*Universidad de las Ciencias Informáticas*

*Facultad 2*

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

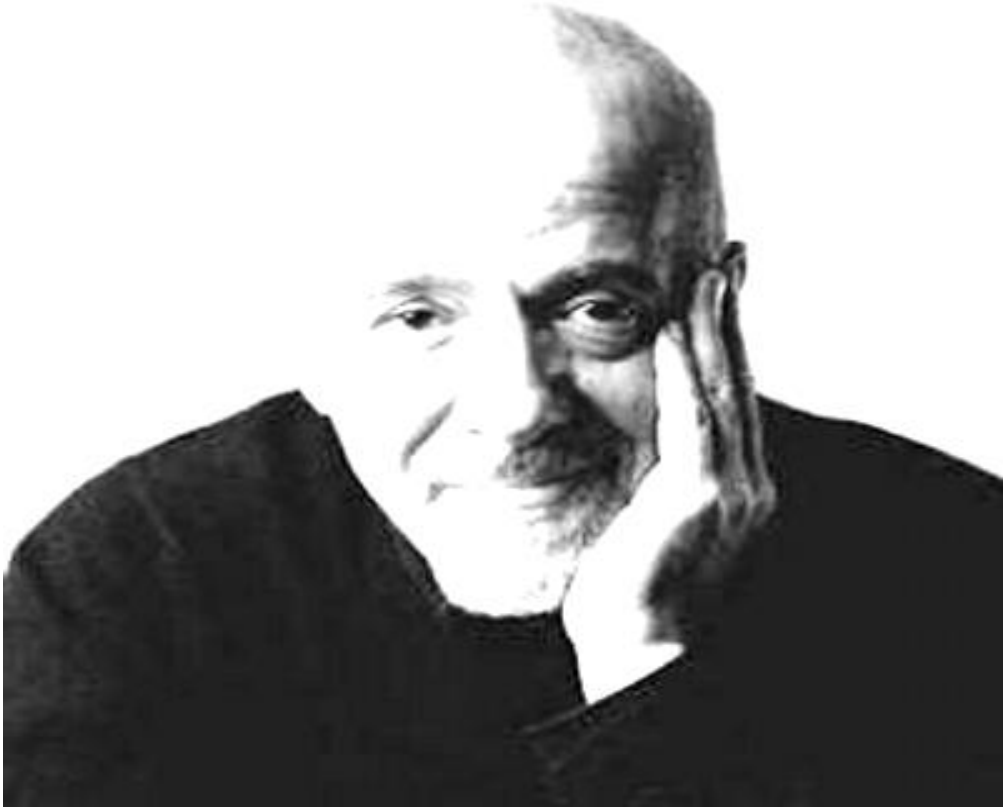

***Título:*** *Aplicación de técnicas de agrupamiento para el proceso de recuperación de información en el sistema RBC*

***Autores:*** *Elizabet Coello Arias*

*Michel Cubertier Sánchez*

***Tutor:*** *Ing. Vladimir Milián Núñez*

La Habana, junio 2017.



*"Recuerda tus sueños y lucha por ellos. Debes saber qué quieres de la vida. Solo hay una cosa que hace tu sueño imposible: el miedo al fracaso."*

## DECLARACIÓN DE AUTORÍA

Declaramos ser los únicos autores de este trabajo y autorizamos a la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

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

| | |
|---|---|
| _____ | _____ |
| Elizabet Coello Arias | Michel Cubertier Sánchez |
| Firma Autor | Firma Autor |

_____

Ing. Vladimir Milián Núñez

Firma Tutor

## Dedicatoria

*Le dedico este trabajo de diploma a Dios primeramente por darme la oportunidad de haber llegado hasta aquí. A la universidad por darme la oportunidad de poder graduarme a pesar de todos los problemas por los que he pasado. A mi abuela Gladys por ser mi segunda madre, aguantar mis malcriadeces todos estos años y regañarme cuando lo necesitaba. A mi mamá por traerme a este mundo y sé que a pesar de no haber crecido a su lado sé que daría la vida por mí. A mi papá por no darse por vencido conmigo y prácticamente obligarme a regresar a la universidad cuando ya no quería estudiar más.*

***Michel***

*Le dedico este trabajo a Dios, la Virgen y todos los santos por darme la fuerza espiritual necesaria para seguir adelante. Y a mis padres por creer y confiar en mí.*

***Elizabet***

## Agradecimientos

*Primeramente, agradezco a mi abuela Gladys porque más que mi abuela para mi es mi segunda madre y siempre se preocupó por mí y trato de llevarme por el buen camino. A mis padres por traerme a este mundo y por lo consejos que me dieron cuando estaba haciendo las cosas mal. A mi hermana Milaydis y a mi cuñado Yusniel por la toda la ayuda que me brindaron en los momentos en que lo necesitaba. En fin, a toda mi familia muchas gracias por estar presentes para mí siempre que lo he necesitado.*

*Agradezco especialmente a estos amigos que han pasado que ya ha dejado de ser amigos para convertirse en mis hermanos: Andy, Abraham, Raidel y Ramón al cual no tengo forma de agradecerle por su amistad y por todo lo que ha hecho por mí y sé que a veces no me comporto de la mejor manera, pero tenlo siempre presente que tú eres y serás siempre mi hermano. También a mis hermanitas: Laura y Betsy.*

*A todas aquellas amistades que he hecho a lo largo de estos años en la universidad; El lucky, El chala, El ruso, Pedro, Manuel, Elio, Oxford, Adrian, Norka, las Camilas, Lily y Lisandra, Yohana, Karel, Eddy, Angela y muchos otros más con los que he compartido en muchos momentos y se preocuparon por mi tesis. A todos los grupos por los que he pasado en la universidad.*

***Michel***

## Agradecimientos

*A Mariela mi mamá por siempre creer en mí y recordarme todo el tiempo que yo sí puedo, por estar pendiente a pesar de la distancia, de cada problema y hacerlo casi suyos, gracias mami, aunque siempre digas que no, pero tú sabes que te amo mucho.*

*A José mi papá, por apoyarme y regañarme cuando no tenía la razón y pensaba que si, al final todo lo que me dijiste fue por mi bien, te amo.*

*A mi madrastra Yamile y mis hermanastras, Aliena y Keylin, por estar pendiente de todos mis problemas y ayudarme en todo lo que podían.*

*A mi padrastro Armando por preocuparse de todo, y estar cerca de mi mamá cuidándola mientras yo me encuentro lejos.*

*A mis abuelas, aunque una no esté físicamente, sé que su espíritu siempre está conmigo.*

*A mis tías, tíos, primos y toda la familia que siempre estuvieron pendiente de mí.*

*A mi amiga y más que amiga hermana Adanaysis, no sé cómo agradecerte todo lo que has hecho y lo que te falta por hacer por mí, gracias voy a estar la vida entera agradecida, y tu familia, Mamita, Adan, Solita, por acogerme y hacerme parte de ella.*

*A mis primeras amigas en la universidad, las Mayi, Mayara y Maryeris, gracias por soportarme y quererme tal cual soy.*

*A Rosi, eres lo único que le agradezco a la programación, gracias a ellas te conocí, gracias por ser mi psicóloga, mi amiga, y ayudarme en todo lo que estaba a tu alcance, aunque al final hiciera lo que me diera la gana con lo que me decías claro porque gracias a ti, conocí a Migue, que fue estuvo implicado en logro de esta tesis.*

*A la que dice ella que es un trapito, Angelina, gracias por correr conmigo, cuidarme, hacerme sopitas que solo yo puedo tomármelas, gracias por pensar que todas mis pesadeces eran un Te Quiero.*

*A Lalo, Yisel por soportarnos mutuamente, si porque tú también tienes que agradecerme a mí.*

*A Yilian la Flaca, por sus tonterías hacerme reír, aunque me dieran ganas de matarla y no podía faltar, por darle el visto bueno al documento.*

*A Leydis por estar ahí cada vez que la necesitaba y ayudarme en todo lo que podía.*

*A Claudia y Yisel Cambra, que, a pesar de la distancia, siempre estuvieron al tanto de todo.*

*A la Dany y Dianelis por compartir los últimos años de esta carrera y ayudarme en todo lo que estuvo en su alcance.*

*A mis niños Manuel, el Moro, Rafa, Joel, Pablo, Alcides, gracias por apoyarme y estar en los buenos y malos momentos.*

*A mi compañero de tesis Cubertir, por que antes de ser mi compañero de tesis ya era mi amigo, gracias cuber por soportarme en todos estos años.*

*A mi tutor que, aunque el mi compañero de tesis casi me proporciona un infarto, y no estuviera localizable, siempre confió en nosotros y nos defendió en lo que estaba a su alcance.*

*Y a todos los que de una forma u otra hicieron posible este logro.*

***Elizabet.***

## Resumen

En la presente investigación se propuso como objetivo implementar un componente para el Sistema de Recomendación de Bibliografía Científica para mejorar el proceso de recuperación de información. El componente propuesto hace uso de técnicas de agrupamiento, las cuales permiten obtener documentos similares mediante la introducción de palabras clave ingresadas por el usuario y a partir de este resultado, obtener los documentos más relevantes y grupos similares para así mejorar la calidad de la información que se le muestra al usuario.

Para el desarrollo del componente se utilizó la minería de texto, donde se seleccionó el algoritmo de agrupamiento CStar. El desarrollo de la investigación estuvo regido por dos metodologías, CRISP-DM para guiar el proceso de minería de texto y AUP –UCI para el desarrollo del componente de búsqueda. El componente se desarrolló usando el marco de desarrollo Django, mediante el lenguaje de programación Python y PostgreSQL como sistema gestor de base de datos.

Para la verificación del correcto funcionamiento de la implementación se utilizaron las pruebas unitarias mediante el método de caja blanca, pruebas de despliegue y pruebas de validación utilizando el método de caja negra, partición equivalente y gráfico de prueba.



**Palabras clave:** bibliografía científica, minería de texto, recuperación de información, volumen de información.

# Índice

# INTRODUCCIÓN

A partir de la expansión y consolidación de Internet, como medio principal de comunicación electrónica de datos, se ha puesto a disposición de casi toda la humanidad un importante volumen de información. La creación y diseminación de información en el World Wide Web, intranets corporativas, y otros medios de comunicación es soportada por un número creciente de herramientas, sin embargo, mientras la cantidad de información disponible está continuamente creciendo, la habilidad de procesarla y asimilarla no presenta el mismo ritmo de crecimiento.

Esta gran aglomeración documental, hace que la gestión de información científica sea cada vez más compleja, al ser las colecciones textuales heterogéneas, crecientes y dinámicas. Vencer estos desafíos, es esencial para proporcionar a los científicos mejores condiciones de trabajo que aseguren una mayor productividad e inviertan un tiempo menor en procesar la información requerida. Es por ello que, para aprovechar todo el potencial de la información disponible, es necesario poseer accesos que permitan que la tarea de recuperación sea eficiente y efectiva. De ahí el constante desarrollo y seguimiento del área de Recuperación de Información en la ciencia de la computación.

La Recuperación de Información (RI), es un área de la ciencia que se encarga del estudio y desarrollo de técnicas y sistemas, que permiten buscar y ofrecer información relacionada con ciertas palabras clave escritas por los usuarios en un buscador mediante una consulta. Para llevar a cabo la RI existe un conjunto de técnicas, entre las que se encuentra el agrupamiento.

Por su parte, el agrupamiento permite organizar la información obtenida y descubrir nuevo conocimiento a partir del resultado de un proceso de recuperación de información. El agrupamiento es una tarea del aprendizaje no supervisado que tiene como objetivo descomponer el conjunto de datos, de forma tal que los objetos que pertenecen al mismo grupo, sean tan similares como sea posible y los objetos que pertenecen a grupos diferentes sean tan disimilares como sea posible. El análisis de grupos es una herramienta para descubrir una estructura previamente oculta en los datos, asumiendo que existe un agrupamiento natural

o cierto entre ellos. Dicho agrupamiento permite que el resultado de la RI sea más ajustado a la necesidad de información y más rápido al reducirse el dominio de búsqueda.

La Universidad de Ciencias Informáticas (UCI) desarrolló en aras de facilitar a los investigadores de la comunidad universitaria el acceso a la bibliografía especializada en el área de ciencias básicas y computación, el Sistema de Representación de Bibliografía Científica (RBC). Esta herramienta le brinda al usuario la posibilidad de introducir un criterio de búsqueda por palabras claves y obtener publicaciones científicas en esta área mencionada anteriormente.

Actualmente el sistema RBC realiza la RI mediante consultas por aparición o no en los documentos de las palabras claves escritas por los usuarios. Esto provoca que en casos donde la cantidad de información sea muy elevada, documentos que son irrelevantes sean mostrados como resultado de la búsqueda, lo que conlleva a que se pierda tiempo para seleccionar los documentos adecuados y no todas las búsquedas cumplan con las expectativas del usuario. Además, los resultados de estas búsquedas están ordenados por relevancia de las palabras clave en el documento y no se muestran en un orden de semejanza entre los documentos. Es por ello que al no hacer uso de técnicas de agrupamiento de la información teniendo en cuenta la distribución por diferentes temáticas, puede dar paso a que no exista una buena precisión y velocidad en la respuesta.

Un ejemplo de lo expuesto anteriormente, es que, si se introduce en el panel de búsqueda la palabra python, se obtiene información sobre el lenguaje de programación Python, la serpiente Python y el grupo humorístico inglés Monty Python de manera simultánea.

Teniendo en cuenta la situación planteada anteriormente se define como **problema a resolver** ¿Cómo utilizar el agrupamiento de documentos para mejorar el proceso de recuperación de información en el sistema RBC?, definiendo como **objeto de estudio** los algoritmos de agrupamiento, enmarcado en el **campo de acción** los algoritmos de agrupamiento aplicados en la recuperación de información.

Para darle respuesta al problema planteado, se definió como **objetivo general:** Implementar un componente para el sistema RBC en el que se apliquen técnicas de agrupamiento durante el proceso de recuperación de información de dicho sistema.

Para darle cumplimiento al objetivo general se definen las siguientes **tareas de la investigación**:

- Revisión de bibliografías con temas vinculados con el objeto de estudio para adquirir los conocimientos y habilidades necesarias para solucionar el problema planteado.
- Selección de las herramientas, tecnologías y metodologías para el desarrollo del sistema.
- Estudio de las técnicas de agrupamiento en el proceso de recuperación de información y del lenguaje de programación para la implementación del componente.
- Diseño del componente a desarrollar, para lograr menor cantidad de errores y mayor calidad del producto.
- Codificación el componente para lograr un producto terminado.
- Realización de pruebas al sistema para garantizar la calidad.

Para el desarrollo de la investigación se emplearon varios métodos científicos de investigación, los cuales fueron:

**Métodos teóricos:**

- **Histórico-Lógico:** Se evidencia en el estudio realizado a bibliografías de trabajos investigativos, artículos científicos, revistas publicadas sobre el campo de recuperación de la información. De los mismos se obtuvieron los aspectos más significativos, de forma tal que facilitara su entendimiento.
- **Analítico-Sintético:** Se evidencia en el análisis de la calidad de software y la búsqueda de información sobre los tipos de prueba de software existentes, así como las herramientas y métodos que se utilizan para la aplicación de las mismas.
- **Modelación:** Utilizado para modelar la arquitectura, crear los artefactos, diagramas y modelos a utilizar en el desarrollo de las nuevas funcionalidades del sistema RBC.

**Métodos empíricos:**

- **Entrevista:** Se emplea para conocer las necesidades del cliente y definir los requisitos y características de la solución propuesta.

Para una mejor comprensión el documento consta de 3 capítulos:

**Capítulo 1:** “Fundamentación Teórica”, en este capítulo se estudian los principales conceptos que se manejan a lo largo del trabajo investigativo. Además de un estudio a los diferentes métodos de agrupamiento. Se definió la metodología de desarrollo de software, las herramientas y el lenguaje de programación a utilizar para darle solución al problema planteado.

**Capítulo 2:** “Características del sistema”, en este capítulo se describen las características de la propuesta de solución y se explica el proceso de la minería de texto para el agrupamiento de documentos similares. Se definen los requisitos funcionales y no funcionales del sistema. Además, se establecen los artefactos requeridos en la planificación definidos por la metodología AUP-UCI.

**Capítulo 3:** “Arquitectura y diseño”, en este capítulo se propone el diseño del componente desarrollado para el sistema RBC, a partir de los diagramas de clases del diseño empleando estereotipos web, los diagramas de colaboración, el diagrama de paquetes y el diseño de la base de datos. Se define el estilo arquitectónico y los patrones de diseño utilizados en el desarrollo de las nuevas funcionalidades.

**Capítulo 4**: “Implementación y prueba”, en este capítulo se describen los procesos de implementación de las nuevas funcionalidades del sistema RBC y los principales resultados obtenidos en la etapa de pruebas, para garantizar su correcto funcionamiento y el cumplimiento con los requisitos definidos por el cliente.

# CAPITULO 1: Fundamentación Teórica

## 1.1 Introducción

En este capítulo se abordan todos los conceptos y elementos teóricos relacionados con la investigación. Se definen diferentes conceptos como: sistemas de recuperación, minería de texto, procesamiento del lenguaje natural, procesamiento de los datos y el agrupamiento de documentos. Por último, se describen las herramientas, tecnologías, metodologías y el lenguaje de programación a emplear en la implementación.

## 1.2 Recuperación de Información

La Recuperación de la Información (RI) es un término que no es nuevo y en la actualidad juega un papel muy importante debido al valor que la misma posee. La información es una necesidad de las personas a diario y es necesario brindar la misma de una manera rápida y precisa, siendo este uno de los principales objetivos de esta rama (F. R. A, 2007).

Según Baeza-Yates (Baeza-Yates y Ribeiro-Neto, 2009), la Recuperación de Información trata con la representación, el almacenamiento, la organización y el acceso a ítems de información.

Croft (Croft, 2011) estima que la recuperación de la información es el conjunto de tareas mediante las cuales el usuario localiza y accede a los recursos de información que son pertinentes para la resolución del problema planteado. En estas tareas desempeñan un papel fundamental los lenguajes documentales, las técnicas de resumen, la descripción del objeto documental, entre otros. Por otro lado, Korfhage (SEDEÑO, 2010) definió la RI como la localización y presentación a un usuario de información relevante a una necesidad de información expresada como una pregunta (RODRÍGUEZ, 2012).

En resumen, la Recuperación de la Información es una rama que tiene que ver con todo lo relacionado con el almacenamiento de diferentes tipos de información, la cual es clasificada y organizada con el objetivo de facilitar a las personas el acceso a esta (BORDIGNON, 2010).

### 1.2.1 Sistemas de Recuperación de la Información

Los Sistemas de Recuperación de Información son una clase de sistemas de información que tratan con bases de datos compuestas por documentos y procesan las consultas de los usuarios permitiendo el acceso a la información relevante en un intervalo de tiempo apropiado (Arco, 2006).

Estos sistemas para la obtención de los diferentes tipos de información se apoyan en la implementación de varias técnicas y algoritmos de búsqueda, dentro de los cuales se encuentran los algoritmos de agrupamiento de documentos, que se han ido perfeccionando debido al aumento de la información y a las necesidades de los usuarios.

### 1.2.2 Teoría de agrupamiento

En el proceso de recuperación de información uno de los principales objetivos es la obtención de la documentación con precisión y eficiencia, por lo que dicha rama investiga y aplica la clasificación de los documentos. La teoría de agrupamiento en resumen plantea que, en las búsquedas de información, aquellos documentos asociados entre ellos, existen grandes posibilidades de que sean relevantes a la misma consulta realizada.

La categorización de documentos dentro de la recuperación está destinada principalmente a mejorar la calidad de los sistemas que la implementan.

Sus principales aplicaciones son:

- Mejorar el rendimiento de los motores de búsqueda de información mediante la categorización previa de todos los documentos disponibles.
- Facilitar la revisión de los resultados por parte del usuario final, agrupando los resultados luego de realizar la búsqueda (Castellano, 2009).

De forma general se mejora la recuperación, si se recuperan no solo los documentos que sean relevantes, sino también los documentos similares a ellos (los que pertenecen al mismo grupo).

## 1.3 CONCEPTOS ASOCIADOS AL OBJETO DE ESTUDIO

### 1.3.1 Minería de Texto

Existen muchas definiciones acerca de la minería de texto. Según IBM, la minería de textos es el proceso de analizar colecciones de materiales de texto con el objetivo de capturar los temas, conceptos clave y descubrir las relaciones ocultas y las tendencias existentes sin necesidad de conocer las palabras o los términos exactos que los autores han utilizado para expresar dichos conceptos.

La minería de textos y la acción de recuperar son conceptos que a veces se confunden, aunque son bastante diferentes. Una recuperación precisa de la información y su almacenamiento supone un reto importante, pero la extracción y administración de contenido de

calidad, de terminología y de las relaciones contenidas en la información son procesos cruciales y determinantes.

El proceso de minería de texto consta de varias etapas, estas son:

- **Pre-procesamiento:** en la primera etapa los textos que se pueden manipular se transforman en una serie de representaciones estructurales de tal manera que se promueva la facilidad de un análisis posterior.
- **Minería de texto o etapa de descubrimiento:** En esta etapa se procede a realizar un análisis de las representaciones intermedias, esta tarea se realiza con el objetivo de así poder descubrir y encontrar patrones interesantes dentro de los textos de interés, así como también se busca obtener nuevos conocimientos.
- **Visualización de resultados:** esta fase es de exploración de los datos guiado para el usuario que sea lo más amigable posible. Las últimas tendencias presentan los resultados mediante graficas o páginas web. Una vez obtenidos los conceptos, los términos o las tendencias, se pueden utilizar métodos automáticos de visualización o bien pueden interpretarse los resultados directamente.

Existen dos vías para aplicar la minería de texto:

- Aprendizaje basado en el lenguaje natural.
- Aprendizaje basado en ejemplos anteriores.

El presente trabajo de investigación se centrará en el problema de mejorar la recuperación de información en el sistema RBC. Como vía para darle solución al problema que antes se menciona, se utilizará el aprendizaje basado en el lenguaje natural, siendo esta una de las técnicas más utilizadas en la transformación de documentos.

### 1.3.2 Procesamiento del lenguaje natural

El lenguaje natural (LN) es el medio utilizado de manera cotidiana por las personas para establecer una comunicación con las demás personas. El LN ha venido perfeccionándose a partir de la experiencia a tal punto que puede ser utilizado para analizar situaciones altamente complejas y razonar muy sutilmente. Los lenguajes naturales tienen un gran poder expresivo y su función y valor como una herramienta para razonamiento (Vásquez Cortez, 2009).

El procesamiento del lenguaje natural consiste en la utilización de un lenguaje natural para comunicarnos con la computadora, debiendo esta entender las oraciones que le sean proporcionadas, el uso de estos lenguajes naturales, facilita el desarrollo de programas que realicen tareas relacionadas con el lenguaje o bien, desarrollar modelos que ayuden a comprender los mecanismos humanos relacionados con el lenguaje (Vásquez Cortez, 2009).

(Refugio Zavala, 2014) define el procesamiento del lenguaje natural es una disciplina que relaciona directamente la computación y la lingüística, la cual tiene como principal objetivo conseguir que el lenguaje humano pueda utilizarse como entrada en un proceso automatizado.

Para extraer información relevante de textos no estructurados, es decir, textos escritos en lenguaje natural, el presente trabajo se centrará en el algoritmo de procesamiento de lenguaje natural Parts Of Speach (POS, siglas en inglés). (Santamaria, 2014), expresa que: mediante este algoritmo se analiza el texto de forma de que una vez comprendido el texto se puedan deducir los conceptos sobre los que trata.

Para un mejor entendimiento se puede resumir que el algoritmo POS permite obtener los conceptos más relevantes de los que trata un documento. A continuación, se explica el pre-procesamiento que deben sufrir los datos obtenidos para que sean de utilidad en tiempo de ejecución.

**Pre-procesamiento de los datos**

Para que los datos puedan ser utilizados en tiempo de ejecución, es necesario que los mismos sufran un pre-procesamiento. El objetivo fundamental del pre-procesamiento de datos es manipular y transformar los datos en bruto de modo que el contenido de la información envuelto en el conjunto de datos puede ser expuesto o accesibles con mayor facilidad (Dorian, 1999).

En el pre-procesamiento de los datos se realiza la selección, la limpieza, el enriquecimiento, la reducción y la transformación de las bases de datos (Sinnexus, 2007). Existen dos pasos para realizar este proceso, estos son la estandarización de textos y la extracción de características de los textos.

**Estandarización de textos**

En el método de estandarización se realizan las siguientes actividades:

1. Eliminación de caracteres extraños y signos de puntuación
2. Eliminación de mayúsculas

3. Bag of words del texto: separa el texto dividiéndolo por palabras y formando una lista con las mismas.
4. Eliminación de stopwords
5. Stemming del texto

Una stopword es una palabra carente de significado para el idioma en que se realiza el pre-procesamiento. Los casos más comunes lo comprenden los artículos, conjunciones, preposiciones, pronombres, entre otros.

Stemming es un método para reducir una palabra a su raíz o (en inglés) a un stem. Hay algunos algoritmos de stemming que ayudan en sistemas de recuperación de información. Stemming aumenta el recall que es una medida sobre el número de documentos que se pueden encontrar con una consulta. Por ejemplo una consulta sobre "bibliotecas" también encuentra documentos en los que solo aparezca "bibliotecario" porque el stem de las dos palabras es el mismo ("bibliotec").

Por lo tanto, se puede concluir que, en la etapa de estandarización del texto, se obtiene como resultado una lista de palabras con repeticiones, es decir, una misma palabra puede estar repetida varias veces.

**Extracción de características de los textos**

Muchas veces después de la estandarización de los textos se obtiene una gran cantidad de datos. El objetivo de la etapa de extracción de características es reducir el tamaño de los datos obtenidos en la etapa anterior, formando una serie de modelos que permitan deducir conceptos de similitud o clasificación de textos. La determinación de estos modelos se logra utilizando algoritmos de minería de texto

A continuación, se realiza un estudio de las técnicas de minería de texto y los algoritmos utilizados en cada una de ellas.

### 1.3.3 Técnicas de Minería de texto

Se pueden clasificar las técnicas de minería de texto en descriptivas y predictivas. Las descriptivas caracterizan las propiedades generales de los datos que se encuentran en las formas intermedias y, por el contrario, las predictivas realizas inferencias en los datos para poder realizar predicciones.

Hay que tener en cuenta lo que se desea obtener para determinar cuál de los dos enfoques se utiliza. En el presente trabajo se van a utilizar las de tipo descriptivas (no supervisadas) en las cuales se encuentran análisis de varianza, que mediante el mismo se evalúa la existencia de diferencias significativas entre las medias de una o más variables en poblaciones distintas.

El análisis discriminante permite la clasificación de individuos en grupos que previamente se han establecido, permite encontrar la regla de clasificación de los elementos de estos grupos y por tanto una mejor identificación con las variables que definan la pertenencia al grupo.

Por último, el agrupamiento, que es un procedimiento de agrupación de una serie de vectores según criterios habitualmente de distancia, tratará de disponer los vectores de entrada de forma que estén más cercanos aquellos que tengan características comunes de tal manera que se maximice la similitud entre los vectores de un mismo grupo y se minimice la similitud entre los grupos, además, esta técnica puede ser combinada fácilmente con cualquier otra. El objetivo fundamental del agrupamiento es determinar el comportamiento de un nuevo vector, a partir de las características del mismo, se define a qué grupo pertenecerá y que acción podrá realizar.

**Selección de la técnica de Minería de texto**

Clustering o agrupamiento es la técnica seleccionada en la investigación, ya que se ajusta a la solución del objetivo general debido a la necesidad de agrupar los documentos similares al que sube el usuario al sistema.

Según (Godoy, 2015) el clustering de documentos es el proceso de buscar un agrupamiento natural en un conjunto de datos basándose en su similitud.

Dentro de las familias de algoritmos de clustering más utilizados se encuentran:

- **Basado en particiones**: crean particiones sucesivas del conjunto de datos.
- **Jerárquicos**: descomposición jerárquica del conjunto de objetos.
- **Basado en grafos**: los cuales se identifican por la creación de sub-grafos en forma de estrella.

El tipo de algoritmo de agrupamiento o clustering que se va a utilizar en el presente trabajo es el basado en grafos. Esta selección se basa en que un documento puede tener pertenencia a más de un grupo (por ejemplo “El padrino”, que puede ser ficción, policíaco, thriller, mafia) y se desea que este documento este en el grupo más característico de la búsqueda, para así evitar

el solapamiento y un máximo cubrimiento del total de documentos devueltos por la búsqueda inicial.

Dentro de los algoritmos basados en grafo, se encuentran las técnicas de agrupamiento Star, dentro de las cuales se encuentra el algoritmo seleccionado para la recuperación de información en el sistema RBC.

## 1.4 Técnicas de agrupamiento Star

Las técnicas de agrupamiento Star están compuestas por un conjunto de algoritmos basados en grafos, los cuales se identifican por la creación de sub-grafos en forma de estrella, de aquí su nombre.

A continuación, se realiza una descripción de algunos algoritmos que componen a dichas técnicas, especificando sus principales características, definiciones, bondades y deficiencias que presentan cada uno de ellos. Para los mismos se necesita tener conocimientos de los siguientes conceptos:

### 1.4.1 Función de semejanza

Se denomina función de semejanza **w** a una función que asocia a cada par de objetos de un universo de objetos **U= {$O_1$, …, $O_n$}** una magnitud que evalúa su semejanza o parecido (Arco y Bello, 2009).

### 1.4.2 Grafo de semejanza

Se llama grafo de semejanzas **G= (V, E, w)**, al grafo completo donde los vértices **V** son los objetos a agrupar y las aristas se etiquetan con las semejanzas entre los objetos **E**, calculada por una función de semejanza **w**.

### 1.4.3 Objeto β-semejantes

Dos objetos cuya semejanza es mayor que un cierto umbral **β** (definido por el usuario) se denominan **β-semejantes**. Si un objeto no es **β-semejante** con ningún otro objeto se denomina **β-aislado** (Arco y Bello, 2009).

### 1.4.4 Grafo de β-semejanza

Un grafo de **β-semejanza** se denota $\mathbf{G_\beta = (V, E_\beta)}$, el cual es un sub-grafo del grafo de semejanzas, donde se eliminan las aristas con peso menor que **β**, donde solamente quedan conectados los objetos semejantes (Arco y Bello, 2009), (Aslam y Pelekhov, 2006).

Entre los algoritmos estudiados para llegar a seleccionar el de la solución se encuentran:

- Star
- CStar
- CStar+

### 1.4.5 Algoritmo Star

El algoritmo Star, propuesto por Aslam (Aslam, 1998), (Aslam, 2004), se basa en la construcción de un grafo de semejanza $\mathbf{G_\beta}$ cuyos vértices representan a los documentos. De este grafo se obtienen los documentos estrellas o centros de clústeres que son los vértices del grafo que tengan mayor cantidad de aristas y el resto de los vértices del grafo son considerados satélites de estas estrellas.

Este algoritmo presenta dos deficiencias significativas, siendo la primera de estas que el resultado de la agrupación está en dependencia del orden en que se realice el análisis de los vértices del grafo.

La segunda deficiencia es que independientemente del orden en que se realice el análisis de los vértices, se obtienen grupos “ilógicos”. Un grupo $\mathbf{g_i}$ se considera ilógico si cumple las siguientes condiciones (Pérez Suárez, 2008):

- Existe un elemento *e* que pertenece a $\boldsymbol{g_i}$ que es más denso que el vértice centro ***c*** que define a $\boldsymbol{g_i}$.
- El elemento ***e*** puede agrupar, si se considera como centro, a los vértices que son agrupados solo por el centro ***c***.
- Estas condiciones vienen dadas debido a que el algoritmo Star no permite que dos vértices adyacentes sean centros.

### 1.4.6 Algoritmo CStar

El algoritmo CStar introduce una nueva definición de sub-grafo, el cual es nombrado sub-grafo en forma de estrella condensada y aplica una heurística, a partir de la cual obtiene un cubrimiento de $\mathbf{G_\beta}$ utilizando sub-grafo de este tipo (Pérez Suárez, 2008). Con este algoritmo se obtienen grupos que pueden tener traslape[1], manteniendo los puntos fuertes de sus predecesores y trabajando sobre las deficiencias anteriores y otras como:

- No obtiene grupos redundantes.
- No consume tanta memoria como los demás algoritmos lo que hace que no sea ineficiente.

La idea principal del algoritmo CStar es determinar un criterio que establezca cuándo un sub-grafo del tipo estrella condensada (**EC**) es más denso que otro y partiendo de este, realizar un cubrimiento del grafo de **β-semejanza** utilizando los sub-grafos **EC** más densos y posteriormente aplicar un proceso de filtrado que reduzca la cantidad de estos.

Un problema que presenta este algoritmo es que puede obtener diferentes agrupamientos cuando se ejecutan sobre una misma colección, debido a que existe una dependencia del orden de análisis de los documentos entre otras características de este algoritmo (Pérez Suárez, 2008).

### 1.4.7 Algoritmo CStar+

El Algoritmo CStar+ se describe como una variante de su antecesor CStar. Este algoritmo utiliza sub-grafos **EC** para realizar un cubrimiento sobre las componentes conexas del grafo de **β-semejanza**. Transformando el problema de determinar un agrupamiento de $\mathbf{G_\beta}$ usando sub-grafos **EC** en el problema de realizar un cubrimiento utilizando sub-grafos **EC** de cada componente conexa.

Es importante tener en cuenta que, aunque obtener un cubrimiento de estas componentes a través de sub-grafos **EC** reduce el encadenamiento, también podría afectar la calidad del

[1] Cubrir parcialmente una cosa a otra.

agrupamiento si dicha componente tiene un alto grado de conexión entre sus vértices, pues se estaría dividiendo en sub-grupos un grupo que ya es altamente cohesionado.

Este algoritmo también presenta el problema de su antecesor de que, se pueden obtener diferentes agrupamientos si se aplican en una misma colección de documentos (Pérez Suárez, 2008).

Después del estudio a los diferentes algoritmos antes mencionado se definió para la implementación del componente propuesto la utilización del algoritmo CStar.

Se seleccionó el algoritmo CStar debido a las siguientes razones:

- Primeramente, tiene como objetivo dar solución a las deficiencias de su antecesor Star, eliminando el problema de que, en dependencia del orden en que se analizaran los vértices del grafo, era la calidad de los resultados obtenidos. Además, elimina la posibilidad de obtener grupos ilógicos y grupos redundantes.
- A diferencia de otros, este algoritmo mantiene un consumo de memoria adecuado, debido a que es relativamente fácil de implementar, no tiene una gran cantidad de funciones asociadas que se tengan que construir para el funcionamiento del mismo.
- Además, hace un proceso de filtrado que reduce la cantidad de grupos densos, lo que influye considerablemente en la calidad de los resultados.

## 1.5 Metodologías de desarrollo

Una metodología de desarrollo se define como un conjunto de actividades organizadas que tienen por objetivo la realización de un trabajo (Moine, 2013). Para cada actividad se define, además de las entradas y salidas, la forma en la que debe llevarse a cabo. En el presente trabajo se hace necesario el empleo de una metodología para el trabajo con los datos y otra para el desarrollo de las nuevas funcionalidades.

**Metodología para guiar el proceso de minería de texto**

Existen varias metodologías para la realización de proyectos de minería de datos entre las cuales se encuentra SEMMA (Sample, Explore, Modify, Model, Assess), KM-IRIS y CRISP-DM (Cross-Industry Standard Process For Data Mining) (Salgueiro, 2012)**,** la cual se seleccionó como la metodología a utilizar en el presente trabajo debido a que está más cercana al concepto real de proyecto, ha sido diseñada como una metodología neutra respecto a la

herramienta que se utiliza para el desarrollo del proyecto de Data Mining y es de distribución libre y gratuita (Salgueiro, 2012)**.**

La metodología CRISP-DM propone seis fases para guiar el desarrollo del proceso de minería de texto. A continuación, se muestra una imagen donde se evidencia cada una de estas fases y el proceso jerárquico que existen entre ellas.

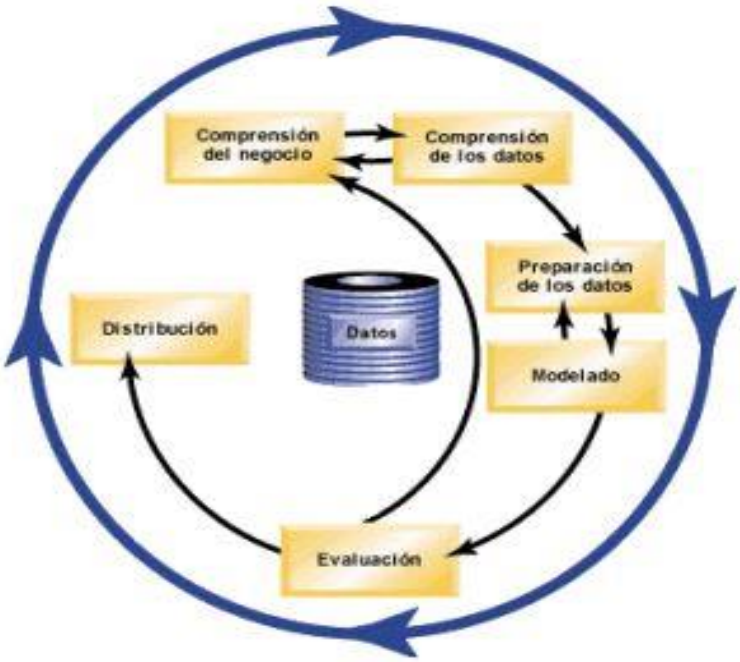



***Ilustración1: Fases de CRISP-DM***

- **Comprensión del negocio**: En esta fase se determinan los objetivos y requerimientos del proyecto desde una perspectiva del negocio, definiendo el problema de minería y el plan de trabajo.
- **Comprensión de los datos:** Fase que consiste en la recolección de datos que se utilizarán en el proyecto y la familiarización con los mismos.
- **Preparación de los datos:** Comprende aquellas actividades de tratamiento de los datos para construir la vista minable o conjuntos de datos finales sobre el cuál se aplicarán las técnicas de minería.
- **Modelado:** En esta fase se aplican las diversas técnicas y algoritmos sobre minería sobre el conjunto de datos para obtener la información oculta y los patrones implícitos en ellos.

- **Evaluación:** Fase en la que se analizan los patrones obtenidos en función de los objetivos organizacionales. En esta etapa se debería determinar si se ha omitido algún objetivo importante del negocio y si el nuevo conocimiento será implementado, es decir, si se pasará a la próxima etapa.
- **Implementación:** consiste en la comunicación e implementación del nuevo conocimiento, el cual debe ser representado de forma entendible para el usuario (Moine, 2013).

CRISP-DM es flexible y se puede personalizar fácilmente, es decir, permite crear un modelo de minería de texto que se adapte a necesidades concretas. Además, profundiza sobre las tareas y actividades a ejecutar en cada etapa del proceso de minería de texto.

**Metodología para guiar el desarrollo del componente de búsqueda**

Para el desarrollo del componente, se decide utilizar la variación que propone la UCI de la metodología Proceso Unificado Ágil (AUP, siglas en inglés), ya que a través de los artefactos que propone, permite lograr una clara compresión del negocio y realizar una descripción más detallada de los requerimientos del componente a desarrollar.

Para seleccionar esta metodología se tuvo en cuenta que, además:

- El equipo de desarrollo es pequeño (2 integrantes).
- Se cuenta con poco tiempo para el desarrollo de la herramienta.

“AUP es una metodología de desarrollo ágil que heredera de otros paradigmas como la programación extrema (XP) y Rational Unified Process (RUP). Consta de principios y prácticas influyentes en la construcción del software en armonía con la documentación esencial de entregables específicos para el entendimiento de la solución. Entre sus objetivos destaca la reducción del costo del cambio en el proyecto sobre la base de procedimientos iterativos (característica propia de RUP), donde la codificación y pruebas del software se llevan a cabo paralelamente (según XP)” (Galindo, 2012).

Al igual que en RUP, en AUP se establecen cuatro fases que transcurren de manera consecutiva (Ervin y Cordero Flores, 2010):

- **Inicio:** permite obtener una comprensión común sobre el alcance del nuevo sistema, entre el cliente y el equipo de desarrollo, definiendo una o varias arquitecturas candidatas para el mismo.

- **Elaboración:** el objetivo es que al equipo de desarrollo se le facilite la comprensión de los requisitos del sistema y pueda validar la arquitectura definida.
- **Construcción:** es la fase en la que el sistema es desarrollado y probado por completo en el entorno de desarrollo.
- **Transición:** el sistema es llevado a entornos de preproducción donde se somete a determinadas pruebas, tanto de validación como de aceptación, siendo desplegado finalmente en los sistemas de producción.

De las 4 fases que propone la metodología AUP (Inicio, Elaboración, Construcción, Transición), la variación AUP-UCI mantiene la fase de Inicio, pero modificando su objetivo. Une las tres restantes fases de AUP en una sola, llamada Ejecución, y se agrega la fase Cierre. Para un mejor entendimiento se muestra la siguiente tabla.

Tabla 1: Fases Variación AUP-UCI

| **Fases AUP** | **Fases Variación AUP-UCI** | **Objetivos de las fases Variación AUP-UCI** |
|---|---|---|
| Inicio | Inicio | En esta fase se realiza un estudio inicial de la organización cliente, que permite obtener información fundamental acerca del alcance del proyecto, realizar estimaciones de tiempo, esfuerzo y costo y decidir si se ejecuta o no el proyecto. |
| Elaboración | Ejecución | En esta fase se ejecutan las actividades requeridas para desarrollar el software, |
| Construcción | | |

| | | |
|---|---|---|
| Transición | | incluyendo el ajuste de los planes del proyecto considerando los requisitos y l arquitectura. Durante el desarrollo se modela el negocio, se obtiene los requisitos, se elabora la arquitectura y el diseño, se implementa y se libera el producto |
| | Cierre | En esta fase se analizan tanto los resultados del proyecto como su ejecución y se realizan las actividades formales de cierre del proyecto. |

Una de las ventajas de AUP-UCI radica en la forma de planificar el proyecto y la estimación de tiempo, factor importante ya que cuenta con poco tiempo para el desarrollo de las nuevas funcionalidades.

La metodología AUP-UCI propone 4 escenarios para la disciplina Requisitos.

- **Escenario1:** proyectos que modelan el negocio con Caso de Uso del Negocio (CUN) solo pueden modelar el sistema con Caso de Uso del Sistema (CUS).

CUN + Modelo Conceptual (MC) = CUS

- **Escenario2:** proyectos que modelen el negocio con MC solo pueden modelar el sistema con CUS.

MC=CUS

- **Escenario3:** proyectos que no modelen el negocio con Diagrama de Procesos del Negocio(DRP).

DPN+MC=DRP

- **Escenario4:** proyectos que no modelen el negocio solo pueden modelar el sistema con Historias de Usuario(HU).

El escenario sobre el cual se trabajará es el Escenario 2, ya que luego de evaluar el negocio se llegó a la conclusión de no es necesario incluir las responsabilidades de las personas que ejecutan las actividades. Además, el objetivo general del presente trabajo corresponde con lo que requiere los resultados obtenidos en la búsqueda de bibliografía científica.

## 1.6 Tecnologías y herramientas de desarrollo

Como se mencionó anteriormente, el componente de búsqueda propuesto por la presente investigación se integrará con el sistema RBC, por lo que se consideró utilizar las mismas herramientas con las que se implementó el sistema RBC. A continuación, se describen las mismas.

**Lenguaje de Programación**

Para la solución propuesta se utiliza el lenguaje de programación Python en su versión 2.7.11 permitiendo el desarrollo de aplicaciones web, rápidas y fáciles. Además de la adaptabilidad con el sistema operativo Windows 8.1 en el cual se está desarrollando la herramienta, y la integración de numerosas bibliotecas. Además de que fue el utilizado para el desarrollo del Sistema RBC.

**Lenguaje de Modelado**

El Lenguaje de Modelado Unificado (UML, por sus siglas en inglés) es una de las herramientas más utilizadas en el mundo actual del desarrollo de software, esto se debe a que permite a los desarrolladores crear diseños que engloben sus propósitos de manera sencilla y fácil de comprender para otras personas. El UML está compuesto por diversos elementos gráficos que se combinan para formar diagramas. Proporciona características que permiten organizar y extender los diagramas. Es necesario resaltar que UML indica que es lo que supuestamente hará el sistema, pero no como lo hará (Schmuller, 2004).

**Framework utilizado**

Un framework es un ambiente de trabajo que contiene librerías de códigos y módulos que pueden ser reutilizados para el rápido desarrollo de aplicaciones. Se definió para el desarrollo del sistema la utilización de Django en su versión 1.6.1

Django es un framework que utiliza Python y que permite el desarrollo rápido de aplicaciones web. Usa una modificación de la arquitectura Modelo-Vista-Controlador (MVC), llamada MTV (Model-Template-View), que sería Modelo-Plantilla-Vista, está forma de trabajar permite que sea pragmático, es decir, que se actúe dando prioridad a las consideraciones prácticas.

**Bibliotecas de soporte para el desarrollo del sistema de recomendación**

NLTK: se usa para el trabajo con texto, pre-procesamiento y limpieza de palabras, como, por ejemplo:

- Reducir todas las palabras a su raíz lexicográfica (Ejemplo: implementando - implementar).
- Eliminar los caracteres raros, los signos de puntuación y dejar todo el texto en minúscula.
- Elimina las palabras comunes de la lengua en la que está escrito (Ejemplo: el, para, de, por, y, un, entre otras).

**Entorno de Desarrollo Integrado**

Un entorno de desarrollo integrado o IDE (siglas provenientes del inglés *Integrated Development Environment*), es un programa informático que brinda un conjunto de componentes que hacen más fácil la programación. Conforma un ambiente favorable para los desarrolladores. Como entorno de desarrollo integrado se utiliza PyCharm en su versión 4.0.4., el cual permite la integración con el framework Django y soporta intérpretes de Python 2.7.11

**Herramienta CASE**

Se selecciona Visual Paradigm for UML 8.0 como herramienta CASE (siglas provenientes del inglés *Computer Assistant Software Engineering*), para el modelado de la propuesta de solución. Esta herramienta permite representar todo tipo de diagramas en el ciclo de vida del desarrollo de software. Además, se integra con Python y PostgreSQL, soportando también el modelado de procesos de negocio con BPMN.

**Sistema Gestor de Base de Datos**

Un Sistema de Gestor de Base de Datos (SGBD) es un conjunto de programas que permiten la creación de base de datos y proporciona herramientas para añadir, borrar, modificar y eliminar

datos de la base de datos, además de mantener la integridad, confidencialidad y seguridad de los datos.

PostgreSQL 9.4 es un SGBD objeto-relacional de código abierto, el cual puede ser ejecutado sobre la mayoría de los sistemas operativos que existen en la actualidad. El sistema es usado para manejar grandes cantidades de información y se basa en el modelo relacional, aunque incorpora conceptos del modelado orientado a objeto. Se destaca por ser robusto y cumplir con los estándares SQL (PostgreSQL, 2013).

Se utiliza PgAdmin en su versión 1.20 como herramienta de código abierto con el propósito general de diseñar, mantener, y administrar las bases de datos de PostgreSQL.

## 1.7 Conclusiones parciales

En este capítulo se analizaron los principales conceptos para una mejor compresión del tema. Se realizó un estudio a las técnicas de agrupamiento Star. Al analizar los diferentes algoritmos de agrupamiento se identifican las principales características de la solución y se decide implementar el algoritmo CStar. El estudio de las tecnologías apropiadas al desarrollo de la solución determinó emplear la metodología CRISP-DM para orientar el trabajo con la minería de datos y AUP-UCI para guiar el desarrollo de las nuevas funcionalidades del componente. Como lenguaje de programación Python en su versión 2.7.11 y, PyCharm en su versión 4.0.4 como entorno de desarrollo integrado para la implementación de la solución.

# Capítulo 2: Características del sistema

## 2.1 Introducción

En este capítulo se exponen los elementos que permiten describir la propuesta de solución. Se aborda el modelo conceptual del negocio, diagramas y descripciones asociadas al proceso de minería de texto para el agrupamiento de documentos. Además, son definidos los requisitos funcionales y no funcionales del sistema. Se establecen los artefactos requeridos en la planificación definidos por la metodología AUP-UCI.

## 2.2 Modelo conceptual

Un modelo conceptual tiene como objetivo identificar y explicar los conceptos significativos en un dominio de problema, identificando los atributos y las asociaciones existentes entre ellos (Software, 2005). En este modelo se definen cuáles son y cómo se relacionan los conceptos relevantes en la descripción del problema, en este caso describe los conceptos relacionados con el negocio del sistema RBC.

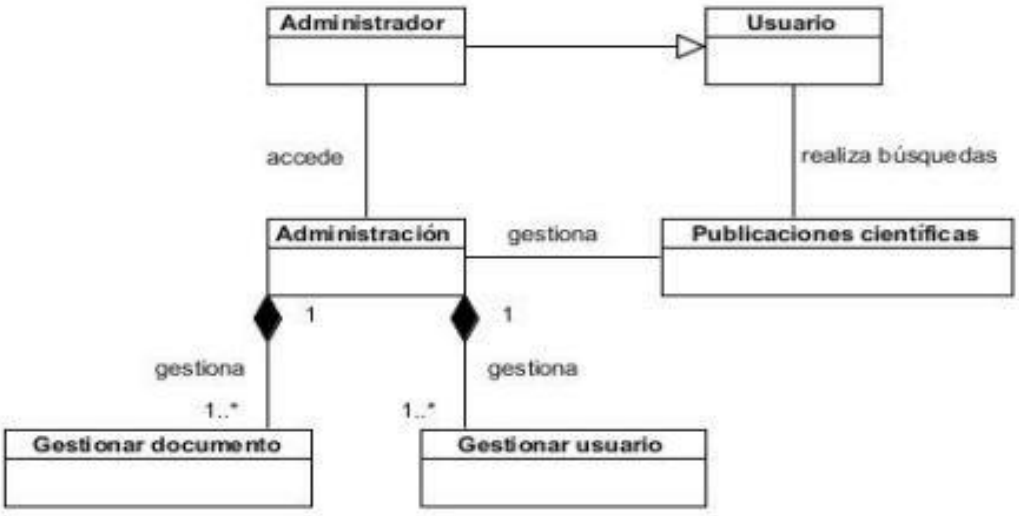



***Ilustración 2: Modelo conceptual***

A continuación, se describen los conceptos fundamentales que componen el modelo conceptual del negocio:

- **Usuario**: Persona que por medio de un ordenador puede acceder a la herramienta para realizar la búsqueda.

- **Publicaciones científicas**: Documentos científicos que el usuario obtendrá como resultado al realizar la búsqueda.
- **Administrador**: Es un usuario del sistema que, además, es el responsable de la gestión de documentos y otros usuarios.
- **Gestionar documento**: Es la funcionalidad donde se podrá gestionar todos los datos referentes a la gestión de documentos.

## 2.3 Propuesta de solución

El sistema propuesto adiciona funcionalidades con respecto al proceso de recuperación de información. Específicamente la recuperación por palabras claves. Actualmente el sistema devuelve todos los documentos que se encuentran en la base datos que contienen las palabras introducidas en el criterio de búsqueda, pero no los devuelve por orden de similitud y por lo que influye en la calidad del resultado. La modificación propuesta es aplicarle a la colección de documentos obtenidas de la base de datos un algoritmo de agrupamiento para obtener los documentos más relevantes y grupos similares entre los que de cierta forma mejora la calidad de la información que se le muestra al usuario.


***Ilustración 3: Flujo de propuesta de solución***

El proceso de Minería de Texto es fundamental para el desarrollo de la solución propuesta, debido a la importancia que requiere el trabajo con los datos, en este caso, con los documentos en la base de datos de publicaciones científicas y las palabras claves por la que va a realizar la búsqueda el usuario.

A continuación, se explica el proceso de minería de texto teniendo en cuenta las etapas que propone la metodología CRISP-DM:

- **Comprensión del negocio:** En esta etapa se realiza una entrevista al cliente para conocer los requerimientos del componente a desarrollar desde una perspectiva del proceso de minería. Y se lleva a cabo un estudio de la herramienta RBC
- **Comprensión de los datos:** Se obtienen los documentos científicos de fuentes de información, en este caso se selecciona el sitio web Revista Cubana de Ciencias Informáticas (disponible en http://rcci.uci.cu). Además, se realiza una exploración de los documentos seleccionados y se comprueba la calidad de los mismos.

**Preparación de los datos**

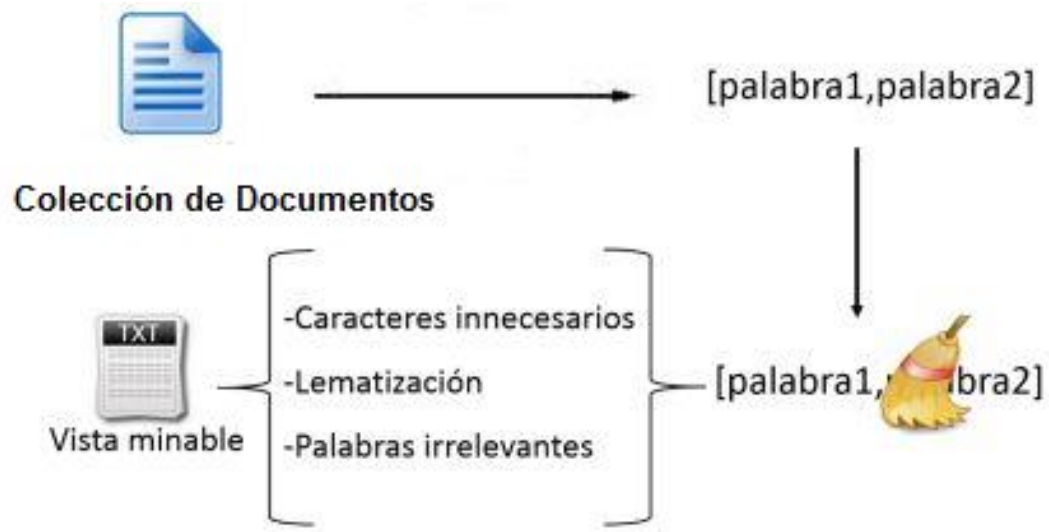



***Ilustración 4: Preparación de los datos***

Como se muestra en la imagen anterior, lo primero que se re realiza en esta etapa es convertir la colección de documentos obtenidos por la consulta realizada a la base de datos. A esta colección se le aplica un procesamiento del lenguaje natural, donde se tokeniza[2] dicha colección obteniendo un vector de palabras por cada documento.

Al vector de palabras obtenido, se le realiza una limpieza del texto, donde se eliminan:

- Mayúsculas (Ejemplo: Método=método).

---

[2] Separar una oración por palabras.

- Tildes (Ejemplo: información=informacion).
- Sinónimos (Ejemplo: implementar=realizar).
- Caracteres innecesarios (Ejemplo:/, *, &, entre otros).
- Se eliminan las palabras irrelevantes, que son las palabras que no aparecen en el diccionario (Ejemplo: las preposiciones, conjunciones, artículos).
- Se radicaliza y/o lematiza[3] la palabra (Ejemplo: desarrollando = desarrollar).

Al finalizar el procesamiento del lenguaje, se obtiene la vista minable de documentos de la base de datos.

**Modelado**

Una vez que se tiene la vista minable de cada uno de los documentos almacenados en la base de datos, se procede a aplicar el algoritmo seleccionado, en este caso el algoritmo de agrupamiento CStar. Para determinar el grado de similitud entre los documentos, se utiliza la función coseno.

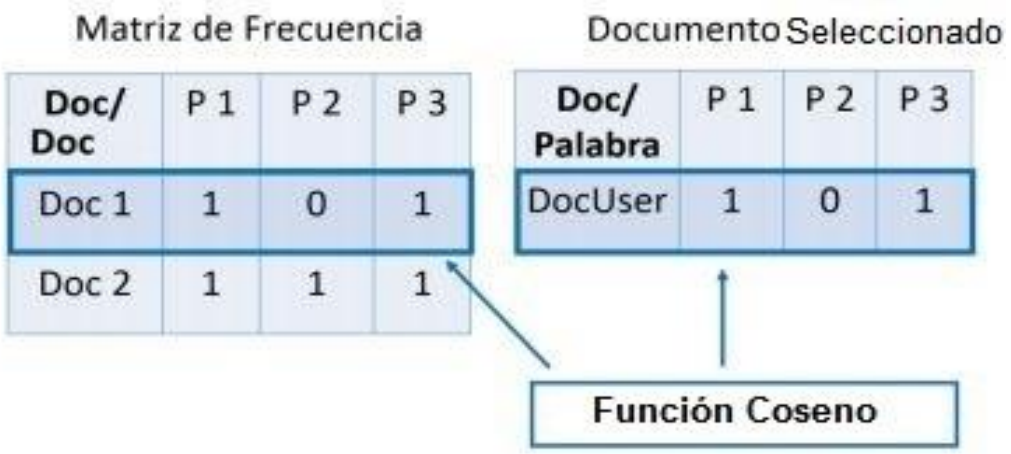



***Ilustración 5: Formación de matriz de semejanza***

El coeficiente del coseno constituye la función utilizada para el cálculo de la similitud para cada par de documentos en la base de datos, la cual se define de la siguiente manera (Pérez Suárez, 2008):

[3] Proceso lingüístico que consiste en, dada una forma flexionada, hallar el lema correspondiente. El lema es la forma que por convenio se acepta como representante de todas las formas flexionadas de una misma palabra.

$$S(d_i, d_j) = \frac{\sum_{h=1}^{k} peso_{ih} * peso_{jh}}{\sqrt{\sum_{h=1}^{k} peso_{ih}^2 * \sum_{h=1}^{k} peso_{jh}^2}}$$

***Ilustración 6: Fórmula para calcular la similitud entre documentos***

Donde $\mathbf{d_i}$ y $\mathbf{d_j}$ son los documentos a comparar, **k** es el número de términos (palabras claves) que caracterizan los documentos, y $\mathbf{peso_{ih/jh}}$ es el peso del término **h** en el documento **i/j**, calculado teniendo en cuenta la frecuencia de aparición de ese término en el documento (Santillán & Ginestá, 2010).

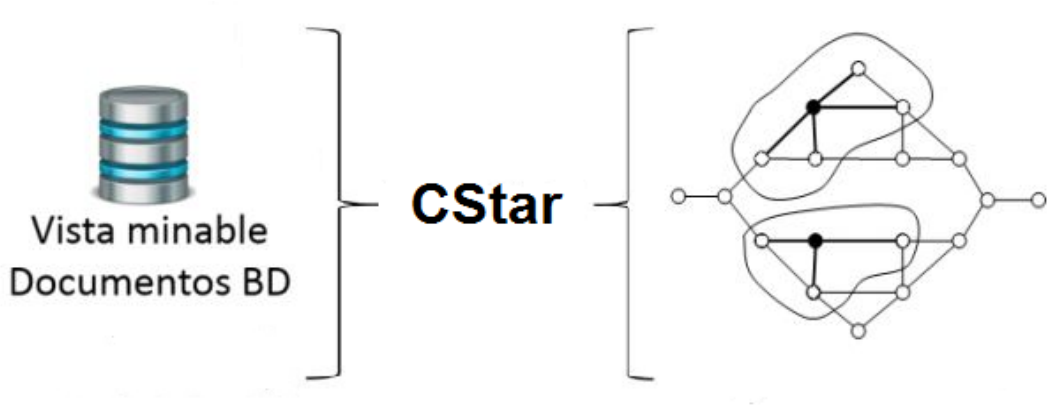



***Ilustración 7: Aplicar minería***

El algoritmo CStar a partir de los documentos presentes en la base datos crea un grafo bidireccional donde la relación entre los documentos se obtiene a partir de una matriz de similitud. Luego de que se obtiene el grafo inicial se van realizando transiciones en las cuales se van eliminando aristas obteniendo un grafo unidireccional (Ilustración 8). Al finalizar las transiciones devuelve un conjunto de vértices candidatos a centro de los grafos relevantes (clúster), que se forman alrededor de los vértices que mayor peso tienen.

Para obtener finalmente los grafos que construye el algoritmo, se crea en un primer momento la lista de vértices candidatos a partir de las transiciones descritas anteriormente, obteniéndose un conjunto reducido de vértices con grado de votación nulo que pueden formar sub-grafo **EC** y que tienen las mayores posibilidades de ser seleccionados como centro. Seguidamente se elabora el conjunto inicial de centros y la lista de vértices dudosos, inicializándose el conjunto inicial de centros con todos aquellos vértices aislados de $\mathbf{G_\beta}$.

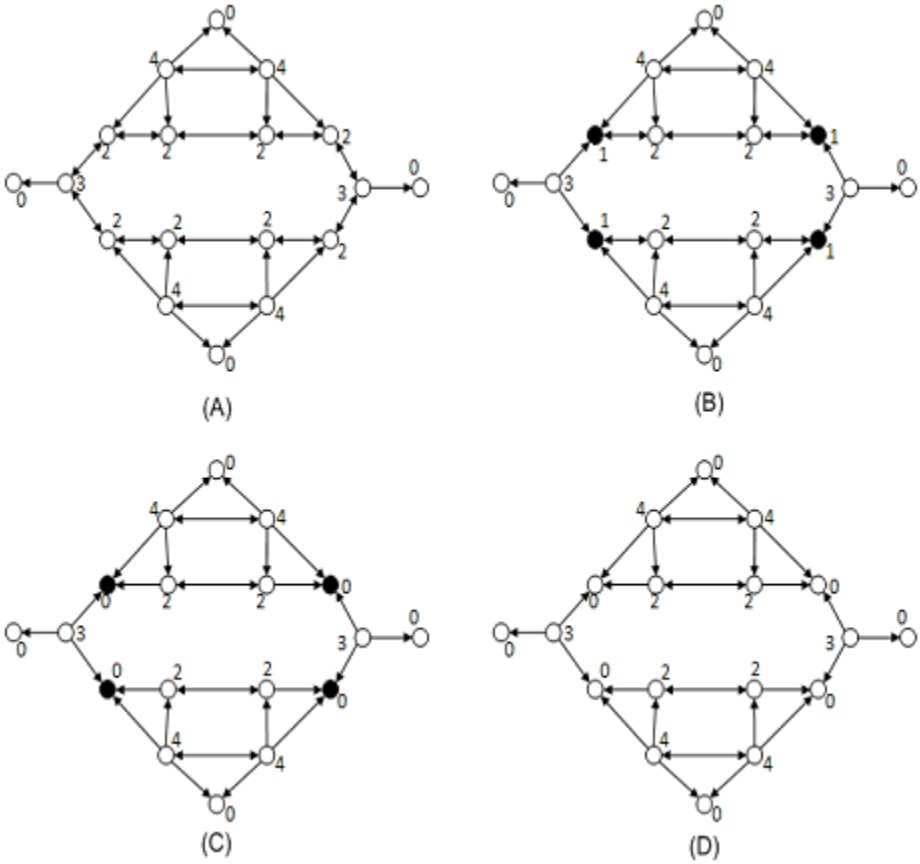



***Ilustración 8: Transiciones del algoritmo CStar***

Después, la lista de vértices candidatos es procesada, garantizando que el vértice **v** seleccionado en cada iteración forma el sub-grafo más denso de los candidatos en esa iteración. Posteriormente se seleccionan nuevos vértices centro en la lista de vértices dudosos, todos estos vértices son verificados en el mismo orden en que fueron insertados, garantizando de esta forma procesar siempre al vértice de la lista que forma el sub-grafo **EC** más denso de todos los posibles, evitando así la obtención de un conjunto redundante (Ilustración 9).

Por último, se elimina del conjunto inicial de centros todo vértice que forme un grupo redundante. Como resultado de la búsqueda se mostrará el autor y un resumen de los documentos obtenidos (Pérez Suárez, 2008). Una visión más detallada sobre el algoritmo propuesto, se puede obtener analizando el pseudo-código del mismo (Ilustración 10).

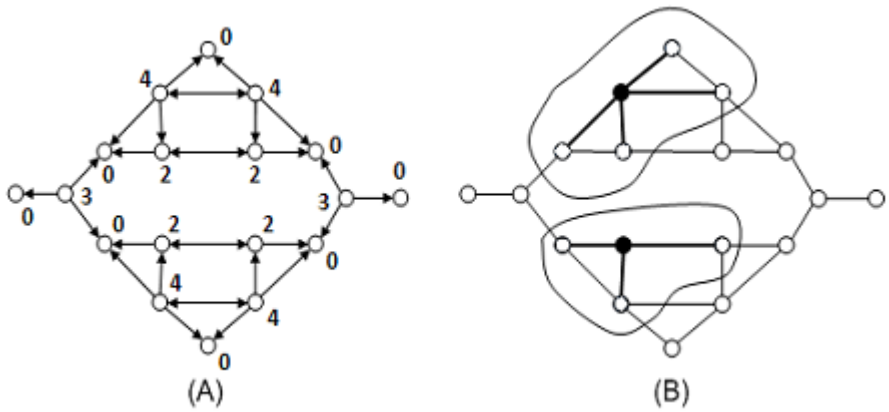



*Ilustración 9: Ejemplo de posibles sub-grafos EC*

Entrada: $D := \{d_1, d_1, \dots, d_N\}$ – colección de documentos
$\beta$ – medida de similitud
Salida: SC - conjunto de clúster

**=== // Fase 1 – Creación de la lista de candidatos L ===**

1. Calcular las semejanzas entre todos los pares de objetos para construir el grafo de β-semejanza.
2. Crear un grafo dirigido $G_\beta^0$ a partir del grafo β-semejante.
3. Mientras número de aristas $G_\beta^n \neq G_\beta^{n-1}$:
    a. Eliminar arista (u,v) si (u.out < v.out) para toda arista (u,v) del grafo $G_\beta^n$
4. Calcular el grado de votación de todos los objetos de $G_\beta^n$
5. Sea L el conjunto de objetos con grado de votación mayor que 0.

**=== //Fase 2 – Crear la lista de centros X y la lista de centros inciertos U ===**

6. Sea X el conjunto de objetos aislados y U conjunto vacío.
7. Mientras existan objetos en L:
    a. Sea $v$ el objeto que tiene el máximo grado de votación en L.
    b. Agregar $v$ a X, si $v.Adj \cap X$ es vacío.
    c. Sino:
        i. Sea F el conjunto de objetos $u$ adyacentes de $v$ tales que $u.Adj \cap X$ es vacío
        ii. Si F no es vacío:
            1. Agregar $v$ a X, si para todo elemento $f$ de F, el grado de votación de $v$ es mayor que el grado de votación de $f$:
            2. Sino, agregar $v$ a U
    d. Eliminamos $v$ de L.

**=== //Fase 3 – Seleccionar nuevos centros de la lista de centros inciertos U ===**

8. Para todo elemento $v$ de U:
    a. Agregar $v$ a X, si para todo elemento $u$ adyacente a $v$, $u.Adj \cap X$ es vacío:

**=== //Fase 4 – Eliminar elementos redundantes de X ===**

9. Ordenar X por grado de votación de menor a mayor
10. Sea SC conjunto vacío
11. Para todo elemento $v$ de X:
    a. Eliminamos $v$ de X, si $v$ es redundante
    b. Sino, agregamos $v$ y sus adyacentes al conjunto SC

*Ilustración 10: Pseudo-código algoritmo CStart*

**Implementación**

En esta etapa se presenta los documentos obtenidos de forma entendible para el usuario. Y se analiza las mejoras que se puedan realizar a la herramienta y la posibilidad de incorporarle nuevas funcionalidades que permitan el crecimiento del sistema RBC.

## 2.4 Especificaciones de los requisitos del sistema

Según (Sommerville, 2005), los requisitos del sistema especifican qué es lo que debe hacer (sus funciones) y sus propiedades esenciales y deseables. En esta etapa el principal objetivo es identificar las necesidades del cliente, con el fin de permitir que el equipo de desarrollo pueda trabajar según los requerimientos especificados y logre un producto terminado con calidad.

A continuación, se especifican los requisitos funcionales y no funcionales del componente de búsqueda a desarrollar.

### 2.4.1 Requisitos Funcionales del sistema

Los requisitos funcionales de un sistema, según (Sommerville, 2005), expresan la naturaleza del funcionamiento del sistema (cómo interactúa el sistema con su entorno y cuáles van a ser su estado y funcionamiento). Es decir que debe hacer el sistema.

**RF1:** Agrupar documentos

**-** Calcular similitud entre documentos

**-** Construir la lista de vértices candidatos

**-** Construir el conjunto inicial de centros y la lista de vértices dudosos

**-** Seleccionar nuevos vértices centro y adicionarlos a al conjunto de centros

**-** Eliminar el conjunto de centros de todo vértice que forme un grupo redundante.

**RF2:** Mostrar documentos

**RF3:** Descargar documentos

### 2.4.2 Requisitos no funcionales

Los requisitos no funcionales del sistema, según (Sommerville, 2005), son las restricciones sobre el espacio de posibles soluciones. Es decir, como debe ser el sistema.

**Usabilidad**

- La interfaz de la aplicación debe permitirle al usuario sin experiencia adaptarse rápidamente, para de esta forma poder interactuar fácilmente con el sistema.

**Disponibilidad**

- El sistema debe estar disponible para el usuario en el momento que lo necesite.
- Debe mantener su funcionamiento con la menor afectación posible en caso de que se presente algún error.

**Rendimiento**

- El funcionamiento del sistema debe ser estable.

**Apariencia e Interfaz**

- Todos los textos y mensajes en pantalla aparecerán en idioma español.
- La aplicación deberá poseer una interfaz fácil de usar por los usuarios.

**Hardware**

- Para las PC clientes:
    1. Requerimientos mínimos 512MB de RAM recomendada o superior.
    2. Tarjeta de red para establecer la conexión.
- Para el servidor:
    1. Computador con procesador Intel Xeon que es el tipo de microprocesador que utilizan los servidores de la UCI, 4GB de memoria RAM, 1Tb de disco duro.
    2. Tarjeta de red para establecer la conexión.

**Software:**

- Para las PC clientes:
    1. Sistema Operativo Windows XP, Windows 7 o Windows 8 y Linux con interfaz gráfica y soporte para conectarse a la red.
    2. Para la utilización del sistema se requerirá el uso de un navegador web, preferentemente Mozilla Firefox en su versión 30.0 o superior.
- Para el servidor:

1. Se requiere el gestor de base de datos PostgreSQL 9.4, el marco de trabajo Django 1.6.1, las bibliotecas necesarias como nltk, gensim, sklearn, entre otras y el Sistema Operativo Windows 8 o Linux.

## 2.5 Definición de los actores

Un actor es un agente, alguien o algo que solicita un servicio al sistema o actúa como catalizador para que ocurra algo. Este representa un rol que es jugado por una persona, un dispositivo hardware, incluso otro sistema (Merseguer, 2010).

***Tabla 2: Actor relacionado con el sistema***

| **Actor** | **Objetivo** |
|---|---|
| Usuario | El usuario buscará por palabras claves en el sistema para obtener los documentos que se encuentren en la base de datos. Además, podrá conocer el resumen de los documentos obtenidos así como su autor y descargarlos. |

## 2.6. Diagrama de caso de uso del sistema

El diagrama de casos de uso permite describir la secuencia de eventos que los actores utilizan para completar un proceso a través del sistema.

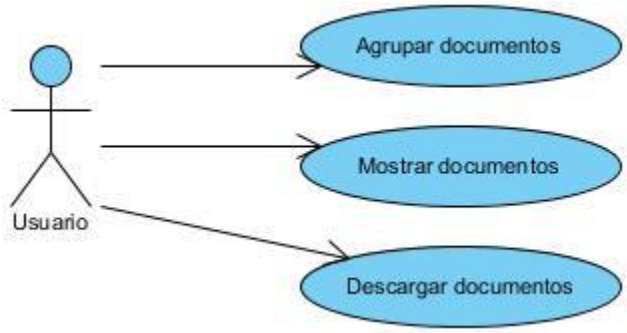
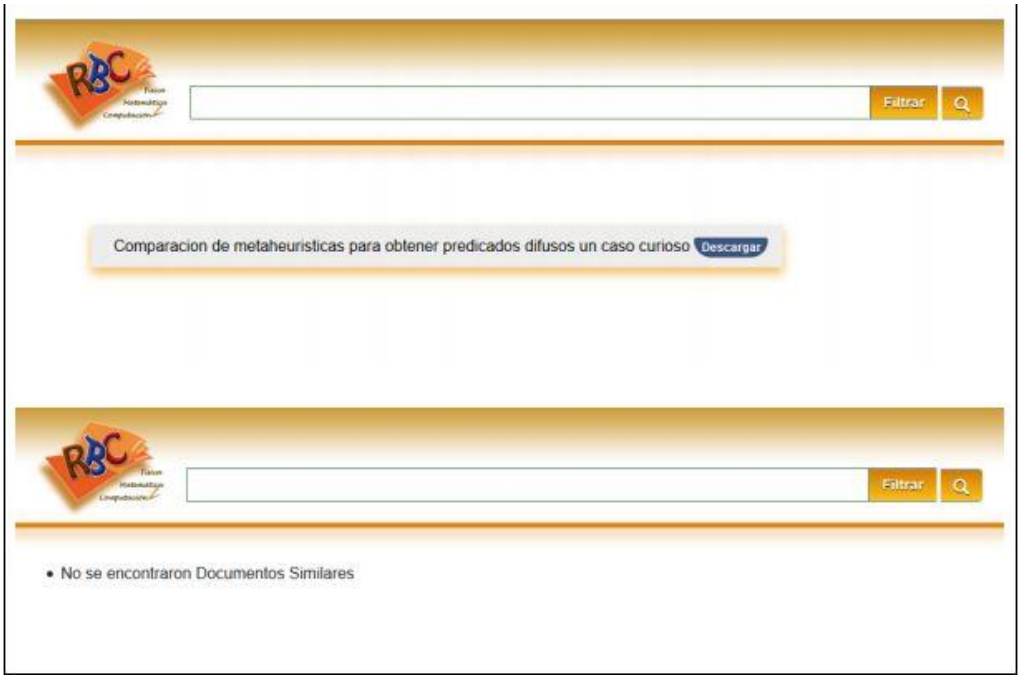



***Ilustración 11: Diagrama de casos de uso del sistema***

### 2.6.1 Descripción de los casos de uso del sistema

***CU1: Agrupar documentos***

| | |
|---|---|
| **Objetivo** | Buscar documentos mediante el algoritmo de agrupamiento |
| **Actores** | Sistema |
| **Resumen** | El caso de uso inicia cuando el usuario introduce los criterios de búsqueda |
| **Complejidad** | Alta |
| **Prioridad** | Alta |
| **Precondiciones** | Seleccionar los criterios de búsqueda |
| **Postcondicion** | |
| **Flujo de eventos** | |
| **Flujo básico Agrupar documentos** | |
| **Actor** | **Sistema** |
| Da click sobre el botón Buscar | |
| | Redirecciona a la página con los resultados de la búsqueda. |
| **Flujos alternos** | |
| En caso de dar click en el botón de buscar y no haber introducido las palabras claves | |
| **Actor** | **Sistema** |
| | Muestra el mensaje de error “Campo Obligatorio” |
| **Prototipo de interfaz gráfica de usuario de la funcionalidad Agrupar documentos** | |



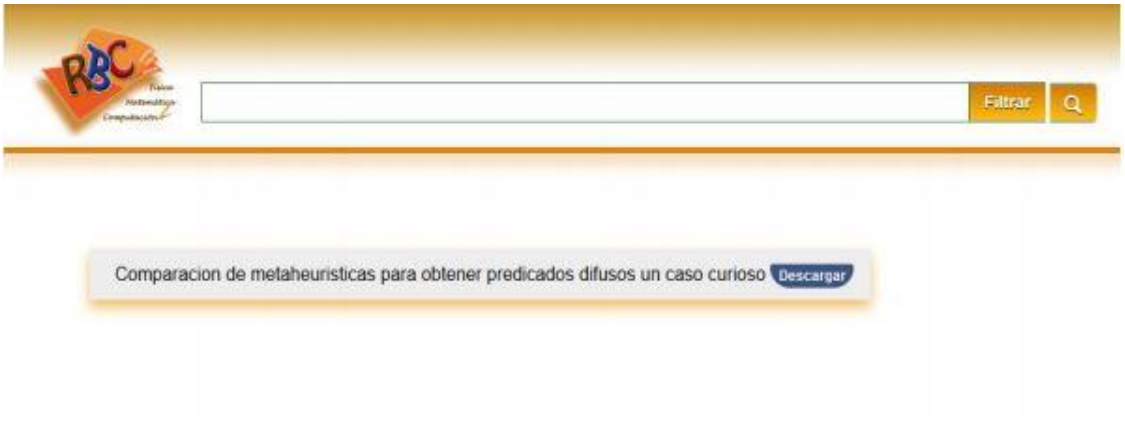

***CU2: Mostrar documentos***

| | |
|---|---|
| **Objetivo** | Obtener los documentos similares a los criterios de búsqueda entrados. |
| **Actores** | Usuario |
| **Resumen** | El caso de uso inicia cuando el usuario selecciona la opción de Buscar |
| **Complejidad** | Media |
| **Prioridad** | Alta |
| **Precondiciones** | |
| **Postcondiciones** | |
| **Flujo de eventos** | |
| **Flujo básico Mostrar documentos** | |
| **Actor** | **Sistema** |
| | |

| | |
|---|---|
| | Muestra una página con el resultado de la búsqueda |
| **Prototipo de interfaz gráfica de usuario de la funcionalidad Mostrar documentos** | |
|  | |

***CU3: Descargar documento***

| | |
|---|---|
| **Objetivo** | Descargar los documentos obtenidos |
| **Actores** | Usuario |
| **Resumen** | El caso de uso inicia cuando el usuario selecciona la opción de Descargar |
| **Complejidad** | Baja |
| **Prioridad** | Media |
| **Precondiciones** | Haber obtenido y seleccionado al menos un documento en la búsqueda |
| **Postcondiciones** | |
| **Flujo de eventos** | |
| **Flujo básico Descargar documento** | |
| **Actor** | **Sistema** |

| | |
|---|---|
| Selecciona la opción Descargar | |
| | Permite guardar el documento |
| **Prototipo de interfaz gráfica de usuario de la funcionalidad Descargar documento** | |
|  | |

## 2.7 Conclusiones parciales

En este capítulo se definieron las características de la propuesta de solución. Se especificó el modelo conceptual asociado al proceso de la minería de texto para el clustering de documentos el cual permitió definir los conceptos fundamentales del negocio. De esta forma, se logró obtener una visión más clara del entorno sobre cual se sitúa el problema a resolver. Se definieron los requisitos funcionales y los no funcionales del sistema. Se modeló el diagrama de casos de uso del sistema y se realizó la especificación de cada caso de uso, permitiendo una mejor compresión del flujo básico de estos.

# Capítulo 3: Arquitectura y diseño

## 3.1 Introducción

En este capítulo se describe el diseño del componente propuesto para el sistema RBC. Se muestran los diagramas de clases del diseño empleando estereotipos web, los diagramas de colaboración, el diagrama de paquetes y el diseño de la base de datos. Se define el estilo arquitectónico y los patrones.

## 3.2 Diseño de la Arquitectura

Una arquitectura de software constituye un modelo comprensible de cómo está estructurado el sistema y cómo trabajan juntos sus componentes (Cervantes, 2010).

Para el desarrollo del componente se empleará la arquitectura cliente-servidor, ya que la aplicación web estará alojada en el servidor de aplicaciones y el usuario podrá acceder a la misma desde cualquier PC cliente.

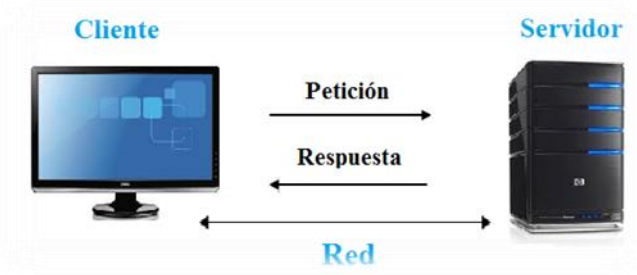



***Ilustración 12: Arquitectura Cliente-Servidor***

### 3.2.1 Patrón Arquitectónico

El framework Django define una modificación de la arquitectura Modelo-Vista-Controlador (MVC), llamada MTV (Model-Template-View), que sería Modelo-Plantilla-Vista. El modelo en Django sigue siendo Modelo, la vista se llama plantilla (Template) y el controlador se llama Vista.

- **Capa Vista (views):** Determina qué datos serán visualizados sin encargarse del estilo de la presentación de los mismos.

- **Capa Modelo (models):** Define los datos almacenados, se encuentra en forma de clases de Python, cada tipo de dato que debe ser almacenado se encuentra en una variable con ciertos parámetros, posee métodos también. Todo esto permite indicar y controlar el comportamiento de los datos.
- **Capa Plantilla (templates):** La plantilla recibe los datos de la vista y luego los organiza para la presentación al navegador web.

La imagen que se muestra a continuación muestra el funcionamiento del MTV en Django.

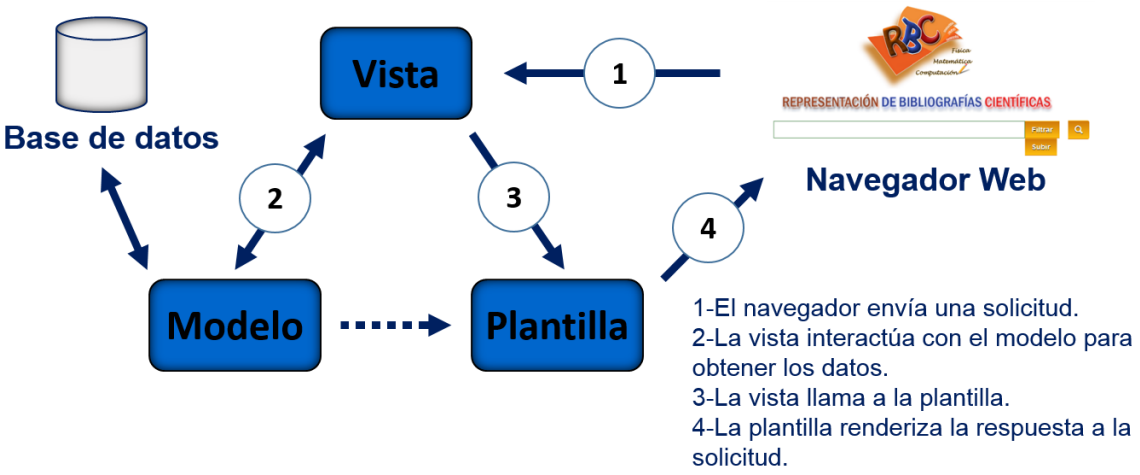



***Ilustración 13: Patrón de Arquitectura Modelo-Plantilla-Vista***

## 3.3 Etapa de Diseño

En esta etapa se especifican los elementos necesarios para lograr la correcta implementación de la solución propuesta. Se muestra el diagrama de paquetes, los diagramas de clases del diseño utilizando estereotipos web.

### 3.3.1 Diagrama de paquetes

El diagrama de paquetes permite un mejor entendimiento del sistema, organizando el mismo a través de paquetes y sus relaciones, conformando así, una estructura lógica del sistema.

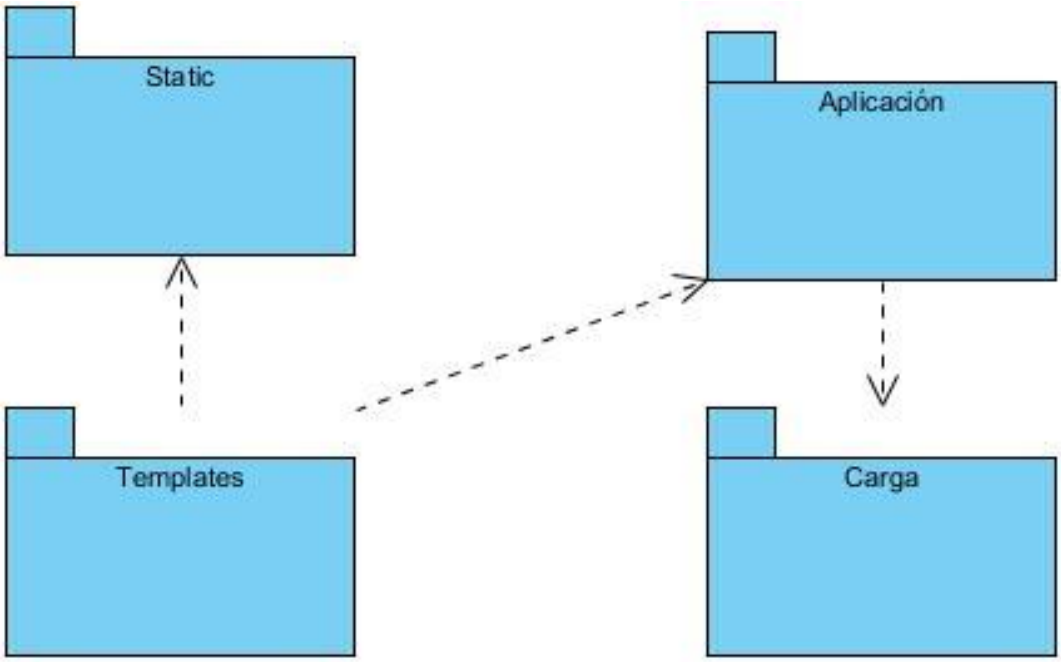



***Ilustración 13: Diagrama de paquetes***

- **Static:** Este paquete contiene los archivos CSS, JavaScript y las imágenes utilizadas para el diseño y estilo de las plantillas HTML.
- **Templates:** Contiene todas las plantillas HTML.
- **Aplicación:** Este paquete contiene las nuevas clases y funcionalidades implementadas.
- **Carga:** Contiene dos subdirectorios, uno llamado Documentos donde se almacenan todos los documentos en distintos formatos que se encuentran en la base de datos. El otro subdirector es Carga, el cual contiene el documento subido por el usuario.

### 3.3.2 Diagrama de clases del diseño utilizando estereotipos web

Los diagramas de clases son una estructura estática donde la representación de los requisitos se lleva a cabo a través de las clases del sistema y sus interrelaciones.

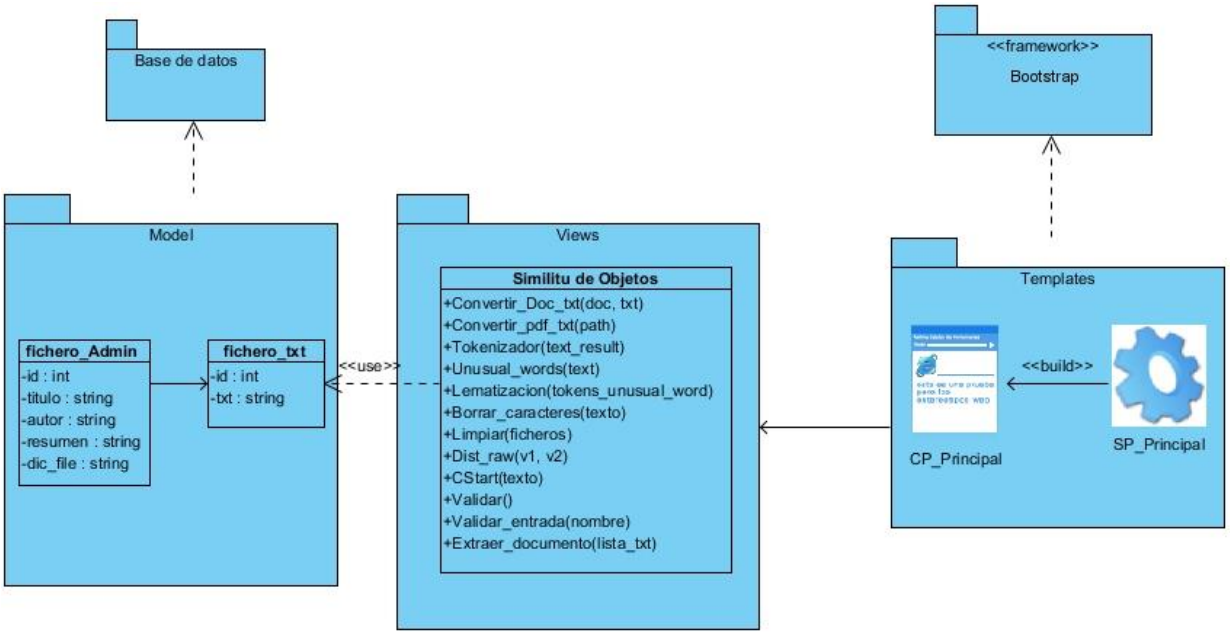



***Ilustración 14: Diagrama de clases del diseño RF Agrupar documentos y Mostrar documentos***

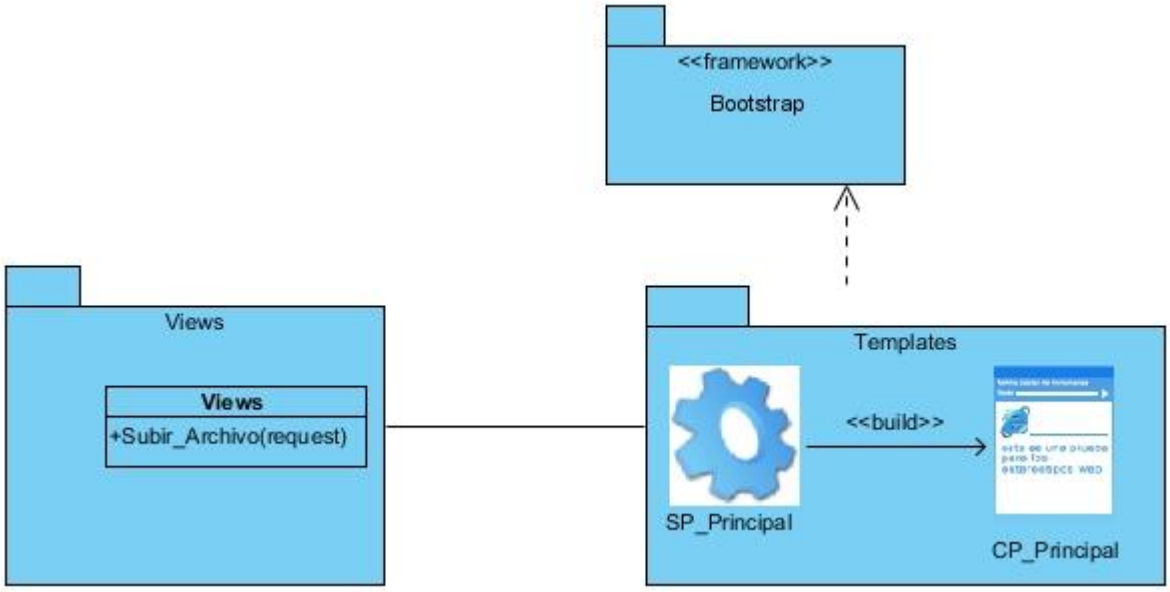



***Ilustración 15: Diagrama de clases del diseño Descargar Documento***

## 3.4 Modelo físico de los datos

El modelo físico de los datos describe las representaciones físicas de los datos utilizados en la herramienta RBC y que serán almacenados en bases de datos. Los elementos esenciales del

diagrama son las entidades, los atributos y las relaciones entre las entidades (Martínez Moreira, 2016).

- **Entidades:** Objetos que el sistema necesita guardar su información, en este caso, las entidades son los documentos y los vectores.
- **Atributos:** Características de las entidades, las cuales se clasifican en obligatorios, opcionales, claves foráneas y claves primarias.
- **Relaciones:** Muestra la relación entre las entidades.

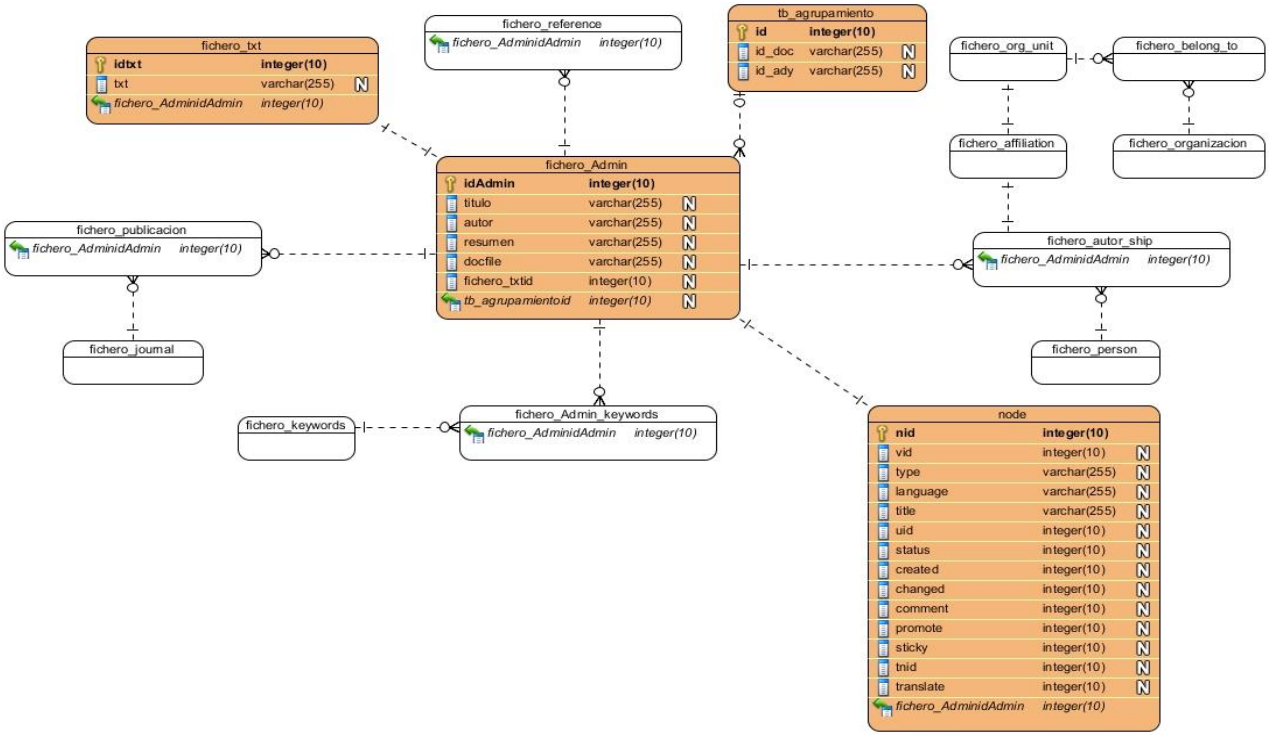



***Ilustración 16: Modelo físico de la base de datos***

Para el desarrollo de las nuevas funcionalidades propuestas, se utiliza la tabla fichero_Admin de la base de datos que posee el sistema RBC inicialmente, además se crea la tabla fichero_txt, la tabla tb_agrupamiento y la tabla node.

- **fichero_Admin:** Contiene toda la información referente a los documentos almacenados que serán comparados entre ellos para ser agrupados.

- **fichero_txt:** Esta tabla contiene información de cada documento almacenado, pero en formato txt.
- **tb_agrupamiento:** La cual almacenará los grupos formados por la aplicación del algoritmo. En esta se guardan los vértices centros en la columna *id_doc* y los vértices satélites (adyacentes) en la columna *id_ady.*
- **node:** El cual almacenará la información de los nodos.

## 3.5 Patrones de diseño

Los patrones de diseño brindan una solución ya probada y documentada a problemas de desarrollo de software que están sujetos a contextos similares (Tedeschi, 2010).

**Patrones GRASP**

Patrones Generales de Software para Asignación de Responsabilidades (GRASP, por sus siglas del inglés *General Responsibility Assignment Software Patterns*). Los patrones GRASP describen los principios fundamentales de la asignación de responsabilidades a objetos expresados en forma de patrones (Larman, 1999).

Existen 9 patrones GRASP: experto, creador, controlador, bajo acoplamiento, alta cohesión, polimorfismo, fabricación pura, indirección y variaciones protegidas. Django que es el framework que se utiliza para el desarrollo de las nuevas funcionalidades del sistema RBC, implementa 3 de los patrones antes mencionados (experto, bajo acoplamiento y alta cohesión).

- **Experto:** Es el encargado de asignar la responsabilidad de la creación de un objeto o la implementación de un método a una clase que contenga toda la información necesaria para cumplir con dicha responsabilidad (Pressman, 2005).

  Ejemplo: el models.py se encarga de la estructura de la base de datos y la lógica de la misma.

- **Bajo acoplamiento:** El acoplamiento es una medida de la fuerza con que una clase está conectada a otras clases, con qué las conoce y con qué concurre a ellas. En tal sentido, el término bajo acoplamiento significa que una clase no depende de muchas clases (Pressman, 2005).

Ejemplo: las URL llaman a funciones y métodos que implementan las Vistas, pero algún cambio que se realice a una función o método, no afecta la URL.

- **Alta cohesión*:*** Es una medida de cuán relacionadas y enfocadas están ñas responsabilidades de una clase. Una alta cohesión caracteriza a las clases con responsabilidades estrechamente relacionadas que no realizan un trabajo enorme (Pressman, 2005).

  Ejemplo: las clases del modelo describen la estructura de las tablas de la base de datos de manera coherente y precisa.

Otro patrón que se evidencia en el desarrollo de las nuevas funcionalidades, es el patrón **Creador**, el mismo tiene como objetivo asignar a la clase B la responsabilidad de crear una instancia de la clase A (Larman, 1999).

Ejemplo: el formulario FicheroForm es responsable de la creación de objetos de tipo formulario.

## 3.6 Conclusiones Parciales

En el presente capítulo se planteó el diseño de la herramienta RBC y el patrón arquitectónico MTV (Modelo-Plantilla-Vista). Se realizó una breve descripción de los patrones de diseño utilizados. Se modelaron los diagramas de paquetes, y el modelo físico de la base de datos, todo esto para lograr un mejor entendimiento de la propuesta de solución.

# Capítulo 4: Implementación y Prueba

## 4.1 Introducción

En este capítulo se describen aspectos relacionados con la implementación y validación del componente desarrollado. Se exponen los principales resultados obtenidos durante la etapa de prueba, para garantizar su correcto funcionamiento y el cumplimiento con los requisitos definidos por el cliente.

## 4.2 Etapa de implementación

En esta etapa se realiza la codificación, se realiza la programación de la solución diseñada, en el lenguaje de programación y la plataforma elegida a tal efecto teniendo en cuenta las restricciones obtenidas en la etapa de análisis (González, 2007).

### 4.2.1 Diagrama de componente

Los diagramas de componentes muestran las piezas del software que conformarán un sistema, donde estas piezas representan todos los tipos de elementos software que se incluyen en la fabricación de aplicaciones informáticas, por tanto, pueden ser simples archivos, paquetes.

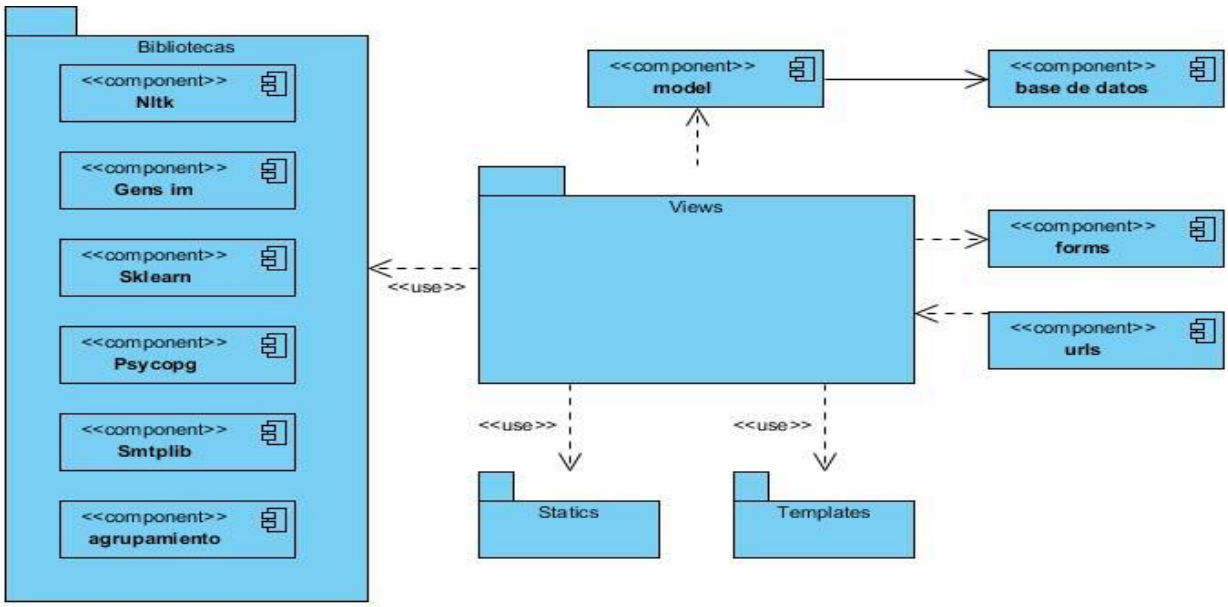



***Ilustración 17: Diagrama de paquete***

El paquete agrupamiento es donde se encuentra el algoritmo CStar.

## 4.3 Estándares de codificación empleados

Los estándares de codificación permiten establecer una forma de programación única, haciendo más entendible el código fuente para otros desarrolladores, facilitando el mantenimiento posterior de las aplicaciones o software desarrollados. Estos estándares definen entre otras cosas, la manera en que se van a escribir los comentarios, etc... Con el objetivo de alcanzar una uniformidad en el código, a continuación, se muestran algunos estándares de codificación utilizados.

- Iniciar el nombre de las variables de las clases con letra inicial mayúscula y en caso de ser un nombre compuesto se utiliza underscore (_).
- Al inicio de cada clase se realiza un comentario explicando el objetivo de la misma. Cada línea de un comentario empieza con # (numeral).
- Antes y después de la declaración de una clase o de una estructura y de la implementación de una función se deja una línea en blanco.
- Entre operadores lógicos y aritméticos se utiliza espacios en blanco.

## 4.4 Diagrama de despliegue

Un diagrama de despliegue modela la arquitectura en tiempo de ejecución de un sistema. Esta muestra la configuración de los elementos de hardware (nodos) y muestra cómo los elementos y artefactos del software se trazan en esos nodos (SOLUS, 2016). Este diagrama modela la topología del hardware del entorno donde se debe ejecutar el sistema, el software necesario para su funcionamiento y los protocolos de comunicación. Para la solución desarrollada tenemos:

1. **PC Cliente**: representa las PC clientes mediante las que se accederá al sistema disponible en el servidor de aplicaciones.
2. **Servidor de Aplicaciones Web**: es el servidor donde estará disponible, además debe estar instalado el lenguaje de programación Python 2.7.11, el framework Django 1.6.1 y el servidor web Apache.
3. **Servidor de Base de Datos:** almacena los datos con los que interactúa el sistema.

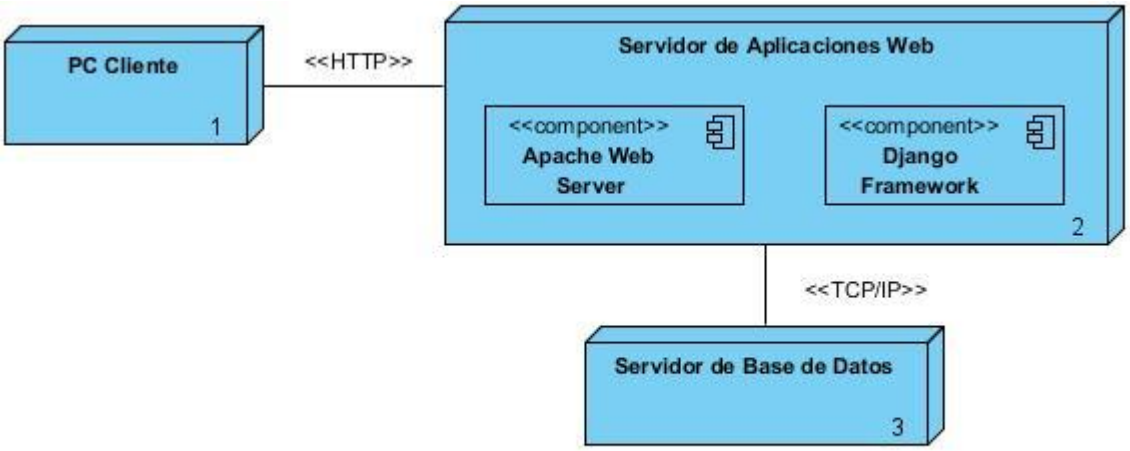



***Ilustración17: Diagrama de Despliegue***

**Descripción de los protocolos de comunicación:**

- **HTTP:** El Protocolo de Transferencia de Hipertexto (HTTP, siglas en inglés), es un sencillo protocolo cliente-servidor que articula los intercambios de información entre los clientes web y los servidores HTTP (Guijarro, 2012). HTTP se basa en sencillas operaciones de solicitud/respuesta. Un cliente establece una conexión con un servidor y envía un mensaje con los datos de la solicitud. En el servidor responde con un mensaje similar, que contiene el estado de la operación y su posible resultado (Guijarro, 2012).
- **TCP/IP:** El Protocolo de Control de Transmisión/Protocolo de Internet (TCP/IP, siglas en inglés), se utiliza para enlazar computadoras que usan sistemas operativos similares o diferentes, incluyendo PC, minicomputadoras centrales sobre redes de área local (LAN). En el caso del componente desarrollado se utiliza para interconectarlo con el servidor de base de datos.

## 4.5 Pruebas

Luego de la implementación de la solución, se realizan un conjunto de actividades para verificar la calidad del producto y su cumplimiento con los requisitos definidos. El objetivo de esta fase es detectar y solucionar los errores que presenta el componente desarrollado, y perfeccionar la solución implementada.

Las pruebas son un componente importante de la calidad del software, proceso de ejecutar un programa para detectar errores (Jovanovi´c, 2008).

### 4.5.1 Métodos de prueba

Los métodos de prueba definen estrategias para descubrir fallos en el sistema. Como métodos de prueba, Pressman propone los siguientes (Pressman, 2005):

**Pruebas de caja blanca**

Mediante los métodos de prueba de caja blanca, el ingeniero del software puede obtener casos de prueba que: garanticen que se ejercite por lo menos una vez todos los caminos independientes de cada módulo; ejerciten todas las decisiones lógicas en sus vertientes verdaderas y/o falsas; ejecuten todos los bucles en sus límites y con sus límites operacionales, y que se ejerciten las estructuras internas de datos para asegurar su validez (Pressman, 2005). Estas pruebas se realizan al código fuente para asegurar que la operación interna se ajuste a las especificaciones.

**Pruebas de caja negra**

Las pruebas de caja negra también conocidos como pruebas funcionales o pruebas de entrada y salida, son las que se ejecutan sobre la interfaz del software. Mediante el uso de estas se examinan todas las funcionalidades. Estas tienen poca relación con el comportamiento interno del software (Pressman, 2005). Estas pruebas, permiten demostrar que las funciones del sistema, sean operativas; que la entrada se acepta de forma adecuada y que se produce una salida correcta.

### 4.5.2 Estrategia de prueba seguida

La estrategia seguida para la realización de las pruebas al componente de búsqueda desarrollado para el sistema RBC, comprende pruebas en los 4 niveles definidos por Roger Pressman (Pressman, 2005).

- **Pruebas unitarias:** Son pruebas de caja blanca que se realizan con el objetivo de detectar errores de implementación en el componente desarrollado. Además, se identifican errores de entrada y salida de datos.

- **Pruebas de integración:** Verifican que cada componente desarrollado no presente errores cuando se integre con los demás.
- **Pruebas de validación:** Son pruebas de caja negra que se enfocan en la satisfacción de las necesidades del cliente, verificando las acciones que el usuario realiza en el sistema y la correcta entrada y salida de datos.
- **Pruebas del sistema:** Son pruebas que confirman el correcto funcionamiento de las funciones desarrolladas.

**Pruebas unitarias**

Para comprobar el correcto funcionamiento de la implementación del componente desarrollado, se realizaron pruebas unitarias. Se utilizó la biblioteca de Python (unit testing) y la definición de las pruebas se realizaron dentro del archivo tests.py, creado por el marco de trabajo Django. Para ejecutar los test o iniciar el servidor de prueba se hace uso del comando python manage.py test.

Se realizaron pruebas unitarias en models a la clase fichero y fichero_txt, y en el views en los métodos vectores, borrar_caracteres y limpiar.

**Pruebas de caja blanca**

```
class Test_models(unittest.TestCase):
    def test_fichero(self):...

    def test_Ficheros_TXT(self):...


class Test_views(unittest.TestCase):
    def test_vectores(self):...

    def test_borrar_caracteres(self):...

    def test_Limpiar(self):...
```

```
Creating test database for alias 'default'...
.....
----------------------------------------------------------------------
Ran 5 tests in 0.003s

OK
Destroying test database for alias 'default'...
```

**Pruebas de validación**

Las pruebas de validación no son más que pruebas de caja negra. Donde se verifica la entrada y salida de los datos y se realizan en el sistema a través de casos de prueba. La técnica utilizada para comprobar el correcto funcionamiento del sistema fue partición equivalente.

Para la realizar esta técnica se entraron datos válidos y no válidos, la cual se comprobó en el escenario de Agrupar Documentos.

**Pruebas de caja negra**

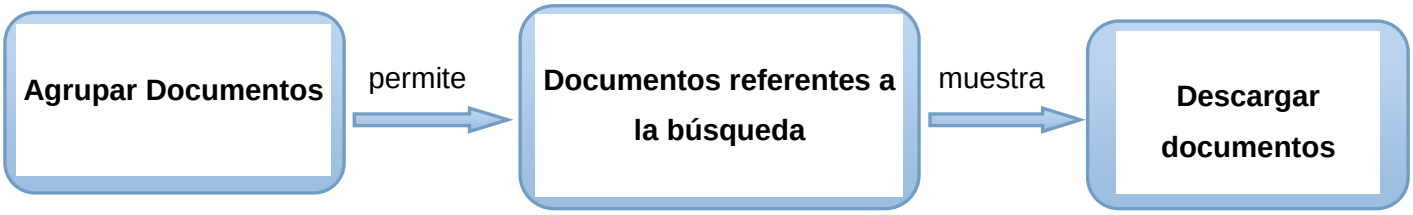



***Ilustración 18: Gráfico correspondiente al caso de prueba del RF3 Descargar Documento.***

Tabla3: Caso de prueba de partición equivalente del RF1 Realizar búsqueda

| **Escenario** | **Descripción** | **Datos** | **Respuesta del sistema** | **Flujo central** |
|---|---|---|---|---|
| EC 1<br>Agrupar documentos. | El usuario introduce las palabras claves con la que desea realizar la búsqueda. | V<br>Cadena de caracteres | El sistema muestra todas los documentos que cumplan con el criterio de búsqueda. | El usuario realiza la búsqueda, las palabras claves y obtiene los documentos que se asemejan con el criterio de búsqueda |

## Pruebas de Despliegue

***Tabla 4:*** *Prueba de despliegue*

| **No** | **Hardware** | | |
|---|---|---|---|
| **1** | Procesador: Dual Core | HDD: 320GB | RAM: 512MB |
| **2** | Procesador: i3 | HDD: 500GB | RAM: 2GB |
| **3** | Procesador: i5 | HDD: 500GB | RAM: 4GB |
| **4** | Procesador: i7 | HDD: 1T | RAM: 8GB |

| **Entorno a operar** | | | | |
|---|---|---|---|---|
| **Navegador web** | **Sistema Operativo** | **Hardware** | **Resultado** | **Problemas** |
| Mozilla Firefox<br>Google Chrome<br>Opera | Linux | 1 | Aceptado | No |
| | | 2,3,4 | Aceptado | No |
| | Windows 8.1 | 1 | Aceptado | No |
| | | 2,3,4 | Aceptado | No |
| | Windows 10 | 1 | Aceptado | No |
| | | 2,3,4 | Aceptado | No |

***Tabla 4:*** Resultado de pruebas

| **Pruebas realizadas** | **Iteración** | **No conformidades** |
|---|---|---|
| Realizar búsqueda | Iteración 1 | Cuando el usuario introduce un valor en el campo no devuelve ningún resultado. |
| | Iteración 2 | Cuando el usuario no introduce nada el sistema devuelve todos los resultados |
| | Iteración 3 | Desarrollada satisfactoriamente. |
| Mostrar Documentos | Iteración 1 | Cuando el usuario da click en el encima de algún documento no se accede al mismo |

| | | |
|---|---|---|
| | Iteración 2 | Desarrollada satisfactoriamente. |
| Descargar Documentos | Iteración 1 | Desarrollada satisfactoriamente. |

## 4.6 Conclusiones parciales

En el presente capítulo se explicó la implementación de las nuevas funcionalidades de del sistema RBC, mediante los diagramas de componente del sistema y el diagrama de despliegue. Se realizaron pruebas unitarias a la implementación del código, pruebas de validación al sistema y de despliegue para un correcto funcionamiento. Además se documentó el resultado de estrategias que garantizan la calidad del desarrollo del componente para el sistema RBC.

# Conclusiones Generales

Una vez finalizada la fundamentación teórica que sustentó la presente investigación, definidas las características de la propuesta de solución y efectuando su desarrollo y validación, se obtuvieron resultados que permiten arribar a las siguientes conclusiones:

- Resultó el algoritmo CStar el candidato para implementar la solución propuesta.
- La selección de las herramientas, tecnologías y metodologías de desarrollo contribuyó a agilizar la implementación del componente para el sistema RBC.
- Mediante el algoritmo CStar se logró la agrupación de los documentos de acuerdo a su similitud.
- El algoritmo se ejecuta correctamente con altos valores de precisión en los grupos formados.
- Las pruebas realizadas arrojaron resultados satisfactorios, pues se identificaron un grupo de no conformidades que fueron corregidas, lo que posibilitó la verificación y validación de las funcionalidades del componente desarrollado para el sistema RBC.

# Recomendaciones

Una vez concluido el desarrollo del componente de búsqueda para el sistema RBC utilizando la recomendación por similitud de objetos, y luego de haber cumplido los objetivos de la presente investigación, se recomienda:

- Implementar una variante multi-hilo para procesamiento en paralelo del algoritmo seleccionado o su factibilidad para procesamiento multi-core o contra gpu.
- Indexar los documentos en la base de datos mediante sus palabras claves y/o relevantes para facilitar el proceso de búsqueda.
- Almacenar los documentos en la base de datos como un vector pre-procesado, lo que facilitaría la aplicación de los algoritmos de agrupamiento del sistema RBC.
- Estudio de la factibilidad de almacenar los resultados de las búsquedas realizadas en bases de datos NoSQL para aumentar la rapidez de respuesta ante futuras consultas similares.

# Bibliografía


1. 2010. Una introducción a APACHE. [En línea] 2010. http://linux.ciberaula.com/articulo/linux_apache_intro/..
2. Airel Pérez Súarez, Gail García Delgado, José E. Medina Pagola, José Fco. Martínez. 2008. Algoritmos de agrupamiento para colecciones. México : s.n., 2008.
3. C.G, Figuerola y J.L.A. 2007. Algunas Técnicas de Clasificación. s.l. : Universidad de Salamanca, 2007.
4. Castellano. 2009. Exactitud y precisión. [En línea] 2009. ttp://www.upaep.cesat.com.mx/index.php?option=com_content&view=article&id=28:exactitud-yprecision&catid=11:metrologia&Itemid=14..
5. Cervantes, Dr.Humberto. 2010. Arquitectura de Software. Arquitectura. [En línea] 2010. http://sg.com.mx/content/view/922.
6. Dorian, Pyle. 1999. Data Preparation for Data Mining. 1999.
7. Ervin y Cordero Flores, Jorge Luis. Metodologías Ágiles, Proceso Unificado Ágil (AUP). Bolivia : s.n. 2010
8. F. R. A, BORDIGNON. 2007. Vol.6, ISBN 1856-4194. RECUPERACIÓN DE INFORMACIÓN: UN ÁREA DE INVESTIGACIÓN EN CRECIMIENTO. Argentina.
9. F.G, RODRÍGUEZ. Experto en Drupal 7 Nivel Avanzado. s.l. : S.L., F. 2012, ISBN.
10. Francisco Refugio Zavala, Hernández. 2014. Buscador de artículos científicos aplicando. Mexico : s.n : s.n., 2014.
11. Galindo, Raúl Miguel Romero. 2012. Análisis, diseño e implementación de un sistema de. Perú : s.n., 2012.
12. Godoy, Daniela. 2015. Minería de Datos Web. [En línea] 2015. http://www.exa.unicen.edu.ar/catedras/ageinweb.
13. González, Carlos Caballero. 2007. Desarrollo de software con calidad para una empresa. 2007.
14. Guijarro, Álvaro Primo. 2012. Protocolo HTTP. 2012.

15. Ian Sommerville. 2005. Ingeniería del Software Séptima edición. Madrid : s.n., 2005.

16. J.A, Aslam y E, Pelekhov. 2006. The Star Clustering Algorithm for Information. 2006.

17. Jovanovi´c, Irena. 2008. Software testing methods and techniques. 2008.

18. Juan Miguel, Moine. 2013. Metodologías para el descubrimiento de conocimiento en bases. 2013.

19. L Arco, R Bello. 2006. Agrupamiento de Documentos Textuales mediante Métodos. Villa Clara : s.n., 2006.

20. L, Arco.R,Bello. 2009. Agrupamiento de Documentos Textuales mediante Métodos. 2009.

21. Larman, Craig. 1999. UML y Patrones. Introducción al análisis y diseño orientado a objetos. México : s.n., 1999. ISBN 970-1 7-0261-1

22. Marelys Martínez Moreira, Alberto Boza García. junio 2016. Sistema de descubrimiento de bibliografía científica. La Habana : s.n., junio 2016.

23. Merseguer, Jose. 2010. Diagramas de casos de uso. Zaragoza : s.n., 2010.

24. PostgreSQL. 2013. The world's most advanced open source database. 2013.

25. Pressman, Roger S. 2005. Ingeniería del software. Un enfoque práctico. Sexta edition. 2005.

26. Ricardo Baeza-Yates y Benito Ribeiro-Neto. Modern Information Retrieval. 2009.

27. R. O. L, SEDEÑO. 2010,vol. 3. Herramientas para un observatorio de información. Granma. [En línea] 2010,vol. 3. http://publicaciones.uci.cu/index.php/SC..

28. Rumbaugh, James, Jacobson, Ivar y Booch, Grady. El Lenguaje Unificado de Modelado.

29. S.A, SOLUS. 2000-2016. Spark Systems. [En línea] 2000-2016. http://www.sparxsystems.com.ar/resources/tutorial/uml2_deploymntdiagram.html.

30. Salgueiro, Armando de Jesús Plasencia. METODOLOGÍA PARA LA REALIZACIÓN DE PROYECTOS DE MINERÍA. La Habana : s.n. 2012

31. Santillán, L.A.C y Ginestá, M.G. 2010. Bases de datos en MySQL. [En línea] 2010. http://ocw.uoc.edu/computer-science-technology-and-multimedia/bases-de-datos/basesde-datos/P06_M2109_02151.pdf..

32. Schmuller, Joseph. Aprendiendo UML en 24 horas. México : Editorial División Computación. 2004

33. Sergio Santamaria, Torres. 2014. Sistema de descubrimiento de bibliografía científica. 2014.

34. Sinnexus. 2007. [En línea] 2007. http://www.sinnexus.com/empresa/index.aspx..

35. Software, Ingeniería de. 2005. Introducción al Modelo Conceptual. 2005.

36. Tedeschi, Nicolás. 2010. ¿Qué es un patrón de diseño? [En línea] 2010. http://msdn.microsoft.com/es-es/library/bb972240.aspx.

37. —. 2010. ¿Qué es un patrón de diseño? [En línea] 2010. [Citado el: 5 de Abril de 2016.] http://msdn.microsoft.com/es-es/library/bb972240.aspx.

38. Vásquez Cortez, Augusto,Huerta Vega y Quispe Patriona,Jaime. 2009. Procesamiento de lenguaje natural. s.l. : Universidad Nacional Mayor de San Marcos : s.n, 2009.