



ESPE

UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA

**DEPARTAMENTO DE CIENCIAS DE LA
COMPUTACIÓN**

CARRERA DE INGENIERÍA EN SISTEMAS E INFORMÁTICA

**TRABAJO DE TITULACIÓN PREVIO A LA OBTENCIÓN DEL
TÍTULO DE INGENIERO EN SISTEMAS E INFORMÁTICA**

**TEMA: DESARROLLO DE UN MODELO PARA EVALUACIÓN
DE GESTORES DE BASE DE DATOS NO RELACIONALES**

**AUTORES: BRAVO MALDONADO, PAULO DENNIS
ÑAUÑAY COLCHA, JORGE LUIS**

DIRECTOR: ING. ALMACHE CUEVA, MARIO

CODIRECTOR: ING. DUQUE, LORENA

SANGOLQUÍ

2018

CERTIFICADO

CERTIFICADO



DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN
CARRERA DE INGENIERÍA EN SISTEMAS E INFORMÁTICA

CERTIFICACIÓN

Certifico que el trabajo de titulación, "**DESARROLLO DE UN MODELO PARA EVALUACIÓN DE GESTORES DE BASE DE DATOS NO RELACIONALES**" realizado por los señores **BRAVO MALDONADO PAULO DENNIS** y **ÑAÑAY COLCHA JORGE LUIS**, ha sido revisado en su totalidad y analizado por el software anti-plagio, el mismo cumple con los requisitos teóricos, científicos, técnicos, metodológicos y legales establecidos por la Universidad de Fuerzas Armadas ESPE, por lo tanto me permito acreditarlo y autorizar a los señores **BRAVO MALDONADO PAULO DENNIS** y **ÑAÑAY COLCHA JORGE LUIS** para que lo sustente públicamente.

Sangolquí, 13 de noviembre del 2017

Atentamente,


Ing. Almaché Cueva Mario
Director del proyecto


Ing. Duque Lorena
Co-Director del proyecto

AUTORÍA DE RESPONSABILIDAD

iii

AUTORÍA DE RESPONSABILIDAD



DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN
CARRERA DE INGENIERÍA EN SISTEMAS E INFORMÁTICA

AUTORÍA DE RESPONSABILIDAD

Nosotros, **BRAVO MALDONADO PAULO DENNIS**, con cédula de identidad N° 1721715314 y **ÑAUÑAY COLCHA JORGE LUIS**, con cédula de identidad N° 1721581062 declaramos que este trabajo de titulación "**DESARROLLO DE UN MODELO PARA EVALUACIÓN DE GESTORES DE BASE DE DATOS NO RELACIONALES**" ha sido desarrollado considerando los métodos de investigación existentes, así como también se ha respetado los derechos intelectuales de terceros considerándose en las citas bibliográficas.

Consecuentemente declaramos que este trabajo es de nuestra autoría, en virtud de ello nos declaramos responsables del contenido, veracidad y alcance de la investigación mencionada

Sangolquí, 13 de noviembre del 2017

BRAVO MALDONADO PAULO DENNIS
C.C. 1721715314

ÑAUÑAY COLCHA JORGE LUIS
C.C 1721581062

AUTORIZACIÓN

iv

AUTORIZACIÓN



DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN
CARRERA DE INGENIERÍA EN SISTEMAS E INFORMÁTICA

AUTORIZACIÓN

Nosotros, **BRAVO MALDONADO PAULO DENNIS** y **ÑAUÑAY COLCHA JORGE LUIS**, autorizamos a la Universidad de las Fuerzas Armadas ESPE publicar en la biblioteca Virtual de la institución el presente trabajo de titulación "**DESARROLLO DE UN MODELO PARA EVALUACIÓN DE GESTORES DE BASE DE DATOS NO RELACIONALES**" cuyo contenido, ideas y criterios son de nuestra autoría y responsabilidad.

Sangolquí, 13 de noviembre del 2017

BRAVO MALDONADO PAULO DENNIS

C.C. 1721715314

ÑAUÑAY COLCHA JORGE LUIS

C.C 1721581062

DEDICATORIA

El presente trabajo va dedicado a Dios en primer lugar por otorgarme fuerza y perseverancia a lo largo de la realización de este trabajo. A mis padres por su amor y apoyo que han sido de aliento a lo largo de mis estudios universitarios

A la Universidad de las Fuerzas Armadas - ESPE y docentes por los conocimientos y tiempo impartidos con el propósito de formar profesionales y personas de bien.

Bravo Maldonado Paulo Dennis

Dedico este trabajo principalmente a Dios, por haberme dado la vida y permitirme haber llegado hasta este momento tan importante de mi formación profesional, a mi amado hijo Mateo Sebastián por ser mi fuente de motivación e inspiración para poder superarme cada día más, a mi madre por ser el pilar más importante y demostrarme siempre su cariño y apoyo incondicional, y mi padre que siempre ha estado presente para guiarme.

Ñauñay Colcha Jorge Luis

AGRADECIMIENTO

Agradezco a Dios por la salud, guía y protección brindada a lo largo de mis estudios universitarios, permitiéndome culminarlos con la realización del presente trabajo.

A mis padres por su confianza y apoyo en los momentos buenos y difíciles.

Al Ing. Mario Almache y a la Ing. Lorena Duque por su guía y conocimientos brindados ya que fueron de gran ayuda para la realización del presente trabajo.

Finalmente a la Universidad de las Fuerzas Armadas – ESPE por los conocimientos adquiridos.

Bravo Maldonado Paulo Dennis

Agradezco a Dios a mis padres y especialmente a mis hermanos Guillermo, Darío, Rocío y mi cuñada Marieta quienes con su ayuda han sido parte fundamental en mi vida.

A mi Director y Codirector de tesis, Ing. Mario Almache e Ing. Lorena Duque por toda la colaboración brindada durante la elaboración de este proyecto.

Esto es para ustedes y para mis amigos que están conmigo, para las personas que siempre han estado ahí para apoyarme con una mirada, un consejo, un aliento, hoy concluyo un ciclo, que me llena de grandes satisfacciones.

Ñauñay Colcha Jorge Luis

ÍNDICE DE CONTENIDOS

CERTIFICADO	ii
AUTORÍA DE RESPONSABILIDAD	iii
AUTORIZACIÓN	iv
DEDICATORIA	v
AGRADECIMIENTO	vi
ÍNDICE DE FIGURAS	xi
RESUMEN	xii
ABSTRACT	xiii
CAPITULO I	1
INTRODUCCIÓN	1
1.1. Antecedentes	1
1.2. Planteamiento del problema	2
1.3. Justificación	3
1.4. Alcance	3
1.5. Objetivos	4
1.5.1. General.....	4
1.5.2. Específicos.....	4
CAPÍTULO II	5
MARCO TEÓRICO	5
2.1. Introducción	5
2.2. Bases de datos no relacionales	5
2.2.1 Definición.....	5
2.2.2. Definiciones de NoSQL.....	5
2.2.3. Qué no es NoSQL.....	6
2.2.4. Características de NoSQL.....	6
2.2.5. Ventajas de las Bases de Datos NoSQL.....	7
2.2.6. Desventajas de las bases de datos NoSQL.....	8
2.2.7. Tipos de NoSQL.....	8
2.2.8. Gestores de Bases de Datos no relacionales.....	19
2.3. Método IQMC	34
2.4. Teorema de Brewer o CAP	35
2.5. Calidad de software y análisis de los modelos de calidad	37
2.5.1. Calidad de software.....	37
2.5.2. Estándar o norma de calidad.....	39
2.5.3. Modelo de calidad.....	40

2.5.4. Tipos de modelos de calidad.....	42
2.5.5. Modelos de calidad de McCall y Bohem.....	43
2.5.6. Modelo de Calidad FURPS.....	47
2.5.7. Estándar ISO/IEC 9126.....	48
2.6. Selección de un modelo de calidad.....	55
2.7. Antecedentes de los modelos de evaluación.....	57
2.7.1. Análisis de los antecedentes.....	60
2.7.2. Conclusión del análisis de antecedentes.....	60
CAPITULO III.....	62
DESARROLLO DEL MODELO.....	62
3.1. Introducción.....	62
3.2. Estudio de ambito de software.....	63
3.3. Determinación de características y subcaracterísticas de calidad.....	65
3.4. Determinación de atributos derivados y básicos.....	74
3.5. Deterinación de métricas para los atributos.....	79
3.6 Selección de nivel de importancia de las características.....	93
CAPITULO IV.....	94
APLICACIÓN Y PRUEBAS.....	94
4.1 Análisis caso de estudio elección de bases de datos No Sql....	94
4.2 Evaluacion de los gestores de bases de datos No Sql.....	101
4.2.1 Aplicación del modelo evaluador de gestores de base de datos NoSQL.....	102
4.3 Resultado del análisis comparativo.....	113
4.4 Comparativa de resultados modelo evaluador planteado Vs. caso de estudio.....	114
CAPITULO V.....	115
CONCLUSIONES Y RECOMENDACIONES.....	115
5.1 CONCLUSIONES.....	115
5.2 RECOMENDACIONES.....	116
REFERENCIAS BIBLIOGRAFICAS.....	117

ÍNDICE DE TABLAS

Tabla 1	Características del motor de base de datos Redis.....	20
Tabla 2	Características del motor de base de datos Memcached.....	21
Tabla 3	Características del motor de base de datos Riak-KV.....	22
Tabla 4	Características del motor de base de datos Mongo-DB.....	23
Tabla 5	Características de motor de base de datos Amazon DynamoDB.....	24
Tabla 6	Características del motor de base de datos Couchbase.....	26
Tabla 7	Características del motor de base de datos CouchDB.....	27
Tabla 8	Características del motor de base de datos Cassandra.....	28
Tabla 9	Características del motor de base de datos HBase.....	29
Tabla 10	Características del motor de base de datos Hypertable.....	30
Tabla 11	Características del motor de base de datos Neo4J.....	31
Tabla 12	Características del motor de base de datos OrientDB.....	32
Tabla 13	Características del motor de base de datos Titan.....	33
Tabla 14	Factores y criterios de McCall.....	45
Tabla 15	Características de FURPS.....	47
Tabla 16	Características importantes para la selección de un modelo de calidad.....	56
Tabla 17	Características base de datos NoSql.....	62
Tabla 18	Características y subcaracterísticas seleccionadas.....	66
Tabla 19	Triangulación de fuentes.....	70
Tabla 20	Características, subcaracterísticas atributos de calidad.....	75
Tabla 21	Métrica de cumplimiento.....	79
Tabla 22	Métrica de cumplimiento por rangos.....	79
Tabla 23	Métrica de cumplimiento de utilidad.....	79
Tabla 24	Métrica de cumplimiento.....	79
Tabla 25	Modelo evaluador de gestores de base de datos no SQL.....	81
Tabla 26	Niveles de importancia por características.....	93

Tabla 27	Aplicación del modelo evaluador de gestores de base de datos no SQL.....	103
Tabla 28	Resultado del modelo evaluador de gestores de base de datos no SQL.....	113

ÍNDICE DE FIGURAS

Figura 1	Árbol de problemas mala calidad de datos.....	3
Figura 2	Aceptación de los tipos de bases de datos.....	9
Figura 3	Modelo Clave-Valor.....	10
Figura 4	Estructura clave-valor documento.....	10
Figura 5	Ejemplo de base de datos orientado columnas.....	12
Figura 6	Estructura de datos tradicional.....	13
Figura 7	Estructura de datos columnar.....	13
Figura 8	Código de modelo orientado a documento.....	14
Figura 9	Base de datos tradicional vs Base de datos orientado a documentos.....	15
Figura 10	Ejemplo de un documento en formato JSON.....	15
Figura 11	Estructura documento con campos.....	16
Figura 12	Ejemplo de base de datos de grafo.....	17
Figura 13	DB-Engines Ranking de gestores clave-valor.....	19
Figura 14	DB-Engines Ranking de gestores tipo documento.....	23
Figura 15	DB-Engines Ranking de gestores orientados a columnas.....	28
Figura 16	DB-Engines Ranking de gestores orientados a grafos.....	31
Figura 17	Calidad del producto software según la ISO/IEC 25010.....	51
Figura 18	Modelo de Calidad en uso.....	52
Figura 19	Estructura usada por modelos de calidad.....	63
Figura 20	Diagrama triangulación de fuentes.....	69
Figura 21	Características seleccionadas en el caso de estudio.....	94
Figura 22	Encuesta de la investigación analizada.....	98
Figura 23	Lista de chequeo de la investigación analizada.....	99
Figura 24	Resultados lista de chequeo de la investigación analizada.....	100
Figura 25	Análisis comparativo entre gestores de base de datos no SQL.....	113

RESUMEN

El volumen de información en el mundo crece exponencialmente con la aparición de las redes sociales y las nuevas tecnologías, que generan información del tipo estructurada y no estructurada, esto crea en los desarrolladores de software y empresas la opción de buscar nuevas alternativas para la persistencia de datos; para esto existe un gran catálogo de gestores de bases de datos no relacionales conocidas como NoSQL, debido a esta gran diversidad es importante conocer qué tipo de base de datos se debe elegir con el propósito de evitar la pérdida de la calidad de datos, posibles pérdidas de información incluso migración de una base de datos relacional a una no relacional. Para ello el presente trabajo presenta el desarrollo de un modelo de evaluación de gestores de bases de datos no relacionales, que se fundamenta en la aplicación del método Individual Quality Model Construction (IQMC) para la construcción de un modelo mixto que una las características y subcaracterísticas propias de gestores de bases de datos NoSQL con características del modelo de calidad de la ISO 25000 que permita evaluar gestores de bases de datos NoSQL dando como resultado la opción que más se ajuste a los criterios utilizados al momento de elegir un gestor de bases de datos no relacional.

PALABRAS CLAVE:

- **BASE DE DATOS**
- **NOSQL**
- **MODELO EVALUACIÓN**
- **IQMC**
- **ISO 25000**

ABSTRACT

The volumen of information in the world grows exponentially with the appearance of social networks and new technologies, what generate structure and unstructured information, this creates in the software developers and companies the option to look for new alternatives for the persistence of data; for that there is a large catalog of database managers known as NoSQL, because of this great diversity it is important to know what kind of database should be chosen in order to avoid loss of data quality, possible losses of information including migration from a relational database to a non-relational database. For this, the present work presents the development of an evaluation model of non relational database managers, which is based on the application of the Individual Quality Model Construction (IQMC) method for the construction of a mixed model that unites the characteristics and subcharacteristics specific to NoSQL database managers with characteristics of the ISO 25000 quality model that allows evaluating NoSQL database managers resulting in the option that best meets the criteria used when choosing a database manager not relational.

KEYWORDS:

- **DATABASES**
- **NOSQL**
- **EVALUATION MODEL**
- **IQMC**
- **ISO 25000**

CAPITULO I

INTRODUCCIÓN

1.1. ANTECEDENTES

La cantidad de información en el mundo está creciendo a gran velocidad, para los próximos años muchas empresas alcanzarán límites en cuanto al almacenamiento de información, es decir, que las bases de datos relacionales tienden a ser finitas en cuanto al almacenamiento y manipulación de la información; por ejemplo, “se proyecta un crecimiento del tráfico IP global, de los centros de datos, que en la actualidad es casi de un zetabyte (1.099.511.627.776 gigabytes) por año, a 4.8 zetabyte en el 2015” [CITATION Tiw11 \l 3082].

La nueva alternativa para el almacenamiento masivo de información son las bases de datos no relacionales o conocidas como bases de datos NoSQL.[CITATION Vai13 \t \l 3082]Estas bases de datos no necesitan tener una estructura en forma de tablas y relaciones entre ellas, por lo tanto son más flexibles y permiten almacenar información en otros formatos teniendo como principal ventaja la escalabilidad horizontal y la velocidad.

Actualmente este enfoque está siendo aplicado a proyectos de gran alcance, en empresas importantes a nivel mundial, con el propósito de almacenar grandes volúmenes de datos, obteniendo logros como: amplia aplicabilidad, escalabilidad, alto rendimiento, alta disponibilidad, etc. siendo usado por más de 60 productos Google incluidos Google Finanzas, Google Earth y Google Analytics.[CITATION FAY13 \l 3082]

Tomando en consideración lo anterior, la complejidad de los sistemas de información van aumentando a tal punto que se dedica mayor importancia a la evaluación de la calidad de las bases de datos[CITATION Loz00 \t \l 3082],y a la utilización de modelos de evaluación que permiten medir la calidad del software como la norma ISO 25000; y así ayudar a las

personas a seleccionar el gestor de base de datos acorde a sus necesidades[CITATION ISO99 \l 3082].

1.2. PLANTEAMIENTO DEL PROBLEMA

Las bases de datos relacionales están diseñadas para ser organizadas en tablas de tal forma que se pueda entender; lastimosamente, cuando estos sistemas fueron diseñados se pensaba en sistemas pequeños, estructurados y centralizados, pero eso cambio y la información dejó de ser tan “estructurada”, los sistemas crecieron a un ritmo exponencial y se hizo necesario distribuir la información, lo que ocasionó que estas bases de datos sean más lentas [CITATION Cas12 \l 12298].

Grandes empresas como Facebook, Twitter, Amazon, Google entre otras, utilizan hoy en día las bases de datos no relacionales, demostrando que este tipo de bases de datos son la alternativa para mejorar la escalabilidad, flexibilidad, velocidad en la manipulación de grandes cantidades de datos y las diferentes formas de almacenamiento de la información[CITATION Ben13 \t \l 3082].

Debido a esto, las empresas deciden cambiar el uso de bases de datos relacionales por bases de datos no relacionales, donde la elección incorrecta de base de datos genera posibles pérdidas de información o baja calidad de los datos almacenados[CITATION Pow14 \t \l 3082], por lo tanto, es necesario identificar y proponer un modelo de evaluación basado en normas y estándares de la ISO 25000 que permita determinar criterios de evaluación al momento de seleccionar una base de datos NoSQL adecuada para el negocio.

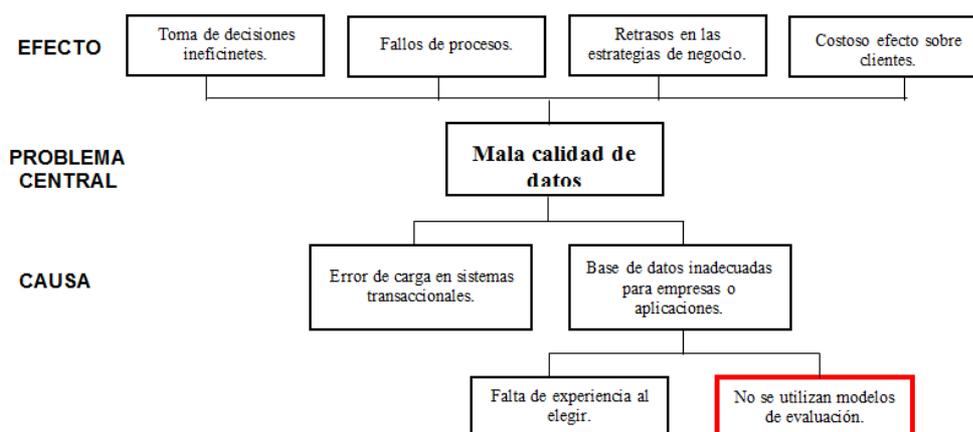


Figura 1 Árbol de problemas mala calidad de datos

Fuente: Autores del trabajo de titulación

1.3. JUSTIFICACIÓN

El desarrollo de un modelo de evaluación para gestores de bases de datos no relacionales, permitirá a las personas seleccionar la opción más adecuada o la que se adapte a las reglas del negocio, para evitar posibles pérdidas económicas o de información, con un modelo que analice los atributos claves obtenidos de las características y subcaracterísticas de las bases de datos no relacionales.

Para la elaboración del modelo se utiliza como referencia normas de evaluación de calidad de software como la norma ISO 25000 para garantizar que el proceso de evaluación sea el correcto.

1.4. ALCANCE

Para el desarrollo del modelo de evaluación se necesitan criterios que permitan valorar las bases de datos no relacionales; para ello se deberán identificar los criterios de evaluación más importantes aplicables a los cuatro tipos de gestores de bases de datos no relacionales:

- Base de datos orientada a Documento
- Base de datos en Grafos
- Base de datos Clave/Valor
- Base de datos Columnas.

Para ello se utilizará la norma ISO 25000 que cuenta con criterios de calidad de software y el método IQMC que permite desarrollar el modelo de calidad.

1.5. OBJETIVOS

1.5.1. General

Desarrollar un modelo de evaluación para gestores de base de datos no relacionales, basados en el método IQMC, que permita a las personas seleccionar la base de datos apropiada para sus aplicaciones o negocios.

1.5.2. Específicos

- Identificar criterios de evaluación de bases de datos NoSQL, en base a la norma ISO 25000.
- Desarrollar el modelo de evaluación para bases de datos NoSQL utilizando el método IQMC.
- Validar el modelo de evaluación, comparando los resultados obtenidos con otros modelos o estándares aplicados a un caso de estudio.

CAPÍTULO II

MARCO TEÓRICO

2.1. INTRODUCCIÓN

En esta parte del documento se hace mención a un grupo de conceptos que son importantes para el desarrollo de la investigación y que serán referenciados en el resto del documento, introduciendo el tema de las bases de datos NoSQL, características y ventajas. Asimismo, se efectúa un conjunto de consideraciones referidas al tema calidad de software y se analizan algunos modelos y estándares de calidad.

2.2. BASES DE DATOS NO RELACIONALES

2.2.1. Definición

En el entorno de las bases de datos existen diferentes tipos, entre ellas las tradicionales bases de datos relacionales y otro grupo llamado bases de datos no relacionales o NoSQL.

Las bases de datos no relacionales son estructuras dinámicas, que no utilizan tablas ni relaciones entre ellas, que permiten agregar atributos a los registros sin la necesidad de rediseñar el modelo de datos utilizado. [CITATION MarcadorDePosición1 \t \l 3082]

Este tipo de bases de datos utiliza el término NoSQL, que proviene de la unión de dos palabras: No y SQL, que significan “No Only SQL”.

2.2.2. Definiciones de NoSQL

“NoSQL es un conjunto de conceptos que permite el procesamiento rápido y eficiente de los conjuntos de datos con un enfoque en el rendimiento, la confiabilidad y la agilidad” [CITATION McC13 \t \l 3082]

Los tipos de bases de datos no relacionales son el conjunto de conceptos que definen a NoSQL, destacados por no seguir un modelo de almacenamiento relacional.

“NoSQL es un término genérico utilizado para referirse a cualquier almacén de datos que no sigue el modelo tradicional RDBMS, los datos no son relacionales y no utilizan SQL como lenguaje de consulta.”[CITATION Vai131 \t \l 3082]

Las anteriores definiciones mencionan que con NoSQL se puede tener un procesamiento rápido y eficiente de datos sin la necesidad de seguir un modelo de datos relacional.

2.2.3. Qué no es NoSQL

Una vez que se conoce lo que es NoSQL, se debe definir lo que no es NoSQL, con el propósito de entender por qué a una base de datos no relacional se le conoce con el término de NoSQL.

“NoSQL no es una base de datos. Ni siquiera es un tipo de base de datos. De hecho es un término que se utiliza para filtrar un conjunto de bases de datos que no son relacionales”[CITATION Vai13 \t \l 3082].

Al utilizar el término “bases de datos NoSQL”, no se menciona a un tipo de base de datos, sino que es un término que representa a un conjunto de bases de datos que no pertenecen a las bases de datos relacionales, incluso se hace referencia a bases de datos que tratan de resolver los problemas de escalabilidad y disponibilidad.[CITATION Vai13 \t \l 3082]

2.2.4. Características de NoSQL

Las bases de datos NoSQL tienen características que permiten el crecimiento y la integración de los datos, que con una base de datos relacional se volvería difícil de realizar, por esa razón se proponen seis

características específicas para poder encasillar a las bases de datos NoSQL

- Escalabilidad horizontal: refiriéndose a la facilidad añadir, eliminar o realizar operaciones con elementos (hardware) del sistema, sin afectar el rendimiento.
- Habilidad de distribución: tiene que ver con la escalabilidad horizontal, pero haciendo énfasis en su soporte; para ello se tiene en cuenta la habilidad de replicar y distribuir los datos sobre los servidores.
- Uso eficiente de recursos: aprovecha las nuevas tecnologías, como los discos en estado sólido, el uso eficiente de recursos como la memoria RAM y los sistemas distribuidos en general.
- Libertad de esquema: Las bases de datos NoSQL al no manejar un modelado de datos entidad-relación tiene como resultado modelo de datos flexibles que permiten con mayor facilidad la integración con los lenguajes de programación orientados a objetos, evitando el proceso de mapeado.
- Modelo concurrencia débil: no implementa ACID (Atomicity, Consistency, Isolation and Durability), que reúne las características necesarias para que una serie de instrucciones puedan ser consideradas una transacción, sin embargo, sí se tienen en cuenta algunas consideraciones para asegurar estos aspectos, pero no son tan estrictas.
- Consultas simples: las consultas requieren menos operaciones y son más naturales, por lo tanto, se gana en simplicidad y eficiencia.

2.2.5. Ventajas de las Bases de Datos NoSQL

Las ventajas y motivos principales para utilizar las bases de datos NoSQL son:[CITATION Gra13 \t \l 12298]

- Almacenamiento de grandes cantidades de datos menos estructurados.

- La escalabilidad horizontal de manera sencilla, sin demasiados problemas de rendimiento.
- Velocidad de desarrollo, evitando complejas sentencias SQL.
- Velocidad de respuesta de las bases de datos

2.2.6. Desventajas de las bases de datos NoSQL

Las desventajas principales que hacen que las bases de datos NoSQL sean una segunda opción de persistencia de datos:

- Falta de ayuda y soporte técnico.
- Falta de madurez.
- Limitaciones para el uso de Inteligencia de Negocios.
- Falta de experiencia.
- Problemas de compatibilidad.

2.2.7. Tipos de NoSQL

El término NoSQL engloba a bases de datos que no utilizan un modelo relacional, sino que utilizan diferentes tipos de almacenamiento de datos los cuales son:

- Clave-Valor
- Orientado a Documentos
- Orientado a columnas
- Grafos

Los sitios web sobre el ranking de motores de bases de datos, concuerdan con los tipos de base de datos NoSQL anteriormente mencionados, debido a que el método de cálculo de puntuación utiliza los siguientes parámetros: número de menciones en sitio web, interés general en el sistema, frecuencia de discusiones técnicas sobre los sistemas, número de ofertas de trabajo donde se mencionan estas bases de datos y relevancia en redes profesionales y sociales. [CITATION Sol121 \l 1033]

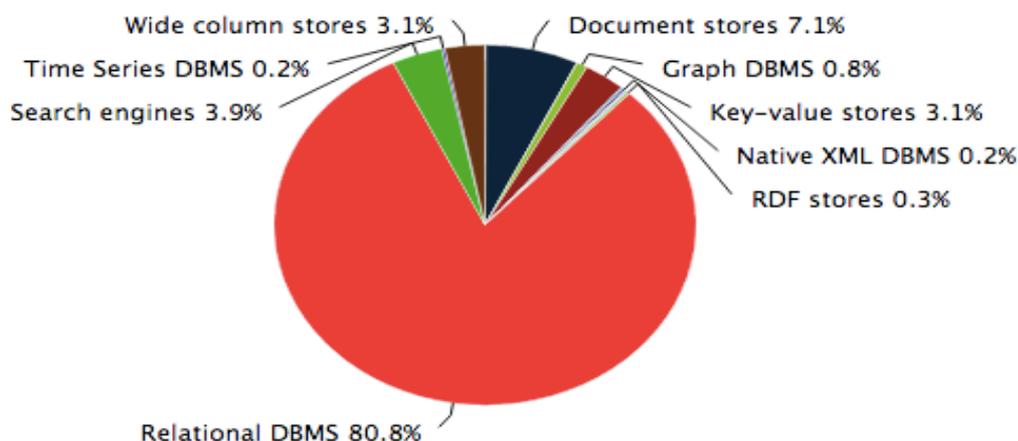


Figura 2 Aceptación de los tipos de bases de datos

Fuente: [CITATION Sol16 \l 3082]

1. Clave-Valor

Los sistemas de base de datos dentro del ámbito NoSQL basados en el modelo clave-valor son posiblemente los sistemas más sencillos utilizados para almacenar información. Se dice que estos modelos almacenan la información en forma de diccionarios, cuyas aplicaciones son altamente funcionales y con un alto grado de rendimiento y curva de escalabilidad ya que la estructura que lo compone tienen una gran simplicidad.[CITATION jfe14 \l 3082]

En este modelo simplemente se guardan tuplas que contienen una clave y su valor. Cuando se quiere recuperar un dato, simplemente se busca por su clave y se recupera el valor, lo que permite la recuperación de la información de forma muy rápida. Es decir la información o valor es almacenado en relación directa con una clave única que identifica la información. (Ver figura 3).

Clave	Valor
nombre	Pedro Ochoa

Figura 3 Modelo Clave-Valor

Fuente: Autores del trabajo de titulación

Las consultas a realizar sobre estos sistemas solo pueden ser ejecutadas en relación a la clave, es decir, no están permitidas las búsquedas por valor por lo que es mandatorio conocer la clave relacionada para poder extraer la información.[CITATION jfe14 \l 3082]

En un sistema clave-valor existen contenedores, también se les llama gabinetes, en cada uno de ellos se puede tener tantas parejas de clave-valor como se pretenda. Hay sistemas que permiten tener claves duplicadas y hay otros que no, o que se puede indicar que no se requiere que se dupliquen. En cada contenedor es posible tener datos de la misma naturaleza (por ejemplo productos, pedidos, clientes, etc.) o totalmente diferentes (puede haber un contenedor por cliente).[CITATION Ing12 \l 3082]

A continuación, un ejemplo de una estructura de clave-valor.

```
tesis => {
    id:231,
    nombre: "Modelo de evaluación para gestores de bases de datos",
    tipo: "Tesis de grado"
}
```

Figura 4 Estructura clave-valor documento

Fuente: Autores del trabajo de titulación

Este tipo de base de datos es similar a la base de datos orientada a documentos, se almacena dentro de una clave, esto permite que las bases de datos orientadas a documentos y clave-valor puedan trabajar juntas originando bases de datos mixtas.[CITATION Gra13 \p 9 \l 12298] Se caracterizan por ser muy eficientes tanto para las lecturas como para las escrituras. Este modelo de datos está diseñado para almacenar grandes volúmenes de información.

Clave-valor es ampliamente utilizado para almacenar información sobre la sesión de un usuario, para datos estadísticos, opciones de configuración y es muy útil para el almacenamiento de información temporal que deba persistir en el tiempo. Se recomienda su uso en casos donde se necesita velocidad en las consultas o se tienen muchos datos con estructura simple que requieren ser procesados una y otra vez y tienen valores cambiantes, por ejemplo, listas de los más vendidos, carritos de la compra, las preferencias cliente, gestión de sesiones, rango de venta y catálogo de productos.[CITATION aIG07 \t \l 3082] Recientemente las bases de datos orientadas a clave/valor han aumentado en popularidad, algunos de las aplicaciones más conocidas que lo utilizan son: Riak, Redis entre otros.

Ventajas

- Registros repartidos entre muchos nodos.
- Colección de pares clave/valor.
- Cada clave es única.
- Difícil de mantener clave única cuando aumentan los datos.

2. Orientado a columnas

Este tipo de bases de datos almacenan los datos en forma de columna en comparación de las bases de datos relacionales que almacenan la información en filas. A primera vista son muy similares a las bases de datos relacionales, pero en realidad son muy diferentes. El uso de este tipo de base de datos permite que el almacenamiento sea de forma efectiva. Esto evita el consumo de espacio cuando se está almacenando datos nulos porque no almacena una columna cuando un valor no existe para esa columna.

Es conveniente el uso de las bases de dato relacionales cuando se necesita crear nuevas filas y recuperar varias columnas para la misma fila al mismo tiempo, mientras que es apropiado utilizar las bases de datos

orientadas a columnas cuando se necesitan crear nuevas columnas con muchos datos ya ingresados en las filas, pero pocas columnas a la vez. Este modelo es básicamente en una tupla compuesta por elementos. (Ver figura 5)

primer_nombre: Pedro
segundo_nombre: Benitez
codigo: 1002
genero: masculino

Figura 5 Ejemplo de base de datos orientado columnas

Fuente: Autores del trabajo de titulación

Cada unidad de datos se puede pensar como un conjunto de par clave-valor, donde la unidad puede ser identificada con una clave primaria.

Otra diferencia entre los sistemas de base de datos relacionales y los orientados a columnas, es la optimización de consultas para mejorar el tiempo de respuesta, donde se obtiene mayor velocidad de consultas en las bases de datos orientadas a columnas.[CITATION Tiw11 \t \l 12298]

Para comprender mejor como se almacenan los datos en una base de datos no relacional orientada a columna observe el ejemplo que se muestra en las Figuras a continuación. Se puede observar claramente como en una base de datos tradicional, (ver figura 6) la información se guarda en forma de filas, mientras que en una base de datos NoSQL orientada a columnas se almacena una forma diferente (ver figura 7). Con este cambio se gana mucha velocidad en lecturas, ya que, si se pretende consultar un número reducido de columnas, es muy rápido hacerlo.

Id	Nombre	Puesto
1	Juan	CTO
2	Maria	CEO
3	Ricardo	CTO

Figura 6 Estructura de datos tradicional

Fuente: Autores del trabajo de titulación

id	Valor	id	Valor
1	Juan	1	CTO
2	Maria	2	CEO
3	Ricardo	3	CTO

Figura 7 Estructura de datos columna

Fuente: Autores del trabajo de titulación

Ventajas

Algunos de los beneficios que ofrecen las bases de datos orientadas a columnas son:

- Alto rendimiento en las consultas de agregación (como COUNT, SUM, AVG, MIN, MAX)
- Alta eficiencia en la compresión y distribución de los datos.
- Verdadera escalabilidad y carga rápida de datos para grandes volúmenes de datos.
- Se usa en la mayoría de herramientas de inteligencia de negocios [CITATION Ing12 \l 3082]

3. Orientado a Documentos

Las bases de datos orientadas a documentos no son sistemas de gestión de documentos. La palabra documento hace referencia a un conjunto estructurado o semiestructurado de pares de clave/valor en documentos estándar como: XML, YAML, JSON y BSON. [CITATION Tiw11 \t \l 12298]

```

{
  Nombre:"Pepe",
  Direccion: "C/ San Juan 15",
  Hijos:[
    {Nombre:"Ana",Edad:"10"},
    {Nombre:"Pedro",Edad:"8"},
    {Nombre:"Juan",Edad:"5"},
    {Nombre:"Felix",Edad:"2"}
  ]
}

```

Figura 8 Código de modelo orientado a documento

Fuente: Autores del trabajo de titulación

A diferencia de una base de datos clave/valor, las bases de datos orientadas a documentos pueden realizar consultas a la clave del documento o a sus elementos, teniendo como resultado partes del documento [CITATION Bro14 \t \l 12298]. Es decir, los documentos se acceden usando una clave, cadena o ruta. Para obtener la información las bases de datos ofrecen una API o lenguaje de consultas que permite obtener el contenido de los documentos.

Una forma de entender a las bases de datos orientadas a documentos, es utilizando como analogía una tabla en una base de datos relacional; los documentos son similares a las filas de la tabla, los elementos que contienen los documentos son similares a las columnas de la tabla, y las colecciones que organizan a documentos con similar estructura son equivalentes a una tabla [CITATION Mon15 \t \l 12298].

Base de datos Relacional

Nombre	Edad	Fecha de nacimiento	Genero
Juan	23	12-02-2016	M
Andres	24	18-05-2016	F

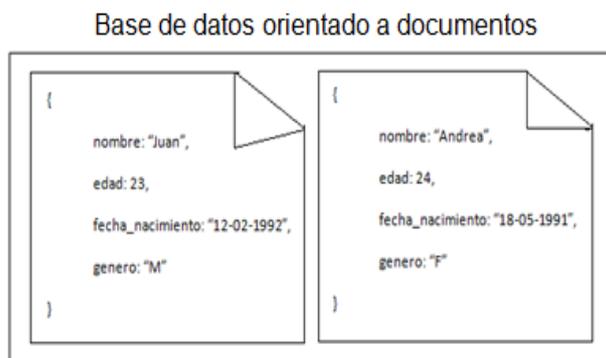


Figura 9 Base de datos tradicional vs Base de datos orientado a documentos

Fuente: Autores del trabajo de titulación

Las bases de datos orientadas a documentos tratan a los documentos como un todo evitando dividir a un documento en pares nombre/valor, permitiendo que una sola colección pueda tener diversos documentos. Las bases de datos orientadas a documentos realizan la indexación de documentos no solo en su identificador primario, sino también en sus propiedades.

```

{
  id: 231,
  nombre: "Modelo de evaluación para gestores de base de datos",
  tipo: "Tesis de grado"
}

```

Figura 10 Ejemplo de un documento en formato JSON

Fuente: [CITATION Vai131 \t \l 12298]

En la figura anterior se puede observar un ejemplo de la estructura de un documento, al estilo *clave: valor* parecido al *valor=dato de campo* de las bases de datos relacionales.[CITATION Gra13 \t \l 12298]

Comparando con una base de datos relacional, las bases de datos orientadas a documentos actúan por sí mismas como registros (filas), estos registros pueden tener diferentes conjuntos de campos o columnas.

```

{
  FirstName: "Jonathan",
  Address: "15 Wanamassa Point Road",
  Children: [
    {Name: "Michael", Age: 10},
    {Name: "Jennifer", Age: 8},
    {Name: "Samantha", Age: 5},
    {Name: "Elena", Age: 2}
  ]
}

{
  FirstName: "Bob",
  Address: "5 Oak St.",
  Hobby: "Sailing"
}

```

Figura 11 Estructura documento con campos.

Fuente. [CITATION Mig14 \l 3082]

Ventajas

- Documentos y arreglos implícitos reducen la necesidad de utilizar grandes joins.
- Uso de esquemas dinámicos.

Características principales

- Alto rendimiento: Los índices ayudan a realizar las consultas de forma rápida, debido a que los índices utilizan claves de documentos o arreglos embebidos.
- Alta disponibilidad: Utilizando las réplicas.
- Los índices pueden ser creados y consultados sin necesidad de tener una estructura definida.

Como beneficios de utilizar bases de datos orientadas a documentos se pueden mencionar el alto rendimiento y escala automática.

4. Grafos

Este modelo como su nombre indica, hace uso de grafos para almacenar la información. Un grafo es una colección de vértices o nodos unidos por aristas que permite relacionar los nodos entre sí.[CITATION jfe14 \l 3082]. Las relaciones que se manejan son solo binarias, donde un nodo solo puede usar una relación para contactar a otro y no más de uno.

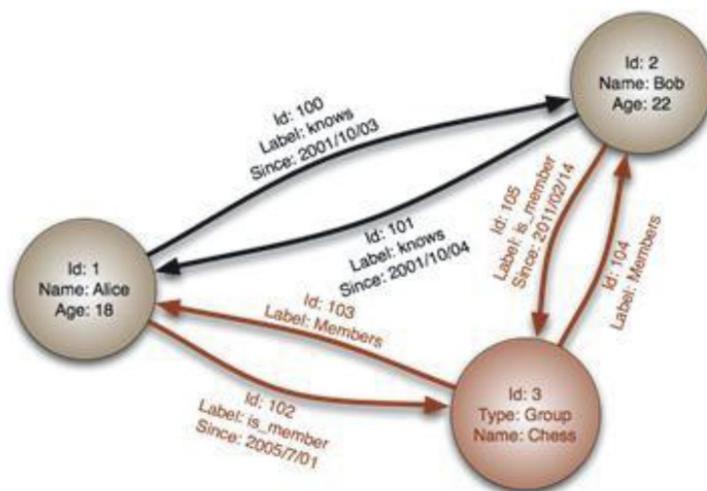


Figura 12 Ejemplo de base de datos de grafo

Fuente: [CITATION Ing12 \l 3082]

Las bases de datos orientadas a grafos representan la información como nodos de un grafo y sus relaciones con las aristas del mismo, de manera que se pueda recorrer la base de datos ya que esta puede describir atributos de los nodos (entidades) y las aristas (relaciones). [CITATION Mig14 \l 12298] Una base de datos orientada a grafos debe estar absolutamente normalizada, esto quiere decir que cada tabla tendría una sola columna y cada relación tan solo dos, con esto se consigue que cualquier cambio en la estructura de la información tenga un efecto tan solo local.[CITATION Ing12 \l 3082]

Características

- Presenta un buen rendimiento con comparación con las bases de datos relacionales.
- Flexibilidad, ya que permiten que las bases de datos crezcan sin problemas.
- Ofrece un alto nivel de escalabilidad.
- Agilidad ya que a medida que la aplicación vaya creciendo, también se precisa que el modelo de datos evolucione para adaptarse a las nuevas necesidades.
- Ofrece rendimiento cuando los datos están interconectados y no tabulares.

Los datos de grafos se almacenan en tuplas con múltiples atributos. Ejemplos de grafos son los datos de mapas de rutas, redes sociales y datos de recursos con sus relaciones.

Ventajas del modelo Grafo

- Son habitualmente más rápidas para conjuntos de datos asociativos[CITATION Mig14 \l 12298].
- Mapean fácilmente en aplicaciones orientadas a objetos[CITATION Mig14 \l 12298].
- Escalan de forma más natural hacia conjuntos de datos ya que no suelen requerir operaciones costosas basadas en joins. [CITATION Mig14 \l 3082]
- Una ventaja es la integridad de datos ya que cualquier cambio que se realice al nodo o relación solo afecta localmente[CITATION Gra13 \l 12298].

En la actualidad son muy utilizadas en la representación de información en las redes sociales. Ejemplos de sistemas de bases de datos de grafos son: Neo4j, FlockDB, Pregel.

2.2.8. Gestores de Bases de Datos no relacionales

En NoSQL existen diversos gestores de bases de datos con licencias de código libre que facilitan su adquisición y uso sin problema alguno.

Lista de SGBD NoSQL

El DB-Engines, clasifica a los gestores de base de datos en función de su popularidad, por tanto, se tomará los mejores gestores puntuados de cada tipo en el año 2016.

Clave valor

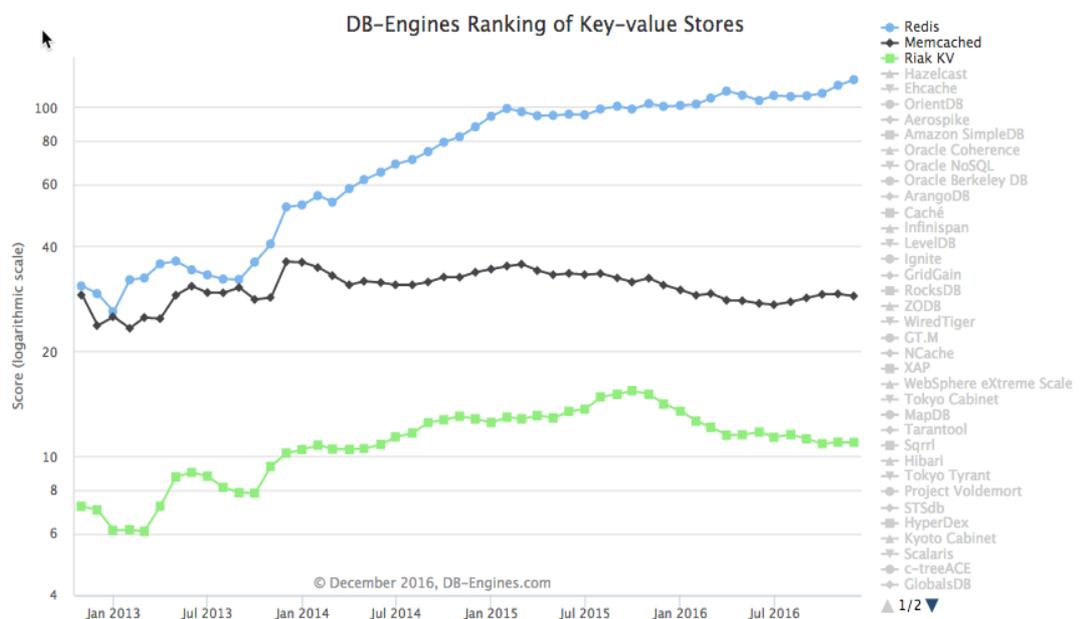


Figura 13 DB-Engines Ranking de gestores clave-valor

Fuente:[CITATION Sol16 \ 3082]

Redis

Es un motor de base de datos en memoria, pero también puede ser usada como una base de datos persistente, se utiliza principalmente en escenarios que requieren un rendimiento a escala o cuando necesita resolver complejos problemas.

Tabla 1
Características del motor de base de datos Redis

Características	
Año versión inicial	2009
Año última versión	2016
Licencia	Open Source
SO de servidor	BSD, Linux, OS X, Windows
Esquema de datos	Esquema-libre
Índices secundarios	No
SQL	No
Lenguajes de programación soportados	C, C# C++, Clojure Crystal, D Dart, Elixir Erlang, Fancy Go, Haskell Haxe, Java JavaScript (Node.js), Lisp Lua, MatLab Objective-C, OCaml Perl, PHP Prolog, Pure Data Python, R Rebol, Ruby Rust, Scala Scheme, Smalltalk Tcl
Procedimientos	Lua
Triggers	No
Particionamiento	Sharding
Replicación	maestro-esclavo
Map Reduce	No
Consistencia	Eventual
Transaccionalidad	No
Concurrencia	Si
Conceptos de usuarios	Basado en contraseñas simples

Fuente: [CITATION Sol16 \l 3082]

Memcached

Libre y de código abierto, de alto rendimiento, sistema de almacenamiento en caché de objetos de memoria distribuida, de carácter genérico, pero que se utilicen en la aceleración de aplicaciones.

Tabla 2
Características del motor de base de datos Memcached

Características	
Año versión inicial	2013
Año última versión	2016
Licencia	Open Source
SO de servidor	FreeBSD Linux, OS X Unix, Windows
Esquema de datos	Esquema Libre
Índices secundarios	No
SQL	No
Lenguajes de programación soportados	.Net, C C ++ , ColdFusion Erlang , Java Lisp, Lua OCaml, Perl PHP, Pitón Rubí, Linux OS X, Unix Windows
Procedimientos	No
Triggers	No
Particionamiento	Ninguno
Replicación	Ninguno
Map Reduce	No
Consistencia	-
Transaccionalidad	No
Concurrencia	Si
Conceptos de usuarios	Si

Fuente[CITATION Sol16 \l 3082]

Riak-KV

Está construida para manejar una variedad de desafíos que enfrentan las aplicaciones de grandes volúmenes de datos, Riak KV automatiza la distribución de datos a través del clúster para lograr un rendimiento rápido y con una arquitectura que garantiza una alta disponibilidad.

Tabla 3
Características del motor de base de datos Riak-KV

Características	
Año versión inicial	2009
Año última versión	2015
Licencia	Open Source
SO de servidor	Linux OS X
Esquema de datos	Esquema Libre
Índices secundarios	No
SQL	No
Lenguajes de programación soportados	.Net, C C ++, ColdFusion Erlang, Java Lisp, Lua Ocaml, Perl PHP, Pitón Rubí
Procedimientos	JavaScript and Erlang
Triggers	No
Particionamiento	Ninguno
Replicación	Ninguno
Map Reduce	No
Consistencia	Eventual
Transaccionalidad	No
Concurrencia	Si
Conceptos de usuarios	No

Fuente[CITATION Sol16 \l 3082]

Orientado a documentos

A continuación una figura que muestra el ranking de los gestores de bases de datos orientadas a documentos en el año 2016 los cuales se toman como muestra: MongoDB, Amazon DynamoDB, Couchbase y CouchDB.

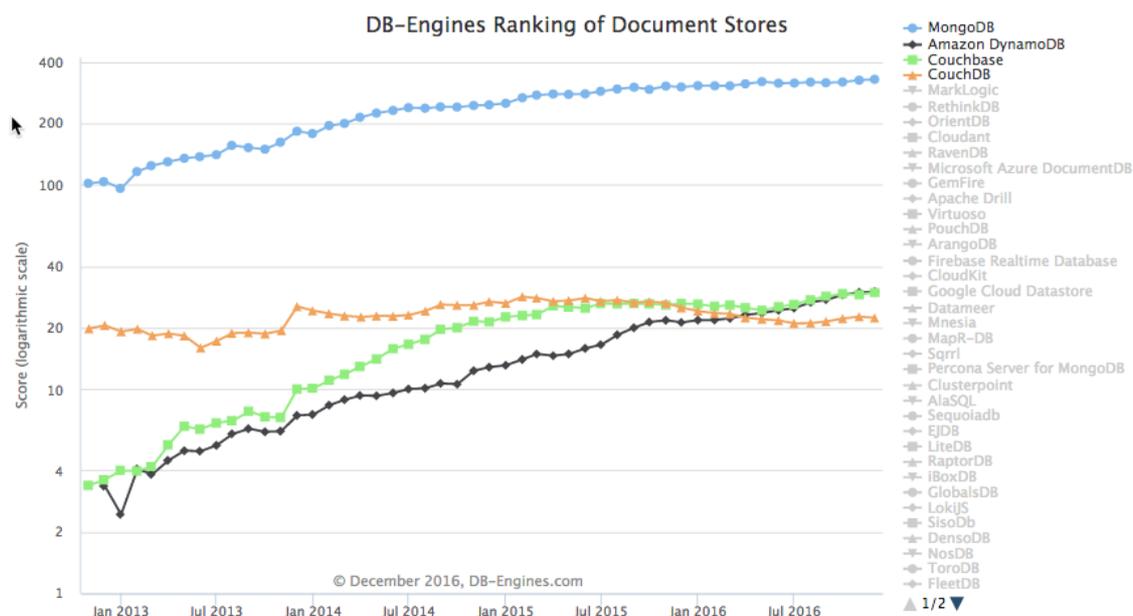


Figura 14 DB-Engines Ranking de gestores tipo documento

Fuente:[CITATION Sol16 \ 3082]

Mongo DB

Aprovechando el poder de los datos, es usada para crear aplicaciones nunca antes posibles, a una fracción del costo de las bases de datos existentes.

Tabla 4
Características del motor de base de datos Mongo-DB

Características	
Año versión inicial	2009
Año última versión	2016

CONTINUA →

Licencia	Open Source
----------	-------------

SO de servidor	Linux OS X Solaris Windows
Esquema de datos	Esquema Libre
Índices secundarios	Si
SQL	No
Lenguajes de programación soportados	Actionscript, C C#, C++, Clojure, ColdFusion D, Dart, Delphi, Erlang Go, Groovy, Haskell, Java JavaScript, Lisp, Lua , MatLab Perl, PHP, PowerShell , Prolog Python, R, Ruby, Scala Smalltalk
Procedimientos	JavaScript
Triggers	No
Particionamiento	Sharding
Replicación	Maestro – Esclavo
Map Reduce	Si
Consistencia	Eventual – Inmediata
Transaccionalidad	No
Concurrencia	Si
Conceptos de usuarios	Derechos de acceso para usuarios y roles

Fuente: [CITATION Sol16 \l 3082]

Amazon DynamoDB

Amazon DynamoDB es una forma rápida y flexible base de datos NoSQL servicio para todas las aplicaciones que necesitan, la latencia de milisegundos consistente de un solo dígito en cualquier escala.

Tabla 5
Características de motor de base de datos Amazon DynamoDB

Características	
Año versión inicial	2012
Año última versión	-
Licencia	Comercial

CONTINUA →

SO de servidor	Alojada
Esquema de datos	Esquema Libre
Índices secundarios	Si
SQL	No
Lenguajes de programación soportados	.Net, ColdFusion Erlang, Groovy Java, JavaScript Perl, PHP Python, Ruby
Procedimientos	No
Triggers	Si
Particionamiento	Sharding
Replicación	Si
Map Reduce	No
Consistencia	Eventual – Inmediata
Transaccionalidad	No
Concurrencia	Si
Conceptos de usuarios	Derechos de acceso para usuarios y funciones pueden ser definidas a través de la Identidad AWS y Access Management (IAM)

Fuente: [CITATION Sol16 \l 3082]

Couchbase

Couchbase Server es una base de datos NoSQL de código abierto diseñado para apoyar el desarrollo ágil y despliegue escalable de la web de la empresa, móviles y aplicaciones de IO.

Tabla 6
Características del motor de base de datos Couchbase

Características	
Año versión inicial	2011
Año última versión	2016
Licencia	Open source

SO de servidor	Linux OS X Windows
Esquema de datos	Esquema Libre
Índices secundarios	Si
SQL	No
Lenguajes de programación soportados	.Net, C Clojure, ColdFusion Erlang, Go Java, JavaScript Perl, PHP Python, Ruby, Scala, Tcl
Procedimientos	Funciones en JavaScript
Triggers	Si
Particionamiento	Sharding
Replicación	Maestro - Maestro, Maestro – Esclavo
Map Reduce	Si
Consistencia	Eventual – Inmediata
Transaccionalidad	No
Concurrencia	Si
Conceptos de usuarios	Usuario y Administrador de separación con la autenticación integrada basada en contraseña y LDAP

Fuente[CITATION Sol16 \l 3082]

CouchDB

Base de datos documental desarrollada con el objetivo de escalar horizontalmente con facilidad.

Tabla 7
Características del motor de base de datos CouchDB

Características	
Año versión inicial	2005
Año última versión	2016
Licencia	Open Source

SO de servidor	Android BSD Linux OS X Solaris Windows
Esquema de datos	Esquema Libre
Índices secundarios	Si
SQL	No
Lenguajes de programación soportados	C, C # ColdFusion, Erlang Haskell, Java JavaScript, Lisp, Lua, Objective-C Ocaml, Perl PHP, PL / SQL Python, Rubí, Smalltalk
Procedimientos	Funciones en JavaScript
Triggers	Si
Particionamiento	Sharding
Replicación	Maestro - Maestro, Maestro – Esclavo
Map Reduce	Si
Consistencia	Eventual
Transaccionalidad	No
Concurrencia	Si
Conceptos de usuarios	Derechos de acceso para los usuarios pueden ser definidos por base de datos

Fuente[CITATION Sol16 \l 3082]

Orientado a columnas

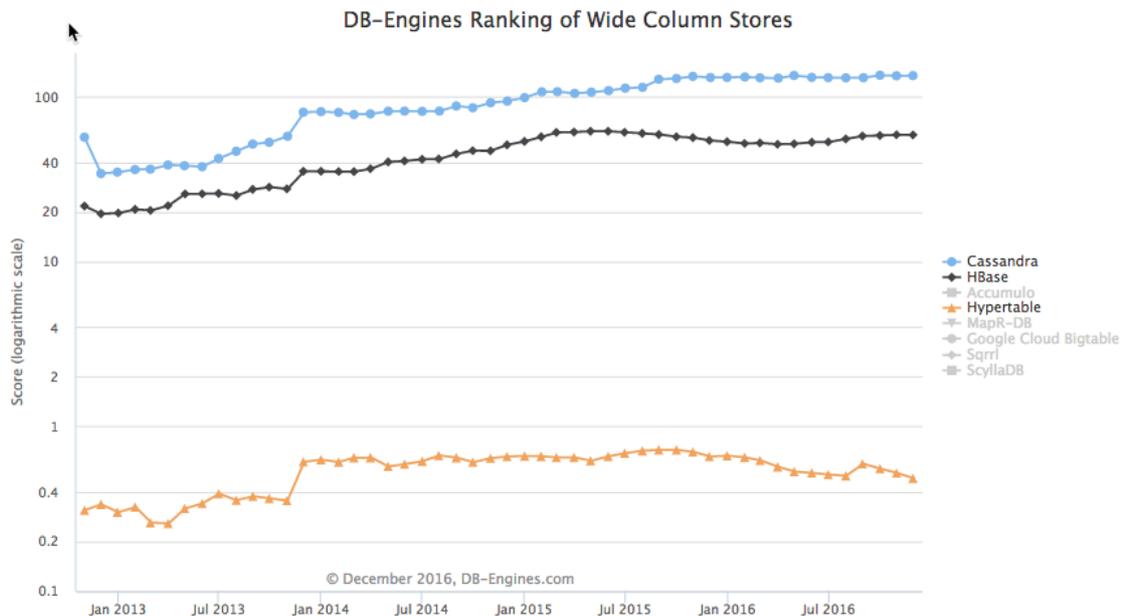


Figura 15 DB-Engines Ranking de gestores orientados a columnas

Fuente:[CITATION Sol16 \l 3082]

Cassandra

Apache Cassandra es el sistema de gestión de base de datos distribuida, ofreciendo una disponibilidad continua, alta escalabilidad y rendimiento, una gran seguridad y simplicidad operacional.

Tabla 8
Características del motor de base de datos Cassandra

Características	
Año versión inicial	2008
Año última versión	2016
Licencia	Open Source
SO de servidor	BSD Linux OS X Windows
Esquema de datos	Esquema Libre
Índices secundarios	Restringido

CONTINUA →

SQL	No
-----	----

Lenguajes de programación soportados	C#, C++, Clojure, Erlang Go, Haskell, Java, JavaScript Perl, PHP, Python, Ruby, Scala
Procedimientos	No
Triggers	Si
Particionamiento	Sharding
Replicación	Seleccionable
Map Reduce	Si
Consistencia	Eventual - Inmediata
Transaccionalidad	No
Concurrencia	Si
Conceptos de usuarios	Derechos de acceso para los usuarios pueden ser definidos por objeto

Fuente[CITATION Sol16 \l 3082]

HBase

Apache HBase es una base de datos NoSQL que se ejecuta en la parte superior de Hadoop como un gran almacén de datos distribuido y escalable. Esto significa que HBase puede aprovechar el paradigma de procesamiento distribuido y beneficiarse del modelo de programación MapReduce de Hadoop.

Tabla 9
Características del motor de base de datos HBase

Característica	
Año versión inicial	2008
Año última versión	2016
Licencia	Open Source
SO de servidor	Linux Unix Windows
Esquema de datos	Esquema libre
Índices secundarios	No
SQL	No
Lenguajes de programación soportados	C, C# C++, Groovy Java, PHP Python, Scala

CONTINUA →

Procedimientos	Si
Triggers	Si
Particionamiento	Sharding
Replicación	Seleccionable
Map Reduce	Si
Consistencia	Inmediata
Transaccionalidad	No
Concurrencia	Si
Conceptos de usuarios	Las listas de control de acceso (ACL)

Fuente [CITATION Sol16 \l 3082]

Hypertable

Fue diseñado con el propósito expreso de resolver el problema de escalabilidad, un problema que no se maneja bien por un RDBMS tradicional.

Tabla 10
Características del motor de base de datos Hypertable

Características	
Año versión inicial	2009
Año última versión	2013
Licencia	Open Source
SO de servidor	Linux OS X Windows
Esquema de datos	Esquema Libre
Índices secundarios	Restringido
SQL	No
Lenguajes de programación soportados	C++, Java Perl, PHP Python, Ruby
Procedimientos	No
Triggers	No
Particionamiento	Sharding
Replicación	Seleccionable en el nivel de sistemas de Archivos
Map Reduce	Si
Consistencia	Inmediata
Transaccionalidad	No
Concurrencia	Si

Conceptos de usuarios	No
-----------------------	----

Fuente[CITATION Sol16 \l 3082]

Grafos

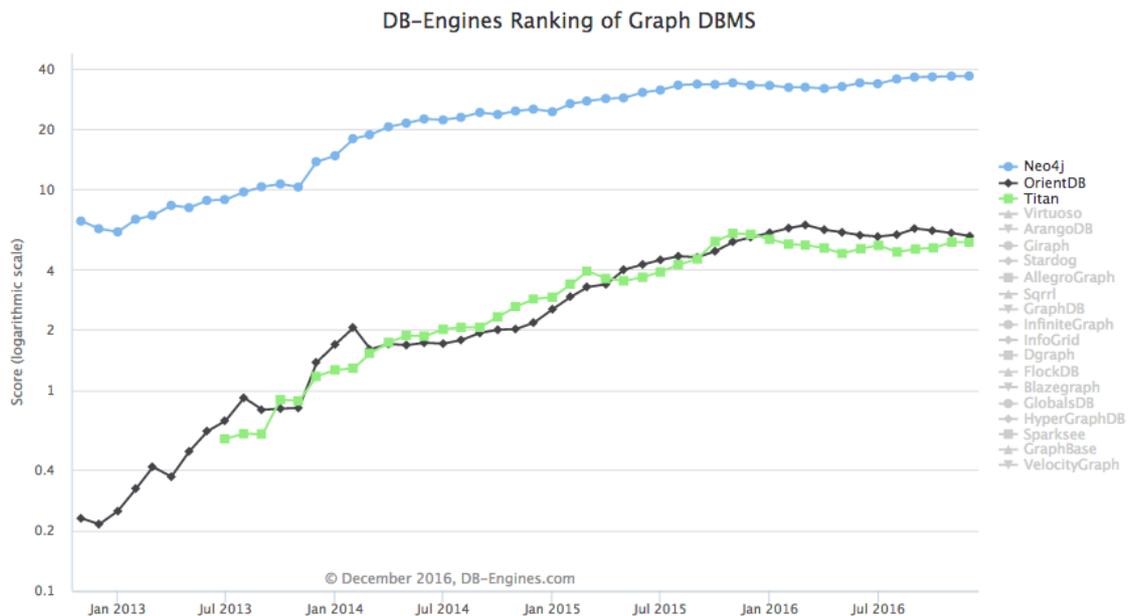


Figura 16DB-Engines Ranking de gestores orientados a grafos

Fuente[CITATION Sol16 \l 3082]

Neo4j

Base de datos transaccional que cuenta con el almacenamiento y procesamiento gráfico nativo para escrituras súper rápidas y lee, además de modelado de datos flexibles y el cumplimiento de ACID.

Tabla11

Características del motor de base de datos Neo4J

Características	
Año versión inicial	2007
Año última versión	2016
Licencia	Open Source
SO de servidor	Linux OS X Windows
Esquema de datos	Esquema Libre
Índices secundarios	Si

CONTINUA →

SQL	No
Lenguajes de programación soportados	.Net, Clojure, Go, Groovy Java, JavaScript, Perl, PHP Python, Ruby, Scala
Procedimientos	Si
Triggers	Si
Particionamiento	Ninguna
Replicación	Maestro Esclavo
Map Reduce	No
Consistencia	Eventual -Inmediata
Transaccionalidad	ACID
Concurrencia	Si
Conceptos de usuarios	No

Fuente[CITATION Sol16 \l 3082]

OrientDB

Con la potencia de un motor de base de datos distribuida Gráfico y la flexibilidad de una base de datos de documentos en un solo producto, es un OrientDB NoSQL, compatible con el ACID, escalable, rápido, seguro, y base de datos operativa flexibles.

Tabla 12
Características del motor de base de datos OrientDB

Características	
Año versión inicial	2010
Año última versión	2016
Licencia	Open Source
SO de servidor	All OS with a Java JDK (>= JDK 6)
Esquema de datos	Esquema Libre
Índices secundarios	Si
SQL	No
Procedimientos	Java, Javascript
Triggers	Hooks
Particionamiento	Sharding
Replicación	Master - Master

CONTINUA →

Map Reduce	No
------------	----

Lenguajes de programación soportados	.Net, C C#, C++ Clojure, Java JavaScript, JavaScript (Node.js) PHP, Python Ruby, Scala
Consistencia	Si
Transaccionalidad	ACID
Concurrencia	Si
Conceptos de usuarios	Derechos de acceso para usuarios y roles; Seguridad a nivel de registro configurable

Fuente[CITATION Sol16 \l 3082]

Titan

Es una base de datos gráfica optimizada para el almacenamiento y consulta de gráficos que contienen cientos de miles de millones de vértices y aristas distribuidos a través de un clúster de varias máquinas. Titán es una base de datos transaccional que puede soportar miles de usuarios simultáneos que ejecutan recorridos de gráficos complejos en tiempo real.

Tabla 13
Características del motor de base de datos Titan

Característica	
Año versión inicial	2012
Año última versión	-
Licencia	Open Source
SO de servidor	Linux OS X Unix Windows
Esquema de datos	Si
Índices secundarios	Si
SQL	No

CONTINUA →

Lenguajes de programación soportados	Clojure, Java Python
Procedimientos	Si
Triggers	Si
Particionamiento	Si
Replicación	Si
Map Reduce	Si
Consistencia	Eventual - Inmediata
Transaccionalidad	ACID
Concurrencia	Si
Conceptos de usuarios	autenticación del usuario y la seguridad a través de Rexster servidor Gráfico

Fuente [CITATION Sol16 \ 3082]

2.3. MÉTODO IQMC

El método IQMC proporciona una serie de directrices y técnicas, con la intención de identificar las características a fin de colaborar en la definición de diversos modelos de calidad, siguiendo la estructura del estándar ISO/IEC 25000 descrito anteriormente.

El método consiste en siete pasos que aunque se presentan como si fueran secuenciales, pueden ser simultaneados o iterados si se considera oportuno. En el primer paso, que puede ser opcional, el ámbito de calidad es explorado en profundidad y seguidamente los seis pasos restantes conducen a la construcción del modelo de calidad partiendo de las características de calidad y su descomposición en subcaracterísticas.

Los pasos a seguir del método son:

Paso 0. Estudio del ámbito del software

Según el método IQMC para desarrollar un modelo que permita evaluar la calidad de un producto de software se debe comenzar por realizar un estudio del ámbito al cual pertenecen los componentes de software para los que se quiere evaluar la calidad. Es un paso opcional que puede evitarse en caso de poseer el conocimiento suficiente.

Paso 1. Determinación de las subcaracterísticas de calidad

Este paso consiste en la selección de subcaracterísticas que se desea evaluar basados en la norma ISO 25000, hay que tomar en cuenta que se podrá agregar nuevas subcaracterísticas, redefinir las existentes y eliminar algunas de acuerdo al interés.

Paso 2. Refinamiento de la jerarquía de subcaracterísticas

Este paso consiste en descomponer las subcaracterísticas del más bajo nivel de abstracción en jerarquías de subcaracterísticas.

Paso 3. Refinamiento de las subcaracterísticas en atributos

El propósito de este paso es tener descompuestas las características en atributos que puedan ser medidos directa e indirectamente.

Paso 4. Refinamiento de atributos derivados en básicos

Este paso consiste en descomponer los atributos en básicos para que puedan ser medidos de forma directa.

Paso 5. Establecimientos de relaciones entre factores de calidad

El propósito de este paso es establecer las relaciones entre los factores de calidad, esto permite determinar las dependencias entre los diferentes factores.

Paso 6. Determinación de las métricas para los atributos.

En este paso se determinan las métricas para medir los atributos.

2.4. TEOREMA DE BREWER O CAP

Surgió en el año 2000 en una charla expuesta por el profesor Eric Brewer, donde exponía que un sistema distribuido no puede cumplir los siguientes 3 requerimientos:

- **Consistencia:** todos los clientes acceden a la misma versión de los datos, incluso durante las actualizaciones de los datos
- **Disponibilidad.** Todos los clientes pueden encontrar siempre al menos una copia de los datos requeridos, incluso cuando alguna de las máquinas del clúster esté caída.
- **Tolerancia a las Particiones.** el sistema mantiene sus características, incluso cuando es desplegado en diferentes servidores, de manera totalmente transparente para el cliente.
[CITATION Equ10 \l 3082]

Este punto es importante, pues permite entender que para ganar velocidad se debe sacrificar por lo menos una de las características mencionadas; pero el enfoque es que el sistema puede estar haciendo cambios entre las combinaciones de estos elementos, teniendo en cuenta las siguientes posibilidades:

CP: el sistema ejecutará las operaciones de forma consistente, aunque se pierda la comunicación entre nodos (partición del sistema), pero no se asegura que el sistema responda (disponibilidad).

AP: el sistema siempre responderá a las peticiones, aunque se pierda la comunicación entre nodos (partición del sistema). Los datos procesados pueden no ser consistentes.

CA: el sistema siempre responderá a las peticiones y los datos procesados serán consistentes. En este caso no se permite una pérdida de comunicación entre nodos (partición del sistema).

2.5. CALIDAD DE SOFTWARE Y ANÁLISIS DE LOS MODELOS DE CALIDAD

2.5.1. Calidad de software

En las últimas décadas el interés por la calidad del software ha crecido de forma constante (incluyendo el interés por evaluar la calidad de las bases de datos como se expone en 1.1), por esta razón los clientes se tornan más selectivos y empiezan a descartar los productos poco fiables o que realmente no se ajustan a sus necesidades.[CITATION EST \l 12298] Si se desea mejorar en cuanto a la calidad del software, entonces es preciso definir la calidad y sobre todo medirla. Todavía el mayor problema en la gestión de la calidad es que el propio término calidad es ambiguo, tanto que comúnmente es mal entendido.

Si se quiere definir calidad del software se pueden adoptar diferentes conceptos. En primer lugar, es importante establecer diferencias entre la calidad del producto software y la calidad del proceso de desarrollo de este. Sin embargo, hay que dejar claro que las metas que se establezcan para la calidad del producto van a determinar los objetivos a establecer para lograr la calidad del proceso de desarrollo, ya que la calidad del primero va a depender, entre otros aspectos, de la calidad del segundo.[CITATION EST \l 12298] Si no se cuenta con un buen proceso de desarrollo es casi imposible obtener un buen producto. Es primordial destacar que la calidad de un producto software se debe tener en cuenta durante todos sus estados de evolución (especificaciones, diseño, implementación y otros). No es suficiente verificar la calidad del producto una vez finalizado su desarrollo, en este momento los problemas de baja calidad ya no tienen solución o su reparación es muy costosa.

A continuación, se exponen diferentes definiciones de calidad del software:

Calidad del software se define como “la concordancia con los requisitos funcionales y de rendimiento explícitamente establecidos, con los estándares de desarrollo explícitamente documentados, y con las características implícitas que se espera de todo software desarrollado profesionalmente”[CITATION Rog02 \l 3082].

Por su parte, la norma ISO/IEC 25010 contenida dentro de la familia de estándares ISO 25000 expresa que “la calidad del producto software se puede interpretar como el grado en que dicho producto satisface los requisitos de sus usuarios aportando de esta manera un valor. Son precisamente estos requisitos (funcionalidad, rendimiento, seguridad, mantenibilidad, etc.) los que se encuentran representados en el modelo de calidad, el cual categoriza la calidad del producto en características y subcaracterísticas” [CITATION Int15 \l 3082].

Una conclusión evidente a la que se puede llegar a partir de estas definiciones es que la calidad es algo relativo. Su relatividad dependerá de los requisitos o necesidades que se desee satisfacer. Por eso, la evaluación de la calidad de un producto siempre va a implicar que se comparen unos requisitos preestablecidos con el producto realmente desarrollado.

Hablar de calidad de un producto de software implica lidiar con diferentes problemas, entre estos se encuentran:

- La propia definición de la calidad del software; se hace difícil encontrar un conjunto de propiedades que den una indicación de su calidad, precisamente para dar solución a este problema surgieron los modelos de calidad.[CITATION EST \l 12298]
- La comprobación de la calidad del software; muchas veces no se sabe medir el grado de calidad de un producto software y para ello juega un papel fundamental el control de calidad.

La mejora de la calidad del software; se torna dificultoso saber utilizar la información disponible acerca de la calidad del producto software para mejorar su calidad a lo largo del ciclo de vida, en este eje aparecen dos conceptos importantes, ellos son la gestión de calidad y el aseguramiento de la calidad.

2.5.2. Estándar o norma de calidad

A la hora de hacer referencia a la calidad del software y a los modelos que permiten evaluarla, es preciso tener en cuenta que en su definición y evaluación desempeñan un papel muy importante el uso de estándares o normas de calidad.

Según la International Organization for Standardization (ISO) “Un estándar es un documento que proporciona requisitos, especificaciones, directrices o características que pueden usarse consistentemente para asegurar que materiales, productos, procesos y servicios sean adecuados para su propósito”[CITATION Int12 \l 3082].

Las normas son un modelo o patrón a seguir. Tienen como finalidad establecer cuales características debe tener un objeto o producto. Por tanto, analizando los conceptos expuestos una norma consta de un conjunto de características y especificaciones que un producto debe cumplir para que sea considerado de calidad. La ISO a lo largo de los años ha desarrollado una infinidad de estándares para poder medir, evaluar y sustentar que un determinado producto tiene calidad. Dentro de la infinidad de estándares desarrollados está la ISO/IEC 25000, la cual proporciona una guía para el uso de la nueva serie de estándares internacionales llamada Requisitos y Evaluación de Calidad de Productos de Software (SQuaRE - System and Software Quality Requirements and Evaluation).

En el desarrollo de este trabajo se tendrá en cuenta el uso de la familia de estándares ISO/IEC 25000, concretamente través de las normas ISO/ IEC 25010 y la división ISO/IEC 2502npor las ventajas que

ofrecen y la flexibilidad que aportan en la construcción de modelos de calidad para la evaluación de productos de software.

2.5.3. Modelo de calidad

Para ayudar a definir la calidad del software de una forma más precisa y provechosa aparecen los modelos de calidad. Diferentes literaturas coinciden en que un modelo de calidad es el conjunto de características y las relaciones entre las mismas, que proporcionan el basamento para especificar requisitos de calidad y evaluar la calidad.

El estándar ISO 8402 define modelo de calidad como el conjunto de factores de calidad, y de relaciones entre ellos, que proporciona una base para la especificación de requisitos de calidad y para la evaluación de la calidad de los componentes software. Los modelos de calidad se estructuran generalmente como una jerarquía (ya sea un árbol, ya sea un grafo dirigido), donde factores de calidad más genéricos, como eficiencia o usabilidad, se descomponen en otros más particulares, como tiempo de respuesta o facilidad de aprendizaje, probablemente en diversos niveles de descomposición [CITATION Int86 \l 3082].

Que las organizaciones sean capaces de desarrollar e implantar modelos de calidad repercute en que se logren desarrollar sistemáticamente tanto productos como bienes y servicios que alcancen mejor calidad, y por consiguiente puedan cumplir con las necesidades y expectativas de los clientes. De ahí que la norma ISO/IEC 25010 señale que “el modelo de calidad representa la piedra angular en torno a la cual se establece el sistema para la evaluación de la calidad del producto. En este modelo se determinan las características de calidad que se van a tener en cuenta a la hora de evaluar las propiedades de un producto software determinado”[CITATION Int15 \l 3082].

Una característica de calidad de un producto software es un conjunto de propiedades mediante las cuales se evalúa y describe su calidad. Una característica de calidad puede ser refinada en múltiples

niveles de subcaracterísticas de calidad. Según [CITATION Car05 \l 12298] las características en un modelo de calidad son factores de calidad no medibles usados para clasificar el nivel superior de subcaracterísticas del modelo. Mientras que las subcaracterísticas en un modelo de calidad son los factores del modelo que pueden ser medidos subjetivamente cuando sea necesario, y que pueden descomponerse en otras subcaracterísticas o alternativamente en atributos que ayudan en su medición.

Por su parte un atributo es una propiedad de calidad a la que puede asignársele una métrica, donde una métrica es un procedimiento que chequea un componente produciendo un dato simple, un número o un símbolo a partir del examen realizado.

En su trabajo Carvallo[CITATION Car05 \t \l 12298] expresa que los atributos pueden ser derivados o básicos y que pueden estar relacionados con más de un factor de calidad en su nivel superior en la jerarquía. Los atributos básicos son factores de calidad objetivamente medibles que no pueden descomponerse en otros atributos.

La calidad del producto de software debe ser evaluada utilizando un modelo de calidad definido. El modelo de calidad se usará al fijar los objetivos de calidad que los productos de software deben cumplir. La especificación y la evaluación de la calidad de un producto de software pueden conseguirse definiendo características de calidad apropiadas, tomando en cuenta el objetivo de uso del producto de software que se evalúa. Es importante tener en cuenta que el modelo de calidad a utilizar depende de los componentes de software a evaluar.

Entre los modelos de calidad uno de los más antiguos y extendidos es el Modelo de McCall, a partir del cual se han derivado modelos como el de Boehm, FURPS y otros, hasta llegar al propuesto por el estándar ISO/IEC 9126, sobre el cual más tarde se basó la familia ISO/IEC 25000. En general, no existe un consenso a la hora de definir y clasificar las características de calidad que un producto de software debe tener, por lo

cual es necesario estudiar diferentes modelos de calidad, sobre los cuales la propuesta estará fundamentada.

2.5.4. Tipos de modelos de calidad

Existen tres tipos de modelos de calidad.

- **Modelos Fijos:** Existe un catálogo de factores de calidad de partida que se usa como base para la evaluación de la calidad. En general, la propuesta típica de un modelo de calidad fijo consiste en una estructuración de los factores en una jerarquía multinivel, con un conjunto de factores de más alto nivel, unos criterios que descomponen dichos factores, y eventualmente métricas para la medida de cada criterio. La ventaja de estos modelos fijos es que proporcionan una vista común y comparable que se reutiliza en cada proyecto, ya que el conjunto de factores de calidad siempre es el mismo. Ahora bien, tiene como inconveniente su poca flexibilidad debido a que asumen que siempre bastará con un subconjunto de sus factores para evaluar la calidad en cualquier proyecto.

Ejemplos de modelos que siguen este enfoque son los modelos de McCall et al. (1997), Boehm et al. (1978), Keller et al. (1990) y el modelo con un enfoque más industrial FURPS (Grady y Caswell, 1987).
[CITATION Jua \t \l 3082]

- **Modelos a la medida:** No existe ningún catálogo de factores de partida, y dichos factores deben ser identificados para cada proyecto. La idea que guía la construcción de estos modelos es que se debe partir de la identificación de los objetivos a alcanzar. Dichos objetivos serían los factores más abstractos que deben descomponerse en factores más concretos hasta llegar a hacer

operativos los objetivos, de forma que pueda ser medida su consecución.

Así, los modelos son creados desde cero para todo nuevo proyecto. Existen diversas propuestas de métodos para crear los modelos de calidad a medida. La ventaja de estos modelos es su total adaptabilidad. Tienen como inconveniente que el coste de su construcción es muy alto comparado con el de los modelos fijos, y la reutilización de modelos de un proyecto a otro es difícil, dado que los factores identificados para un proyecto no tienen por qué ser adecuados para otro.

Ejemplos: GQM (Goal-Question-Metric)[CITATION Jua \t \l 3082]

- **Modelos Mixtos:** Se intenta combinar las ventajas de los dos tipos anteriores de modelos. La idea es que exista un conjunto de factores de calidad más abstractos que sean reutilizados en virtualmente todos los proyectos posibles, y que puedan ser refinados y operacionalizados para un proyecto particular.

Ejemplos:

El modelo propuesto en el estándar ISO/IEC 9126-1(2001)[CITATION Jua \t \l 3082]

2.5.5. Modelos de calidad de McCall y Bohem

McCall y Cavano definieron un juego de once factores de calidad como los primeros pasos hacia el desarrollo de métricas de la calidad del software.[CITATION EST \l 12298] El modelo de McCall organiza los factores en tres ejes o puntos de vista, desde los cuales el usuario puede contemplar la calidad de un producto, y cada uno de estos factores se descompone a su vez en criterios.[CITATION EST \l 12298]

Otro de los modelos de calidad es el presentado por Barry Boehm en 1978. Este modelo introduce características de alto nivel,

características de nivel intermedio y características primitivas, cada una de las cuales contribuye al nivel general de calidad [CITATION Fre10 \l 3082].

Las características de alto nivel representan requerimientos generales de uso, las características de nivel intermedio representan los factores de calidad de Boehm y el nivel más bajo corresponde a características directamente asociadas a las métricas de calidad [CITATION Fil07 \l 3082].

Tanto el modelo de McCall como el de Boehm son modelos fijos, esto significa que no hay posibilidades de que sean modificados o adaptados. Por tanto, los criterios y factores de calidad son determinados y fijados de forma tal que la medida de calidad debe ajustarse estrictamente a lo definido y a las relaciones entre criterios y factores de calidad que cada modelo propone.

El modelo de McCall se centra en tres aspectos importantes de un producto de software:

- Características operativas/Operación del Producto
- Capacidad para soportar los cambios/Revisión del Producto
- Adaptabilidad a nuevos entornos/Transición del producto

El modelo de McCall organiza los factores en tres ejes o puntos de vista desde los cuales el usuario puede contemplar la calidad de un producto,[CITATION EST \l 12298] basándose en once factores de calidad organizados en torno a los tres ejes y a su vez cada factor se desglosa en otros criterios:[CITATION EST \l 12298]

Tabla 14
Factores y criterios de McCall

Ejes	Factor	Criterios
Operación del producto	Facilidad de uso	- Facilidad de operación: Atributos del software que determinan la facilidad de operación del software. - Facilidad de comunicación: Atributos del software que proporcionan entradas y salidas fácilmente asimilables.

		<ul style="list-style-type: none"> - Facilidad de aprendizaje: Atributos del software que facilitan la familiarización inicial del usuario con el software y la transición del modo actual de operación. - Formación: El grado en que el software ayuda para permitir que nuevos usuarios apliquen el sistema.
	Integridad	<ul style="list-style-type: none"> - Control de accesos. Atributos del software que proporcionan control de acceso al software y los datos que maneja. - Facilidad de auditoría: Atributos del software que facilitan la auditoría de los accesos al software. - Seguridad: La disponibilidad de mecanismos que controlen o protejan los programas o los datos.[CITATION EST \l 12298]
	Corrección	<ul style="list-style-type: none"> - Completitud: Atributos del software que proporcionan la implementación completa de todas las funciones requeridas.[CITATION EST \l 12298] - Consistencia: Atributos del software que proporcionan uniformidad en las técnicas y notaciones de diseño e implementación. - Trazabilidad o rastreabilidad: Atributos del software que proporcionan una traza desde los requisitos a la implementación con respecto a un entorno operativo concreto.[CITATION EST \l 12298]
Operación del producto	Fiabilidad	<ul style="list-style-type: none"> - Precisión: Atributos del software que proporcionan el grado de precisión requerido en los cálculos y los resultados.[CITATION EST \l 12298] - Consistencia. - Tolerancia a fallos: Atributos del software que posibilitan la continuidad del funcionamiento bajo condiciones no usuales.[CITATION EST \l 12298] - Modularidad: Atributos del software que proporcionan una estructura de módulos altamente independientes. - Simplicidad: Atributos del software que posibilitan la implementación de funciones de la forma más comprensible posible. - Exactitud: La precisión de los cálculos y del control.[CITATION EST \l 12298]
	Eficiencia	<ul style="list-style-type: none"> - Eficiencia en ejecución: Atributos del software que minimizan el tiempo de procesamiento. - Eficiencia en almacenamiento: Atributos del software que minimizan el espacio de almacenamiento necesario.[CITATION EST \l 12298]

CONTINÚA →

Revisión del producto	Facilidad de mantenimiento	<ul style="list-style-type: none"> - Modularidad. - Simplicidad. - Consistencia. - Concisión: Atributos del software que posibilitan la implementación de una función con la menor cantidad de códigos posible. [CITATION EST \1 12298] - Auto descripción: Atributos del software que proporcionan explicaciones sobre la implementación de las funciones. [CITATION EST \1 12298]
	Facilidad de prueba	<ul style="list-style-type: none"> - Modularidad. - Simplicidad. - Auto descripción. - Instrumentación: Atributos del software que posibilitan la observación del comportamiento del software durante su ejecución para facilitar las mediciones del uso o la identificación de errores.
	Flexibilidad	<ul style="list-style-type: none"> - Auto descripción. - Capacidad de expansión: Atributos del software que posibilitan la expansión del software en cuanto a capacidades funcionales y datos. - Generalidad: Atributos del software que proporcionan amplitud a las funciones implementadas. [CITATION EST \1 12298] - Modularidad.
	Reusabilidad	<ul style="list-style-type: none"> - Auto descripción. - Generalidad. - Modularidad. - Independencia entre sistema y software: Atributos del software que determinan su dependencia del entorno operativo. [CITATION EST \1 12298] - Independencia del hardware: Atributos del software que determinan su dependencia del hardware. [CITATION EST \1 12298]
	Interoperabilidad	<ul style="list-style-type: none"> - Modularidad. - Compatibilidad de comunicaciones: Atributos del software que posibilitan el uso de protocolos de comunicación e interfaces estándar. [CITATION EST \1 12298] - Compatibilidad de datos: Atributos del software que posibilitan el uso representaciones de datos estándar. [CITATION EST \1 12298] - Estandarización en los datos: El uso de estructuras de datos y de tipos estándar a lo largo de todo el programa. [CITATION EST \1 12298]
	Portabilidad	<ul style="list-style-type: none"> - Auto descripción. - Modularidad. - Independencia entre sistema y software. - Independencia del hardware.

Fuente[CITATION Cer16 \1 3082]

2.5.6. Modelo de Calidad FURPS

Los factores de calidad descritos por McCall y sus colegas representan solo una de las muchas listas de comprobación sugeridas para la calidad del software. Hewlett – Packard (HP) ha desarrollado un conjunto de factores de calidad del software al que se le ha dado el acrónimo de FURPS: funcionalidad, facilidad de uso, fiabilidad, rendimiento y capacidad de soporte.[CITATION MAR03 \l 3082].

Basado en el acrónimo de FURPS+: Funcionalidad (Functionality), Usabilidad (Usability), Confiabilidad (Reliability), Desempeño (Performance) y Soportabilidad (Supportability). La siguiente tabla presenta la clasificación de los atributos de calidad que se incluyen en el modelo, junto con sus características (Pressman, 2002).

Tabla 15
Características de FURPS

Tipo de Requerimiento		Descripción
F	Funcionalidad	Características, capacidades y algunos aspectos de seguridad
U	Usabilidad	Factores Humanos (Interacción), ayuda, documentación
R	Confiabilidad	Frecuencia de fallos, capacidad de recuperación de un fallo y grado de previsión
P	Desempeño	Tiempos de respuesta, productividad, precisión, uso de recursos.
S	Soporte	Adaptabilidad, facilidad de mantenimiento, internacionalización, facilidad de configuración.

Fuente[CITATION 16En1 \l 3082]

Este modelo incluye otros elementos además de los factores de calidad y los atributos, como son restricciones de diseño y requerimientos

de implementación tanto físicos como de interfaz. Sin embargo, entre sus limitaciones se destaca que no considera la portabilidad de los productos de software que se están evaluando.

2.5.7. Estándar ISO/IEC 9126

El estándar de calidad ISO/IEC 9126 divide el modelo de calidad para los productos de software en dos partes:

- a) Calidad interna y externa
- b) calidad durante el uso.

En la primera parte se definen seis características para la calidad interna y externa, las cuales a su vez son divididas en subcaracterísticas que se manifiestan de manera externa cuando se usa el software como parte del sistema informatizado, y que además son el resultado de los atributos internos del software. La segunda parte del modelo de calidad define cuatro características para la calidad durante el uso que representan para el usuario, la combinación de las seis características de calidad del producto de software.

Este estándar es el punto de partida para la posterior aparición de la familia de normas ISO/IEC 25000, expresadas como la concatenación, unificación y revisión de las cuestiones abordadas por los estándares ISO / IEC 9126 e ISO/IEC 14598.

Familia de estándares ISO/IEC 25000

Teniendo en cuenta la importancia creciente que se le ha otorgado tanto a la calidad del proceso como del producto recientemente ha aparecido la familia de normas ISO/IEC 25000, que proporciona una guía para el uso de la nueva serie de estándares internacionales llamada Requisitos y evaluación de calidad de productos de software (SQuaRE - System and Software Quality Requirements and Evaluation).

ISO/IEC 25000 constituye una serie de normas basadas en ISO/IEC 9126 y en ISO/IEC 14598 cuyos objetivos principales son:

- Guiar el desarrollo de los productos de software mediante la especificación de requisitos y evaluación de características de calidad[CITATION Int15 \l 3082].
- Crear un cerco de trabajo común para evaluar la calidad del producto software, sustituyendo a las anteriores ISO/IEC 9126 e ISO/IEC 14598 y convirtiéndose así en la piedra angular de esta área de la Ingeniería del Software.[CITATION GIS15 \l 3082]

Las normas de la familia ISO/IEC 25000 proponen nuevas normas partiendo de partes relevantes de los estándares precedentes que amplían el espectro de características de calidad a tener en cuenta para una evaluación.

ISO/IEC 25000 está formada por las divisiones siguientes:

- **ISO/IEC 2500n:** División de gestión de calidad. Los estándares que forman esta división definen todos los modelos comunes, términos y referencias a los que se alude en las demás divisiones de SQuaRE.[CITATION Ecu15 \l 12298]
- **ISO/IEC 2501n:** División del modelo de calidad. El estándar que conforma esta división presenta un modelo de calidad detallado, incluyendo características para la calidad interna, externa y en uso. [CITATION Ecu15 \l 12298]
- **ISO/IEC 2502n:** División de mediciones de calidad. Los estándares pertenecientes a esta división incluyen un modelo de referencia de calidad del producto software, definiciones matemáticas de las métricas de calidad y una guía práctica para su aplicación.

Presenta aplicaciones de métricas para la calidad de software interna, externa y en uso.[CITATION Ecu15 \l 12298]

- **ISO/IEC 2503n:** División de requisitos de calidad. Los estándares que forman parte de esta división ayudan a especificar los requisitos de calidad. Estos requisitos pueden ser usados en el proceso de especificación de requisitos de calidad para un producto software que va a ser desarrollado o como entrada para un proceso de evaluación. El proceso de definición de requisitos se guía por el establecido en la norma (ISO, 2003).[CITATION Ecu15 \l 12298]
- **ISO/IEC 2504n:** División de evaluación de la calidad. Estos estándares proporcionan requisitos, recomendaciones y guías para la evaluación de un producto software, tanto si la llevan a cabo evaluadores, como clientes o desarrolladores.[CITATION Ecu15 \l 12298]
- **ISO/IEC 25050–25099:** Estándares de extensión SQuaRE. Incluyen requisitos para la calidad de productos de software “Off-The-Self” y para el formato común de la industria (CIF) para informes de usabilidad. [CITATION Ecu15 \l 3082]

Ventajas que ofrece la norma ISO/IEC 25000[CITATION Pie14 \l 3082]

- El modelo representa la calidad esperada del producto de software.
- Permite mayor eficacia.
- Plantea la evaluación de productos intermedios.
- Propone una calidad final a través de las evaluaciones intermedias.
- Permite efectuar un rastreo entre las expectativas, requerimientos y medidas de evaluación.
- Mejora la calidad del producto.[CITATION Ecu15 \l 12298]

Dentro de la familia de estándares ISO/IEC 25000 y para los propósitos de esta investigación destacan las normas ISO/IEC 25010 y la ISO/IEC 2502n.

El modelo de calidad que se plantea en la norma ISO/IEC 25010 clasifica la calidad del software en características, las cuales a su vez se dividen en subcaracterísticas y estas posteriormente en atributos de calidad.[CITATION Ecu15 \l 12298] El modelo de calidad consiste en dos partes: el modelo para la calidad interna y externa y el modelo para la calidad en uso.

La calidad en uso es una medida de la calidad del sistema en su ambiente de operación para un grupo de usuarios específicos, llevando a cabo tareas específicas. La calidad externa proporciona una vista de caja negra del software, se orienta a las propiedades relacionadas con la ejecución del software sobre un tipo de hardware y sistema operativo determinados. Por su parte la calidad interna proporciona una vista de caja blanca y se orienta a las propiedades del producto software que típicamente están disponibles durante el desarrollo. La calidad interna está principalmente relacionada con propiedades estáticas del software y tiene un impacto sobre la calidad externa, la cual a su vez tiene un impacto sobre la calidad en uso.

La norma ISO/IEC 25010 define un modelo de calidad para la calidad interna y externa que se encuentra compuesto por las ocho características de calidad que se muestran en la siguiente figura.



Figura 17 Calidad del producto software según la ISO/IEC 25010

Fuente [CITATION ISO15 \l 3082]

Mientras que el modelo de calidad en uso define 5 características a nivel de sistema. Las características de calidad se descomponen en subcaracterísticas y a su vez el modelo permite a que estas subcaracterísticas sean descompuestas en una estructura jerárquica. (Ver figura 17)



Figura 18 Modelo de Calidad en uso

Fuente [CITATION ISO15 \l 3082]

Las características de calidad cubren todos los aspectos de interés con respecto a la calidad para la mayoría de los productos de software y como tal pueden usarse como una lista de chequeo para asegurar una cobertura de calidad completa. La norma ISO/IEC 25010 propone un esquema flexible que en su utilización dependerá de las características del producto de software que se desee evaluar.

Por tanto, el modelo de calidad sirve como un marco de trabajo para asegurar que todos los aspectos de calidad sean tenidos en cuenta desde los puntos de vista de calidad interna, externa y en uso. En este sentido la calidad del producto software puede ser evaluada mediante la medición de los atributos internos (típicamente medidas estáticas de productos intermedios), mediante la medición de atributos externos (típicamente midiendo el comportamiento del código cuando es ejecutado) o mediante los atributos de calidad en uso (cuando el producto está bajo un uso real o simulado).

Para el análisis de las métricas de calidad en el modelo de evaluación se aplicará la ISO/IEC 2502n la cual se descompone en las siguientes normas:

- ISO/IEC 25020 - Measurement reference model and guide: presenta una explicación introductoria y un modelo de referencia común a los elementos de medición de la calidad. También proporciona una guía para que los usuarios seleccionen o desarrollen y apliquen medidas propuestas por normas ISO.[CITATION Pie14 \l 12298]
- ISO/IEC 25021 - Quality measure elements: define y especifica un conjunto recomendado de métricas base y derivadas que puedan ser usadas a lo largo de todo el ciclo de vida del desarrollo software. [CITATION Pie14 \l 12298]
- ISO/IEC 25022 - Measurement of quality in use: define específicamente las métricas para realizar la medición de la calidad en uso del producto.[CITATION Pie14 \l 12298]
- ISO/IEC 25023 - Measurement of system and software product quality: define específicamente las métricas para realizar la medición de la calidad de productos y sistemas software.[CITATION Pie14 \l 12298]
- ISO/IEC 25024 - Measurement of data quality: define específicamente las métricas para realizar la medición de la calidad de datos. [CITATION ISO15 \l 3082]

De forma general los modelos de calidad proporcionan a los evaluadores un marco de trabajo medible, el cual precisamente define y consolida las diferentes vistas que se tienen sobre calidad, lo cual es requerido para poder evaluar el producto software. Sin embargo, la selección de las características de calidad que se deben incluir en el modelo no es una tarea trivial. Según Kontio [CITATION Kon96 \l 3082] cuando se definen los criterios de evaluación raras veces se definen bien.

Los criterios se pueden expresar con una o dos palabras, por ejemplo: facilidad de uso, uso de memoria o fiabilidad. Esto deja el significado exacto de cada criterio muy abierto a la interpretación del evaluador. Este problema se vuelve mucho más relevante cuando hay más de un evaluador involucrado. Como consecuencia la evaluación puede ser engañosa y a menudo poco fiable.

Un hecho real es que la construcción de modelos de calidad se ve amenazada por diferentes factores, entre los que sobresalen factores relacionados con el equipo, relacionados con el alcance de la calidad y factores metodológicos. La falta de experiencia en la construcción de modelos de calidad y el desconocimiento total o parcial sobre el alcance de la calidad se encuentran entre los factores relacionados con el equipo que son determinantes.

Entre los factores relacionados con el alcance de la calidad se encuentran la falta de una terminología común y el gran número de atributos con fuertes relaciones entre ellos que pueden aparecer en la evaluación de un producto software. Las dificultades en la identificación del nivel de detalle sobre el cual el modelo de calidad debe ser construido o cuándo el modelo de calidad ya ha sido finalizado emergen como algunos factores metodológicos. Por tanto, se necesita de la adopción de un método bien estructurado que sirva de apoyo en la identificación y categorización de las características de calidad que se requieren: el método IQMC.

En su implementación el método IQMC adopta un enfoque mixto, por lo tanto, se debe seleccionar un marco de trabajo de calidad (incluyendo un catálogo de partida). Según [CITATION Car05 \t \l 12298] la selección del catálogo de partida no puede ser arbitraria, esta debe cumplir al menos los siguientes principios básicos:

- Se deben ajustar algunas características de calidad de alto nivel. Este es un punto crucial porque los modelos de calidad pueden diferir dramáticamente de un dominio de aplicación a otro, así su

refinamiento en subcaracterísticas de calidad de bajo nivel debe ser también notablemente diferente.

- Se debe permitir la creación de jerarquías de características de calidad, lo cual es esencial en la construcción de modelos de calidad.
- Esta jerarquía de características de calidad debe permitir el solapamiento, puesto que algunas características de calidad pueden contribuir con otras de diferentes maneras.
- Debe ser lo más extendida posible.

2.6. SELECCIÓN DE UN MODELO DE CALIDAD

Se han descrito algunos modelos de calidad más conocidos, ya sean por ser históricos como el de McCall por ser el primer modelo de calidad conocido o por ser un estándar internacional como la ISO 25000. Tomando en cuenta que existen una gran cantidad de modelos de calidad y que en esta investigación solo se analizaron algunos de ellos, se presenta la siguiente problemática: ¿Cuál modelo de calidad sería el adecuado para las necesidades de la presente investigación?

Para la selección del modelo de calidad se considera oportuno definir varios criterios para comparar los distintos modelos, aspectos como estructura del modelo, organización del modelo, sus características son aspectos importantes para la selección. A continuación, se detallan los elementos más importantes tomados en cuenta para seleccionar el modelo:

- Tipos de modelos de calidad: Como se definió en el apéndice 2.5.4 existen tres tipos de modelos de calidad. Se tomará como más apropiado para la construcción del modelo de evaluación los modelos mixtos por razón de que brindan características pre definidas y permiten extenderse en lo referente al tema tratado.

- Organización del modelo: Las relaciones entre elementos y la estructura es un aspecto importante a tener en cuenta, ya que un modelo bien estructurado y con relaciones coherentes entre sus elementos, ayudará a una mejor aplicación del mismo.
- Características y subcaracterísticas: Es un elemento esencial en la selección del modelo, ya que estas serán la guía a seguir para determinar cuál base de datos NoSQL de las existentes será la más adecuada para determinada persona o entidad.

Además de emplear estos criterios; para la selección de un modelo de calidad fue imprescindible realizar una comparación entre los explicados en los apéndices 2.5.5, 2.5.6 y 2.5.7, analizando algunas particularidades de cada uno de ellos.

Tabla 16
Características para la selección de un modelo de calidad

Modelo	Año de publicación	Tipo de modelo	Elementos estructurales	Descripción del modelo
McCall	1977	Fijo	3 ejes de calidad, 11 factores, 23 criterios.	Elementos definidos, métricas a definir por el evaluador.
Bohem	1978	Fijo	3 características de alto nivel, 7 características intermedias y 15 características primitivas.	Elementos definidos, métricas a definir por el evaluador.
FURPS	1987	Fijo	5 Tipo de requerimientos, un funcional y 4 no funcionales, 16 características.	Requerimientos definidos, características definidas, métricas a definir por el evaluador.
ISO/IEC 9126	2001	Mixto	6 características, 27 subcaracterísticas definidas.	Características definidas, subcaracterísticas definidas, guía para la elaboración de las métricas.
ISO/IEC 25000	2005	Mixto	13 características de alto nivel y 41 subcaracterísticas definidas.	Características definidas, subcaracterísticas definidas, guía para

				la elaboración de las métricas.
--	--	--	--	---------------------------------

Fuente[CITATION Ore13 \t \l 3082]

Se puede observar que el modelo más conveniente para el tema en cuestión es el estándar ISO/IEC 25000, ya que es el más actual que ha sido publicado, es un tipo de modelo mixto, ofrece una guía para la elaboración de las métricas de calidad, fue desarrollado por la Organización Internacional de Normalización después de un extenso estudio de cuales características debía cumplir un software y es, como se explicó en el apéndice 2.5.7 una serie de normas basadas en ISO/IEC 9126 y en ISO/IEC 14598, es decir es la sucesora de la norma 9126.

Teniendo en cuenta todas estas consideraciones, se selecciona como catálogo el estándar ISO/IEC 25000. No solo es el estándar que representa los esfuerzos colectivos más avanzados en términos de modelos de calidad de software, sino que para la evaluación de la calidad del producto define 8 características de calidad de alto nivel y su descomposición en subcaracterísticas, mientras que para la evaluación de la calidad en uso propone cinco características que también se descomponen en subcaracterísticas. Por tanto, proporciona un modelo de referencia muy flexible que puede ser adaptado a cualquier dominio específico, en este caso solo es necesario proporcionar un mapeo entre el modelo adaptado y el modelo que propone el estándar. Esto se logra agregando, adaptando, descomponiendo o desechando algunas de las subcaracterísticas, y completando la jerarquía con las características de calidad y las métricas requeridas para la evaluación de un cierto componente.

2.7. ANTECEDENTES DE LOS MODELOS DE EVALUACIÓN

Para analizar si existe algún indicio de lo que se está realizando y fundamentar la presente propuesta se hace un estudio del ámbito de los modelos de evaluación de calidad y de las bases de datos no relacionales. Para ello se realizó una búsqueda exhaustiva de autores o empresas que hayan aplicado y/o desarrollado algo semejante a lo que se

pretende efectuar en la investigación. A continuación, se vislumbra una descripción de lo encontrado:

- a) Evaluación técnica informática del sistema integrado de operaciones y negocios-SION de la empresa pública Correos del Ecuador mediante la aplicación de la norma ISO/IEC 25000 [CITATION Rep15 \l 3082]

La empresa Correos del Ecuador CDE-EP es una institución pública dedicada al servicio postal de calidad a nivel local, nacional e internacional. En esta institución se elaboró un modelo de evaluación para aplicarlo a un sistema realizado por el grupo de desarrollo de dicha entidad, nombrado SION. Este modelo se elaboró con el objetivo de determinar la calidad del producto y de establecer conclusiones y recomendaciones de mejora para la aplicación. Para ello se plantea su evaluación técnica mediante la Norma ISO/IEC 25000 y con la aplicación del método IQMC.

Una vez desarrollado el modelo y aplicado, las métricas arrojaron los siguientes resultados: El sistema SION cumple con el 69,62% de calidad, por lo que se considera un producto SATISFACTORIO técnicamente. [CITATION Rep15 \l 3082]

Como se puede constatar la elaboración de un modelo de evaluación empleando la ISO 25000 y el método IQMC resulto de modo satisfactorio para evaluar la calidad de un producto de software.

- b) Evaluación de frameworks realizados en java para aplicaciones on-line [CITATION Ore131 \l 3082]

En la Universidad del Azuay en la Facultad de Ciencias de la Administración se desarrolló un trabajo diploma para evaluar todos los frameworks realizados en java para aplicaciones on-Line. Para ello el autor se basa en analizar todos los modelos de calidad existente y selecciona entre ellos al estándar ISO/IEC 9126 para desarrollar un

modelo de evaluación. Para su aplicación se apoya además en el método IQMC.[CITATION Ore131 \l 12298]

La investigación concluyo una vez analizados varios frameworks aplicándole el modelo de calidad, arrojando como resultado que Seam es el mejor frameworks para desarrollar aplicaciones web empresariales. Una vez más la aplicación de modelos de calidad para analizar algún tipo de software arroja resultados satisfactorios.[CITATION Ore131 \l 12298]

- c) Estudio del uso de MongoDB como alternativa a las bases de datos relacionales tradicionales en aplicaciones web que requieren rapidez de lectura/escritura de los datos almacenados. [CITATION Bri11 \l 3082]

En diciembre del 2011 en Cuenca, Universidad Tecnológica Israel, facultad de Sistemas Informáticos desarrollo una investigación donde la autora Diana Marisela Brito Zhunio establece un conjunto de parámetros para evaluar cual base de datos relacional o no relacional se ajusta mejor a sus necesidades. Para ello primeramente establece características generales para sistemas relacionales. Luego establece parámetros a considerar para las bases de datos NoSQL finalizando este acápite con un compendio entre ambas consideraciones, visualiza cuales parámetros se podrían cumplir a la hora de aplicar una base de datos no relacional, o si se deseara aplicar un sistema mixto.[CITATION Bri11 \l 3082]

- d) Análisis comparativo de dos bases de datos SQL y dos bases de datos NoSQL.[CITATION SAL14 \l 3082]

En la Universidad Tecnológica de Pereira en el año 2014, Facultad de ingenierías se desarrolló un estudio para comparar dos bases de datos relacionales vs dos no relacionales. Para ello se determinaron un conjunto de variables y criterios de comparación. Tales como: Modelo de almacenamiento de datos, rendimiento, flexibilidad, esquema, escalado,

tolerancia a fallas. Durante el desarrollo de la investigación se analizan aspectos generalizados de las bases de datos nosql como son: Control de concurrencia, posibilidades de consulta, replicaciones y concurrencia. [CITATION SAL14 \l 12298]

2.7.1. Análisis de los antecedentes

En las aplicaciones descritas anteriormente se efectúa la evaluación de algún tipo de software a través de normas, estándares o indicadores.

En general las investigaciones antes mencionadas han realizado un estudio previo para seleccionar alguna norma, estándar o indicadores para determinar la calidad en determinados tipos de software. Otras determinaron a través de parámetros o indicadores cual base de datos no relacional se ajustaba mejor a sus necesidades. Se ha demostrado además que la utilización de la norma ISO/IEC 25000 ha resultado efectiva para determinar el grado de calidad de determinados software.

Después de una exhaustiva búsqueda de algún modelo para evaluar bases de datos no relacionales, hasta el momento no se encontró ningún resultado semejante a lo que se pretende realizar.

2.7.2. Conclusión del análisis de antecedentes

Una vez analizado los modelos de evaluación aplicados a distintos tipos de software y de analizar diferentes perspectivas en las investigaciones sobre las bases de datos no relacionales, se puede constatar que resultó positivo aplicar tanto modelos de evaluación basados en las normas ISO y el uso del método IQMC, así como el empleo de bases de datos no relacionales partiendo de parámetros y diferentes criterios de evaluación para solucionar necesidades propias de la investigación.

Se considera oportuno desarrollar un modelo de evaluación de gestores de bases de datos no relacionales fundamentado en los siguientes puntos:

- La aplicación de modelos basados en normas ISO e indicadores arrojó en todos los casos vistos, resultados satisfactorios.
- El empleo de parámetros o de características de bases de datos no relacionales sirvió como base para seleccionar cuál base de datos NoSQL resultó acorde para las necesidades de la investigación.
- Los sistemas analizados en algunas de las investigaciones antes mencionadas, así como las bases de datos no relacionales son todos en general productos de software.

CAPITULO III DESARROLLO DEL MODELO

3.1. INTRODUCCIÓN

En este capítulo se abordará el desarrollo del Modelo de Evaluación a través de la Norma ISO 25000 y del método IQMC para evaluar bases de datos NoSQL a través de los siete pasos que lo conforman, el modelo permitirá comparar y evaluar los diferentes gestores para determinar el más apropiado para poder usar.

En la siguiente tabla se lista las características indispensables halladas en el capítulo 2, con las que debe contar un Sistema Gestor de base de datos no SQL.

Tabla 17
Características base de datos NoSQL

CARACTERISTICAS BD NOSQL
Escalabilidad horizontal
Habilidad de distribución
Uso eficiente de recursos
Libertad de esquema
Concurrencia débil
Consultas simples
CARACTERISTICAS DB-ENGINES
Año versión inicial
Año última versión
Licencia
SO de servidor
Esquema de datos
Índices secundarios
SQL
Lenguajes de programación soportados
Procedimientos
Triggers

CONTINÚA →

Particionamiento

Replicación
Map Reduce
Consistencia
Transaccionalidad
Concurrencia
Conceptos de usuarios

Fuente: Autores del trabajo de titulación

Es importante resaltar que el método IQMC consiste de siete pasos que, aunque se presentan como si fueran secuenciales, pueden ser simultaneados y/o iterados si se considera oportuno. [CITATION Jua \t \l 3082].

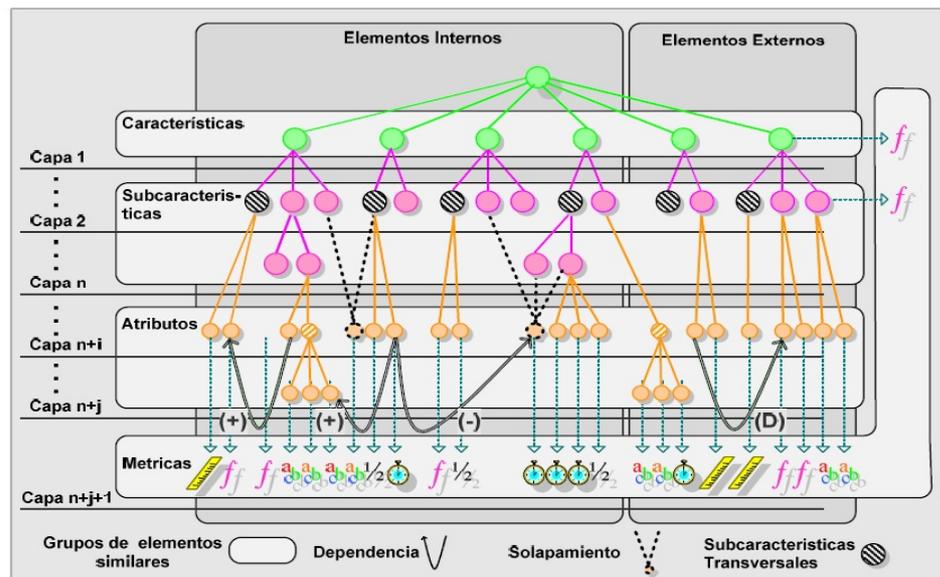


Figura 19 Estructura usada por modelos de calidad

Fuente[CITATION Jua \t \l 3082]

3.2. ESTUDIO DE ÁMBITO DE SOFTWARE

En la actualidad la información aumenta a un ritmo sin precedentes, y cada vez se hace más compleja su administración y las empresas no solo desean almacenar esta información, sino quieren sacarle el mayor provecho; los usuarios piden cada vez más velocidad en

las consultas, y la arquitectura de los sistemas ha tenido un cambio considerable. El porqué de las bases de datos NoSQL se puede resumir en los siguientes tres aspectos:

Tamaño y cantidad de la información

Estos dos aspectos: el aumento tanto en el tamaño de los archivos como en su cantidad ha incidido en el crecimiento del interés hacia tecnologías NoSQL; además, hay que tener en cuenta que las bases de datos relacionales (soluciones actuales) presentan serios problemas en cuanto a escalabilidad en el manejo de la información se refiere, lo que genera que a medida que aumentan los datos, el desempeño disminuye y se hacen menos intuitivas (las consultas son cada vez más largas y complejas).

Velocidad

Las aplicaciones modernas (especialmente aplicaciones Web) tienen muchos usuarios al mismo tiempo que exigen respuestas razonablemente ágiles, el usuario final no está interesado en qué tecnología de bases de datos se use, sino en que la respuesta a sus consultas se dé en un tiempo razonable, como se demuestra en un experimento realizado para medir la paciencia de los usuarios en las consultas, el cual indicó que demoras de medio segundo tienen serias consecuencias en las métricas del negocio.

Falta de innovación

Las bases de datos tradicionales fueron creadas para la misma necesidad general de almacenamiento y manejo de la información, pero se diseñaron teniendo en cuenta las características de la época: grandes computadores y datos estructurados, sin embargo, aunque la necesidad

es la misma, la época ha cambiado, y por eso se hace necesario innovar en el tema para crear soluciones que respondan a las necesidades actuales.

En este punto se puede concluir que el cambio del paradigma en el manejo de la información motivó el auge del movimiento NoSQL.

Las últimas tendencias tanto en la red como en la computación en la nube están haciendo de NoSQL el modelo más deseable. La necesidad de (ampliar rápidamente) escalabilidad, desvincular el hardware del modelo de datos y proporcionar bases de datos más eficientes, son todos factores que contribuyen a esta transición

Por último, debe destacarse que el tema no es nuevo, pero históricamente se puede decir que en la última década se ha hablado mucho de él.

Las tres grandes causas del interés que está causando NoSQL son: los cambios en el tamaño y la cantidad (e incluso estructura) de los datos, la necesidad de velocidad por parte de los usuarios y la falta de innovación en el área. [CITATION Ale12 \t \l 12298]

3.3. DETERMINACIÓN DE CARACTERÍSTICAS Y SUBCARACTERÍSTICAS DE CALIDAD

Se tomará como base la ISO 25010 el cual determina un conjunto de características y subcaracterísticas donde algunas de las existentes deban reformularse ligeramente para adaptarla al dominio de interés, o eliminarse en el caso de no ser necesarias, siguiendo los pasos del método IQMC

Tabla 18
Características y subcaracterísticas seleccionadas

CARACTERÍSTICAS	SUBCARACTERÍSTICAS	OBSERVACIÓN
------------------------	---------------------------	--------------------

Funcionalidad	Complejidad funcional	
	Corrección funcional	No aplica
	Pertinencia funcional	No aplica
Eficiencia	Comportamiento	
	Utilización de recursos	
	Capacidad	
Compatibilidad	Co-Existencia	
	Interoperabilidad	
Usabilidad	Adecuación	No aplica
	Aprendizaje	
	Operatividad	
	Protección de errores de usuario	No aplica
	Estética	
	Accesibilidad	
Fiabilidad	Madurez	
	Disponibilidad	
	Tolerancia a fallos	
	Capacidad de recuperación	

CONTINUA →

Seguridad	Confidencialidad	No aplica
	Integridad	
	No repudio	No aplica
	Responsabilidad	No aplica
	Autenticidad	
Mantenibilidad	Modularidad	No aplica
	Reusabilidad	No aplica

	Capacidad de ser analizado	
	Capacidad de ser modificado	
	Capacidad de ser probado	No aplica
Portabilidad	Adaptabilidad	
	Facilidad de Instalación	
	Capacidad de ser reemplazado	

Fuente: Autores del trabajo de titulación

En la anterior tabla se lista las características y subcaracterísticas de la ISO 25000, que se utilizara para la construcción del modelo, algunas subcaracterísticas no se seleccionaron debido a que los gestores de bases de datos no SQL no cuentan con atributos o cualidades que cumplan con lo especificado, a continuación se describe cada una de las subcaracterísticas que no aplican.

Corrección funcional

Capacidad del producto o sistema para proveer resultados correctos con el nivel de precisión requerido.

Pertinencia funcional

Capacidad del producto software para proporcionar un conjunto apropiado de funciones para tareas y objetivos de usuario especificados.

Adecuación

Capacidad del producto que permite al usuario entender si el software es adecuado para sus necesidades.

Protección contra errores de usuario

Capacidad del sistema para proteger a los usuarios de hacer errores.

Confidencialidad

Capacidad de protección contra el acceso de datos e información no autorizados, ya sea accidental o deliberadamente.

No repudio

Capacidad de demostrar las acciones o eventos que han tenido lugar, de manera que dichas acciones o eventos no puedan ser repudiados posteriormente.

Responsabilidad

Capacidad de rastrear de forma inequívoca las acciones de una entidad.

Modularidad

Capacidad de un sistema o programa de ordenador (compuesto de componentes discretos) que permite que un cambio en un componente tenga un impacto mínimo en los demás.

Reusabilidad

Capacidad de un activo que permite que sea utilizado en más de un sistema software o en la construcción de otros activos.

Capacidad para ser probado

Facilidad con la que se pueden establecer criterios de prueba para un sistema o componente y con la que se pueden llevar a cabo las pruebas para determinar si se cumplen dichos criterios.[CITATION iso \l 12298]

Para sustentar la selección de características y subcaracterísticas se utilizó la técnica de investigación llamada triangulación de fuentes o datos, en donde “se utiliza una variedad de fuentes de información o informante, respecto a un determinado problema o situación o hecho a analizar. La triangulación se produce cuando existe la concordancia o discrepancia entre estas fuentes. Se pueden triangular informantes/personas, tiempos y espacios/contextos”. [CITATION Mor04 \l 12298]

Para aplicar esta técnica se utiliza tres fuentes: las características de las bases de datos no relaciones, las características de la norma ISO 25000 desarrollado y las características de los gestores de bases de datos nosql según DB-Engines, todos estos temas desarrollados en el capítulo 2.

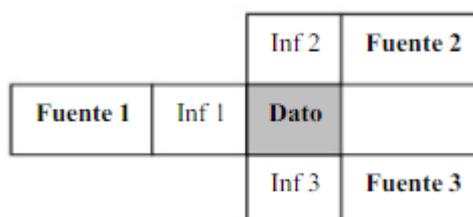


Figura 20 Diagrama triangulación de fuentes

Fuente [CITATION Mor04 \l 12298]

**Tabla 19
Triangulación de fuentes**

Características	Sub características	Fuente 1 ISO 25000	Fuente 2 NoSQL	Fuente 3 DB Engines
Funcionalidad	Compleitud funcional	Grado en el cual el conjunto de funcionalidad	Cada tipo de base de datos contienen	Tiene lista de funciones para

		es cubre todas las tareas y los objetivos del usuario especificados .	diferentes características para que el usuario tenga una noción global del grado de cumplimiento	diferente tipo de base de datos, donde el usuario puede determinar el grado de cumplimiento
Eficiencia	Comportamiento	Tiempos de respuesta	Velocidad en ejecución de consultas	Uso de map-reduce
	Utilización de recursos	Uso eficiente de recursos	Uso eficiente de recursos	Clúster
	Capacidad	Grado de cumplimiento de requisitos	Las bases nosql tiene la característica de manejar un gran volumen de información, siendo el mayor requisito de los usuarios	Especificaciones técnicas del gestor

CONTINÚA →

Compatibilidad	Co-Existencia	Co-existencia con otros software	El modelo de datos flexible hace que la base pueda ser compatible con sistemas y lenguajes de programación ya que maneja estándares como JSON	Lenguaje de programación soportados
	Interoperabilidad	Sistemas que intercambian información		Sincronización de datos
Usabilidad	Aprendizaje	Facilidad de aprendizaje	Todo es dependiente de los desarrolladores de los gestores	Capacitaciones y cursos
	Operatividad	Uso con facilidad		Concepto de usuario
	Estética	Interfaz de usuario		
	Accesibilidad	Capacidad de uso por otro usuario		Licencias libre y pagada
Fiabilidad	Madurez	Desempeña funciones específicas	Funciones especificadas por el desarrollador del gestor	Funciones técnicas
	Disponibilidad	Capacidad del software para estar	Disponibilidad alta por el uso de	Consistencia

		operativo y en uso	replicas	
--	--	-----------------------	----------	--

CONTINÚA →

Seguridad	Tolerancia a fallos	Sistema sigue en funcionamiento a pesar de presentar fallos	La escalabilidad horizontal permite que la base de datos funcione a pesar de los fallos	Replicación
	Capacidad de recuperación	Capacidad para recuperar datos fallidos	Las bases de datos orientadas a columnas tiene como ventaja la facilidad de recuperación de datos	Replicación
	Integridad	Prevenir accesos o modificaciones no autorizados a datos	Consistencia eventual como característica de las NoSQL	Consistencia de datos
	Autenticidad	Capacidad de demostrar la identidad de un sujeto o un recurso.	Todo es dependiente de los desarrolladores de los gestores	Niveles de acceso

CONTINÚA →

Mantenibilidad	Capacidad de ser analizado	Facilidad para diagnosticar deficiencias o fallos	Funcionalidades de los gestores	Conceptos de usuarios
	Capacidad de ser modificado	Facilidad para ser modificado	Modelo de datos es flexible debido a que no utiliza modelo entidad-relación	Esquema de datos libre
Portabilidad	Adaptabilidad	Capacidad software que permite ser adaptado de forma efectiva y eficiente a cualquier entorno	Especificaciones técnicas del gestor	Sistemas operativos soportados
	Facilidad de Instalación	Facilidad para instalar en un determinado entorno	Especificaciones técnicas del gestor	Tutoriales en sitios web oficial
	Capacidad de ser reemplazado	Capacidad software para ser utilizado en vez de otro	Existencia de base de datos nosql que pueden ser sustituidas por otras o por bases de datos relacionales	Tipo de base de datos

Fuente: Autores del trabajo de titulación

3.4. DETERMINACIÓN DE ATRIBUTOS DERIVADOS Y BÁSICOS

Este apartado del método IQMC es importante ya que se pueden tener atributos que aun necesiten ser descompuestos en atributos básicos y que puedan ser medibles.

A continuación una tabla con las características, subcaracterísticas y atributos planteados para usarlo en el modelo de evaluación planteado.

Tabla 20
Características, subcaracterísticas atributos de calidad

CARACTERÍSTICAS Y SUBCARACTERÍSTICAS Y ATRIBUTOS DE CALIDAD			
CARACTERÍSTICAS	SUBCARACTERÍSTICAS	ATRIBUTOS	
Funcionalidad	Complejidad funcional	Manipulación de datos	Ejecución de operaciones básicas
			Herramientas de administración
		Funcionalidades del Sistema	Índices secundarios
			Procedimientos
			Vistas
			MapReduce
			Triggers
		Habilidad de distribución	Copias de seguridad
Esquema de datos	Ambientes distribuidos		
		Esquema Libre	
Eficiencia	Comportamiento	Tiempo de respuesta	
	Utilización de recursos Capacidad	Recursos de Hardware Manejo de Información	Escalabilidad horizontal
			Escalabilidad vertical
			Requisitos de Hardware para instalación
		Manejo a gran escala	
Compatibilidad	Co-Existencia	Interacción con otros sistemas	Generadores de reportes

CONTINÚA →

	Interoperabilidad	Interoperabilidad con otros gestores	Sincronizar datos
Usabilidad	Aprendizaje	Facilidad de aprendizaje	Capacitación o seminarios
			Cursos específicos del gestor
			Cursos on-line
			Video tutoriales
			Video tutoriales claros y entendibles
		Sitio web oficial	
		Información del sitio oficial útil	
		Documentación Disponible	Cuenta con documentación
			Documentación entendible
			Cuenta con casos de estudios e investigaciones
	Casos de estudio e investigaciones útiles		
	Existen proyectos de ejemplo		
	Ejemplos entendibles.		
Operatividad	Facilidad de administración		
Estética	Interfaz gráfica		
	Facilidad de navegación		
Accesibilidad	Ayuda en línea		
	Comunidad de soporte		
	Facilidad para desarrolladores con habilidades especiales		
Fiabilidad	Madurez	Tiempo en el mercado	

CONTINÚA →

	Disponibilidad	Actualizaciones disponibles	
		Base de conocimiento	
		Obtención del gestor	Disponibilidad de descarga Descargar del sitio oficial Descargar de sitios no oficiales
	Tolerancia a fallos	Recuperación Automática	Trabaja en clúster
		Tolerancia a fallos de gestor	Maneja Transaccionalidad
		Tolerancia a fallos de datos	Respaldos y recuperación de datos.
	Capacidad de recuperación	Facilidad para restaurar datos	
		Facilidad para recuperar datos eliminados	
	Seguridad	Integridad	Perfiles de acceso a esquemas
Autenticidad		Administración del gestor Nivel de acceso	
Mantenibilidad	Capacidad de ser analizado	Detección de errores	
	Capacidad de ser modificado	Modificaciones en caliente	
Portabilidad	Adaptabilidad	Sistema operativo	Funciona sobre Windows
			Funciona sobre Unix
			Funciona sobre Linux
			Funciona sobre BSD

CONTINÚA →

			Funciona sobre OS X
		Idiomas disponibles	Ingles
			Español
	Capacidad de Instalación	Facilidad de instalación	Otros
			Manuales de instalación
			Ayuda en Línea
	Capacidad de ser reemplazado	Migración	Tiempo de instalación
Migrar base a un nuevo gestor			

Fuente: Autores del trabajo de titulación

3.5. DETERMINACIÓN DE MÉTRICAS PARA LOS ATRIBUTOS

Para medir los atributos se va usar métricas objetivas también conocidas como medidas directas, estas permitirán determinar el nivel de cumplimiento de cada gestor de los atributos encontrados en el desarrollo del paso 3 y 4 del método IQMC.

Métrica de cumplimiento: Permite determinar si un gestor cumple o no con un atributo con las medidas que se detalla en la siguiente tabla.

Tabla 21
Métrica de cumplimiento

Cumplimiento	Equivalente
Si	1
No	0

Fuente: Autores del trabajo de titulación

Métrica de cumplimiento por rango: Permite evaluar en que magnitud el gestor que se está evaluando cumple con el atributo las medidas se detallan en la siguiente tabla.

Tabla 22
Métrica de cumplimiento por rangos

Intervalos	Equivalente	Cumplimiento
0	0	Nulo
1	0,25	Medio
2	0,5	Bajo
3	0,75	Alto
4	1	Completo

Fuente: Autores del trabajo de titulación

Tabla 23
Métrica de cumplimiento de utilidad

Cumplimiento	Equivalente
1	Muy útil
0,75	Útil
0,50	Poco Útil
0	Nada Útil

Fuente: Autores del trabajo de titulación

Tabla 24
Métrica de cumplimiento

Cumplimiento	Equivalente
1	Alto
0,50	Medio
0	Bajo

Fuente: Autores del trabajo de titulación

Modelo de Evaluación de Base de Datos No SQL

Tabla 25
Modelo evaluador de gestores de base de datos no SQL

ISO 25000		
Características / Subcaracterísticas	Métrica	Descripción
FUNCIONALIDAD		
Capacidad de la base de datos en proporcionar las funcionalidades necesarias para satisfacer las necesidades del usuario.		
COMPLETITUD FUNCIONAL		Capacidad de la base de datos para proporcionar un conjunto apropiado de funciones en tareas específicas.
Manipulación de datos		Provee las funciones básicas para la administración de base de datos.
1	Ejecución de operaciones básicas	Si = 1 / No = 0 Permite realizar las operaciones básicas insert, update y delete.
2	Herramientas de manipulación	Si = 1 / No = 0 Cuenta con herramientas que apoyen en la manipulación de datos.
Funcionalidades del sistema		Capacidad de la base de datos para proporcionar un conjunto apropiado de funciones en tareas específicas.
1	Índices secundarios	Si = 1 / No = 0 Permite el manejo de índices secundarios.
2	Procedimientos	Si = 1 / No = 0 Permite crear procedimientos.

CONTINÚA →

3	Vistas	Si = 1 / No = 0	Permite crear vistas.
4	MapReduce	Si = 1 / No = 0	Soporta MapReduce.
5	Triggers	Si = 1 / No = 0	Permite el uso de Triggers.
6	Copias de seguridad	Si = 1 / No = 0	Permite realizar copias de seguridad.
Habilidad de distribución			Capacidad de la base de datos para proporcionar un conjunto apropiado de funciones en tareas específicas.
1	Ambientes distribuidos	Si = 1 / No = 0	La base de datos soporta ambientes distribuidos.
Esquema de datos			Existe libertad de esquema al modelar los datos.
1	Esquema libre	Si = 1 / No = 0	El esquema de modelado es flexible.
EFICIENCIA			
Capacidad de la base de datos de responder en el menor tiempo posible a una petición de una funcionalidad.			
COMPORTAMIENTO			Se refiere a los tiempos de respuesta cuando lleva a cabo sus funciones bajo condiciones determinadas.
1	Tiempo de respuesta	Alto = 1 / Medio = 0,50 / Bajo = 0	Los tiempos de respuesta de una transacción son óptimos.
UTILIZACION DE RECURSOS			Cantidad y recursos adecuados para que la

CONTINÚA →

			base de datos tenga un correcto funcionamiento.
	Recursos de Hardware		Cuantos recursos de hardware debe tener para funcionar correctamente.
	1 Escalabilidad horizontal	Si = 1 / No = 0	Añadir más nodos particulares del sistema.
	2 Escalabilidad vertical	Si = 1 / No = 0	Añadir más recursos a un nodo particular del sistema.
	3 Requisitos de Hardware para instalación	Si = 1 / No = 0	Necesita más de un recurso de Hardware para instalar un gestor de base de datos.
	CAPACIDAD		Capacidad del producto de software para proporcionar rendimiento apropiado.
	Manejo de información		Capacidad de procesar grandes volúmenes de información.
	1 Manejo a gran escala	Alto = 1 / Medio = 0,50 / Bajo = 0	Permite el manejo de grandes cantidades de información.
COMPATIBILIDAD	Capacidad de la base de datos para intercambiar información.		
	COEXISTENCIA		Capacidad de la base para coexistir con otro software independiente.
	Interacción con otros sistemas		Capacidad de comunicación con otros productos software.
	Generadores de reportes	Si = 1 / No = 0	Puede comunicarse con generadores de reportes.

CONTINÚA →

INTEROPERABILIDAD		Capacidad de la base de datos que permite que otro sistema pueda operarlo y controlarlo.	
Interoperabilidad con otros gestores		Capacidad de intercambiar información con otro gestor.	
1	Sincronizar datos.	Si = 1 / No = 0	Permite sincronizar los datos con otro servidor.
USABILIDAD			
Capacidad de las base de datos para ser entendida aprendida y usada por el usuario.			
APRENDIZAJE		Capacidad de la base de datos que permite al usuario aprender su aplicación.	
Facilidad de aprendizaje		Provee las funciones básicas para la administración de base de datos.	
1	Capacitación o seminarios	Si = 1 / No = 0	Existen eventos de capacitación.
2	Cursos específicos del gestor	Si = 1 / No = 0	Existe preparación para los desarrolladores por parte del fabricante.
3.	Cursos on-line	Si = 1 / No = 0	Existen cursos en la web para reforzar el conocimiento.
4.	Video tutoriales.	Si = 1 / No = 0	Existen videos explicativos del gestor
5.	Utilidad de video tutoriales	Muy útil = 1 / Útil = 0,75 / Poco útil = 0,50 / Nada útil = 0 / NA = No aplica	Si existen videos tutoriales que representan apoyo para una nueva implantación.

CONTINÚA →

6.	Sitio web oficial	Si = 1 / No = 0	Tiene un sitio web oficial del fabricante.
7.	Utilidad del sitio web oficial	Muy útil = 1 / Útil = 0,75 / Poco útil = 0,50 / Nada útil = 0 / NA = No aplica	Si existe el sitio web este representa un apoyo para búsqueda de información.
Documentación disponible			Provee las funciones básicas para la administración de base de datos.
1	Cuenta con documentación	Si = 1 / No = 0	Existen manuales sobre el uso del gestor.
2.	Documentación entendible	Muy útil = 1 / Útil = 0,75 / Poco útil = 0,50 / Nada útil = 0	Si existe documentación, la documentación disponible es útil y entendible al momento de usarla.
3.	Cuenta con estudios e investigaciones	Si = 1 / No = 0	Existen investigaciones o estudios del gestor que apoyen al desarrollador.
4.	Casos de estudio e investigación útiles	Muy útil = 1 / Útil = 0,75 / Poco útil = 0,50 / Nada útil = 0 / NA = No aplica	Si cuentan con casos de estudios e investigaciones, estos representan una ayuda los casos de estudio e investigación en caso de dudas.
5.	Existen proyectos de ejemplo	Si = 1 / No = 0	Se cuenta con proyectos ejemplos como apoyo para el desarrollador.
6.	Ejemplos entendibles	Muy útil = 1 / Útil = 0,75 / Poco útil = 0,50 / Nada útil = 0 / NA = No aplica	Los ejemplos disponibles son útiles para realizar nuevas implementaciones.

CONTINÚA →**OPERABILIDAD**

Capacidad de la base de datos que permite al usuario operarlo y controlarlo.

FIABILIDAD

Capacidad de la base de datos de realizar funciones específicas bajo ciertas condiciones.

MADUREZ

Capacidad del gestor para mantenerse operativo en correcto funcionamiento.

CONTINÚA →

1	Tiempo en el mercado	0 : 4	Años que el gestor ha estado en el mercado si es mayor a 5 toma el valor de 1.
2	Actualizaciones disponibles	Si = 1 / No = 0	En los últimos 12 meses ha existido una nueva versión.
3	Base de conocimiento	Si = 1 / No = 0	Se cuenta con una base de conocimiento de los errores presentados y posibles soluciones.
DISPONIBILIDAD			Se refiere si el sistema es accesible para su uso cuando lo requiere.
Obtención del Gestor			Disponibilidad del gestor para su uso.
1	Disponibilidad de descarga	Si = 1 / No = 0	Se puede descargar el gestor desde el internet.
2	Descargar del sitio oficial	Si = 1 / No = 0	La versión está disponible en el sitio oficial.
3	Descargar de sitios no oficiales	Si = 1 / No = 0	La versión está disponible en sitios no oficiales.
3	Replicación de datos	Si = 1 / No = 0	Proceso de copiar y mantener

		actualizados los datos en varios nodos de bases de datos.
	TOLERANCIA A FALLOS	Capacidad para operar ante fallos de hardware y software.

CONTINÚA →

	Recuperación Automática		Capacidad del gesto para mantener un nivel de prestaciones en caso de fallos en su funcionalidad.
	1 Trabaja en clúster	Si = 1 / No = 0	Permite manejo de clúster.
	Tolerancia a fallos del gestor		Capacidad del gestor en recuperarse luego de un fallo en los datos.
	1 Maneja Transaccionalidad	Si = 1 / No = 0	Provee mecanismos para recuperar el gestor en caso de fallas.
	Tolerancia a fallos en datos.		Capacidad del gestor de recuperarse luego de fallos en la información.
	1 Respaldos de datos.	Si = 1 / No = 0	Provee mecanismos de respaldos y recuperación de datos.
	CAPACIDAD DE RECUPERACIÓN		Capacidad del gestor para reestablecer un nivel de prestaciones y recuperar datos afectados en caso de fallas.
	1 Facilidad para restaurar datos	Si = 1 / No = 0	Cuenta con mecanismos para restaurar los datos.
	2 Facilidad para recuperar datos eliminados	Si = 1 / No = 0	Cuenta con mecanismos para recuperar datos borrados.
SEGURIDAD			

Capacidad de la base de datos para proteger la información de manera que usuarios no autorizados puedan acceder a ella.

INTEGRIDAD

Capacidad del gestor para prevenir accesos o modificaciones no autorizadas a los datos.

CONTINÚA →

1 Perfiles de acceso a esquemas

Si = 1 / No = 0

Proporciona mecanismos para prevenir el acceso no autorizado a los datos almacenados.

AUTENTICIDAD

Capacidad del gestor de demostrar la identidad de un usuario que desea acceder al gestor.

1 Administración del gestor

Si = 1 / No = 0

Proporciona mecanismos para prevenir el acceso no autorizado al gestor.

2 Nivel de acceso

Si = 1 / No = 0

Permite parametrizar a un usuario para acceder a las funcionalidades en función a perfiles.

MANTENIBILIDAD

Capacidad de la base de datos para ser modificada efectiva y eficientemente.

CAPACIDAD DE SER ANALIZADO

Capacidad del gestor para identificar problemas y fallos.

1 Detección de errores

Si = 1 / No = 0

Permite identificar errores al ejecutar una acción en caso de no ser ejecutada con éxito.

CAPACIDAD DE SER MODIFICADO

Capacidad del gestor para soportar una modificación.

	1	Modificaciones en caliente	Si = 1 / No = 0	Permite realizar modificaciones en el gestor en caliente sin afectar su rendimiento.
--	---	----------------------------	-----------------	--

CONTINÚA →

PORTABILIDAD

Capacidad de la base de datos para ser transferido de un entorno a otro de hardware o software.

ADAPTABILIDAD			Capacidad del gestor para adaptarse a diferentes entornos.
Sistema operativo			Capacidad del gestor para funcionar correctamente sobre un determinado sistema operativo.
1	Funciona sobre Windows	Si = 1 / No = 0	Capacidad de funcionar sobre Windows.
2	Funciona sobre Unix	Si = 1 / No = 0	Capacidad de funcionar sobre Unix.
3	Funciona sobre Linux	Si = 1 / No = 0	Capacidad de funcionar sobre Linux.
4	Funciona sobre BSD	Si = 1 / No = 0	Capacidad de funcionar sobre BSD.
3	Funciona sobre OS X	Si = 1 / No = 0	Capacidad de funcionar sobre OS X.
Idiomas disponibles			Capacidad del gestor para funcionar correctamente sobre un determinado sistema operativo.

	1	Inglés	Si = 1 / No = 0	Disponible en idioma inglés.
--	---	--------	-----------------	------------------------------

CONTINÚA →

	2	Español	Si = 1 / No = 0	Disponible en idioma español.
	3	Otros	Si = 1 / No = 0	Disponible otros idiomas.
CAPACIDAD DE INSTALACIÓN				
Facilidad de instalación			Cuenta con mecanismos que faciliten la instalación del gestor.	
	1	Manuales de instalación	Si = 1 / No = 0	Se cuenta con manuales de instalación.
	2	Ayuda en Línea	Si = 1 / No = 0	Se cuenta con línea de apoyo durante la instalación.
	3	Tiempo de instalación	0 : 4	Cuanto tiempo tarda en instalar el gestor, si es menor a 4 el valor será 1.
CAPACIDAD DE SER REEMPLAZADO				
Migración			Migrar los datos a un nuevo gestor de base de datos.	
	1	Migrar base a un nuevo gestor	Si = 1 / No = 0	Se cuenta con las herramientas para poder migrar los datos a un nuevo gestor.

Fuente: Autores del trabajo de titulación

3.6 SELECCIÓN DE NIVEL DE IMPORTANCIA DE LAS CARACTERÍSTICAS

La norma ISO/IEC 25010 determina que es importante establecer niveles de importancia a las características, para la evaluación del modelo el nivel de importancia se colocara en dependencia de las necesidades de un negocio particular.

Tabla 26
Niveles de importancia por características

CARACTERÍSTICA ISO/EIEC 25010	NIVEL DE IMPORTANCIA	PESO (%)
Adecuación Funcional	Primordial	15,87
Eficiencia	Primordial	7,94
Compatibilidad	Medio	3,17
Usabilidad	Primordial	30,16
Fiabilidad	Medio	19,05
Seguridad	Primordial	4,76
Mantenibilidad	Medio	3,17
Portabilidad	Medio	15,87
TOTAL		100

Fuente: Autores del trabajo de titulación

CAPITULO IV APLICACIÓN Y PRUEBAS

El presente capítulo presenta la parte final y práctica de la investigación, donde se pretende aplicar el modelo de evaluación desarrollado en el capítulo anterior.

4.1 ANÁLISIS CASO DE ESTUDIO DE ELECCIÓN DE BASES DE DATOS NO SQL

Para realizar la evaluación se ha tomado en cuenta un trabajo de grado con el tema: “ANÁLISIS DE LAS BASES DE DATOS COMO ALTERNATIVA A LAS BASES DE DATOS SQL”, [CITATION Peñ12 \l 12298] este tiene como objetivo final identificar una alternativa viable NoSQL, para ello han tomado en cuenta 4 gestores Cassandra, MongoDB, Neo4j y DynamoDB. [CITATION Peñ12 \l 12298]

Las características analizadas en el trabajo de grado son:

Características		
Mantenibilidad	Características de la máquina	Instalación/Implementación
Usabilidad	Versión	Soporte y solución de inquietudes
Madurez	Documentación	Ventajas/Desventajas/Limitaciones

Figura 21 Características seleccionadas en el caso de estudio

Fuente [CITATION Peñ12 \l 12298]

La búsqueda se realiza en internet debido a que es un tema que pocos conocen y que es relativamente nuevo, por lo que encontrar información física es difícil; con el objeto de complementar la información obtenida se procederá a indagar en trabajos de grado, tesis o noticias sobre temas relacionados que puedan corroborar la veracidad de los datos encontrados en las diferentes páginas web durante la búsqueda. Paralelamente a esta actividad se comienza a obtener contactos en

diferentes empresas para la realización de una encuesta que será el núcleo del trabajo de grado sobre el que giren las decisiones a tomar de la alternativa NoSQL. La encuesta se realizó inicialmente en Microsoft Word 2007 y luego se adaptó a Google forms, un servicio de formularios que presta Google para la realización de encuestas por medio de internet. [CITATION Peñ12 \l 12298]

La encuesta consta de 16 preguntas basadas en su mayoría en conocer qué bases de datos utiliza la empresa para la que trabaja el encuestado y las actividades que involucra poseerlas, como mantenimiento, políticas de uso, requisitos, entre otras; y la parte restante, en decisiones sobre el uso de los motores relacionadas también con las políticas de la empresa y conocimiento breve de NoSQL. [CITATION Peñ12 \l 12298]

Seguidamente se analizarán las respuestas dadas a cada pregunta en base a la muestra de empresas encuestadas y con esta estructura se permitirá la realización de los objetivos siguientes e influencia sobre el concepto NoSQL. Con las dos formas de encuesta (archivo en Microsoft Word y Google forms) se enviaron las encuestas por medio de correos electrónicos a los contactos en cinco diferentes empresas.[CITATION Peñ12 \l 12298]

En esta lista se identifican los puntos clave a considerar de acuerdo a las respuestas obtenidas de los encuestados para optar por una alternativa de solución a los motores SQL. Terminada la realización de las encuestas, se completa la lista de chequeo, sobre la cual las cuatro bases de datos NoSQL se evalúan, de tal forma que se permita seleccionar y analizar por qué se escogió determinada opción, dando por completado el objetivo específico 3, y también el objetivo de este trabajo de grado

A continuación, se muestra una lista de chequeo con la estructura Características vs. Bases de datos NoSQL la cual se ha presentado en el trabajo de grado que se está analizando para realizar la comparación con el modelo propuesto en el capítulo 3.

ANÁLISIS DE LAS BASES DE DATOS COMO ALTERNATIVA A LAS BASES DE DATOS SQL

1. ¿Qué bases de datos ha adquirido/usado la organización para el almacenamiento de información en los últimos 5 años?
 - a. MySQL
 - b. Microsoft SQL Server
 - c. Oracle
 - d. PostgreSQL
 - e. Otro, ¿cuál/cuáles? _____

2. ¿Qué características se tuvieron en cuenta para la utilización de las bases de datos anteriores?
 - a. Soporte del proveedor
 - b. Integración con otras aplicaciones
 - c. Facilidad de uso
 - d. Velocidad en las transacciones
 - e. Funcionalidad
 - f. Precio

3. ¿Qué factor es más importante cuando una aplicación hace uso de la base de datos?
 - a. Almacenamiento de datos.
 - b. Tiempos de respuesta

4. ¿Qué funcionalidades son indispensables en un motor para que pueda hacer parte de la infraestructura de almacenamiento en la empresa?
 - a. Stored Procedures
 - b. Vistas (views)
 - c. Triggers
 - d. Funciones
 - e. Event Scheduler (eventos programados)
 - f. Otros, (¿cuáles?) _____

CONTINÚA →

5. ¿Sobre qué sistema operativo corre el servidor de bases de datos?

6. ¿Cada cuánto tiempo realizan respaldos a la información contenida en la base de datos?
 - a. Varias veces a la semana
 - b. Cada semana
 - c. Cada mes

7. ¿En alguna aplicación es necesaria la sincronización de datos en otro(s) servidor(es)?
 - a. Si
 - b. No
8. ¿Tienen soporte por parte del proveedor de la base de datos?
 - a. Si
 - b. No
9. ¿Cuánto es el costo mensual de mantenimiento y administración de una base de datos en un servidor local?
 - a. Menos de \$500.000
 - b. Entre \$500.000 y \$1.000.000
 - c. Más de \$1.000.000
10. Cuando sale al mercado una nueva versión del motor, ¿lo adquieren inmediatamente?
 - a. Si
 - b. No
11. ¿Las aplicaciones actuales son en su mayoría desarrollos internos o realizados por terceros?
 - a. Desarrollado en casa (interno)
 - b. Desarrollado por terceros
12. ¿De forma general, ¿qué consideraciones tienen presentes para la migración de los datos a una nueva versión u otro motor de base de datos?
13. ¿Existe alguna política en la empresa que obligue a almacenar la información en instalaciones de la empresa?
 - a. Si
 - b. No

CONTINÚA →

14. ¿Estaría dispuesto a utilizar algún sistema de almacenamiento ubicado en instalaciones de un proveedor especializado que brinde garantías de seguridad disponibilidad e integridad?
- a. Si
 - b. No
15. ¿Conoce el término NoSQL?
- a. Si
 - b. No
16. ¿Conoce alguna de las siguientes bases de datos?
- a. Cassandra
 - b. BigTable
 - c. Dynamo
 - d. SimpleDB
 - e. Ninguna de las anteriores

Figura 22 Encuesta de la investigación analizada

Fuente [CITATION Aná121 \t \l 3082]

En base a los resultados de la encuesta el autor ha realizado lista de chequeo:

		Bases de datos NoSQL			
Características		Cassandra	MongoDB	Neo4j	DynamoDB
1	¿Tiene marco de trabajo gráfico?	Si	Si	Si	Si
2	¿Se tiene soporte por parte del proveedor?	No	No	Si	Si
3	¿Se puede integrar con otras aplicaciones?	Si	Si	Si	Si
4	¿Optimiza el tiempo de repuesta en las transacciones?	Si	Si	Si	Si
5	¿Tiene una versión estable de por lo menos este año?	Si	Si	Si	Si
6	¿Almacena los datos de forma óptima?	Si	Si	Si	Si
7	¿Contiene procedimientos almacenados, vistas, triggers, etc.?	Si	No	Si	Si
8	¿Se pueden particionar las tablas que contienen los datos?	Si	Si	Si	Si
9	¿Permite la replicación de los datos?	Si	Si	Si	Si
10	¿Puede funcionar bajo sistema operativo Windows?	Si	Si	Si	Si
11	¿Puede funcionar bajo sistema operativo AIX?	No	No	Si	No
12	¿Se pueden realizar procesos de back up varias veces a la semana?	Si	Si	Si	Si
13	¿Es posible sincronizar los datos con otro servidor?	Si	Si	Si	Si
14	¿Se asegura la integridad, confidencialidad y disponibilidad de los datos?	Si	Si	Si	Si
15	¿Tiene medidas de aseguramiento de recuperación de desastres?	Si	Si	Si	Si
16	¿Tiene herramientas que apoyen la administración de los datos?	Si	Si	No	Si
17	¿Los datos son almacenados en servidores propios del usuario?	Si	Si	Si	No
18	¿Es usuario es el encargado de la gestión y mantenimiento de los datos?	Si	Si	Si	No

Figura 23 Lista de chequeo de la investigación analizada

Fuente [CITATION Aná121 \t \l 3082]

Una vez analizadas las encuestas, arrojo los siguientes resultados.

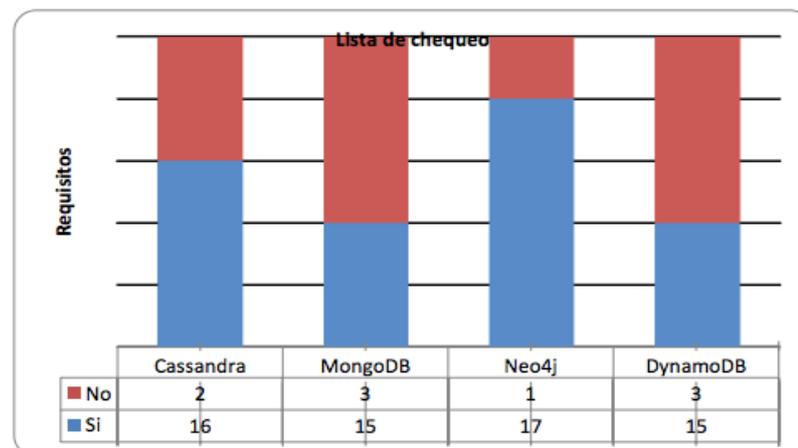


Figura 24 Resultados lista de chequeo de la investigación analizada

Fuente[CITATION Aná121 t \ 3082]

Para la alternativa NoSQL, se considerará de acuerdo con la figura anterior, la que tenga más respuestas “Si”, ya que satisfacen los requisitos analizados después de realizar las encuestas a las organizaciones. Según esto el orden sería el siguiente:

1. Neo4j
2. Cassandra
3. DynamoDB
4. MongoDB.

DynamoDB, de Amazon, aunque obtuvo un tercer lugar, consideramos que debería ser el primer acercamiento NoSQL para las empresas ya que es un concepto que aún no es muy reconocido y con esta se facilita la operación y mantenimiento de los elementos del motor, contienen un costo directo asociado para utilizarlo, pero no sería un problema debido a las ventajas de integración, soporte, respaldo, entre otros atributos vistos en los numerales anteriores. Se debe tener en cuenta que en este sistema los datos son almacenados y administrados por un tercero, es decir que para las políticas de la compañía en la cual se restringe el almacenamiento externo, no sería posible; pero sí sería viable y posiblemente a considerar como opción en aquellas que respondieron

afirmativamente a la pregunta número 14 de la encuesta (80% de los encuestados), en la que se preguntaba sobre la disposición a utilizar instalaciones de un proveedor con garantías de confidencialidad, integridad y disponibilidad sobre los datos. [CITATION Peñ12 \l 12298]

Como segunda alternativa a considerar para las empresas que no sea posible almacenar ninguna información fuera de la empresa es Neo4j, preferiblemente la versión Empresarial que tiene soporte y mayor disponibilidad, aunque es de resaltar que esta solución NoSQL está más enfocada a actividades del negocio donde se utilicen nodos e interacciones entre estos como lo son los grafos, de igual forma el resultado o el sistema con el que interactúa el usuario final son gráficas de datos geospaciales. Para las compañías cuyo propósito no es el anteriormente mencionado, esta base de datos no es una opción viable. [CITATION Peñ12 \l 12298]

4.2 EVALUACION DE LOS GESTORES DE BASES DE DATOS NO SQL

La evaluación se aplicará a los gestores Cassandra, MongoDB, Neo4j y DynamoDB los cuales fueron analizados en el caso de estudio del acápite 4.1

Para poder realizar la evaluación se ha recolectado la información necesaria de los gestores de sitios oficiales, esto permite dar fundamento a los atributos que se desea evaluar.

A continuación, se muestra la matriz de evaluación aplicada a los gestores antes mencionados, posteriormente se obtiene un cuadro con las ponderaciones de cada una de las características y su calificación final, para poder seleccionar el gestor mejor puntuado según la evaluación realizada.

4.2.1 Aplicación del modelo evaluador de gestores de base de datos NoSQL.

A continuación, se procede a aplicar el modelo de evaluación propuesto en base a las características, subcaracterísticas, atributos y métricas elegidas.

APLICACIÓN DEL MODELO EVALUADOR PROPUESTO

Tabla 27
Aplicación del modelo evaluador propuesto

ISO 25000						
Características / Subcaracterísticas	Métrica	CASSANDRA	MONGODB	NEO4J	DYNAMODB	
FUNCIONALIDAD						
COMPLETITUD FUNCIONAL						
Manipulación de datos						
1	Ejecución de operaciones básicas	Si = 1 / No = 0	1	1	1	1
2	Herramientas de manipulación	Si = 1 / No = 0	1	1	1	1
Funcionalidades del sistema						
1	Índices secundarios	Si = 1 / No = 0	0	1	1	1
2	Procedimientos	Si = 1 / No = 0	0	1	1	0
3	Vistas	Si = 1 / No = 0	1	0	1	1

CONTINÚA →

4	MapReduce	Si = 1 / No = 0	1	1	0	0
5	Triggers	Si = 1 / No = 0	1	0	1	1
6	Copias de seguridad	Si = 1 / No = 0	1	1	1	1
Habilidad de distribución						
1	Ambientes distribuidos	Si = 1 / No = 0	1	1	1	1
Esquema de datos						
1	Esquema libre	Si = 1 / No = 0	1	1	1	1
TOTAL FUNCIONALIDAD (10)			8	8	9	8
EFICIENCIA						
COMPORTAMIENTO						
1	Tiempo de respuesta	Alto = 1 / Medio = 0,50 / Bajo = 0	1	1	1	1
UTILIZACION DE RECURSOS						
Recursos de Hardware						
1	Escalabilidad horizontal	Si = 1 / No = 0	1	1	1	1
2	Escalabilidad vertical	Si = 1 / No = 0	0	1	0	1
3	Requisitos de Hardware para instalación	Si = 1 / No = 0	1	1	1	1

CONTINÚA →

CAPACIDAD						
Manejo de información						
1	Manejo a gran escala	Alto = 1 / Medio = 0,50 / Bajo = 0	1	1	1	1
TOTAL EFICIENCIA (5)			4	5	4	5
COMPATIBILIDAD						
COEXISTENCIA						
Interacción con otros sistemas						
	Generadores de reportes	Si = 1 / No = 0	0	1	0	0
INTEROPERABILIDAD						
Interoperabilidad con otros gestores						
1	Sincronizar datos.	Si = 1 / No = 0	0	1	1	1
TOTAL COMPATIBILIDAD (2)			0	2	1	1
USABILIDAD						
APRENDIZAJE						
Facilidad de aprendizaje						
1	Capacitación o seminarios	Si = 1 / No = 0	1	1	1	1
2	Cursos específicos del gestor	Si = 1 / No = 0	0	1	0	0

CONTINÚA →

3.	Cursos on-line	Si = 1 / No = 0	0	0	0	0
4.	Video tutoriales.	Si = 1 / No = 0	1	1	1	1
5	Utilidad de video tutoriales	Muy útil = 1 / Útil = 0,75 / Poco útil = 0,50 / Nada útil = 0 / NA = No aplica	1	0.75	1	0
6.	Sitio web oficial	Si = 1 / No = 0	1	1	1	1
7	Utilidad sitio web oficial	Muy útil = 1 / Útil = 0,75 / Poco útil = 0,50 / Nada útil = 0	1	1	1	1
Documentación disponible						
1	Cuenta con documentación	Si = 1 / No = 0	1	1	1	1
2	Documentación entendible	Muy útil = 1 / Útil = 0,75 / Poco útil = 0,50 / Nada útil = 0 / NA = No aplica	1	1	1	1
3.	Cuenta con estudios e investigaciones	Si = 1 / No = 0	0	1	1	1

CONTINÚA →

4	Casos de estudio en investigaciones útiles	Muy útil = 1 / Útil = 0,75 / Poco útil = 0,50 / Nada útil = 0 / NA = No aplica	0	0	0	0
5	Existen proyectos de ejemplo	Si = 1 / No = 0	1	1	1	0
6	Ejemplos entendibles	Muy útil = 1 / Útil = 0,75 / Poco útil = 0,50 / Nada útil = 0 / NA = No aplica	1	1	1	1
OPERABILIDAD						
1	Facilidad de administración		1	1	1	1
ESTETICA						
1	Interfaz grafica	Si = 1 / No = 0	1	1	1	1
2	Facilidad de navegación	Si = 1 / No = 0	0	1	1	1

CONTINÚA →

ACCESIBILIDAD						
1	Ayuda en línea	Si = 1 / No = 0	0	0	0	0
2	Comunidad de soporte	Si = 1 / No = 0	1	0	1	0
3	Facilidad para desarrolladores con habilidades especiales	Si = 1 / No = 0	1	1	1	1
TOTAL USABILIDAD (14)			9	11	11	9
FIABILIDAD						
MADUREZ						
1	Tiempo en el mercado	0 : 4	1	1	1	1
2	Actualizaciones disponibles	Si = 1 / No = 0	1	1	1	0
3	Base de conocimiento	Si = 1 / No = 0	0	1	1	1
DISPONIBILIDAD						
Obtención del Gestor						
1	Disponibilidad de descarga	Si = 1 / No = 0	1	1	1	1
2	Descargar del sitio oficial	Si = 1 / No = 0	1	1	1	0

CONTINÚA →

3	Descargar de sitios no oficiales	Si = 1 / No = 0	0	1	1	0
3	Replicación de datos	Si = 1 / No = 0	1	1	1	0
TOLERANCIA A FALLOS						
Recuperación Automática						
1	Trabaja en clúster	Si = 1 / No = 0	0	1	1	1
Tolerancia a fallos del gestor						
1	Maneja Transaccionalidad	Si = 1 / No = 0	0	0	1	0
Tolerancia a fallos en datos.						
1	Respaldo de datos.	Si = 1 / No = 0	1	1	1	1
CAPACIDAD DE RECUPERACIÓN						
1	Facilidad para restaurar datos	Si = 1 / No = 0	1	1	1	1
2	Facilidad para recuperar datos eliminados	Si = 1 / No = 0	0	1	1	1
TOTAL FIABILIDAD (12)			7	11	12	7

CONTINÚA →

SEGURIDAD						
INTEGRIDAD						
1	Perfiles de acceso a esquemas	Si = 1 / No = 0	1	1	1	1
AUTENTICIDAD						
1	Administración del gestor	Si = 1 / No = 0	1	1	1	1
2	Nivel de acceso	Si = 1 / No = 0	1	1	1	1
TOTAL SEGURIDAD (3)			3	3	3	3
MANTENIBILIDAD						
CAPACIDAD DE SER ANALIZADO		Capacidad del gestor para identificar problemas y fallos.				
1	Detección de errores	Si = 1 / No = 0	0	0	1	1
CAPACIDAD DE SER MODIFICADO		Capacidad del gestor para soportar una modificación.				
1	Modificaciones en caliente	Si = 1 / No = 0	1	1	1	1
TOTAL MANTENIBILIDAD (2)			1	1	2	2
PORTABILIDAD						
ADAPTABILIDAD						

CONTINÚA →

Sistema operativo						
1	Funciona sobre Windows	Si = 1 / No = 0	1	1	1	0
2	Funciona sobre Unix	Si = 1 / No = 0	0	0	0	0
3	Funciona sobre Linux	Si = 1 / No = 0	1	1	1	0
4	Funciona sobre BSD	Si = 1 / No = 0	1	0	0	0
3	Funciona sobre OS X	Si = 1 / No = 0	1	1	1	0
Idiomas disponibles						
1	Inglés	Si = 1 / No = 0	1	1	1	1
2	Español	Si = 1 / No = 0	1	0	0	0
3	Otros	Si = 1 / No = 0	1	0	1	1
CAPACIDAD DE INSTALACIÓN						
Facilidad de instalación						
1	Manuales de instalación	Si = 1 / No = 0	0	1	1	0
2	Ayuda en Línea	Si = 1 / No = 0	0	0	0	0
3	Tiempo de instalación	0 : 4	1	1	1	0

CONTINÚA →

CAPACIDAD DE SER REEMPLAZADO						
Migración						
1	Migrar base a un nuevo gestor	Si = 1 / No = 0	0	0	0	0
TOTAL PORTABILIDAD (10)			7	6	6	1

Fuente: Autores del trabajo de titulación

4.3 RESULTADO DEL ANÁLISIS COMPARATIVO

Una vez aplicado el modelo de evaluación a aplicar las métricas propuestas en el paso 6 del método IQMC se sumaron todos los puntos asignados y se obtuvieron los resultados de se detallan en la siguiente tabla y gráfico.

Tabla 28
Resultado del modelo evaluador propuesto

CARACTERISTICA ISO/IEC 25010	PUNTOS	PESO (%)	CASSANDRA		MONGO DB		NEO4J		DYNAMODB	
			PUNTOS	%	PUNTOS	%	PUNTOS	%	PUNTOS	%
Funcionalidad	10	15,87	8	12,70	8	12,70	9	14,29	8	12,70
Eficiencia	5	7,94	4	6,35	5	7,94	4	6,35	5	7,94
Compatibilidad	2	3,17	0	0,00	2	3,17	1	1,59	1	1,59
Usabilidad	19	30,16	13	20,63	14,75	23,41	15	23,81	11	17,46
Fiabilidad	12	19,05	7	11,11	11	17,46	12	19,05	7	11,11
Seguridad	3	4,76	3	4,76	3	4,76	3	4,76	3	4,76
Mantenibilidad	2	3,17	1	1,59	1	1,59	2	3,17	2	3,17
Portabilidad	10	15,87	7	11,11	6	9,52	6	9,52	1	1,59
TOTAL	63	100	43	68,25	50,75	80,56	52	82,54	38	60,32

Fuente: Autores del trabajo de titulación

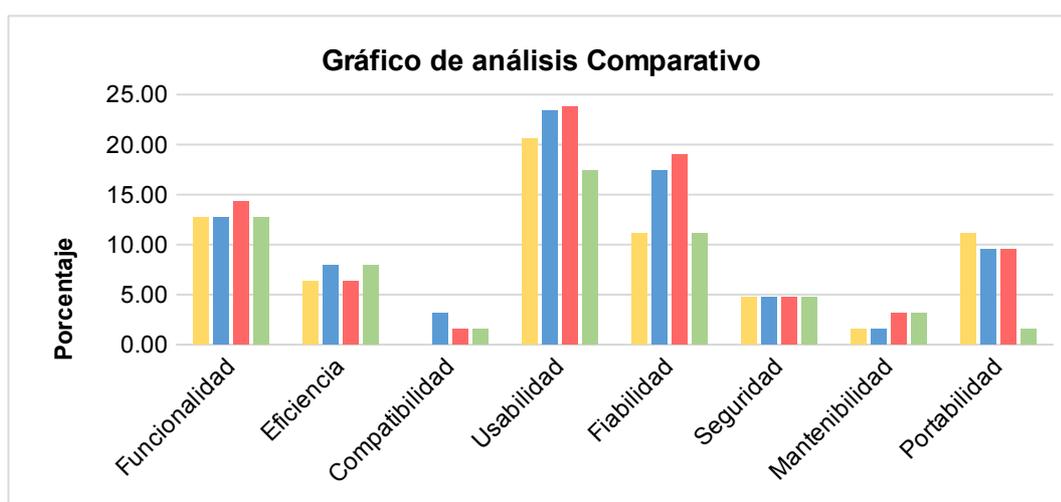


Figura 25 Análisis comparativo entre gestores de base de datos no SQL

Fuente: Autores del trabajo de titulación

Se puede observar que el gestor de base de datos que cumple con la mayoría de los indicadores de modelo de evaluación es Neo4J, con una puntuación 52 sobre 63 con un valor porcentual de 82,54 siendo así el gestor más opcionado para ser utilizado. Teniendo un mayor puntaje en las características de funcionalidad, usabilidad y fiabilidad.

4.4 COMPARATIVA DE RESULTADOS MODELO EVALUADOR PLANTEADO VS. CASO DE ESTUDIO.

En el caso de estudio seleccionado los resultados arrojados fueron que NEO4J es el gestor de base de datos no SQL mejor puntuado y el más apto para ser usado, realizando la comparación con el modelo propuesto en esta investigación coinciden de forma parcial los resultados, para justificar la razón de esta diferencia hay que tomar en cuenta que esta comparativa del caso de estudio fue realizada en el año 2012 hasta la presente fecha son 4 años, otro punto importante es que solamente se tomaron en cuenta 3 características para la comparación, y es claro que en la tecnología las cosas avanzan de una manera acelerada y el cambio es constante, es por esa razón que los resultados son en cierta parte similares.

CAPITULO V CONCLUSIONES Y RECOMENDACIONES

5.1 CONCLUSIONES

- La base de datos NoSQL documental es la más utilizada en la actualidad, por las compañías debido a las múltiples ventajas, como la funcionalidad intuitiva.[CITATION DBE16 \l 3082].
- Para desarrollar el modelo de evaluación se realizó un estudio de los modelos de calidad, y comparación de sus características seleccionando como base la norma ISO 25000.
- En la investigación se evidencia como el modelo de calidad ISO 25000 se adecuó de forma correcta en la elaboración del modelo de evaluación, como medio importante para determinar las características, subcaracterísticas y atributos de calidad.
- Para obtener mejores resultados en la elaboración del modelo de evaluación se empleó un enfoque mixto empleando la norma ISO/IEC 25000 y el método IQMC, arrojando resultados satisfactorios.
- La realización de un modelo de evaluación es una guía para determinar el gestor de base de datos no relacional adecuado a las necesidades de los desarrolladores y empresas
- El modelo de evaluación planteado se puede utilizar como base para aumentar su alcance con características de bases de datos relacionales.

5.2 RECOMENDACIONES

- Se recomienda tener un conocimiento previo sobre características de gestores de base de datos no relacionales, con el fin obtener mayores resultados al momento de aplicar el modelo planteado.
- Experimentar el mismo proceso de desarrollo de modelo de evaluación, pero utilizando nuevos estándares, o combinando estándares de calidad con el propósito de mejorar la propuesta planteada.
- Identificar y mejorar las métricas para los atributos inmersos en el modelo de evaluación.
- Someter el modelo de evaluación propuesto a más casos de estudio con el fin de detectar y corregir falencias.
- Someter el modelo de evaluación propuesto a nuevas iteraciones del método IQMC para obtener nuevas versiones del modelo.
- Plantear nuevos atributos con la información de nuevos tipos de bases de datos no relacionales.

REFERENCIAS BIBLIOGRAFICAS

ACENS. (Febrero de 2014). *quantummode*. Recuperado el 15 de Noviembre de 2015, de Bases de datos NoSQL. Qué son y tipos que nos podemos encontrar: <https://www.acens.com/wp-content/images/2014/02/bbdd-nosql-wp-acens.pdf>

Benitez, C. A. (2013). *Sistemas Web Escalables*. Buenos Aires: Fox Andina S.A.

Brito Zhunio, D. M. (2011). *Estudio del uso de MongoDB como alternativa a las bases de datos relacionales tradicionales en aplicaciones web que requieren rapidez de lectura/escritura de los datos almacenados*. Cuenca – Ecuador.

Brooks, C. (2014). *Enterprise NoSQL for Dummies*. Nueva Jersey: John Wiley & Sons, Inc.

Carvallo, J. P. (2005). *Systematic construction of quality models for Cots-Based Systems*. UNIVERSITAT POLTECNICA DE CAYALUNYA, Barcelona.

Carvallo, J. P., Franch, X., & Quer, C. (2009). *Calidad de componentes software*.

Castro Romero, A., González Sanabria, J. S., & Callejas Cuervo, M. (2012). *Utilidad y funcionamiento de las bases de datos NoSQL*.

DB-ENGINES. (s.f.). Recuperado el 3 de Enero de 2016, de <http://db-engines.com/en/ranking>

DeCandia, G. (Oct. 2007). Dynamo: amazon's highly available key-value store.

- ECURED. (2017). *ISO/IEC 25000*. Recuperado el 16 de Noviembre de 2015, de ISO/IEC 25000: http://www.ecured.cu/index.php/ISO/IEC_25000
- Fay Chang, J. D. (2013). *Bigtable: A Distributed Storage System for Structured Data*.
- Fillottrani, P. R. (2007). *Calidad en el Desarrollo de Software*. Recuperado el 2 de Octubre de 2015, de <http://www.cs.uns.edu.ar/~prf/teaching/SQ07/>
- Forero, M. M. (19 de marzo de 2014). *Slideshare*. Recuperado el 15 de Noviembre de 2015, de <http://es.slideshare.net/Cycle-IT/introduccion-a-no-sql/10>
- GISI-UNLa. (2015). *Revista latinoamericana de ingeniería de software*. Red de Ingeniería de Software de Latinoamérica (RedISLA).
- GNOSS, E. (27 de Marzo de 2010). *Next Web*. Recuperado el 15 de Noviembre de 2015, de <http://red.gnoss.com/comunidad/nextweb/recurso/basicos-de-internet-teorema-de-brewer-o-cap/0ae5ad19-8f63-422d-98d6-34bec3eba38a>
- Gracia del Busto, H. Y. (2012). *Revista Telem@tica*. *Revista Telem@tica Vol. 11*, 21-33.
- Graterol, Y. (2013). *Mongo DB*. Obtenido de Mongo DB en español: <http://libromongodb.com/#description>
- International Organization for Standardization (ISO). (2 de Octubre de 2007). *ISO25000*. Obtenido de www.iso25000.com

International Organization for Standardization (ISO). (2010). *Portal ISO 2502n*. Recuperado el 2 de Octubre de 2015, de <http://iso25000.com/index.php/normas-iso-25000/8-iso-iec-2502n>

International Organization for Standardization (ISO). (1986). *ISO International Standard 8402: Quality Management and Quality Assurance Vocabulary*.

International Organization for Standardization (ISO). (1999). *Software Product Evaluation*. Ginebra.

International Organization for Standardization (ISO). (2012). *International Organization for Standardization*. Recuperado el 2 de Octubre de 2015, de <http://www.iso.org/iso/home/standards.htm>

International Organization for Standardization (ISO). (Septiembre de 2017). *ISO 25000*. Obtenido de ISO 25000: <http://iso25000.com/index.php/normas-iso-25000/iso-25010?limit=3&limitstart=0>

Kontio, J. (1996). A case study in applying a systematic method for COTS selection. *Proceedings of the 18th international conference on Software engineering: IEEE Computer Society*.

López Peña, C. A. (2012). *Metabiblioteca*. Recuperado el 12 de 01 de 2016, de Metabiblioteca: <http://eia-dspace.metabiblioteca.com/bitstream/11190/4111/1/INFO0050.pdf>

Lozano Pérez, M. D., & Ramos Salaver, I. (2000). *Ingeniería del software y bases de datos: tendencias actuales*. Salamanca: Universidad de Castilla-La Mancha.

- Marquez Garcia, M. (2003). *Ensayo del modelo de calidad aplicado a software de minería de datos*. Tesis, INSTITUTO POLITECNICO NACIONAL, México DF. Obtenido de <http://www.saber.cic.ipn.mx/cake/SABERsvn/trunk/Repositorios/webVerArchivo/365/1>
- McCreary, D., & Kelly, A. (2013). *Making Sense of NoSQL a guide for managers and the rest of us*. Shelter Island: Manning Publications Company.
- Mongo DB. (2015). *MongoDB Architecture Guide*. New York.
- Mora, H. (2004). *Criterios de validez y triangulación en la investigación social "cualitativa"*. Obtenido de Academia: http://www.academia.edu/1099259/Validez_y_triangulaci%C3%B3n_en_investigaci%C3%B3n_cualitativa
- Nuñez Moraleda, B. M., & Cervera-Paz, Á. (21 de Octubre de 2000). *El modelo de mccall como aplicación de la calidad a la revision del software de gestion empresarial*. Recuperado el 15 de Enero de 2016, de Universidad de Cádiz: <http://rodin.uca.es/xmlui/handle/10498/16054>
- Orellana Cordero, M. G. (2013). *Evaluación de Frameworks realizados en java para aplicaciones on-line*. Cuenca, Ecuador.
- Pérez, F. J., & Martínez, W. (2010). *Slideshare*. Obtenido de Modelos de calidad del software: <http://www.slideshare.net/rolmary/modelo-decalidaddelsoftware1>

Piedra Rodríguez, J. F., & Cordoba Egas, H. D. (20 de Mayo de 2014). *ISO IEC 25000*. Recuperado el 16 de Noviembre de 2015, de Slideshare: <http://es.slideshare.net/dano1917/estandar-isoiec-25000>

PowerData - Especialistas en Gestión de Datos. (26 de Mayo de 2014). *La Calidad de Datos - Los factores imprescindibles a tener en cuenta en una corporación*. Obtenido de PoweData: http://landings.powerdata.es/factores-en-la-calidad-de-datos-de-una-corporacion?&__hssc=&__hstc&hsCtaTracking=d7997de7-0953-4518-8321-d65c956ff1f7%7C52c8b321-26e0-4218-a938-e48be0c7001e

Pressman, R. S. (2002). *Ingeniería de Software, un enfoque práctico* (5ta ed.).

Salazar Cardenas, J. E. (2014). *Análisis comparativo de dos bases de datos sql y dos bases de datos no sql*. Universidad Tecnologica de Pereira .

Scalone, F. (Junio de 2006). *Estudio comparativo de los modelos y estandares de calidad del software*. Obtenido de Universidad Tecnológica Nacional: <http://laboratorios.fi.uba.ar/lsi/scalone-tesis-maestria-ingenieria-en-calidad.pdf>

Solano, H., & Torres, I. (2013). *Análisis de frameworks para el desarrollo de aplicaciones móviles en la plataforma Android*. Recuperado el 3 de Enero de 2016, de Universidad Azuay: dSPACE.uazuay.edu.ec/bitstream/datos/3139/1/09914.pdf

Solid IT. (2012). *DB-Engines*. Obtenido de DB-Engines: http://db-engines.com/en/ranking_definition

Solid IT. (2016). *DB-ENGINES*. Recuperado el 24 de 12 de 2016, de DB-ENGINES: <http://db-engines.com/en/ranking>

Tiwari, S. (2011). *Professional NoSql*. Indianapolis: John Wiley & Sons, Inc.

Vaish, G. (2013). *Getting Started with NoSQL*. Birmingham: Packt Publishing.

Vivanco Méndez, M. C., & Ganán Culqui, P. V. (Agosto de 2012). *Evaluación Técnica Informática del Sistema Integrado de Operaciones y Negocios - Sión de la empresa pública Correos del Ecuador CDE EP. mediante la aplicación de la norma ISO/IEC 25000*. Recuperado el 15 de Diciembre de 2015, de Repositorio Digital ESPE: <http://repositorio.espe.edu.ec/handle/21000/5574>