectura. A continuación, se muestra como están distribuidas físicamente las capas de la arquitectura:

Figura 2: distribución de la arquitectura base

Fuente: elaboración propia

Capa de presentación:es la responsable de la presentación visual de la aplicación. Esta enviará mensajes a los objetos de esta capa de negocios o intermedia, la cual o bien responderá entonces directamente o mantendrá un diálogo con la capa de la base de datos, la cual proporcionará los datos que se mandarían como respuesta a la capa de presentación.

Se puede decir que es la que se presenta al usuario,llamada también formulario o interfaz de presentación,esta captura los datos del usuario en el formulario e invoca a la capa de negocio, trasmitiéndole los requerimientos del usuario, ya sea de almacenaje, edición,o de ecuperación de la información para la consulta respectiva (Research Gate, 2014).

Capa de negocio: es la responsable del procesamiento que tiene lugar en la aplicación y contendrá los objetos que se corresponden con las entidades de la aplicación, además de que es la que conlleva capacidad de mantenimiento y de reutilización.Contendrá objetos definidos por clases reutilizables que se

pueden utilizar una y otra vez en otras aplicaciones. Estos objetos se suelen llamar objetos de negocios y son los que contienen la gama normal de constructores, métodos para establecer y obtener variables, métodos que llevan a cabo cálculos y métodos,normalmente privados, en comunicación con la capa de la base de datos.

Es en esta capa donde se reciben los requerimientos del usuario y se envían las respuestas tras el proceso, a requerimiento de la capa de presentación. Se denomina capa de negocio o lógica del negocio, es aquí donde se establecen todas las reglas que deben cumplirse.Esta capa interactúa con la capa de presentación para recibir las solicitudes y presentar los resultados, y con la capa de datos, para solicitar al manejador de base de datos que realice una operación de almacenamiento, edición, eliminación, consulta de datos u otra (Research Gate, 2014).

Capa de datos: esta capa se encarga de acceder a los datos, se debe usar la capa de datos para almacenar y recuperar toda la información de sincronización del sistema.Es aquí donde se implementa las conexiones al servidor y la base de datos propiamente dicha, se invocaa los procedimientos almacenados los cuales reciben solicitudes de almacenamiento o recuperación de información desde la capa de negocio.Todas estas capas pueden residir en un único ordenador (no debería ser lo usual), pero es lo más frecuente.

Seguridad:está muy relacionada a las tecnologías que se seleccionen para la implementación, pero tiene aspectos generales que son fundamentales y deben tenerse en cuenta. La seguridad se trata de manera diferente en cada nivel físico, teniendo en cuentas sus especificaciones. En el cliente se garantiza por mecanismos de Autenticación y Autorización, que son los encargados de verificar la validez de un usuario en el sistema y de autorizar el acceso a varias o ninguna de las funcionalidades del sistema, bloqueo de operaciones no deseadas y técnicas de validación de los datos que se introducen en las aplicaciones.(Research Gate, 2014)

En el nivel servidor se aplica por un mecanismo de publicación de los usuarios que están autenticados y autorizados en los clientes y todas las peticiones son verificadas que tengan los usuarios correctos. En el nivel de datos está protegido por el SGBD que ofrece funcionalidades para la protección de los datos, el mismo será accedido únicamente desde el servidor de aplicaciones, es decir, ningún cliente tendrá acceso al servidor de datos, así se puede aislar este de cualquier red pública y obtener mayores niveles de

seguridad. Existen, además, mecanismos de seguridad para la red manteniendo las conexiones seguras dentro de redes seguras(Research Gate, 2014).

2.5. Patrones de diseño

Los patrones de diseño son técnicas para resolver problemas comunes en el desarrollo de software y otros ámbitos referentes al diseño de interacción o interfaces. Para que una solución sea considerada un patrón debe poseer ciertas características. Una de ellas es que debe haber comprobado su efectividad resolviendo problemas similares en ocasiones anteriores. Otra es que debe ser reutilizable, lo que significa que es aplicable a diferentes problemas de diseño en distintas circunstancias (Pressman, 2010).

2.5.1. Patrones GRASP(General Responsibility Assignment Software Patterns)

Controlador: el objetivo de este patrón es asignar la responsabilidad de controlar el flujo de eventos del sistema, a clases específicas, o sea, delega la responsabilidad en otras clases con las que mantiene un modelo de alta cohesión. Este patrón se evidencia en la solución a través de la clase gestora de negocio *GtrReporte.java*, la cual agrupa funcionalidades y toma decisiones con información afín a una entidad determinada (Docsity, 2017).

Experto: este patrón define que la clase experta en información es la que debe llevar la responsabilidad, ya que esta última es la que contiene toda la información para cumplir con la responsabilidad. Este patrón se pone en práctica en la solución en las clases entidades que son las expertas en información y las encargadas de manejar la lógica del negocio que comprenden cada uno de los conceptos que se engloban en la propuesta de solución (Docsity, 2017). En la siguiente figura se muestra la clase entidad *Natributo.java*, la cual es la experta en información para la creación de esta entidad.

```java
@Entity
@Table(name = "natributo")
@XmlRootElement
@NamedQueries({
    @NamedQuery(name = "Natributo.findAll", query = "SELECT n FROM Natributo n")
    , @NamedQuery(name = "Natributo.findByIdnatributo", query = "SELECT n FROM Natributo n WHERE n.idnatributo = :idnatributo")
    , @NamedQuery(name = "Natributo.findByDenominacion", query = "SELECT n FROM Natributo n WHERE n.denominacion = :denominacion")
    , @NamedQuery(name = "Natributo.findByActivo", query = "SELECT n FROM Natributo n WHERE n.activo = :activo")})
public class Natributo implements Serializable {

    private static final long serialVersionUID = 1L;
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Basic(optional = false)
    @Column(name = "idnatributo")
    private Long idnatributo;
    @Basic(optional = false)
    @Column(name = "denominacion")
    private String denominacion;
    @Basic(optional = false)
    @Column(name = "activo")
    private boolean activo;
    @JoinTable(name = "ratributo_operador", joinColumns = {
        @JoinColumn(name = "idnatributo", referencedColumnName = "idnatributo")}, inverseJoinColumns = {
        @JoinColumn(name = "idnoperador", referencedColumnName = "idnoperador")})
    @ManyToMany
    private List<Noperador> noperadorList;
    @OneToMany(cascade = CascadeType.ALL, mappedBy = "idnatributo")
    private List<NatributoClasificacion> natributoClasificacionList;
    @JoinColumn(name = "idnconcepto", referencedColumnName = "idnconcepto")
```


Figura 3:experto en las clases entidades Natributo.java

Fuente: elaboración propia

Bajo acoplamiento: este patrón está estrechamente relacionado con los patrones Experto o Alta Cohesión. Permitirá incitar la asignación de responsabilidades de forma que su colocación no incremente el acoplamiento tanto que produzca los resultados negativos propios de un alto acoplamiento. Al obtener una alta cohesión en el diseño de las clases del sistema, se garantiza un bajo acoplamiento ya que permite tener clases más independientes, donde la realización de cambios sea más sencilla. Este patrón se observa en las clases gestoras de negocio y las de interfaz de presentación (Docsity, 2017).

Alta cohesión: este patrón tiene como propósito asignar responsabilidades de manera que la cohesión siga siendo alta.Este patrón se muestra en las clasesde interfaz de presentacióny las gestoras de negocio garantizando que solo existan en ellas las características y funcionalidades necesarias (Docsity, 2017).

2.5.2. PatronesGoF (*Gang of Four*)

Instancia única o Singlenton: permitirá la creación de una única instancia de las clases que sean necesarias (Docsity, 2017).En la siguiente figura se muestra la clase entidad *Dresumen.java*, una de las cuales utiliza el patrón instancia única.

```
    , @NamedQuery(name = "Dresumen.findByDatos", query = "SELECT d FROM Dresumen d WHERE d.datos = :datos")})
public class Dresumen implements Serializable {

    private static final long serialVersionUID = 1L;
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Basic(optional = false)
    @Column(name = "idresumen")
    private Long idresumen;
    @Basic(optional = false)
    @Column(name = "fechacreacion")
    @Temporal(TemporalType.DATE)
    private Date fechacreacion;
    @Basic(optional = false)
    @Column(name = "datos")
    private String datos;
    @JoinTable(name = "rresumen_asociado", joinColumns = {
        @JoinColumn(name = "idresumen_asociado", referencedColumnName = "idresumen")}, inverseJoinColumns = {
        @JoinColumn(name = "idresumen", referencedColumnName = "idresumen")})
    @ManyToMany
    private List<Dresumen> dresumenList;
    @ManyToMany(mappedBy = "dresumenList")
    private List<Dresumen> dresumenList1;
    @JoinColumn(name = "idnconcepto", referencedColumnName = "idnconcepto")
    @ManyToOne(optional = false)
    private Nconcepto idnconcepto;

    public Dresumen() {
    }
```


Figura 4: clase entidad Dresumen.java

Fuente: elaboración propia

Decorador: el principal objetivo de este patrón es añadir responsabilidades a objetos concretos de manera dinámica y transparente sin afectar a otros objetos. Este patrón brinda más flexibilidad que la herencia estática y evita que las clases más altas en la jerarquía estén demasiado cargadas de funcionalidad y sean complejas. Este patrón se evidencia en la clase *Reporte.rdato_operador.java*(Docsity, 2017).

2.6. Diagrama de clases dediseño

Un diagrama de clases es un tipo de diagrama estático que describe la estructura de un sistema, muestra los elementos que lo forman (clases e interfaces) y las relaciones entre ellos (asociaciones). Las clases se representan mediante una caja subdividida en tres partes: en la superior se muestra el nombre, en el medio los atributos y en la inferior las operaciones o métodos.

Las asociaciones entre dos clases se representan mediante una línea que las une y esta puede tener una serie de elementos gráficos que expresan características particulares de la asociación (Pressman, 2010).

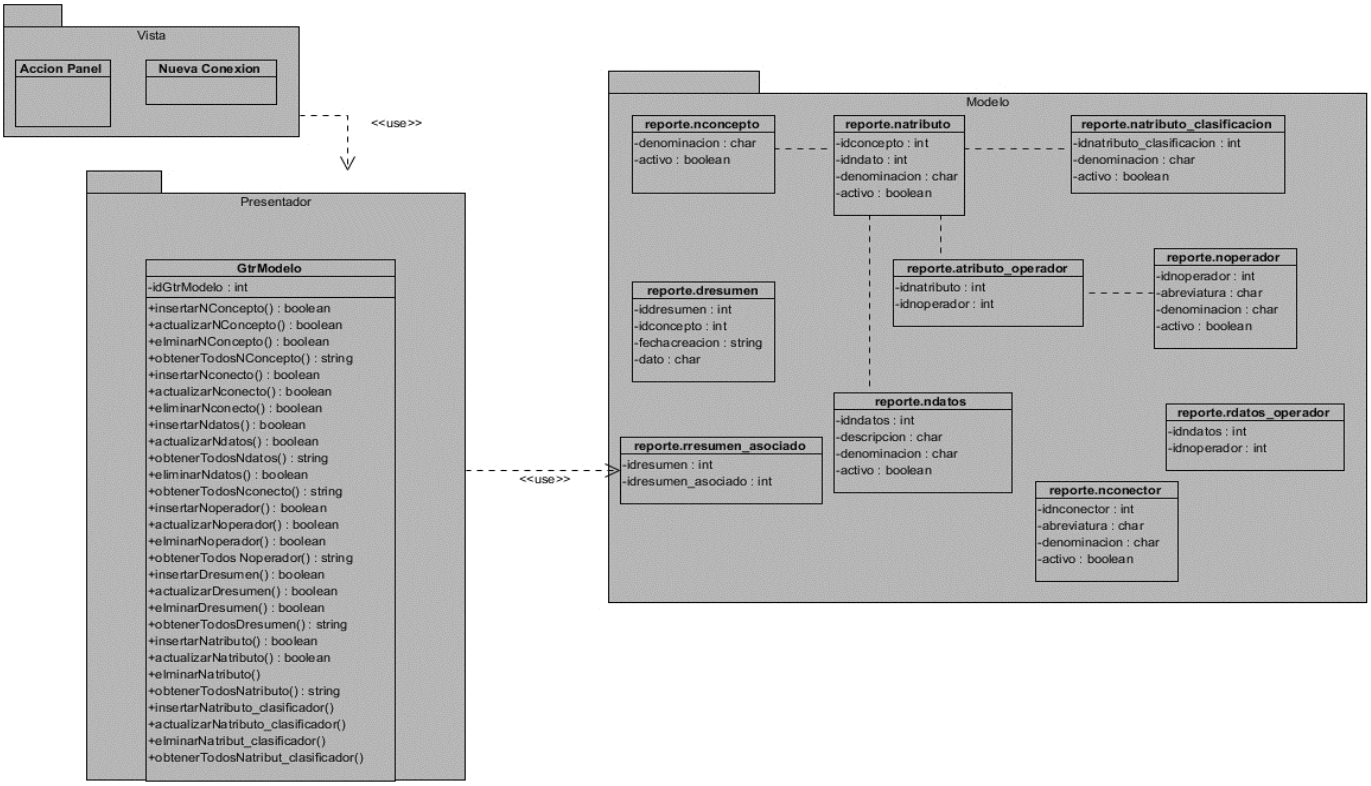


Figura 5: diagrama de clases de diseño

Fuente: elaboración propia

En el diagrama anterior se observa la relación entre las clases que intervienen en la implementación de la propuesta de solución, las cuales se encuentran organizadas en paquetes que se corresponden con la arquitectura del componente.

2.7. Modelo de datos

El modelo de datos se utiliza para describir la estructura lógica y física de la información persistente gestionada por el sistema. Se utiliza para definir la correlación entre las clases de diseño persistentes y las estructuras de datos persistentes, y para definir las estructuras de datos persistentes (Tecnología, 2018).

A continuación, se muestra el modelo de datos de la solución propuesta, el cual es un diseño basado en conceptos y la semántica de la forma de organizar la información en los modelos de datos relacionales.El modelode datos obtenido cuenta con un total de 10 tablas, las cuales son:*reporte.natributo; reporte.natributo_clasificador; reporte.ratributo_operador; reporte.noperador; reporte.rdato_operador; reporte.nconector; reporte.ndato; reporte.nconcepto; reporte.dresumen; reporte.rresumen_asociado*.

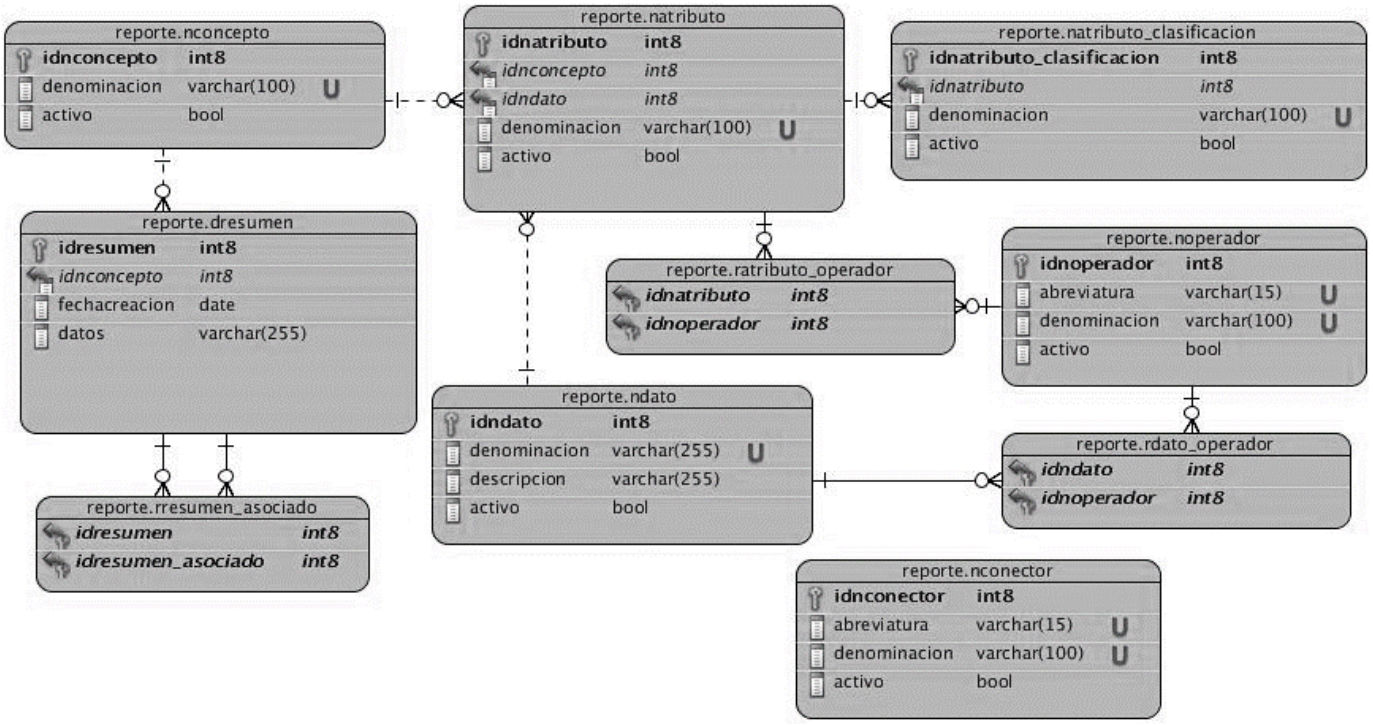


Figura 6: modelo de datos

Fuente: elaboración propia

Descripción de las tablas

La descripción de las tablas muestra de forma sintetizada el objetivo que persigue larepresentación en la base de datos de cada entidad. En la Tabla 4 se documentan lastablas de la base de datos relacionadas con la descripción de los requisitos analizados.

Tabla 4:tablas de la base de datos asociadas al modelo de búsquedas dinámicas.

Nombre	Descripción
dresumen	Entidad que almacena la información de los conceptos por fechas.
nconcepto	Entidad que almacena la información de los conceptos de negocio definidos para los reportes.
Natributo	Entidad que almacena la información de los atributos asociados a cada concepto.
natributo_clasificacion	Entidad que almacena la información de la denominación de los atributos indicadores asociados a los conceptos.
nconector	Entidad que almacena la información de los conectores de los atributos. Ejemplo: AND, OR, etcétera.
noperador	Entidad que almacena la información de los operadores. Ejemplo: =, =!, LIKE, etcétera.
Ndato	Entidad que almacena la información de los tipos de datos que se asocian a los atributos. Ejemplo: INT, TEXT, DATE, etcétera.
rdato_operador	Entidad que almacena la información de loa datos asociados por operadores.
ratributo_oporador	Entidad que almacena la información de los atributos asociados por operador.
rresumen_asiciado	Entidad que almacena la información de los conceptos correlacionados entre sí.

Fuente: elaboración propia

2.8. Diagrama de despliegue

El diagrama de despliegue es un tipo de diagrama del Lenguaje Unificado de Modelado que se utiliza para modelar el hardware utilizado en las implementaciones de sistemas y las relaciones entre sus componentes. Describen la topología del sistema, la estructura de los elementos de hardware y el software que ejecuta cada uno de ellos. Los diagramas de despliegue representan a los nodos y sus relaciones. Los nodos son conectados por asociaciones de comunicación tales como enlaces de red,

conexiones TCP/IP(Systms, 2020).

A continuación, se muestra el diagrama de despliegue correspondiente al sistema:

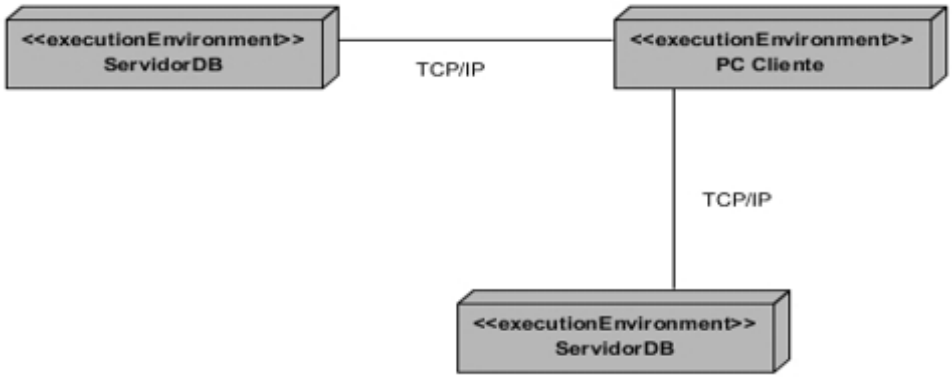


Figura 7: diagrama de despliegue.

Fuente: elaboración propia

La imagen anterior representa el diagrama de despliegue para el componente. El mismo estará compuesto por una PC cliente que representa las computadoras que utilizarán los usuarios para interactuar con la aplicación de escritorio instalada que se comunicará con cada uno de los servidores de bases de datos de cada proyecto mediante la extensión ODBC.

2.9. Estándares decodificación

Los estándares de codificación son normas que deben seguirse durante todo el ciclo de implementación del sistema (ohmyroot, 2017). Los mismos fueron seleccionados por el equipo de arquitectura del proyecto. A continuación, se describen algunos de los estándares más utilizados en la implementación del componente de configuración para las búsquedas dinámicas.

Declaración de clases:

- Las clases deben ser nombradas de acuerdo a la convención de nombres.
- Las clases persistentes comenzarán con las siglas definidas en la base dedatos para cada tabla.
- Los formularios de tipo ventanas emergentes comienzan con las siglas frm, ylos de tipo panel con las siglas pnl.

- Los gestores de negocio comienzan con las siglas Gtr. Las clases de la lógica de negocio de acceso a datos comienzan con las siglas Rtry.

A continuación, se muestra un ejemplo de una declaración de clase que es permitida:

```
public FrmPrincipal() throws Exception {
    initComponents();
    gestor = new GtrXtreme();
    btnEliminar.setEnabled(false);
    btnEditar.setEnabled(false);
    btnEliminarBD.setEnabled(false);
    generarMenu();

}
```

Figura 8: declaración de las clases

Fuente: elaboración propia

Componentes de formularios:

- Los botones (*JButton*) comienzan con las siglas btn.
- Los campos de texto (*JTextField*) comienzan con las siglas txt.
- Los campos de fecha (*Date*) comienzan con las siglas fch.
- Las listas desplegables (*JComboBox*) comienzan con las siglas cmb.
- Las áreas de texto (*JTextArea*) comienzan con las siglas txa.
- Las etiquetas (*JLabel*) comienzan con las siglas lbl.
- Los cuadros de chequeo (*JCheckBox*) comienzan con las siglas chb.
- Los botones de opción (*JRadioButton*) comienzan con las siglas rbt.
- Las tablas (*JTable*) comienzan con las siglas tbl.

Funciones y métodos:

- Los nombres de funciones pueden contener únicamente caracteres alfanuméricos. Los guiones bajos (_) no están permitidos. Los números están permitidos en los nombres de función, pero no se aconseja en la mayoría de los casos.

- Los nombres de funciones deben empezar siempre con una letra minúscula. Cuando un nombre de función consiste en más de una palabra, la primera letra de cada nueva palabra debe estar en mayúsculas. Esto es llamado comúnmente como formato "*camelCase*".
- Por norma general, se recomienda la elocuencia. Los nombres de función deben ser lo suficientemente elocuentes como para describir su propósito y comportamiento.

A continuación, se muestra un ejemplo de nombres de funciones admisibles:

```
public void limpiarBaseDatos() {
    try {
        em = emf.createEntityManager();
        em.getTransaction().begin();
        Query consulta = em.createNamedQuery("");
        consulta.executeUpdate();
        em.getTransaction().commit();
    } catch (Exception e) {
        em.getTransaction().rollback();
        throw e;
    } finally {
        em.close();
    }
}
```

Figura 9: nombre de función admisible

Fuente: elaboración propia

Constantes:

Las constantes pueden contener tanto caracteres alfanuméricos como barras bajas (_). Los números están permitidos. Todas las letras pertenecientes al nombre de una constante deben aparecer en mayúsculas. Las palabras dentro del nombre de una constante deben separarse por barras bajas (_). A continuación, se muestra un ejemplo delas constantes y su inicialización:

```
public ERGeneral(String nombre, String club, int edad, Double puntos) {
    this.campo1 = nombre;
    this.campo2 = club;
    this.campo11 = edad;
    this.campo14 = puntos;
}
```

Figura 10: declaración de las constantes

Fuente:elaboración propia

Comentarios

Comentarios de comienzo: Todos los ficheros fuente deben comenzar con un comentario en el que se lista el nombre de la clase, información de la versión, fecha, y copyright:

```
4   /*
5   * Nombre de la clase
6   *
7   * Información de la versión
8   *
9   * Fecha
10  *
11  * Copyright
12  */
13
```

Figura 11: comentarios de comienzo

Fuente: elaboración propia

Comentarios en las funciones:

Todas las funciones deben tener un comentario, antes de su declaración, explicando qué hacen. Ningún programador debería tener que analizar el código de una función para conocer su utilidad. Tanto el nombre como el comentario que acompañe a la función deben bastar para ello.

Variables

Nombres de variables: los nombres deben ser descriptivos y concisos. No usar ni grandes frases ni pequeñas abreviaciones para las variables. Siempre es mejor saber qué hace una variable con solo conocer su nombre. Los nombres de las variables inician con letra minúscula, pero si estas tienen más de una palabra, cada nueva palabra debe iniciar con letra mayúscula. Las variables globales deben de tener como prefijo el nombre de la clase a la que pertenecen.

Se recomienda una declaración por línea, ya que facilita los comentarios. Se debe intentar inicializar las variables locales donde se declaran. La única razón para no inicializar una variable donde se declara es si el valor inicial depende de algunos cálculos que deben ocurrir. Además, es necesario poner las declaraciones solo al principio de los bloques y no esperar al primer uso para declararlas ya que puede

confundir a programadores no preavisados y limitar la portabilidad del código dentro de su ámbito de visibilidad. También se debe evitar las declaraciones locales que ocultan declaraciones de niveles superiores. En la figura 9 que se muestra a continuación se evidencia el uso de las reglas descritas:

```
private void jTPuertoKeyTyped(java.awt.event.KeyEvent evt) {
    char c=evt.getKeyChar();

    if(c<'0'|| c>'9')evt.consume();

}
```

Figura 12: declaraciones

Fuente: elaboración propia

2.10. Conclusiones parciales

- La descripción general de la propuesta de solución permite adquirir una visión de los flujos de información y las actividades que engloban el negocio.
- El uso de la metodología AUP en su variación para la UCI, permite organizar el desarrollo de la solución.
- A través de la especificación de los requisitos e historias de usuarios, se documenta toda la información relativa a la propuesta de solución.
- El diseño de dicha propuesta permite concebir los elementos necesarios para la implementación de la misma.
- Queda establecida la arquitectura para el componente identificando cómo interactúa cada una de las capas dentro del estilo arquitectónico definido.
- La definición de los estándares de codificación establece las normas a seguir durante todo el ciclo de la implementación para obtener un código legible y fácilmente actualizable.

CAPÍTULO 3: VALIDACIÓN DEL COMPONENTE PARA BÚSQUEDAS DINÁMICAS

Introducción

En este capítulo se evalúa el nivel de calidad y fiabilidad de los resultados obtenidos en el desarrollo del componente para búsquedas dinámicas mediante estructuras JSON en PostgreSQL para los proyectos de CEGEL. Dicha valoración se lleva a cabo a partir de la validación de los requisitos a través de las técnicas: Revisiones de los requisitos, Construcción de prototipos y Generación de casos de prueba. Además, se validará el diseño a partir de las métricas: Tamaño Operacional de las Clases y Relación entre Clases. Por último, se utilizan las disciplinas de pruebas internas que define la metodología que guía el desarrollo de la solución propuesta, con el fin de verificar y revelar la calidad del producto antes de su entrega al cliente.

3.1. Validación de requisitos

La calidad de los productos de trabajo procedentes de requisitos se evalúa durante un paso de validación. La validación de requisitos examina las especificaciones para asegurar que todos los requisitos del sistema han sido establecidos sin ambigüedad, sin inconsistencias, sin omisiones, que los errores detectados hayan sido corregidos, y que el resultado del trabajo se ajusta a los estándares establecidos para el proceso, el proyecto y el producto(Pressman, 2010). Para la validación de los requisitos se tendrán en cuenta las técnicas de:

Revisiones de los requisitos:los requerimientos se analizan sistemáticamente usando un equipo de revisores que verifican errores e inconsistencias(Sommerville, 2011).

Se realizaron revisiones a cada uno de los requisitos por parte del equipo de desarrollo. Las revisiones internas generaron un total de cinco no conformidades de tipo técnicas y de ortografía, las cuales fueron corregidas satisfactoriamente. Con el equipo de análisis y líder de proyecto se realizó la revisión en la cual se añadieron detalles a 3 requisitos funcionales y se aprobaron finalmente los mismos, generándose de este encuentro la firma del documento "*Especificación_de_requisitos_de_software"* por parte del cliente.

Calidad de la Especificación

Para medir la calidad de la especificación de los requisitos de software se aplicará la métrica Calidad de la Especificación (CE). Según Pressman (2010), el empleo de esta métrica permitirá obtener un alto nivel de entendimiento y precisión de los requisitos,para llegar a ello se debe primeramente calcular el total de requisitos de software como se muestra a continuación:

Nr= Nf + Nnf

Nr: el total de requisitos de especificación.

Nf: cantidad de requisitos funcionales.

Nnf: cantidad de requisitos no funcionales.

Como resultado de la sustitución de los valores, se obtiene:

Nr = 45+ 11Nr = 56

Para determinar, finalmente, la Calidad de la Especificación (CE) de los requisitos y ausencia de ambigüedad en los mismos se realiza la siguiente operación:

Q1 = Nui / Nr

Donde ***Nui*** es el número de requisitos para los cuales todos los revisores tuvieron interpretaciones idénticas. Mientras más cerca de 1 esté el valor de ER, menor será la ambigüedad.

Para el caso de los requisitos obtenidos, cinco produjeron contradicción (RF2, RF4, RF7, RF8 y RF11) en sus interpretaciones. Sustituyendo las variables se obtiene:

Q1 = 51/ 56Q1 = 0.91

Es importante aclarar que mientras más cerca de 1 está el valor de Q1, menor es la ambigüedad. Teniendo en cuenta el resultado anterior, igual a 0.91, se concluye que en el 91% de los requisitos se obtuvo una sola interpretación. Los cinco requisitos identificados como ambiguos fueron modificados y validados para garantizar su correcta interpretación, llegando al resultado ideal de Q1=1.

Construcción de prototipos: en esta aproximación a la validación, se muestra un modelo ejecutable del sistema en cuestión a los usuarios finales y clientes. Así, ellos podrán experimentar con este modelo para comprobar si cubre sus necesidades reales(Sommerville, 2011).

Se mostró al cliente un modelo factible que permitió tener una perspectiva preliminar de cómo se vería este componente en los sistemas y mediante la interacción con el mismo se comprobó la satisfacción y aprobación del cliente, estos prototipos fueron aprobados satisfactoriamente.En el epígrafe 2.3 se hace referencia a las Historias de Usuario como producto de trabajo de la disciplina de Requisitos, en la misma se detallan los prototipos obtenidos para la propuesta de solución para el caso del requisito que se toma como ejemplo en este documento.

Generación de casos de prueba:los requerimientos deben ser comprobables. Si las pruebas para los requerimientos se diseñan como parte del proceso de validación, esto revela con frecuencia problemas en los requerimientos. Si una prueba es difícil o imposible de diseñar, esto generalmente significa que los requerimientos serán difíciles de implementar, por lo que deberían reconsiderarse(Sommerville, 2011).

Como parte del proceso de validación de los requisitos funcionales fueron diseñados casos de pruebas para cada requisito. En el presente capítulo se relacionan los casos de pruebas generados en esta técnica.

3.2. Métricas para la evaluación del diseño

Un elemento clave de cualquier proceso de ingeniería es la medición. Pueden usarse medidas para entender mejor los atributos de los modelos que se crean y para valorar la calidad de los productos o sistemas sometidos a ingeniería que se construyen(Pressman, 2010)

Para validar el diseño que se obtiene en la presente investigación, se define aplicar un conjunto de métricas definidas por (Lorenz, y otros, 1994):

Tamaño Operacional de las Clases: las métricas basadas en el Tamaño Operacional de las Clases (TOC) se basan en contar la cantidad de atributos y la cantidad de operaciones que tiene una clase individual y el promedio que presenta el sistema en su totalidad. Se evalúa a partir de los siguientes atributos de calidad:

- Responsabilidad
- Complejidad de implementación
- Reutilización

Una vez analizado el indicador tamaño de clase, si el valor resultante tiende al crecimiento, es probable que la clase posea un alto grado de responsabilidad; en consecuencia, el nivel de reutilización sería mínimo y la implementación altamente compleja. Para medir los atributos de calidad se definen los umbrales que se muestran en la siguiente tabla, en la misma se emplea la abreviatura Prom (promedio).

Tabla 5: métrica TOC. Categoría por atributos y criterio de evaluación.

Atributo de calidad	Categoría	Criterio
Responsabilidad	Baja	TOC < =Promedio (Prom).
	Media	TOC Entre Prom. y 2* Prom.
	Alta	TOC >2*Prom.
Complejidad de implementación	Baja	TOC < =Prom.
	Media	Entre Prom. y 2* Prom.
	Alta	TOC >2*Prom.
Reutilización	Baja	TOC >2*Prom.
	Media	TOC Entre Prom. y 2* Prom.
	Alta	TOC < =Prom.

Fuente: (Lorenz, y otros, 1994)

Relación entre Clases: el resultado de la aplicación de la métrica Relación entre Clases (RC) está dado por el número de relaciones de uso que se establecen entre una clase y las demás clases existentes. Se evalúa a partir de los siguientes atributos de calidad(Lorenz, y otros, 1994):

- Acoplamiento: un aumento del RC implica un aumento del acoplamiento de la clase.
- Complejidad de Mantenimiento: un aumento del RC implica un aumento de la complejidad del mantenimiento de la clase.

- Reutilización: un aumento del RC implica una disminución en el grado de reutilización de la clase.
- Cantidad de Pruebas: un aumento del RC implica un aumento de la cantidad de pruebas necesarias para probar una clase.

Para obtener un nivel óptimo de relación entre clases, el valor obtenido al aplicar dicha métrica debe ser directamente proporcional al acoplamiento y a la complejidad de mantenimiento; además debe ser inversamente proporcional al nivel de reutilización del código. Para aplicar la métrica RC es necesario categorizar cada una de las clases según la cantidad de relaciones que esta contenga.

A continuación, se muestra una tabla con las categorías para clasificar cada uno de los atributos de calidad anteriormente mencionados, así como el criterio de evaluación. En la misma se emplean la abreviatura Prom (promedio).

Tabla 6: métrica RC. Categoría por atributos y criterio de evaluación.

Atributo	Categoría	Criterio
Acoplamiento	Ninguno	RC = 0
	Bajo	RC = 1
	Medio	RC = 2
	Alto	RC > 2
Complejidad de Mantenimiento	Baja	RC <= Prom
	Media	Prom<=RC<= 2* Prom
	Alto	RC >2* Prom
Reutilización	Baja	RC >2* Prom
	Media	Promedio< RC <= 2 *Promedio
	Alto	RC <= Promedio
Cantidad de pruebas	Baja	RC <= Promedio
	Media	Promedio<= RC <2*Promedio
	Alto	RC >= 2*Promedio

Fuente:(Lorenz, y otros, 1994)

3.3. Validación del diseño

Para validar el diseño realizado en la propuesta de solución se tuvo en cuenta las métricas Tamaño Operacional de Clases y Relación entre Clases descritas en el epígrafe anterior.

Para el caso de la métrica TOC a continuación, se muestran las medidas de los parámetros de calidad obtenidas luego de ser aplicada al diseño de clases del componente de búsquedas dinámicas, teniendo en cuenta las categorías por atributos y los criterios de evaluación de la tabla descrita anteriormente. En el caso de la siguiente tabla se utilizaron las abreviaturas:

- **Rp** para el atributo de Responsabilidad
- **Ci**para el atributo de Complejidad de implementación
- **R** para el atributo de Reutilización

Tabla 7: resultados obtenidos luego de aplicada la métrica TOC.

No.	Clase	Cantidad de procedimientos	Rp	Ci	R
1	GtrReporte()	0	Baja	Baja	Alta
2	ERGeneral	5	Baja	Baja	Alta
3	buscaratibuto()	0	Baja	Baja	Alta
4	btnDesgloseActionPerformed()	4	Baja	Baja	Alta
5	btnEliminarBDActionPerformed()	0	Baja	Baja	Alta
6	rbEditarActionPerformed()	5	Baja	Baja	Alta
7	btnEditarActionPerformed()	9	Media	Media	Media
8	btnEliminarActionPerformed()	1	Baja	Baja	Alta
9	rbEliminarActionPerformed()	4	Baja	Baja	Alta
10	btnAgregarActionPerformed()	0	Baja	Baja	Alta
11	btnCerrarActionPerformed()	9	Media	Media	Media
12	btnActualizarActionPerformed	3	Baja	Baja	Alta
13	*Reporte.rdato_operador()*	6	Baja	Baja	Alta

Fuente: elaboración propia

Al aplicar la métrica TOC a todas las clases del modelo de diseño de los nomencladores del componente, se obtuvieron para cada atributo de calidad los siguientes resultados:

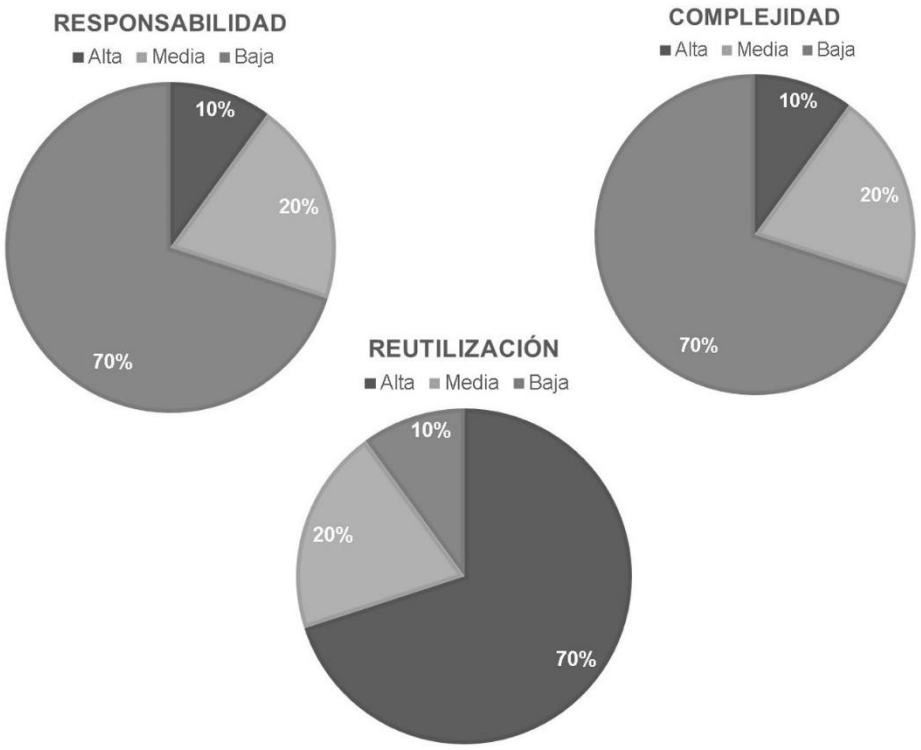


Figura 13: resultado de la métrica TOC

Fuente: elaboración propia

Después de evaluar los resultados obtenidos de cada uno de los atributos de calidad que mide esta métrica, se tiene que:

- el 70% de las clases poseen una baja responsabilidad y complejidad de pruebas.
- el 70% de las clases poseen una alta reutilización.

Estos resultados demuestran la calidad del diseño de la solución propuesta, pues al obtener bajos índices de responsabilidad y complejidad, unidos a un alto grado de reutilización de las clases se facilita en gran medida la implementación del componente como resultado de la propuesta de solución.

Por otra parte, se muestran las medidas de los parámetros de calidad obtenidas luego de aplicar la métrica RC al diseño de clases del componente de búsquedas dinámicas, teniendo en cuenta las categorías por atributos y los criterios de evaluación de la tabla descrita en el epígrafe anterior. En el caso de la siguiente tabla se utilizaron las abreviaturas:

- **Ac** para el atributo de Acoplamiento.
- **Cm** para el atributo de Complejidad de mantenimiento.
- **R** para el atributo de Reutilización.
- **Cp**para el atributo Cantidad de prueba.

Tabla 8: resultados obtenidos luego de aplicada la métrica RC.

No.	Clase	Cantidad de relación de uso	Ac	Cm	R	CP
1	GtrReporte()	1	Baja	Baja	Alta	Baja
2	ERGeneral	2	Media	Media	Media	Media
3	buscaratributo()	0	Ninguno	Baja	Alta	Baja
4	btnDesgloseActionPerformed()	0	Ninguno	Baja	Alta	Baja
5	btnEliminarBDActionPerformed()	0	Ninguno	Baja	Alta	Baja
6	rbEditarActionPerformed()	1	Baja	Baja	Alta	Baja
7	btnEditarActionPerformed()	2	Media	Media	Media	Media
8	btnEliminarActionPerformed()	0	Ninguno	Baja	Alta	Baja
9	rbEliminarActionPerformed()	0	Ninguno	Baja	Alta	Baja
10	btnAgregarActionPerformed()	0	Ninguno	Baja	Alta	Baja
11	btnCerrarActionPerformed()	0	Ninguno	Baja	Alta	Baja
12	btnActualizarActionPerformed	0	Ninguno	Baja	Alta	Baja
13	*Reporte.rdato_operador()*	0	Ninguno	Baja	Alta	Baja

Fuente: elaboración propia

Después de haber aplicada la métrica, se tomaron todos los resultados obtenidos individualmente y se agruparon para ser analizados de manera general, promediando los valores obtenidos por categoría en cada atributo, los resultados se muestran en la siguiente figura:

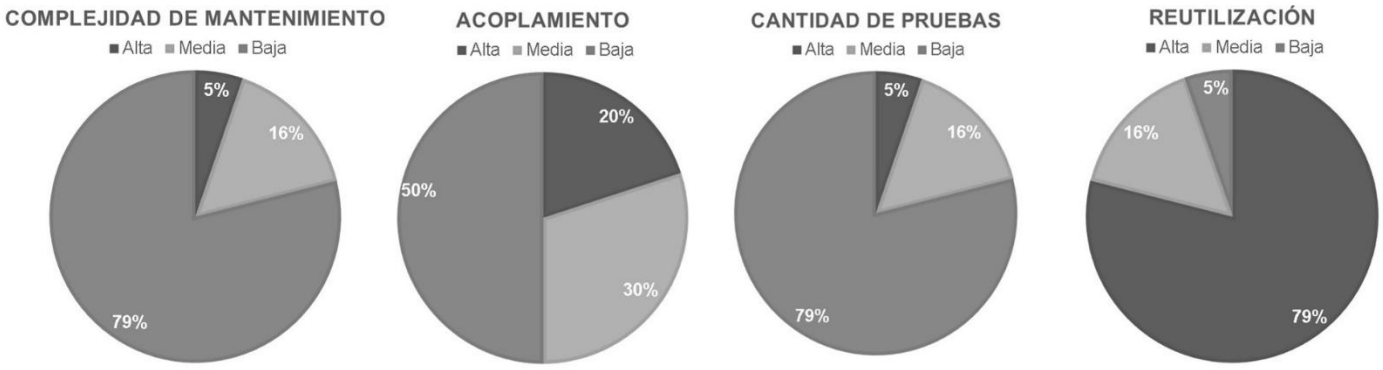


Figura 14: resultado de la métrica RC

Fuente: elaboración propia

Después de evaluar los resultados obtenidos de cada uno de los atributos de calidad que mide esta métrica, se tiene que:

- el 79% de las clases tienen una baja complejidad de mantenimiento y una baja cantidad de pruebas.
- el 50% de las clases tiene un bajo acoplamiento.
- el 79% de las clases poseen una alta reutilización.

Como se puede observar en la figura el acoplamiento posee un nivel bajo por lo que se tiene poca dependencia entre las clases trayendo como consecuencia una alta probabilidad de reutilización. También se puede apreciar que existe un bajo nivel de complejidad de mantenimiento, por lo que a la hora de optimizar métodos y demás operaciones no es necesario realizar una gran cantidad de pruebas, lo que minimiza el tiempo de implementación y prueba del componente propuesto.

3.4. Validación de la implementación

Las pruebas y validaciones de software son de vital importancia para corroborar que la aplicación desarrollada no presente problemas de ejecución y se encuentre libre de errores, ya que durante el proceso de desarrollo de software es frecuente que los desarrolladores, al programar las diferentes funcionalidades, obvien algunas posibilidades que pueden variar el resultado esperado durante la ejecución del código; por tanto, estas pruebas y validaciones constituyen la vía a través de la cual se asegura que el producto desarrollado está listo para ser entregado a los usuarios finales. Los errores

más frecuentes pueden suceder a causa del mal uso de estructuras de datos, errores lógicos, entre otras (Pressman, 2010)

3.4.1. 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 (Rodríguez Sánchez, 2015). A continuación, se detallan las pruebas llevadas a cabo durante esta disciplina, con el objetivo de asegurar la calidad del resultado de la implementación.

3.4.1.1. Pruebas funcionales

Las pruebas funcionales son pruebas diseñadas tomando como referencia las especificaciones funcionales de un componente o sistema. Se realizan para comprobar si el software cumple las funciones esperadas (Pressman, 2010). Para llevar a cabo estas pruebas se tuvo en cuenta el método de Caja blanca y el método de Caja negra. A continuación, se describe como se llevaron a cabo ambos métodos.

Método de Caja blanca

Con el empleo de este método es posible desarrollar casos de prueba que garanticen la ejecución, al menos una vez, de los caminos independientes[7](Pressman, 2010). Para aplicar este método se define la técnica de ruta básica.

Técnica de ruta básica: esta técnica tiene como objetivo comprobar que cada camino se ejecute independiente de un componente o programa, obteniéndose una medida de la complejidad lógica del diseño. Debe ser utilizada para evaluar la efectividad de los métodos asociados a una clase, con el objetivo de asegurar que cada camino independiente sea ejecutado por lo menos una vez en el sistema (Pressman, 2010).

Pressman propone como estrategia para aplicar la ruta básica, realizar un análisis de la complejidad ciclomática de cada procedimiento que componen las clases del sistema. Una vez concluido este paso se selecciona el método con valor de contener errores, además de que ofrece una medida del número de pruebas que deben diseñarse para validar la correcta implementación de una determinada función. Esta métrica se calcula sobre un grafo y se puede realizar mediante tres formas distintas:

1. **V(G)** = R

[7] Camino que recorre por lo menos una nueva arista en el grafo de flujo.

2. **V(G)** = E – N + 2
3. **V (G)** = P + 1

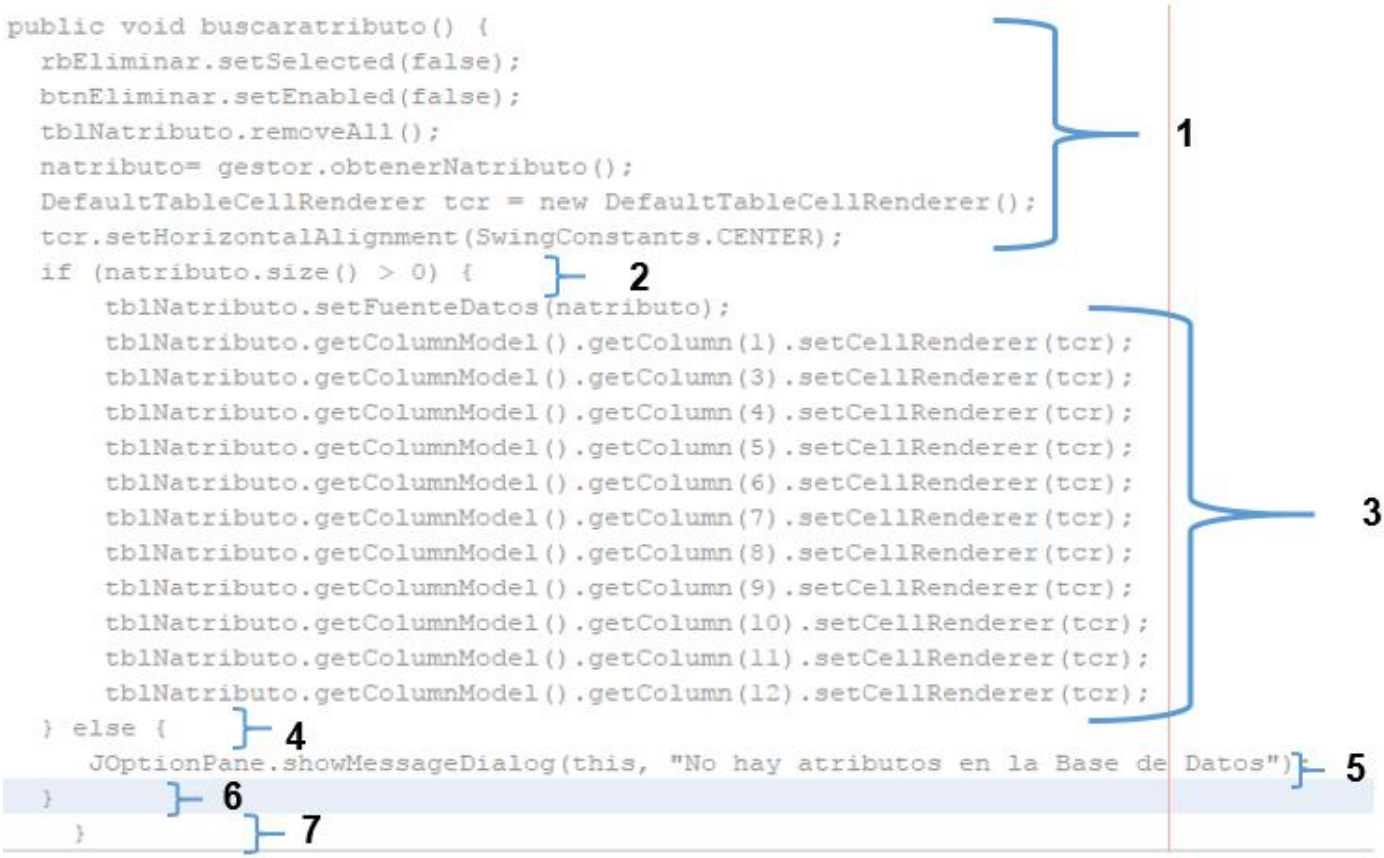
Conociendo que:

- **G:** Grafo de flujo (grafo).
- **R:** El número de regiones contribuye a estimar el valor de la complejidad ciclomática.
- **E:** Número de aristas.
- **V(G):** Complejidad ciclomática.
- **N:** Número de nodos del grafo.
- **P:** Número de nodos predicados [8] incluidos en el grafo.

Una vez calculada la complejidad ciclomática, el valor obtenido representa el límite superior de pruebas que deberán aplicarse (Pressman, 2010).

Para obtener los casos de prueba a partir de la técnica seleccionada se debe construir el grafo de flujo correspondiente al código del método *buscaratributor()*. La prueba fue aplicada a diez (10) métodos logrando así validar el código aplicando el método de caja blanca.

[8]Está caracterizado porque dos o más aristas emergen de él.



A continuación, se muestra el código del método *buscaratributo()*:

Figura 15: código del método buscar()para la obtención del grafo de flujo

Fuente: elaboración propia

A continuación, se procede a realizar el grafo de flujo para el método previamente descrito:

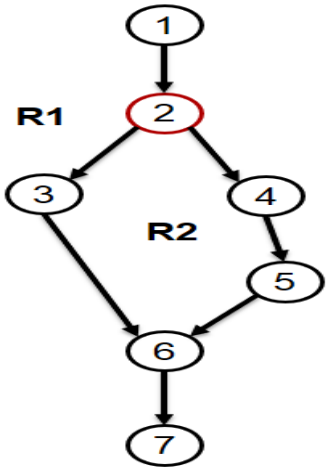


Figura 16: grafo de flujo a partir del método antes descrito.

Fuente: elaboración propia

A dicho grafo de flujo se le calculó la complejidad ciclomática, calculada por las tres vías conocidas, obteniéndose los siguientes resultados:

- . **V(G)** = R = 2
- **V(G)** = 7 – 7 + 2 = 2
- **V(G)**= 1 + 1 = 2

Después de calcular la complejidad del grafo, se pudo comprobar que los resultados obtenidos son iguales a 2, por tanto, se deben realizar 2 casos de pruebas, uno por cada ruta independiente. Las rutas independientes resultantes fueron:

- 1-2-3-6-7
- 1-2-4-5-6-7

Cada ruta independiente es un caso de prueba a realizar, de forma que los datos introducidos provoquen que se visiten las sentencias vinculadas a cada nodo del camino. A continuación, se muestran el caso de prueba para la ruta independiente 1:

Tabla 6. Caso de prueba. Ruta independiente #1.

Caso de prueba:	**Devolver nombre**
Camino	1-2-3-6-7
Entrada	
Resultados esperados	La búsqueda debe devolver si hay o no atributos en la base de datos.
Condiciones	Que existan elementos en la base de datos.

Fuente: elaboración propia

Resultados de la ejecución de los casos de prueba

Luego de realizada la prueba para los métodos seleccionados se obtuvo, en una primera iteración, un total de ocho (8) no conformidades, las cuales fueron resueltas. En una segunda iteración se encontraron dos (2) no conformidades que fueron resueltas antes de la tercera iteración, en la cual no se encontraron nuevas no conformidades, obteniéndose finalmente un código funcional.

Las no conformidades estuvieron relacionadas principalmente con fragmentos de código que podían ser optimizados y condiciones en los métodos con estructuras *If/else* que no estaban declaradas correctamente.

Método de Caja negra

El método de caja negra permite al ingeniero del software obtener conjuntos de condiciones de entrada que ejerciten completamente todos los requisitos funcionales de un programa. La misma no es una alternativa a las técnicas del método de caja blanca. Más bien se trata de un enfoque complementario que intenta descubrir diferentes tipos de errores que los métodos de caja blanca.Estas pruebas permiten encontrar:

- Funciones incorrectas o ausentes.
- Errores de interfaz.
- Errores en estructuras de datos o en accesos a las bases de datos externas.

- Errores de rendimiento.
- Errores de inicialización y terminación (Pressman, 2010).

Para llevar a cabo el método de Caja negra se utilizan las técnicas: Partición de equivalencia y Análisis de valor de frontera, que a continuación se describen.

Partición de equivalencia: esta técnica divide el dominio de entrada de un programa en clases de datos a partir de las cuales pueden derivarse casos de prueba. El método se esfuerza por definir un caso de prueba que descubra ciertas clases de errores, reduciendo así el número total de casos de prueba que deben desarrollarse. Una clase de equivalencia representa un conjunto de estados válidos y no válidos para las condiciones de entrada (Pressman, 2010).

Análisis de valor de frontera: un mayor número de errores ocurre en las fronteras del dominio de entrada y no en el "centro". Por esta razón es que el análisis de valor de frontera se desarrolló como una técnica de prueba que conduce a una selección de casos de prueba que revisan los valores de frontera. Es una técnica de diseño de casos de prueba que complementa la partición de equivalencia. En lugar de seleccionar algún elemento de una clase de equivalencia, esta conduce a la selección de casos de prueba en los "bordes" de la clase (Pressman, 2010).

Para aplicar estas técnicas, se debe primeramente realizar el Diseños de Casos de Prueba (DCP) con el objetivo de obtener un conjunto de pruebas que tengan la mayor probabilidad de descubrir los defectos del software.

En ingeniería del software, un DCP es un conjunto de condiciones o variables bajo las cuales un analista determinará si una aplicación o una característica de éstos es parcial o completamente satisfactoria (Pressman, 2010). Como primer paso en el DCP, se muestra a continuación la descripción de las variables para el caso del requisito funcional

Para el caso del RF1. Adicionar conexión que se toma como ejemplo en el presente documento, a continuación, se muestra el DCP asociado a este requisito.

Tabla 9: DCP del RF1. Adicionar conexión

Escenario	Descripción	Denominación	Respuesta del sistema	Flujo central
EC 1.1 Datos	Adicionar	**V**	1) Al dar clic en el botón	Inicio/ Nueva

correctos	conexión en el sistema introduciendo los datos de forma correcta	conexión **V** nueva conexión	probar conexión, el sistema muestra el siguiente mensaje "Conexión establecida". 2) Al dar clic en el botón aceptar se debe listar dicha conexión.	conexión
EC 1.2 Datos vacíos	Adicionar conexión en el sistema introduciendo datos vacíos	**I** (vacío)	1) Al dar clic en el botón probar conexión, el sistema muestra el siguiente mensaje:"conexión fallida" 2) Al dar clic en el botón aceptar, el sistema muestra el mensaje: “Hay campos obligatorios vacíos”.	
EC 1.3 Datos duplicados	Adicionar conexión en el sistema introduciendo datos duplicados	**I** conexión	1) El sistema muestra un mensaje de error "La conexión que intenta realizar ya existe"	
EC 1.4 Cancelar operación	Cancelar operación dando clic en el botón cancelar.	**N/A**	El sistema descarta los cambios realizados y oculta el área para adicionar los datos de la conexión.	

Fuente: elaboración propia

Para aplicar el método de caja negra a la solución, se efectuaron un total de 3 iteraciones para poder alcanzar resultados satisfactorios, atendiendo al correcto comportamiento del sistema ante diferentes situaciones. En la figura que se muestra a continuación, se realiza un resumen mediante una gráfica con el número total de No conformidades (NC) por iteración.

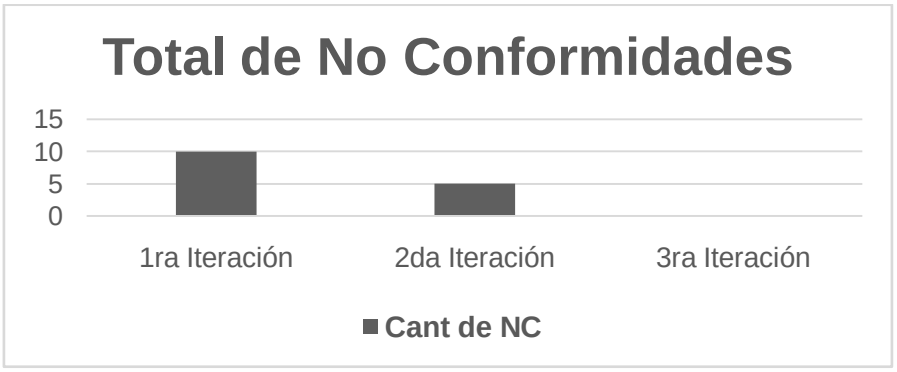


Figura 17: cantidad de NC por iteración

Fuente: elaboración propia

Estas NC se clasificaron en: Opciones que no funcionan, Ortografía y Validación. En cada una se corrigieron las NC dando paso a que en la tercera iteración no se detectaran NC, lo que trajo consigo que se procediera a la disciplina de pruebas de liberación propuesta por la metodología AUP-UCI. A continuación, se muestra un resumen de las NC por tipo de clasificación en cada iteración de prueba:

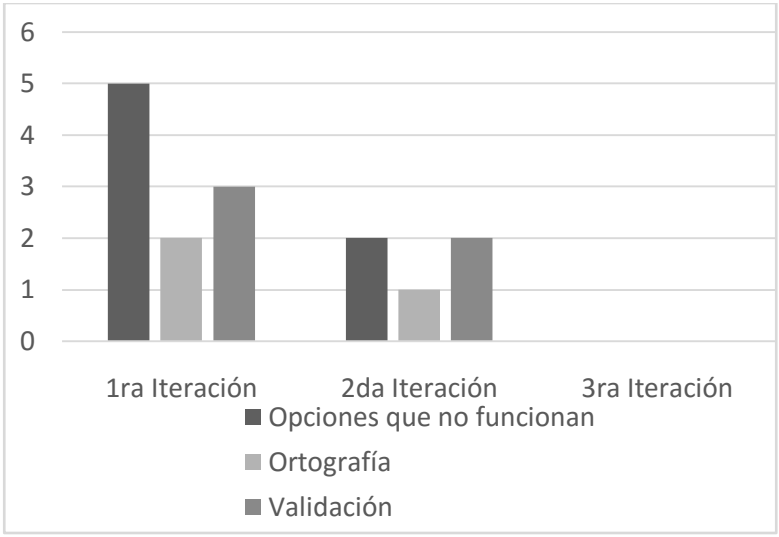


Figura 18: cantidad de NC por tipo de clasificación en cada iteración

Fuente: elaboración propia

3.5. Conclusiones parciales

- Las métricas de validación del diseño permiten definir de forma cuantitativa la calidad del mismo en el desarrollo del componente de búsquedas dinámicas mediante estructura JSON en PostgreSQL para los proyectos de CEGEL.
- La ejecución de pruebas funcionales evidencia que las funciones externas son operativas, que las entradas se aceptan de forma adecuada y que se producen salidas correctas, obteniendo finalmente una aplicación que satisface los requisitos definidos.

CONCLUSIONES GENERALES

Con el desarrollo del Componente para búsquedas dinámicas mediante estructuras JSON en PostgreSQL para los proyectos de CEGELse puede afirmar que:

1) La sistematización del marco teórico de la investigación sustentó los principales referentes teóricos en los que se enmarca el desarrollo de la propuesta desolución.
2) La identificación de los requisitos, así como el análisis y diseño del componente para búsquedas dinámicas, permitieron obtener una aproximación y sentar las bases para la implementación de dicho componente.
3) La implementación del componente para búsquedas dinámicas permitió obtener un sistema funcional que contribuye a la mejora del rendimiento y la estandarización de los reportes en los proyectos de CEGEL.
4) La validación de los resultados obtenidos a través de las métricas de validación del diseño y las pruebas de software permitió comprobar de forma cuantitativa la calidad de los artefactos obtenidos durante el desarrollo delsistema.

RECOMENDACIONES

Para futuras investigaciones relacionadas con la presente se recomienda:

1) Optimizar el componente de búsqueda a partir de la posibilidad de establecer conexiones con diferentes tipos de gestores de base de datos.

2) Dotarlo de mayor seguridad en la salva de la información.

3) Proveer al servicio con nuevos filtros sobre la informaciónque puedan enriquecer el espectro de búsqueda.

REFERENCIAS BIBLIOGRÁFICAS

Alegsa, Leandro. 2016.*Alegsa.* [En línea] 21 de 6 de 2016. [Citado el: 27 de 11 de 2019.] http://www.alegsa.com.ar/m/Dic/busqueda_avanzada.php.

Aprendiendoarduino. 2017. [En línea] 2017. [Citado el: 29 de 11 de 2019.] https://aprendiendoarduino.wordpress.com/2017/06/18/ide-arduino-y-configuracion/.

AprendiendoArduino. 2014. AprendiendoArduino. *AprendiendoArduino.* [En línea] 2014. [Citado el: 26 de 11 de 2019.] https://aprendiendoarduino.wordpress.com/2016/03/29/entorno-de-programacion-de-arduino-ide/.

calidad, Asociacion española para la. CAE. *CAE.* [En línea] https://www.aec.es/web/guest/home.

CCM. 2018. CCM. *CCM.* [En línea] 2018. https://es.ccm.net/.

CEGEL. 2020. UCI. Documento de despliegue y acompañamiento. 2020.

—. 2020. UCI. Ficha de costo de proyectos. 2020.

—. 2020. Universidad de las Ciencias Informáticas. Expedientes de proyectos. 2020.

Conogasi.org. 2018. [En línea] 23 de 4 de 2018. [Citado el: 29 de 11 de 2019.] http://conogasi.org/articulos/lenguaje-de-programacion/.

EDteam. 2019. [En línea] 5 de 3 de 2019. https://ed.team/comunidad/que-es-un-json.

Empresariales, Intituto Europeo de Estudios. 2016. [En línea] 2016. [Citado el: 29 de 11 de 2019.] https://revistadigital.inesem.es/informatica-y-tics/los-gestores-de-bases-de-datos-mas-usados.

Garibal, Cadenas Victor. 2016. JSON marcando tendencias. [En línea] 2016. [Citado el: 27 de 11 de 2019.] https://medium.com/@victor.garibayy/qu%C3%A9-es-y-para-qu%C3%A9-sirve-json-be05fe02e67d.

Ibrugo. 2014.*Ibrugo.* [En línea] 11 de 6 de 2014. [Citado el: 27 de 11 de 2019.] https://www.ibrugor.com/blog/apache-http-server-que-es-como-funciona-y-para-que-sirve/.

LIGP. 2017. Laboratorio de Investigaciones en Gestión de Proyectos. Ecosistema de software para la

Dirección Integrada de Proyectos. 2017.

miGabeta. 2017. [En línea] 12 de 1 de 2017. [Citado el: 28 de 11 de 2019.] https://migabeta.wordpress.com/2017/01/12/arquitectura-de-capas/.

NetBeans.org. 2018. [En línea] 2018. [Citado el: 29 de 11 de 2019.] https://netbeans.org/kb/docs/ide/git.html.

—. 2018. NetBeans. *NetBeans.* [En línea] 2018. [Citado el: 27 de 11 de 2019.] https://netbeans.org/kb/docs/ide/git.html.

Nex_U. 2018. Nex_U. *Nex_U.* [En línea] 2018. [Citado el: 27 de 11 de 2019.] https://www.nextu.com/blog/conoce-las-ventajas-y-desventajas-de-javascript/.

Oracle. 2018. netbeans.org. NetBeans IDE Features. *netbeans.org. NetBeans IDE Features.* [En línea] 2018. [Citado el: 28 de 11 de 2019.] https://netbeans.org/features/index.html.

—. 2019. Oracle. *Oracle.* [En línea] 2019. https://www.oracle.com/ar/database/what-is-a-relational-database/.

—. 2019. Oracle. *Oracle.* [En línea] 2019. [Citado el: 27 de 11 de 2019.] https://www.oracle.com/ar/database/what-is-a-relational-database/.

org, Conogasi. 2018. [En línea] 23 de 4 de 2018. [Citado el: 29 de 11 de 2019.] http://conogasi.org/articulos/lenguaje-de-programacion/.

Pérez , Porto Julián y Gardey, Ana. 2018. Definicion.de. *Definicion.de.* [En línea] 2018. https://definicion.de/estandarizacion/.

PostgreSQL, El equipo de desarrollo de.PostgreSQL_JSON. 2019. [En línea] 2019. [Citado el: 29 de 11 de 2019.] http://www.postgresqltutorial.com/postgresql-json/.

Pressman, Roger S. 2010.*La Ingeniería del Software, un enfoque práctico.* Mexico : s.n., 2010. 978-607-15-0314-5.

Rock , Content Redator . 2019. [En línea] 5 de 6 de 2019. [Citado el: 27 de 11 de 2019.] https://rockcontent.com/es/blog/que-es-java/.

—. 2019. [En línea] 5 de 6 de 2019. https://rockcontent.com/es/blog/que-es-java/.

Rodríguez, Peña Alina Dolores y Silva , Rojas Luis Guillermo . 2016. Revista Cubana de Informática Médica. 2016, Vol. vol.8 no.1.

Rouse, Margaret . 2019. SearchDataCenter en Español. *SearchDataCenter en Español.* [En línea] 9 de 2019. https://searchdatacenter.techtarget.com/es/definicion/Almacen-operacional-de-datos-ODS.

Rouse, Margaret . 2015. SearchDataCenter en Español. *SearchDataCenter en Español.* [En línea] enero de 2015. [Citado el: 27 de 11 de 2019.] https://searchdatacenter.techtarget.com/es/definicion/SQL-o-lenguaje-de-consultas-estructuradas.

Somos_tu_aliado_tecnologico. 2015. Somos tu aliado tecnologico. *Somos tu aliado tecnologico.* [En línea] 6 de 2015. [Citado el: 29 de 11 de 2019.] http://programaenlinea.net/que-es-java-hibernate/.

UCI. Universidad de las Ciencias Informaticas. [En línea] [Citado el: 13 de 1 de 2020.] https://www.uci.cu/investigacion-y-desarrollo/productos/xedro/gespro-1305.

Universidad de las Ciencias Informaticas . [En línea] [Citado el: 10 de 12 de 2019.] https://www.uci.cu/vida-universitaria/informacion-cientifico-tecnica.

Valderrama, Johan Sebastián. 2016. [En línea] 2016. [Citado el: 27 de 11 de 2019.] https://www.mindmeister.com/es/742289673/herramientas-case.

Virtual, Ingenio. 2018. Ingenio Virtual. *Ingenio Virtual.* [En línea] enero de 2018. http://www.ingeniovirtual.com/.

w3resource. 2019. w3resource. [En línea] 9 de 11 de 2019. https://www.w3resource.com/PostgreSQL/pl-pgsql-tutorial.php.

ANEXOS

ANEXO 1: Encuesta realizada a los especialistas del proyecto CEGEL.

Esta encuesta es considerada anónima y personal, dirigida a los especialistas del proyecto CEGEL, que han interactuado en la confecciónde productos y soluciones de software que tienen como objetivo la gestión de las entidades.

El objetivo de la encuesta es determinar el tiempo promedio que demoran habitualmente en realizar las actividades comprendidas dichos especialistas, de ahí que su criterio sobre el rendimiento es muy importante para el perfeccionamiento de lainvestigación.

Por favor complete la encuesta cuidadosamente al leerla primero, y luego señale sus respuestas con una "X".

1. ¿Qué tiempo demora al buscar una determinada información?

   _Menos de 1 minuto.

   _De 1 a 5 minutos.

   _De 5 a 10 minutos.

   _Otro ¿Cuál?_____

2. ¿Qué tiempo demora en recopilar una x cantidad de datos necesarios?

   _Menos de 20 minutos.

   _De 20 a 30 minutos.

   _De 30 minutos a 1 hora.

   _De 1 a 2 horas.

   _Otro ¿Cuál?_____

3. ¿Qué tiempo demoran en emitir un reporte?

_Menos de 30 minutos.

_1 hora

_2 horas

_3 horas

_Otro ¿Cuál?_____

ANEXO 2: Entrevista realizada alespecialista: Yosvany Gómez Perdomo

Objetivo: Identificar los principales elementos que componen la búsqueda.

Preguntas:

1. ¿Qué datos componen la búsqueda?
2. ¿Qué detalles se van a mostrar?
3. ¿Qué detalles se van a mostrar al emitir un reporte?
4. ¿Qué datos se deben registrar al adicionar nueva información a la base d datos?