UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

FACULTAD 1

TRABAJO DE DIPLOMA PARA OPTAR POR EL TÍTULO DE INGENIERO EN CIENCIAS INFORMÁTICAS

COMPONENTE PARA LA BÚSQUEDA POR IMÁGENES EN LA PLATAFORMA C.U.B.A.

Autor:
Heikel Andrés Molina González

Tutores:
Ing. Yuneldis Reyes Velázquez
Ing. Eyeris Rodríguez Rueda

A mis padres, por su cariño y su ejemplo.

Declaración de autoría

Declaro por este medio que yo, Heikel Andrés Molina González, con carné de identidad 94020134102, soy el autor principal del trabajo titulado **“Componente para la búsqueda por imágenes en la plataforma C.U.B.A."** y autorizo a la Universidad de las Ciencias Informáticas a hacer uso de la misma en su beneficio, así como los derechos patrimoniales con carácter exclusivo.

Declaro que todo lo anteriormente expuesto se ajusta a la verdad y asumo la responsabilidad moral y jurídica que se derive de este juramento profesional.

Y para que así conste, firmo la presente declaración de autoría en La Habana, a los ____ días del mes de _____ del año 2018.

Autor:

_____

Heikel Andrés Molina González

Tutores:

_____

Ing. Yuneldis Reyes Velázquez

_____

Ing. Eyeris Rodríguez Rueda

Resumen

El objetivo de la presente investigación es desarrollar un componente que se integre a la plataforma c.u.b.a. y que permita realizar búsqueda de imágenes similares a partir de una imagen. Esto aporta valor agregado a la plataforma y aumenta la fidelización de los usuarios al usarla. El estudio y análisis del estado del arte permitieron identificar las funcionalidades y tecnologías para el diseño y desarrollo de la solución propuesta. Todo el proceso se hizo bajo las pautas que establece la metodología AUP en su variación para la Universidad de las Ciencias Informáticas y se seleccionó como principales tecnologías el marco de trabajo *Spring*, *Java* como lenguaje de programación y el entorno integrado de desarrollo *NetBeans*. Se utilizó además *Solr* como componente de indexación y librerías como *TensorFlow* y *openCV* para el procesamiento de las imágenes. Para comprobar el correcto funcionamiento del sistema y su capacidad de responder a una alta concurrencia de usuarios se aplicaron diferentes tipos de pruebas: funcionales, integración y carga y estrés. El método *Delphi* se aplicó a la consulta de expertos, validando que la solución desarrollada tiene un alto valor para los usuarios de la plataforma c.u.b.a. y permite encontrar imágenes y su localización en la web cubana.
Palabras clave: recuperación de información, procesamiento de imágenes, sistema de recuperación de información.

Índice general

Introducción **1**

1. Fundamentos teóricos **5**
1.1. Fundamentos teóricos asociados al dominio de la investigación 5
1.1.1. Recuperación de información 5
1.1.2. Sistemas de recuperación de información 7
1.1.3. Procesamiento digital de imágenes 9
1.1.4. Recuperación de imágenes 10
1.2. Herramientas existentes que realizan búsqueda a partir de imágenes en la web 11
1.2.1. Resultado del estudio de los sistemas homólogos 13
1.3. Metodología de desarrollo de software 14
1.4. Herramientas, lenguajes y tecnologías 15
1.4.1. Indexador de información 16
1.4.2. Lenguajes de programación 17
1.4.3. Tecnologías 19

2. Análisis y diseño **20**
2.1. Descripción de la propuesta de solución 20
2.2. Modelo de dominio 21
2.3. Modelo de Casos de Uso del Sistema 23
2.3.1. Especificación de casos de uso 23
2.4. Especificación de requisitos de Software 24
2.4.1. Requisitos funcionales 25
2.4.2. Requisitos no funcionales 25
2.5. Estilo arquitectónico 26
2.6. Patrones de diseño 26

2.7. Diagrama de Clases de Diseño 28
2.8. Diagrama de interacción 29
2.9. Modelo de despliegue 30

3. Implementación y validación **32**
3.1. Modelo de componentes 32
3.1.1. Diagrama de componentes 33
3.2. Estándares de codificación 34
3.3. Validación de la propuesta de solución 35
3.3.1. Pruebas funcionales 35
3.3.2. Pruebas de integración 37
3.3.3. Pruebas de carga y estrés 38
3.3.4. Validación de la hipótesis de la investigación 39

Conclusiones **45**

Bibliografía **46**

Índice de figuras

1. Etapas del proceso de RI 2

1.1. Componentes de un SRI. 8
1.2. Etapas del procesamiento digital de imágenes. 9

2.1. Descripción de la propuesta de solución 21
2.2. Modelo del dominio 22
2.3. Caso de uso inicializados por el Usuario 23
2.4. Especificación de caso de uso 24
2.5. Creación de objetos 27
2.6. Diagrama de clases 29
2.7. Diagrama de secuencia 30
2.8. Diagrama de despliegue 31

3.1. Diagrama de componentes 34
3.2. Descripción de las variables para el Caso de Prueba 1 35
3.3. Caso de Prueba del RF1. Cargar imagen 36
3.4. Caso de Prueba del RF2. Extraer características de la imagen 36
3.5. Resultado de las pruebas funcionales 37
3.6. Caso de Prueba del RF3. Comparar imagen 38
3.7. Caso de Prueba del RF4. Mostrar imágenes similares 38

Índice de tablas

1.1. Resumen del análisis de los sistemas homólogos. 14

2.1. Asignación de responsabilidades 27

3.1. Resultados de prueba de carga y estrés 39
3.2. Nivel de conocimiento de posibles expertos 40
3.3. Coeficiente de Argumentación o Fundamentación 41
3.4. Niveles de competencia 42
3.5. Expertos seleccionados 42
3.6. Resultado de la encuesta realizada 43
3.7. Frecuencia acumulada de los datos primarios 43
3.8. Frecuencia relativa de los datos primarios 44
3.9. Imagen de la frecuencia relativa acumulativa 44

Introducción

Con la llegada de Internet y la aparición de bibliotecas virtuales, redes sociales, revistas electrónicas y otro gran cúmulo de información se hizo necesario crear herramientas que facilitaran la localización y recuperación de la misma. Por ello se crearon los sistemas de recuperación de información. Estos son aplicaciones que dado un criterio de búsqueda que pueden ser textos, urls, documentos, localizaciones geográficas, imágenes, y otros; devuelven un resultado y su localización en la red de redes(Castells y col., 2011).
Los sistemas de recuperación de información están compuestos por tres componentes fundamentales:

1. Subsistema de recolección(Spider).
2. Subsistema de indexación.
3. Subsistema de visualización.

Estos subsistemas se encargan de encontrar todo lo existente en la red, almacenarlo y luego mostrarlo a través de interfaces o servicios según la búsqueda solicitada por los usuarios(Leyva y col., 2016). El proceso de recuperación de información tiene diferentes etapas: localizar, procesar y presentar la información recuperada al usuario(Alarcón, 2006). La etapa de localización es presenciada cuando un sistema de recolección rastrea la red en busca de información que es almacenada en forma de índices en el componente de indexación. Esta información se procesa o analiza para ser comparada a través de un cálculo de similitud y se devuelve al usuario lo que es relevante según el criterio de búsqueda utilizado. El proceso se puede visualizar en la **figura 1**.

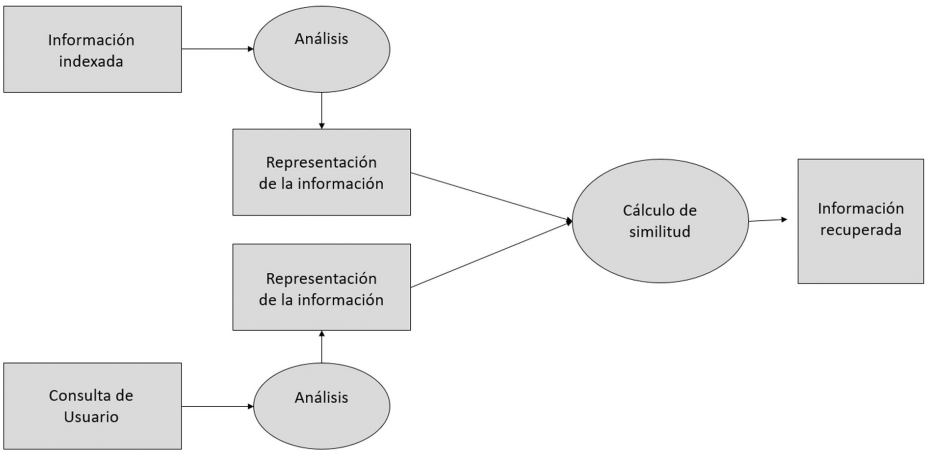

Figura 1: Etapas del proceso de Recuperación de Información. Creación propia.

Este proceso de localización puede ser engorroso si no se cuenta con herramientas que faciliten la búsqueda y permitan acceder de forma directa a lo que se necesita encontrar. Grandes empresas como Google, Microsoft con su buscador Bing y Yahoo, entre otras, han sabido satisfacer las necesidades de sus clientes y brindan un mejor servicio gracias a la búsqueda por imágenes que son introducidas por el usuario en el buscador, sin embargo estos manipulan la información y el posicionamiento de acuerdo a intereses personales o de los gobiernos a los que representan. Además, mucho de los servicios que brindan son bloqueados a Cuba, debido principalmente al bloqueo y a sanciones tecnológicas impuestas por el gobierno norteamericano(*Colocándonos en la web*, 2012)(*Ciberguerra contra Cuba: Mentiras en la red*, 2011).
Desde el año 2007 se contaba con el primer buscador cubano conocido como 2x3, fue presentado por el Stand Cuba en la XII Convención y Expo Internacional Informática 2007. Esta herramienta no tenía las características propias de un buscador, sino de un directorio y no satisfacía adecuadamente las necesidades de los usuarios(Santovenia Díaz y col., 2007). Bajo la idea de informatización de la sociedad cubana, desde el año 2013 en nuestro país se desarrolla el proyecto Orión en la Universidad de las Ciencias Informáticas, herramienta libre concebida para el trabajo con sitios cubanos y accesible desde la web nacional, que no depende de servicios brindados por buscadores privados.
El avance en la ciencia y la tecnología ha posibilitado una evolución en la concepción del buscador, conocido actualmente como plataforma de Contenidos Unificados para Búsqueda Avanzada(c.u.b.a.). Tiene una arquitectura basada en componentes y cada componente realiza consultas al subsistema de indexación y almacenan los resultados en una base de datos. Estos resultados almacenados están disponibles para cuando el usuario introduce un criterio de búsqueda, el componente lo analiza y devuelve una respuesta. La

herramienta se encuentra en funcionamiento y se encarga de buscar y visualizar la información alojada en la web cubana, conformada a las alturas de 2018 por 6883 sitios bajo el dominio .cu[1]. Aunque el buscador tiene varias funcionalidades, no dispone de una que le permita al usuario introducir una imagen y utilizarla como criterio de búsqueda para encontrar otras similares.

El hombre ha encontrado en las imágenes una forma de comunicarse y así expresar lo que es y lo que siente, incluso antes de la aparición del lenguaje articulado(Vasquez, 1997). La tendencia a usar teléfonos celulares para realizar fotografías ha permitido que la existencia de estas sea cada vez mayor, en lugares y situaciones diferentes.*"La práctica fotográfica no solo registra imágenes de lugares y situaciones, sino que también les da forma, expresando el imaginario tanto de aquello que se fotografía, como de quien lo fotografía y de quienes interpretan tal fotografía."*(Dávila Legerén, 2015)

La plataforma c.u.b.a. carece de la funcionalidad de búsqueda introduciendo un fichero gráfico ya que no se desarrollan todas las etapas del proceso de recuperación de información en las imágenes, estas son: procesar y presentar al usuario. La situación resulta en una experiencia frustrante para el usuario cuando realiza una búsqueda de información relacionada con una imagen.

Dada la situación problemática expuesta es posible definir el siguiente **problema de investigación**: ¿Cómo mejorar el proceso de recuperación de información en la búsqueda de imágenes de la plataforma c.u.b.a.?

Se define como **objeto de estudio**: el proceso de recuperación de información y como **campo de acción**: el proceso de recuperación de información de imágenes.

Se plantea el siguiente **objetivo general** para darle solución al problema: Desarrollar un componente para la plataforma c.u.b.a. que permita encontrar imágenes similares, sus datos y localización en la web, utilizando una imagen como criterio de búsqueda.

Para alcanzar lo propuesto se plantean los siguientes **objetivos específicos**:

1. Describir la referencia teórica referente al desarrollo de herramientas para la realización de búsquedas utilizando una imagen como criterio de búsqueda.
2. Diseñar las funcionalidades del componente para la plataforma c.u.b.a. que permita utilizar una imagen como criterio de búsqueda.
3. Implementar las funcionalidades del componente para la plataforma c.u.b.a. que permita utilizar una imagen como criterio de búsqueda.
4. Validar el componente para la plataforma c.u.b.a. que permita utilizar una imagen como criterio de búsqueda.

Con posterioridad a la realización de un estudio de la literatura y desarrollado el marco teórico se formula la siguiente **hipótesis de investigación**: el desarrollo de un componente para la búsqueda por imágenes en

[1]Según el Centro Cubano de Información de Red, disponible en http://nic.cuba.cu/estadisticas.php.

la plataforma c.u.b.a., mejorando el proceso de recuperación de información, permitirá encontrar imágenes similares, sus datos y localización en la web cubana.
Para la realización de la investigación se ha utilizado un grupo de métodos teóricos y empíricos de la investigación científica, agrupados en dos niveles:
Teóricos:

- Analítico-Sintético: El método fue empleado para analizar el estado del arte de los procesos de búsqueda de imágenes y de esta forma obtener conocimiento procediendo a sintetizarlo.
- Histórico-lógico: Se realiza una revisión exhaustiva del desarrollo evolutivo del objeto de investigación a lo largo del tiempo con el objetivo de definir las limitaciones actuales del conocimiento: El método permitió reconocer los avances teórico-prácticos y problemas que actualmente existen en el área de investigación tratada.
- Hipotético-deductivo: Se establecen una serie de verdades supuestas que han sido comprobadas o rechazadas dando como resultado una hipótesis de investigación.

Empíricos:

- Modelación: Es empleado en la representación mediante diagramas de las características, procesos y componentes del sistema propuesto, así como la relación existente entre ellos.
- Simulación: Se realiza una aproximación a la realidad del fenómeno dado. Esto significa que para emular dicho fenómeno es necesario un conjunto de datos que si bien no son reales se aproximan a los reales. Este método es utilizado para realizar las pruebas de la implementación del componente resultante.

En función de orientar al lector, el presente trabajo se encuentra estructurado de la siguiente forma:

1. **Fundamentos teóricos.** Se puntualizan los principales conceptos en torno al dominio de la investigación y se realiza un estudio del estado del arte de las soluciones existentes. Se describen las herramientas, metodologías y técnicas utilizadas para dar solución al problema planteado.
2. **Análisis y diseño.** Se centra en el desarrollo de la investigación. Se describe de forma general la solución propuesta y su funcionamiento.
3. **Implementación y validación.** Se detalla la propuesta de solución al problema planteado. Se realizan las estrategias de prueba definidas y se valida la hipótesis.

Capítulo 1

Fundamentos teóricos de las búsquedas de imágenes a partir de imágenes introducidas en el buscador.

Con el objetivo de contrastar el panorama del conocimiento actual sobre el objeto de estudio expresado, se exponen y analizan las teorías, conceptos y antecedentes relacionados con el proceso de búsqueda de imágenes a partir de una imagen como criterio de búsqueda en los SRI.

1.1. Fundamentos teóricos asociados al dominio de la investigación

La teoría, antes de ser comprobada, consiste en una hipótesis, una formulación provisional que establece cómo se comportan ciertas variables. La hipótesis nos obliga a verificar si la realidad se comporta conforme dice la teoría(Del Cid y col., 2011). A continuación se mencionan y relacionan una serie de conceptos que permitirán entender la teoría enunciada.

1.1.1. Recuperación de información

Internet es una de las fuentes de información más grande que se conoce, constantemente crece y se genera gran cantidad de esta en diferentes formatos haciéndose impreciso un cálculo estimado de su tamaño(Martínez Espadas, 2015)(*Internet Live Stats*, 2018). Ante este crecimiento los usuarios no encuentran lo que buscan de forma fácil y eficiente. Por esto se hace necesario enfrentar la desorganización de la información almacenada en internet, para que sea accesible y localizable(Kobayashi y col., 2000).

Recuperación de información(RI) es un término que describe el proceso que hace posible filtrar la gran cantidad de información disponible y que el usuario solo acceda a la que es relevante para él y deseche la que resulte irrelevante. Puede definirse como el problema de selección, luego de analizar un documento e identificada la necesidad de información, para realizar una comparación y obtener resultados satisfactorios(Marchionini, 1997)(Díaz, 2002)(Salton, 1989). Este no es un término nuevo y viene desarrollándose desde principios del siglo pasado cuando en 1920 Emanuel Goldberg patentó un aparato mecánico que realizaba consultas sobre un catálogo almacenado en un rollo de microfilm[1](Eliot y col., 2009). Sin embargo, no es hasta la década de 1950 que comienzan a aparecer las primeras técnicas de recuperación de información con el auge de dos nuevos conocimientos: cómo indexar documentos y cómo recuperarlos(Sanderson y col., 2012).

En la actualidad la RI ocupa un rol más importante debido a las características más relevantes de la información: su valor y la necesidad de disponer de ella en el momento en que se solicite(Hechevarría-Kindelán, 2002). Varios algoritmos han surgido para mejorar las diferentes técnicas de recuperación de información, de selección de documentos y de la obtención del nivel de relevancia de los mismos. Aunque existen novedosas técnicas basadas en inteligencia artificial: redes neuronales (RN), algoritmos genéticos (AG), procesamiento del lenguaje natural (PLN), los modelos clásicos de RI son: modelo Booleano, modelo Espacio Vectorial, modelo Probabilístico y modelo Booleano Extendido o modelo Difuso(L. G. Jaimes y col., 2005)(Cacheda, 2008).

Modelo Booleano

Este se considera el primer modelo teórico, se basa en la teoría del álgebra de Bole[2] y trata con proposiciones, elementos de circuitos de dos estados, etc., asociados por medio de operadores como **AND**, **OR**, **IF**... **THEN**. Muchas herramientas de búsqueda en la red se basan en este modelo por ser de desarrollo sencillo. Fue introducida en 1847 por George Boole y se le conoce así en honor al famoso matemático(Amati y col., 2002).

La ventaja del modelo Booleano radica en el hecho de que es simple, basado en el Álgebra de Boole, lo que da un marco teórico sólido. Su principal desventaja es el criterio de recuperación binario que suele resultar estricto y definitivo. Algunos lo consideran mayormente como un sistema de recuperación de datos y no como uno de información(Cacheda, 2008).

[1] Un microfilm es una película en la cual se copian principalmente documentos o manuscritos pero, al ser de tamaño pequeño, permite poder almacenar en poco espacio grandes cantidades de rollos y ampliarlas posteriormente para su uso, ya sea imprimiéndolas, proyectándolas o escaneándolas.

[2] Se denomina Álgebra de Bole o Álgebra Booleana a las reglas algebraicas, basadas en la teoría de conjuntos, para manejar ecuaciones de lógica matemática

Modelo Espacio Vectorial

El primero en proponer un SRI basado en Espacio Vectorial fue Salton, dentro del marco del proyecto SMART a finales de los 1960(Salton y McGill, 1986). La idea es que se pueden representar los documentos como vectores de términos y podrán situarse en un espacio vectorial de *n* dimensiones, por lo que se pueden representar tantas dimensiones como elementos tenga el vector. Situado en ese espacio vectorial, cada documento cae entonces en un lugar determinado por sus coordenadas[3]. Creándose grupos de documentos que quedan próximos entre sí a causa de las características de sus vectores. Estos grupos están formados en teoría por documentos similares, de forma que serían relevantes para la misma necesidad de información(Salton y McGill, 1986).
La consulta, cuando es formulada una pregunta, también ocupa un lugar en este espacio vectorial siendo más relevantes para la misma los elementos que queden próximos a ella. La representación de los documentos y las consultas se realiza mediante la asociación de un vector de peso no binario(un peso por cada término de índice), por ejemplo: $d_i = (t_{i1}, t_{i2}, t_{i3}..., t_{in})$.
Este sistema está dotado de una gran potencialidad gracias a que tanto los documentos como las consultas tienen la misma representación(Comeche, 2006).

Modelo Probabilístico

El modelo probabilístico está compuesto por conjuntos de variables, operaciones con probabilidades y el Teorema de Bayes.
Está basado en el llamado "Principio de la ordenación por probabilidad". Este principio, formulado por Robertson, asegura que el rendimiento óptimo de la recuperación se consigue ordenando los documentos según sus probabilidades de ser juzgados relevantes con respecto a una consulta, siendo estas probabilidades calculadas de la forma más precisa posible a partir de la información disponible(Robertson, 1977).
Este modelo es capaz de ordenar los documentos según su posibilidad de ser juzgados como relevantes con respecto a una consulta, calculando estas probabilidades de la forma más precisa posible a partir de la información que se tenga(Van Rijsbergen, 1986).

1.1.2. Sistemas de recuperación de información

Los Sistemas de Recuperación de Información(SRI) o buscadores, son una fuente de acceso a la información que se encuentra distribuida en el mundo de la web, así como a los servicios que esta brinda. La importancia de estos sistemas está determinada por su capacidad de rastrear la información para luego de

[3]Al igual que en un espacio de tres dimensiones cada objeto queda bien ubicado si se especifican sus tres coordenadas espaciales

ser almacenada en forma de índices, permitir su acceso de acuerdo al nivel de relevancia que se le asigne, correspondiendo con los distintos criterios de búsqueda.

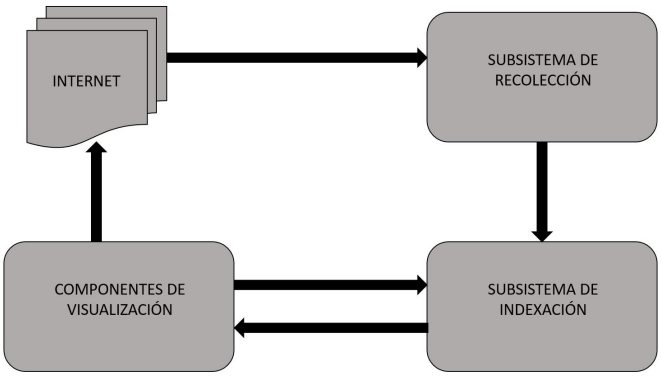

Figura 1.1: Componentes principales de un Sistema de Recuperación de Información. Creación propia.

Como muestra la **figura 1.1**, los SRI están compuestos generalmente por tres componentes fundamentales y la interacción de los mismos permiten el rastreo de toda la web, el resultado de este proceso es almacenado para luego ser mostrado a los internautas a través de interfaces o servicios en respuesta a los criterios de búsqueda definidos(Leyva y col., 2016).

Mecanismo de rastreo

El mecanismo de rastreo o *spider* web se utiliza para de forma metódica y automatizada inspeccionar todas las páginas de internet siguiendo su estructura hipertextual. La información extraída de la búsqueda es almacenada para su posterior análisis.

Mecanismo de indexación

El mecanismo de indexación es el encargado de recepcionar la información rastreada por el *spider* y luego procesarla y almacenarla(Cleverdon, 1997). Suele asignarse palabras claves o frases a cada página web y son guardados en campos de meta-etiquetas. Estos campos pueden ser añadidos de forma personalizada y depende de lo que el mecanismo de rastreo sea capaz de identificar en cada lugar que recolecta información.

Componente de visualización

El componente de visualización es usado por los usuarios con necesidades de información para interactuar con el sistema, posibilita introducir consultas y especificaciones de lo que desea encontrar. Puede ser una

interfaz de escritorio o página web, así como la combinación de ambas.

Los SRI pueden clasificarse en directorios, metabuscadores y buscadores si se analiza su forma de operar, alcance que poseen y tipo de documento que recuperan. La implementación mas idónea de sistemas autónomos dedicados a la recopilación, procesamiento y recuperación de grandes volúmenes de información pudiera aplicarse a uno clasificado como buscador(Kuna y col., 2014).

1.1.3. Procesamiento digital de imágenes

El procesamiento digital de imágenes como se muestra en la **figura 1.2**, sigue una serie lógica de etapas que son necesarias para obtener resultados deseados. Este proceso se inicia con la etapa de adquisición de imágenes, la siguiente etapa es el preprocesamiento, que se realiza con el fin de detectar y eliminar las fallas que puedan existir en la imagen para mejorarla. Las técnicas más utilizadas en esta etapa son: a) mejora del contraste, b) eliminar el ruido, y c) restauración. En la etapa de segmentación, la imagen se divide en sus partes constituyentes u objetos, con el fin de separar las partes necesarias de procesamiento del resto. Las técnicas básicas en esta etapa son aquellas orientadas a: a) el píxel, b) a los bordes, y c) a las regiones. Sin embargo, las técnicas no son excluyentes sino que se combinan de acuerdo al tipo de aplicación(Palomino y col., 2009).

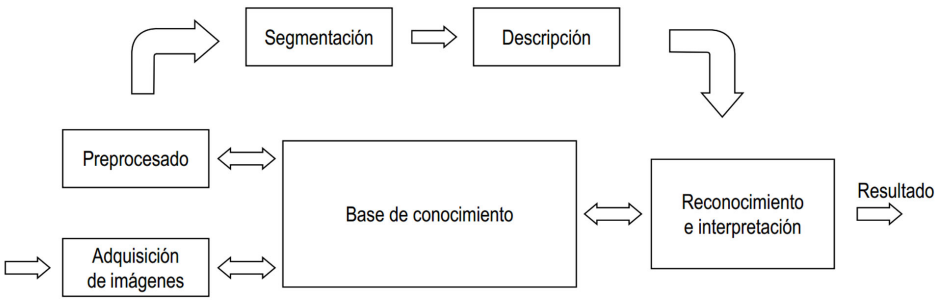

Figura 1.2: Etapas del procesamiento digital de imágenes. Palomino y col., 2009

La etapa de descripción o extracción de características consiste en extraer las que contengan alguna información cualitativa de interés o que sean fundamentales para diferenciar una clase de objetos de otra. La

etapa de reconocimiento es cuando se asigna una etiqueta a un objeto basándose en la información que proporcionen sus descriptores, esto implica dotar de significado al conjunto de objetos reconocidos.
En el presente trabajo solo se abordarán las etapas de segmentación, descripción y reconocimiento e interpretación, ya que no se realizan capturas de imágenes con medios electrónicos y no se modifican estas para mejorar su calidad o para resaltar detalles que interesen.

1.1.4. Recuperación de imágenes

Los seres humanos como individuos son capaces de analizar la similitud entre imágenes, aunque no manejen criterios generales o universales para hacerlo. Por lo tanto las decisiones o conclusiones que una persona toma puede diferir en otras y no se puede definir qué opinión es la más adecuada. Debido a esto, es altamente complicado para un sistema computacional emular una aproximación de lo que la razón humana puede ser y dejar a los usuarios satisfechos. Para que este proceso se realice de forma efectiva se han propuesto varias técnicas que trabajan a distintos niveles(López, 2007):

Técnicas de bajo nivel

Suelen conocerse como técnicas de Recuperación de Imágenes Basada en Contenidos (CBIR: *Content-Based Image Retriveral*). Se agrupan varias técnicas clásicas y modernas de tratamiento de imágenes, reconocimiento de patrones y recuperación de información; tales como segmentación, clasificación, extracción de características, reducción de información, entre otras. Estas pretenden ser independientes del dominio de la aplicación final(Ortega Gonzalez, 2008).
La semejanza entre 2 imágenes x_i y x_j se puede medir utilizando una función de semejanza $d(f_i, f_j)$ que describa la distancia entre los vectores de características f_i y f_i. La elección de la función de semejanza es crítica y dependiente del dominio. El problema de recuperación se puede plantear como sigue: dada una imagen de consulta q recuperar un subconjunto M de imágenes de la BD de imágenes X, $M \subset X$ tal que:
$d(T(q), T(m)) \leq t, m \in M$
donde t es un umbral definido por el usuario(Yoo y col., 2002).

Técnicas de alto nivel

Realizan los procedimientos de comparación ocupando entidades que no son propiamente las imágenes, sino objetos que pretenden describir el significado, la importancia, el contenido y el papel de sus elementos visuales(Ortega Gonzalez, 2008). Muchas de las desventajas del empleo de estas técnicas son evidentes, tales como la necesidad de participación humana especializada en la construcción de las entidades descriptoras y no todos los individuos manejan los mismos criterios. Se usan principalmente para forzar o auxiliar

al sistema y ofrecer resultados más apegados a la percepción del usuario. Esto siempre se cumple si es el propio usuario o personas cercana en criterios a él, quién ha participado en la construcción de las entidades descriptoras.

Utilizando descriptores

Se incorporan descriptores o anotaciones textuales que pueden ser globales o locales en el momento de la indexación(Chen, 2005).

Utilizando ontologías

Se construyen estructuras ontológicas para describir el contenido semántico de las imágenes. Al realizar la búsqueda se comparan estos vectores con los almacenados y clasificados previamente en la estructura ontológica y en la base de datos de imágenes indexadas. Así se obtienen las similares a las de entrada(L. y col., 2007).
Una propuesta muy interesante es la combinación de varias técnicas que trabajan a distintos niveles, intentando aproximarse lo más posible a resultados esperados por los usuarios.

1.2. Herramientas existentes que realizan búsqueda a partir de imágenes en la web

Existen en la actualidad varias herramientas en la red de redes que realizan búsqueda de imágenes a partir de una imagen introducida en el buscador por el usuario. A continuación se examinan algunas de las más conocidas y utilizadas.

Google

Google Inc. es una empresa multinacional que brinda servicios relacionados con internet, software, dispositivos móviles y otras tecnologías. Su motor de búsqueda ha sido desde 1999 uno de los más recomendables(López Córdova y col., 2015). La búsqueda de imágenes de Google permite introducir una imagen y realizar una consulta a partir de esta, mostrando resultados semejantes. La interfaz es intuitiva y amigable y además de mostrar un icono que permite la carga del archivo de referencia de forma local o desde una url[4] en internet, cuenta con la opción de arrastra y suelta (del inglés *drag and drop*). Los resultados muestran una recopilación de las imágenes con similitudes en cuanto a color, textos, reconocimiento de caras y objetos

[4]Localizador de recursos uniforme por su siglas es inglés. Es la dirección o localizador de un recurso en la red.

entre otros. Tiene la opción de agregar etiquetas en forma de texto que permiten hacer una aproximación más exacta de lo que se quiere encontrar. El buscador combina las técnicas de recuperación de imágenes a bajo y alto nivel. Compara el color predominante, tamaño y textura, reconoce personas, rostros y objetos en las imágenes.

Bing

Bing es el buscador web de Microsoft anteriormente conocido como Live Search, Windows Live Search y MSN Search. Permite realizar búsqueda de imágenes a partir de imágenes introducidas en el buscador y cuenta con las opciones de cargar un archivo desde internet por su url o de forma local, este no cuenta con la opción arrastra y suelta. Los resultados recogen las imágenes similares en cuanto a color predominante, textura, textos, objetos y caras. Además se pueden agregar etiquetas en forma de texto lo que posibilita una aproximación más exacta de lo que se quiere encontrar.

Yandex

Yandex es una empresa multinacional de tecnología rusa especializada en servicios y productos relacionados con Internet. El buscador que cuenta con una interfaz intuitiva y fácil de usar que permite la selección de imágenes de referencia tanto desde internet a través de su url o desde el ordenador; cuenta con la opción de arrastrar y soltar. Los resultados muestran imágenes relacionadas en cuanto a color predominante, tamaño, objetos y texto que contengan. No permite introducir etiquetas alternativas en forma de texto para mejorar los resultados de la búsqueda.

Otros

Existen otras herramientas en la red que se especializan únicamente en la búsqueda de imágenes, alguna de ellas buscan características específicas como son la textura y el color mientras otras localizan imágenes idénticas[5].

TinEye

Se puede buscar por imagen, realizando lo que llamamos una búsqueda inversa. Puede hacerse subiendo el archivo o buscando por url. No tiene la opción arrastra y suelta desde el ordenador pero permite arrastrar una imagen desde otra página que esté abierta. El sistema busca parecidos que de alguna forma pueden haber sido alterados en comparación con la imagen original(*TinEye Reverse image search*, 2018). Esta herramienta realiza la búsqueda utilizando técnicas de bajo nivel y no busca imágenes similares sino idénticas.

ImageBrief

[5]Estas pueden estar alteradas de alguna manera.

El propósito principal es facilitar el intercambio de fotos de calidad, sin embargo, como añadido, ImageBrief permite buscar imágenes con imágenes de la misma manera que Google Imágenes, pulsando en el icono en forma de cámara de fotos. El objetivo es el de encontrar fotografías o ilustraciones similares o de mayor calidad que las que se encuentran en internet y que se pueden comprar para propósitos comerciales o profesionales(*ImageBrief Similar image search*, 2018). Usa técnicas de bajo nivel y busca exactitud en el parecido de la imagen usada como criterio y el resultado.

1.2.1. Resultado del estudio de los sistemas homólogos

Luego del estudio realizado de los sistemas existentes que realizan búsqueda de imágenes a partir de imágenes introducidas por el usuario en el buscador, se comparan estos en la **tabla 1.1** para seleccionar las que le den solución de manera apropiada al problema de investigación. Se han seleccionado los siguientes parámetros de comparación:

- Búsqueda de imágenes idénticas (BII).
- Búsqueda de imágenes similares (BIS).
- Detección de personas (DP).
- Detección de rostros (DR).
- Detección de objetos (DO).
- Detección de transparencia (DT).
- Comparación de color predominante (CCP).
- Comparación por tamaño(alto y ancho) (CT).

Tabla 1.1: Resumen del análisis de los sistemas homólogos.

Característica	Google	Bing	Yandex	TinEye	ImageBrief
BII	Sí	Sí	Sí	Sí	Sí
BIS	Sí	Sí	Sí	No	Sí
DP	Sí	Sí	Sí	No	No
DR	Sí	Sí	Sí	No	No
DO	Sí	Sí	Sí	No	No
DT	Sí	Sí	Sí	Sí	Sí
CCP	Sí	Sí	Sí	No	Sí
CT	Sí	Sí	Sí	Sí	Sí

Luego del análisis antes expuesto y las características que se han analizado en las herramientas existentes se puede concluir que:

1. El análisis del color predominante, la detección de rostros y la detección de objetos son parámetros clave en la búsqueda de imágenes similares.
2. El uso final de la aplicación está en estrecha relación con las funcionalidades que esta posee. TinEye no analiza algunas características porque su objetivo es buscar las imágenes que sean idénticas. La propuesta de solución está orientada a encontrar imágenes similares.

Cada una de las características antes comparadas brindan una mayor panorámica del estado actual de las herramientas existentes y posibilitaron identificar que la búsqueda de imágenes idénticas, detección de personas, detección de objetos, detección de transparencia, comparación de color predominante y comparación por tamaño pueden ser usadas en el componente que se ha propuesto como solución planteada en la investigación.

1.3. Metodología de desarrollo de software

El Proceso Unificado Ágil (AUP) es una versión simplificada del Proceso Unificado de Rational (RUP). Este describe una manera simple, fácil de entender y de manera ágil la forma de desarrollar aplicaciones de software de negocio usando conceptos que aún se mantienen validos en RUP. El AUP aplica una serie de técnicas ágiles que incluye:

- Desarrollo Dirigido por Pruebas.
- Modelado ágil.
- Gestión de Cambios ágil.
- Refactorización de Base de Datos para mejorar la productividad.

Se propone una variación de la metodología AUP unida al modelo CMMI-DEV[6] v1.3 para garantizar que la actividad productiva en la UCI use las buenas prácticas en función de un software de calidad(Sánchez, 2015). c.u.b.a. se desarrolla bajo la metodología antes mencionada y por ser la propuesta de solución un componente que debe integrarse a esta, se decide usar la misma en su desarrollo. Se escoge el **escenario 2**[7] de la metodología para generar los artefactos que se exponen en el desarrollo del **Capítulo 2**.

1.4. Herramientas, lenguajes y tecnologías

Para desarrollar la propuesta de solución, surgida del estudio de los sistemas homólogos, se hace necesario el estudio de varias tecnologías, lenguajes y herramientas con el fin de cumplir el objetivo de la investigación, las mismas se exponen a continuación.

Lenguaje y herramienta de modelado

El lenguaje unificado de modelado (UML por sus siglas en inglés) es un lenguaje estándar para describir planos de software. UML se puede utilizar para visualizar, especificar, construir y documentar los artefactos de un sistema que involucra una gran cantidad de software(Ó. A. Jaimes y col., 2015).
Las herramientas CASE (Computer Aided Software Engineering) suelen inducir a sus usuarios a la correcta utilización de metodologías que implican una reducción tanto en el costo del proyecto, como en el tiempo de desarrollo del producto de software final. Para esto, es importante considerar que tipo de herramienta CASE es necesaria al momento de llevar adelante una actividad relacionada con la Ingeniería de Software(Battaglia y col., 2017).
En la Universidad de las Ciencias Informáticas (UCI) se ha estandarizado el uso del Visual Paradigm for UML en su distribución libre como herramienta CASE para el modelado de los procesos de desarrollo de software que en ella se llevan a cabo, dado por la gran cantidad de ventajas que posee, las cuales están en

[6] Integración de modelos de madurez de capacidades o Capability Maturity Model Integration es un modelo para la mejora y evaluación de procesos para el desarrollo, mantenimiento y operación de sistemas de software(Jezreel y col., 2017).

[7] Este escenario propone que si el negocio se modela con Modelo conceptual solo se puede encapsular los requisitos con Casos de Uso del Sistema.

concordancia con los intereses y políticas establecidas en la institución. Entre sus principales características se encuentran que es multiplataforma, posee interoperabilidad, facilita la colaboración en equipo y brinda apoyo al ciclo de vida completo del desarrollo de software(ROSALES, 2013).

Herramienta para el control de versiones

El control de versiones es un sistema que registra los cambios realizados sobre un archivo o varios archivos a lo largo del proceso de desarrollo, de modo que puedas recuperar versiones específicas en otro momento. Para garantizar el control de versiones y la correcta integración del subsistema propuesto como solución de la investigación se emplea el repositorio GitLab y se propone además el cliente Git. Las ventajas principales son expuestas a continuación(Paredes y col., 2015):

- Es un sistema de control de versiones distribuido.
- No depende de acceso a la red o un repositorio central.
- Está enfocado a la velocidad, uso práctico y manejo de proyectos grandes.

1.4.1. Indexador de información

Estos sistemas almacenan metadatos en una base de datos local extraídos previamente por los sistemas de recolección. Sobre esta base de datos se implementan los servicios de búsqueda y recuperación de recursos digitales.

Solr

Apache Solr es un motor de búsqueda de código abierto que proporciona potentes funcionalidades de búsqueda y explora la infracción desde diferentes perspectivas, lo que se conoce como navegación por facetas. Estas funcionalidades son difíciles de implementar sobre una base de datos relacional, por lo que se utilizan bases de datos no relacionales. Está implementado en Java y es un producto maduro que utilizan grandes empresas como Netflix. Algunas de las características de Solr se enuncian a continuación:

- Servidor con interfaz tipo REST.
- Esquema de datos configurable.
- Interface web de administración.
- Navegación de resultados por facetas.

- Escalable a varios servidores para búsquedas distribuidas.
- Módulos de importación de datos desde bases de datos, email y archivos de texto enriquecido[8].
- Análisis de texto.

El esquema en el que se combina el uso de una base de datos con Solr suele ser el adecuado para la mayoría de las aplicaciones y gracias a su interfaz de tipo REST y amplia variedad de formatos de salida, es fácil de integrar en una gran variedad de ambientes y entornos de desarrollo como Java, JavaScript, PHP, Ruby, Phython y .NET(Ramos, 2017).
Por lo anteriormente planteado y añadiendo que Solr es el mecanismo de indexación utilizado por la plataforma C.U.B.A., donde se almacena la información extraída de las imágenes de la web cubana, se ha determinado utilizarlo como mecanismo de indexación.

1.4.2. Lenguajes de programación

A continuación se exponen los lenguajes de programación que se han seleccionado para implementar la propuesta de solución al problema de investigación.

Lado del servidor

Los lenguajes de programación son lenguajes formales diseñados para la realización de procesos que pueden ser llevados a cabo por computadoras. Son un conjunto de reglas semánticas y sintácticas que se usan para la codificación de instrucciones de un programa o algoritmo(Cunalata Miranda y col., 2016).
Java
Java es un lenguaje de programación de propósito general, concurrente y orientado a objetos. Es sin dudas el lenguaje más usado en computadoras, teléfonos y televisores. Las características que distinguen Java como lenguaje robusto se citan a continuación(Lugo y col., 2016):

- Lenguaje simple. Una de las cosas más importantes de Java es que no es para nada complejo.
- Lenguaje Orientado a Objetos. Los objetos se encargan de encapsular información, clases y funciones que se pueden reutilizar en distintos programas.
- Aplicaciones distribuidas. Permite el desarrollo de aplicaciones distribuidas.
- Seguro. Es un lenguaje de código abierto que permite la creación de aplicaciones web sin necesidad de tener problemas con filtros de seguridad.

[8]Documentos con formato PDF, RTF, Word.

Lo anteriormente visto, agregando que la herramienta **Solr** está programada en Java nos ayuda a definir que este será el lenguajes a utilizar del lado del servidor.

Marco de trabajo para Java: Spring Boot

Spring Boot es un *framework* del lenguaje de programación java que se basa en el patrón Modelo-Vista-Controlador. Es el resultado de la evolución de la ingeniería del software. Se utiliza en el desarrollo de aplicaciones para estandarizar el trabajo, resolver, agilizar y manejar los problemas y complejidades que aparecen a medida que las exigencias del propio proyecto crecen. Spring comprende diversos módulos que proveen un rango de servicios(Johnson y col., 2004):

- Contenedor de inversión de control.
- Programación orientada a aspectos.
- Acceso a datos.
- Administración Remota.
- Convención sobre Configuración.
- Procesamiento por lotes.

El marco de trabajo puede ser utilizado en la construcción de servicios web, permitiendo con el uso de los servicios anteriormente listados, una aplicación robusta y funcional. Se decide hacer uso de Spring en el desarrollo de la propuesta de solución.

Bibliotecas

Diariamente avanza la tecnología y las comunidades desarrollan sus software y herramientas con el objetivo de que las tecnologías usadas estén a la par de los avances. Actualmente existen librerías, bibliotecas, módulos y *frameworks* que agilizan el desarrollo y la producción de software. Java cuenta con bibliotecas que brindan sus funcionalidades y ventajas para el trabajo y procesamiento de imágenes, alguna de ellas son: JavaAdvanceImaging, Image4J, OpenCV.
TensorFlow es una biblioteca de código abierto para aprendizaje automático a través de un rango de tareas, y desarrollado por Google para satisfacer sus necesidades de sistemas capaces de construir y entrenar redes neuronales para detectar y descifrar patrones y correlaciones, análogos al aprendizaje y razonamiento usados por los humanos. Hasta hace poco esta biblioteca era exclusiva de lenguajes como Phyton y C++ pero

recientemente desarrolladores han liberado versiones para el uso con Java, facilitando así el reconocimiento de objetos en las imágenes y su preprocesamiento(*Image Recognition.* 2018).
Para la presente investigación se hará uso de OpenCv para la detección de rostros y Tensorflow para la detección de objetos.

1.4.3. Tecnologías

Debido a que el subsistema que se propone como solución se debe integrar a la plataforma framework y luego de analizar la documentación existente, se seleccionan las mismas herramientas que fueron utilizadas para el desarrollo del buscador.

Entorno Integrado de Desarrollo

NetBeans es un entorno de desarrollo integrado libre, hecho principalmente para el lenguaje de programación Java. Existe además un número importante de módulos para extenderlo y son libres de uso. Soporta el desarrollo de todos los tipos de aplicación Java (J2SE, web, EJB y aplicaciones móviles). Entre sus características se encuentra un sistema de proyectos basado en Ant, control de versiones y refactoring(*NetBeans IDE.* 2018)(*Oracle's NetBeans IDE is the smarter and faster way to code.* 2018).

Capítulo 2

Análisis y diseño del componente para la búsqueda por imágenes en la plataforma c.u.b.a.

En el capítulo se analizan un grupo de consideraciones técnicas y de implementación referentes a la propuesta de solución para el problema de mejorar el proceso de recuperación de información de imágenes en la plataforma c.u.b.a. Se utilizan artefactos que propone la metodología AUP en su variación para la UCI en el modelado del negocio y la comprensión de las entidades, conceptos, relaciones y otros elementos relativos al ambiente en que se desenvuelve la investigación. Se presentan los patrones de diseño que se evidencian a partir de un modelado ágil. La propuesta de solución se encuentra fundamentada sobre la base de los conceptos y procesos abordados en el capítulo 1.

2.1. Descripción de la propuesta de solución

El componente propuesto es un servicio de tipo REST implementado con Spring Boot que permite al usuario a través de una petición POST enviar una imagen. Esta es procesada mediante técnicas de bajo nivel, se hace uso de bibliotecas como Tensorflow y OpenCV para la extracción de características. Luego se compara mediante una función de comparación con las que se encuentran en Solr, usando el modelo probabilístico. El componente devuelve en formato *json*, a través de una petición GET, las imágenes que resultaron seleccionadas. Este servicio es independiente del dispositivo y plataforma que lo consuma, lo que garantiza su integración con aplicaciones sin importar su arquitectura. La propuesta se muestra gráficamente en la **Figura 2.1**.

Figura 2.1: Descripción de la propuesta de solución

2.2. Modelo de dominio

Un modelo de dominio es una representación de las entidades, procesos, usuarios y sus relaciones que tienen existencia normalmente dentro del ambiente que se analiza, en este caso, para la posterior informatización de un proceso. Representa los conceptos más significativos en el sistema o la unidad organizacional que se estudia y puede constituir un medio para comprender e identificar interrelaciones comprendidas en el ámbito del dominio del problema. La metodología de desarrollo de software Rational Unfied Process (RUP), describe un modelo de dominio como *"un modelo de objeto de negocio incompleto, que se enfoca en productos explicativos, entregables, o eventos que son importantes para el dominio comercial. Tal modelo no incluye las responsabilidades que las personas tienen"*(Paul, 2002). En la **Figura 2.2** se presenta el diagrama de clases del Modelo de dominio sobre el que se trabaja en la presente investigación.

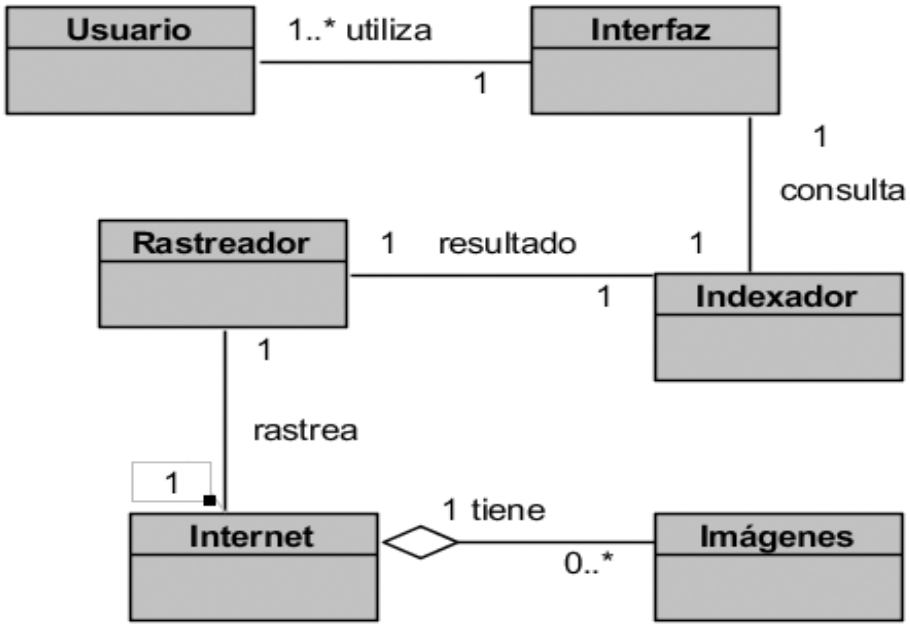

Figura 2.2: Diagrama de clases del Modelo de dominio

Conceptos del modelo de dominio

- **Usuario:** Persona que introduce un criterio de búsqueda en la aplicación web.
- **Interfaz:** Componente del motor de búsqueda encargado de recibir las peticiones de los usuarios, consultar los datos en los indexadores y mostrarle al usuario los resultados de su búsqueda.
- **Indexador:** Componente del motor de búsqueda que almacena los resultados del rastreo en forma de índice para una mejor selección de los datos solicitados por la aplicación web. Pueden existir varios componentes de indexación.
- **Rastreador:** Componente del motor de búsqueda que accede a la información pública en la red y la procesa. Pueden existir varias instancias de este componente escaneando la red.
- **Internet:** Red informática que almacena los documentos.
- **Imágenes:** Imágenes que se encuentran en internet.

2.3. Modelo de Casos de Uso del Sistema

La técnica de modelado de casos de uso se debe usar para la especificación de requisitos del sistema, no para el diseño del sistema. Las relaciones de casos de uso es opcional y no es necesaria para realizar un documento de especificación de requisitos(Portillo, 2002).

Diagrama de Caso de Uso del Sistema

En la **Figura 2.3** se muestra el caso de uso y el actor que lo inicializa.

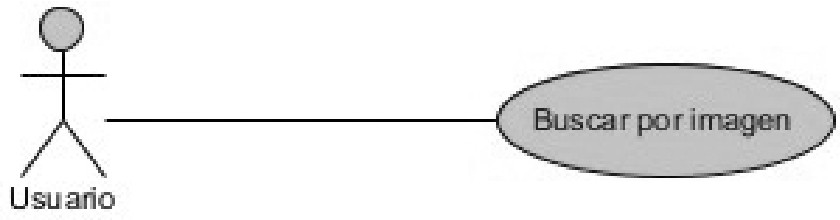

Figura 2.3: Caso de uso inicializados por el Usuario

A continuación se presenta y describe brevemente el caso de uso representado.

- **Actor Usuario:** Es el actor que interactúa con el sistema y realiza la búsqueda a partir de imágenes.
- **CU 1 Buscar por imagen:** Permite al usuario realizar una búsqueda utilizando una imagen como criterio.

2.3.1. Especificación de casos de uso

En la **Figura 2.4** se presenta la descripción del caso de uso del sistema.

Objetivo	Buscar por imagen.	
Actores	Usuario.	
Resumen	El usuario introduce una imagen que es utilizada como criterio de búsqueda para encontrar otras similares.	
Complejidad	Alta.	
Prioridad	Alta.	
Precondiciones	El administrador ha configurado el sistema y ha iniciado el rastreo.	
Postcondiciones	Se han extraído los metadatos de las imágenes y se han indexado en Solr.	
Flujo de eventos		
Flujo básico buscar por imagen		
No.	Actor	Sistema
1.	Introduce la imagen.	
2.		Extrae las características de la imagen introducida.
3.		Consulta las imágenes indexadas en Solr.
4.		Compara la imagen introducida con las indexadas.
5.		Selecciona las imágenes más relevantes
6.		Devuelve las imágenes seleccionadas.
7.	Visualiza el resultado	
Relaciones	CU incluidos	
	CU extendidos	
Requisitos no funcionales		
Prototipo de interfaz de usuario		No aplica.

Figura 2.4: Descripción de caso de uso buscar por imagen

2.4. Especificación de requisitos de Software

Los requisitos del software permiten establecer lo que el sistema debe hacer, sus características fundamentales y las restricciones en el funcionamiento del sistema y los procesos de desarrollo del software. Los requisitos expresan las necesidades objetivas que presentan los usuarios, ante un sistema que resuelve un problema en particular de un determinado dominio(Sommerville, 2005).

2.4.1. Requisitos funcionales

En un sistema, los requisitos funcionales describen lo que el sistema debe hacer(Sommerville, 2005). A continuación se enumeran y se expone la prioridad que representa cada uno de ellos.

1. Cargar imagen. Prioridad media.
2. Extraer características de la imagen. Prioridad alta.
3. Comparar imagen con las existentes. Prioridad alta.
4. Mostrar imágenes similares. Prioridad media.

Las características que se extraen de la imagen introducida por el usuario son: color predominante, objetos, cantidad de rostros, transparencia y tamaño.

2.4.2. Requisitos no funcionales

Los requisitos no funcionales son aquellos que no se refieren directamente a las funciones específicas que proporciona el sistema, sino a las propiedades emergentes de este como la fiabilidad, el tiempo de respuesta y la capacidad de almacenamiento(Sommerville, 2005).
Se agrupan los requisitos no funcionales obtenidos a criterio del autor.

Requerimientos de hardware

- RNF 1. El servidor de servlets Tomcat y servidor maestro para Solr con Tomcat deben poseer como mínimo un CPU Core i3 a 2.50 GHz con 4 GB de RAM DDR3.

Requerimientos de software

- RNF 2. Se requiere del sistema operativo Linux Mint 16.4 o superior.
- RNF 3. Se requiere la instalación de la Máquina Virtual de Java para el correcto funcionamiento del componente.
- RNF 4. Se requiere la instalación del servidor web y de servlets Tomcat 7 para el correcto funcionamiento del servidor de Solr y del componente.
- RNF 5. Debe contar con la portabilidad necesaria para ser transferido de un ambiente a otro o reemplazado por nuevas versiones.

- RNF 6. Se requiere el uso de herramientas y recursos de software libre, las que se podrán usar, modificar y distribuir libremente.

Requerimientos de seguridad

- RNF 8. El campo de entrada debe ser validado.

2.5. Estilo arquitectónico

Un estilo es un concepto descriptivo que define una forma de articulación u organización arquitectónica. El conjunto de los estilos cataloga las formas básicas posibles de estructuras de software, mientras que las formas complejas se articulan mediante composición de los estilos fundamentales(Larman, 2003)(Reynoso, 2004).

Modelo Vista Controlador

Spring es un Framework que puede usarse en todas las capas de la aplicación. Se adapta perfectamente a la arquitectura Modelo-Vista-Controlador(MVC)(Schaefer y col., 2014). MVC separa la lógica del negocio de la interfaz de usuario y es uno de los más utilizados en el desarrollo de aplicaciones web ya que facilita la mantenibilidad y escalabilidad del sistema de forma simple y sencilla. Aunque la propuesta de solución no incluye la Vista, se decide hacer uso de este patrón, atendiendo a que el marco de trabajo seleccionado está regido por este tipo de patrón arquitectónico. En la propuesta de solución el estilo se refleja en la clase controladora, que maneja las peticiones de los usuarios y procesa la información para devolver los resultados. La clase modelo es la encargada de manejar las peticiones al indexador. La vista no aplica en la propuesta de solución.

2.6. Patrones de diseño

Los patrones de diseño son una representación de la descripción de un problema específico y recurrente. Presenta un esquema genérico demostrado con éxito para la solución del problema. La solución se especifica mediante la descripción de los componentes que la construyen y la forma en que estos colaboran entre sí(Canós y col., 2012)(Larman, 2003).
En el diseño del componente para la búsqueda por imágenes se tuvieron en cuenta una serie de patrones GRASP (Patrones Generales de Software para Asignación de Responsabilidades) que se presentan a continuación. Describen los principios fundamentales de la asignación de responsabilidades.

Experto: siguiendo el patrón Experto en Información se le asignaron responsabilidades a las clases que se pueden ver en la **Tabla 2.1**. Se mantiene el encapsulamiento de la información ya que los objetos utilizan su propia información para llevar a cabo sus tareas. Esto conlleva a un bajo acoplamiento, construyendo un sistema robusto y fácil de mantener.

Tabla 2.1: Relación de asignación de responsabilidades

Clase objeto	**Responsabilidad**
extraerColor	Conocer el color predominante de una imagen.
cantRostros	Conocer la cantidad de rostros en una imagen.
determinarDimension	Conocer la proporción de una imagen.
determinarTransparencia	Conocer si existe transparencia en una imagen.
extraerCaracteristicas	Conocer todas las características de una imagen.

Creador: la clase *extraerCaracteristicas* es la encargada de analizar el contenido de la imagen pasada por parámetro y crea instancias de las clases *extraerColor*, *cantRostros*, *determinarDimension* y *determinar-Transparencia*. Estas realizan el procesamiento necesario para detectar el color predominante, la cantidad de rostros que contiene, las dimensiones y determinan si existe transparencia o no. En la **Figura 2.5** se muestra el flujo de la creación de objetos.

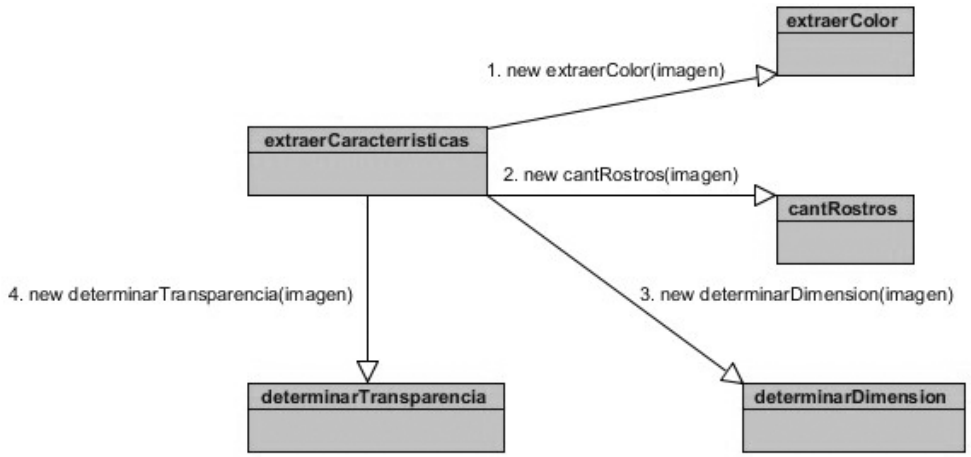

Figura 2.5: Creación de objetos en la clase *extraerCaracteisticas*

Alta cohesión: en la clase *extraerCaracteisticas* se manifiesta este patón de diseño. Esta clase colabora y a su vez delega responsabilades en las clases *extraerColor*, *cantRostros*, *determinarDimension* y *determinar-*

Transparencia. La utilización de este patrón se evidencia además en el diseño de la clase *extraerColor* que colabora y delega responsabilidades de las operaciones en los espacios de color en la clase *espacioColor*.
Controlador: este patrón tiene como objetivo asignar la responsabilidad a una clase de recibir o manejar un mensaje de evento del sistema generado por un actor externo, por lo general de la interfaz gráfica de usuario a la que accede un usuario para realizar ciertas operaciones en el sistema(Larman, 2003). La utilización de este patrón se evidencia en la clase *uploadController* que es la encargada de atender y ofrecer respuesta a cada una de las peticiones realizadas por el usuario mediante peticiones GET y POST que son enviadas desde la interfaz.

2.7. Diagrama de Clases de Diseño

El diagrama de clases del diseño describe gráficamente las especificaciones de las clases de software de una aplicación. Contiene las definiciones de las entidades del software, a diferencia del modelo conceptual que define conceptos del mundo real(Larman, 2003). En la **Figura 2.6** se muestra el diagrama de diseño perteneciente al caso de uso Buscar por imagen donde se presenta el controlador principal, que luego de introducida una imagen implementa una fase para analizarla y compararla con las que se encuentran indexadas en Solr. La clase *extraerCaracteristicas* es la encargada de extraer las características a la imagen subida. La clase *compararImagen* es la encargada de comparar las imágenes previamente indexadas con la imagen que se utiliza como criterio de búsqueda. La clase *ordenarLista* ordena la lista de las imágenes que resultaron seleccionadas por el algoritmo de comparación para ser mostradas al usuario.

Diagrama de clases del diseño

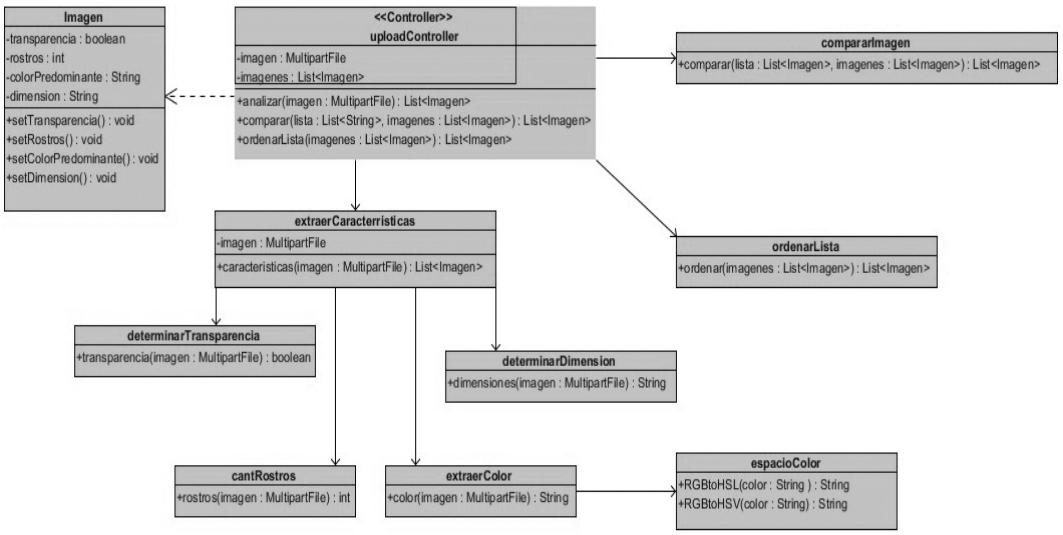

Figura 2.6: Diagrama de clases del diseño del caso de uso *Buscar por imagen*

2.8. Diagrama de interacción

En la **Figura 2.7** se muestra el diagrama de secuencia correspondiente al caso de uso *Buscar por imagen*. Modela en comportamiento dinámico que caracteriza al sistema. Representa un conjunto de objetos y clases, sus relaciones y los mensajes que se envían entre ellos.

Diagrama de secuencia

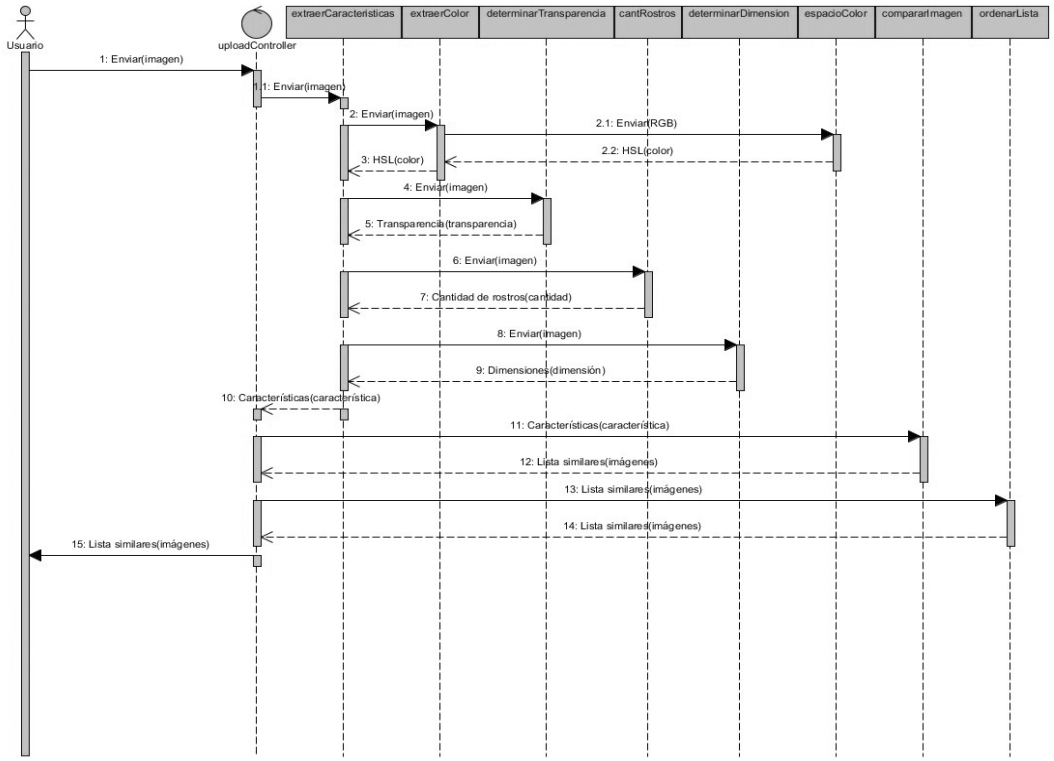

Figura 2.7: Diagrama de secuencia del caso de uso *Buscar por imagen*

2.9. Modelo de despliegue

Un diagrama de despliegue modela la arquitectura en tiempo de ejecución de un sistema. Esto muestra la configuración de los elementos de hardware (nodos) y muestra cómo los elementos y artefactos de software se relacionan en estos nodos(Canós y col., 2012).

En la **Figura 2.8** se aprecia el nodo *Dispositivo cliente*, representa un dispositivo utilizado por el usuario desde el que se realiza la búsqueda por imágenes a través de protocolo HTTPS. El nodo *Servidor de servicios con Tomcat* es el encargado de atender y ofrecer respuesta a cada una de las solicitudes del cliente. Se observa además, un nodo *Servidor maestro para Solr con Tomcat* como contenedor de servlets.

Figura 2.8: Diagrama de despliegue del componente para la búsqueda por imágenes

Capítulo 3

Implementación y validación del componente para la búsqueda por imágenes en la plataforma c.u.b.a.

En el presente capítulo se describe la etapa de implementación y codificación del componente para la búsqueda por imágenes en la plataforma c.u.b.a. La fase de implementación del sistema es una imprescindible dentro del proceso de desarrollo de software. Comprende la materialización, en forma de código, de todos los artefactos, descripciones y arquitectura propuestos en la etapa de análisis y diseño. Esto se hace según (Larman, 2003) para conformar el producto final requerido por el cliente. Para corroborar la correspondencia entre el producto y los requisitos definidos en las etapas anteriores, este debe ser sometido a determinadas pruebas. En el proceso se seleccionan determinadas pruebas en función del objetivo de la validación del sistema.

3.1. Modelo de componentes

El modelo de componentes representa la forma en que un sistema informático está estructurado, atendiendo a las diferentes partes que lo componen. (Sommerville, 2005) expone que cada componente debe ser tratado como una unidad de composición independiente e indispensable dentro de un sistema. Los archivos, módulos, librerías, ejecutables y binarios son ejemplos de componentes físicos.

3.1.1. Diagrama de componentes

El diagrama de componentes permite obtener una vista de la estructura general del sistema y el comportamiento de las funcionalidades que cada uno de estos componentes proporcionan y utilizan entre sí. Los principales paquetes que componen el diagrama expuesto en la **Figura 3.1** corresponden al componente para la búsqueda por imágenes y se describen a continuación:

- **Extractor:** Contiene las clases encargadas de extraer las principales características de la imagen introducida por el usuario.
- **ColorDetection:** Contiene las clases que se encargan de determinar el color predominante de la imagen
- **ObjectRecognition:** Alberga las clases que manejan el uso de Tensorflow y la detección de objetos en la imagen.
- **FacesDetector:** Compuesta por la clase que detecta la cantidad de rostros contenidos en la imagen.
- **Operations:** Tiene las clases encargadas de manejar el comportamiento de las operaciones solicitadas por el controlador, estas pueden ser: recibir una nueva imagen para ser procesada, extraer características y comparar.
- **Controller:** Contiene la clase controladora encargada de procesar las peticiones de los usuarios.

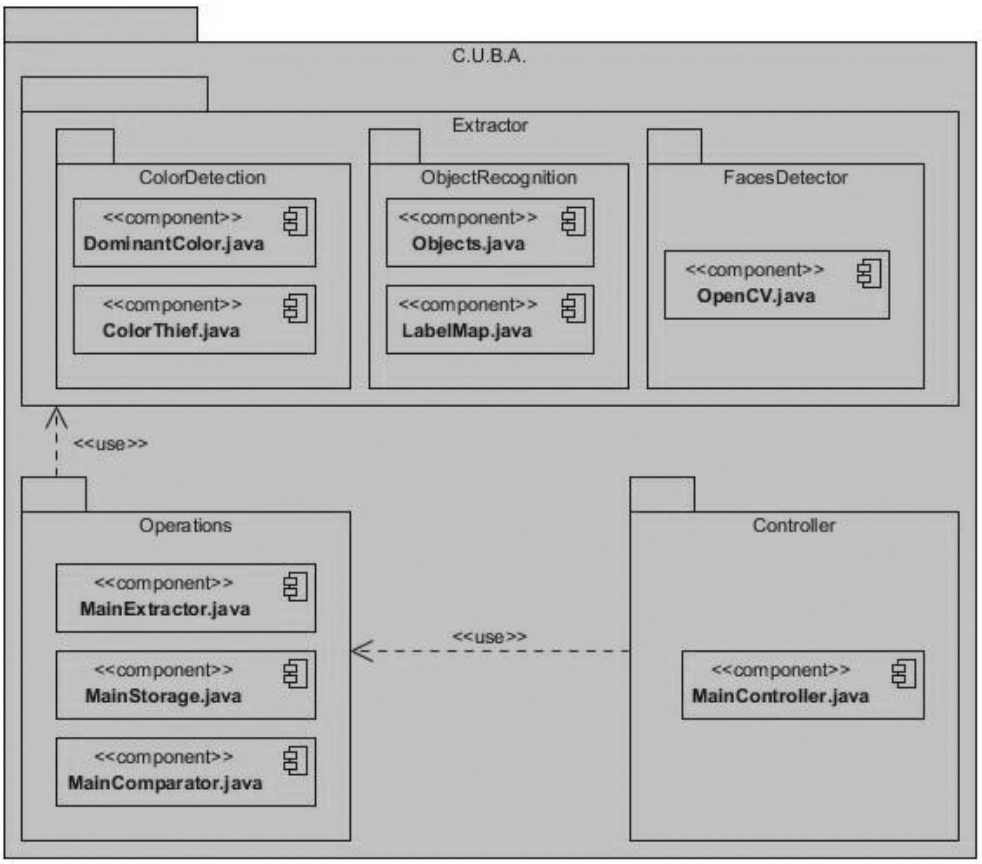

Figura 3.1: Diagrama de componentes

3.2. Estándares de codificación

La forma en que se estructura un código de Java no implica un mal funcionamiento de la aplicación final. Esto está demostrado en la práctica y aunque seguir una serie de directrices lleva a largo plazo a mejores programas, más legibles y entendibles, ninguna de ellas forma parte de la especificación Java. Oracle publica una guía llamada *Code Conventions for the Java programming Language* con muchas convenciones que están ampliamente extendidas entre la comunidad de programadores Java (*CCJL.* 2018). Entre sus elementos relevantes se encuentran:

- Añadir un espacio después de cada delimitador coma ’,’.
- El nombre de las clases se realiza en *UpperCamelCase*, lo que quiere decir, que comienza por mayúscula.

- Usa notación *camelCase* sin guiones bajos en variables, funciones, métodos y argumentos.
- Añadir un único espacio a ambos lados de un operador =, ==, etc.
- En los array multilínea, añade una coma al final de cada elemento, incluido el último.
- Añade un salto de línea antes de una sentencia *return*, a menos que este se encuentre solo en un bloque de sentencias, y un salto después de cada llave de sentencia, excepto después de la llave de cierre de clases.

3.3. Validación de la propuesta de solución

A continuación se detallan los tipos de pruebas de software aplicados a la herramienta implementada. El objetivo fundamental es la detección de las no conformidades respecto a las funcionalidades de la aplicación, la medición del grado de usabilidad y la correcta integración de los diferentes componentes del sistema.

3.3.1. Pruebas funcionales

Las pruebas funcionales se aplican a un sistema o software determinado para validar que las funcionalidades implementadas estén de acuerdo a las especificaciones de los requisitos definidos con anterioridad. Existen dos métodos principales para llevar a cabo las pruebas funcionales, estos son el método de Caja Blanca y el método de Caja Negra. El primero está centrado en las pruebas al código de las aplicaciones, mientras que el segundo permite a los probadores enfocarse en el funcionamiento de la interfaz, analizando los datos de entrada y salida. Para aplicar este tipo de pruebas, se selecciona la técnica de Caja Negra y se diseñaron dos Casos de Prueba (CP) que corresponden a los requisitos funcionales 1. Cargar imagen, de prioridad media y 2. Extraer características de la imagen, de prioridad alta.

Nº	Nombre del campo	Clasificación	Valor nulo	Descripción
1	Buscar	Campo de archivo	No	Solo permite archivos de tipo imagen.

Figura 3.2: Descripción de las variables

Caso de prueba 1: SC RF1_Cargar imagen.				
Condiciones de ejecución: No requiere condiciones de ejecución para realizar la búsqueda.				
Escenario	**Descripción**	**1**	**Respuesta del sistema**	**Flujo central**
EC 1.1 Realizar búsqueda por imágenes de forma correcta.	El sistema realiza la búsqueda de forma correcta.	V Imagen	El sistema procesa la imagen y devuelve las que son similares que estén indexadas.	1. El usuario introduce la imagen utilizando el botón seleccionar que se encuentra en la aplicación cliente. 2. Se presiona la tecla "enter" del teclado o se da click al botón "Buscar" para obtener el resulado.
EC 1.2 Realizar búsqueda por imágenes de forma incorrecta.	El sistema no realiza la búsqueda con archivos no admitidos.	I Archivo PDF	El sistema notifica al usuario "El archivo que ha introducido no es de tipo imagen".	

Figura 3.3: Caso de Prueba del RF1

Caso de prueba 2: SC RF2_ Extraer características de la imagen.				
Condiciones de ejecución: Requiere que el usuario haya cargado una imagen al sistema.				
Escenario	**Descripción**	**1**	**Respuesta del sistema**	**Flujo central**
EC 1.1 Extraer todas las características de la imagen cargada de forma correcta.	El sistema extrae todas las características de la imagen cargada de forma correcta.	V Imagen con objetos detectables	El sistema extrae las características de la imagen.	1. El sistema analiza la imagen introducida por el usuario y detecta que características deben extraerse. 2. El sistema extrae las características.
EC 1.2 Extraer todas las características de la imagen cargada de forma incorrecta.	El sistema intenta detectar los objetos pero la imagen no tiene objetos detectables.	I Imagen sin objetos detectables	El sistema utiliza un algoritmo diferente para realizar la extracción de características.	

Figura 3.4: Caso de Prueba del RF2

Resultado de las Pruebas funcionales

En una primera iteración se generaron 6 no conformidades, 2 de ellas son funcionales, relacionadas con la ausencia de la notificación al usuario cuando el archivo introducido no es una imagen o cuando no se pudo almacenar correctamente la imagen para su procesamiento. Una de excepción, esta no conformidad se produce en la validación del campo imagen, al dejar este campo vacío se lanzaba un error fatal que interrumpía el flujo del proceso. Fue resuelta al validar el campo *file* como *NOT NULL*. Se detectaron 3 no

conformidades de ortografía que fueron resueltas en una segunda iteración. En una tercera iteración no se detectaron no conformidades obteniendo resultados satisfactorios, los que son expuestos en la siguiente figura:

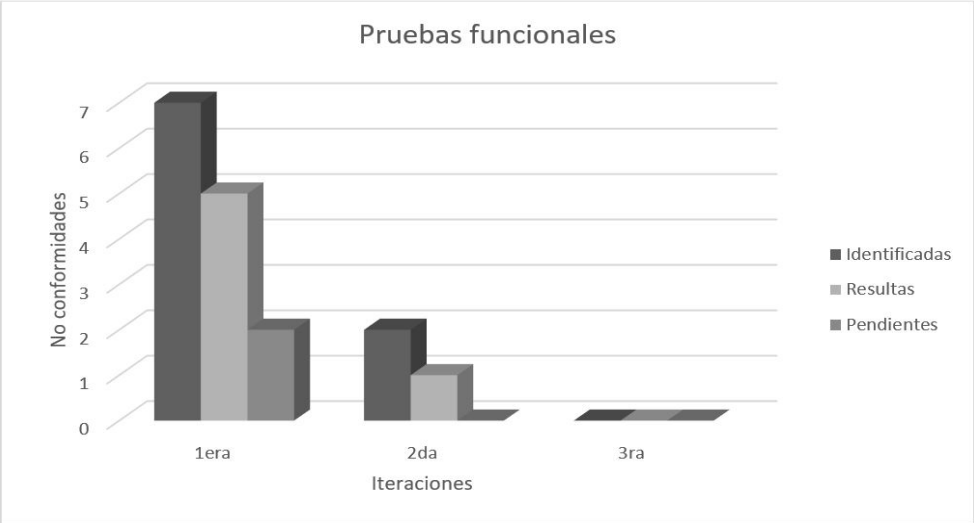

Figura 3.5: Resultado de las pruebas funcionales

3.3.2. Pruebas de integración

Las pruebas de integración aseguran que los distintos componentes de un software interactúan entre sí de forma correcta. Con el objetivo de validar la compatibilidad y el funcionamiento de las distintas partes que componen el Componente para la búsqueda por imágenes en la plataforma c.u.b.a. se toman las acciones relacionadas con:

- Validación de la conexión de Spring Boot y Solr. Ver **Figura 3.6**.
- Comprobar la salida de datos desde el componente hacia cualquier dispositivo cliente. Ver **Figura 3.7**.

Caso de prueba 3: SC RF3_Comparar imagen con las existentes				
Condiciones de ejecución: Requiere que el sistema haya extraído las características de la imagen.				
Escenario	**Descripción**	**1**	**Respuesta del sistema**	**Flujo central**
EC 1.1 Comparar la imagen cargada con las indexadas de forma correcta.	El sistema realiza la comparación de forma correcta.	V Imagen	El sistema compara la imagen con las imágenes indexadas.	1. El sistema carga las imágenes de Solr. 2. El sistema compara cada una de las características de la imagen introducida con las que se obtuvieron de Solr.
EC 1.2 Comparar la imagen cargada con las indexadas de forma incorrecta.	El sistema no encuentra imágenes que sean similares a la cargada.	I Imagen sin contenido	El sistema no compara las imágenes al no tener elementos para efectuar la comparación.	

Figura 3.6: Caso de Prueba del RF3

Caso de prueba 3: SC RF4_Mostrar imágenes similares				
Condiciones de ejecución: Requiere que el sistema haya comparado las imágenes				
Escenario	**Descripción**	**1**	**Respuesta del sistema**	**Flujo central**
EC 1.1 Mostrar las imágenes similares de forma correcta.	El sistema devuelve las imágenes que son similares.	V Imagen	El sistema devuelve las imágenes similares.	1. El sistema envía información al usuario que puede ser las imágenes similares o una notificación.
EC 1.2 Mostrar las imágenes similares de forma incorrecta.	El sistema no encuentra imágenes que sean similares a la cargada.	I Imagen sin contenido	El sistema notifica al usuario: “No existen imágenes similares”.	

Figura 3.7: Caso de Prueba del RF4

La aplicación de estas pruebas no arrojaron no conformidades, lo que valida que existe una correcta integración de los componentes internos del módulo.

3.3.3. Pruebas de carga y estrés

Las pruebas de carga consisten en probar el funcionamiento del software bajo condiciones extremas, mientras que las pruebas de estrés enfrentan al programa a condiciones anormales. Las pruebas ejecutan un

sistema que demanda recursos en cantidad, frecuencia, o volúmenes extremos. Es necesario aplicar este tipo de pruebas al componente propuesto como solución ya que se debe comprobar el rendimiento del sistema soportando una cantidad máxima de usuarios que soliciten este servicio en la web. Para la realización de las pruebas se utiliza la herramienta JMeter en su versión 2.3.1.
La prueba realizada consistió en definir 2 iteraciones de 800 y 1300 hilos de concurrencia, las que simulan 1000 y 2000 accesos de usuarios respectivamente. Fueron realizadas en un servidor Intel Core-i7 de 6ta generación a 2.6Ghz de velocidad de procesador, con una memoria RAM de 8GB y los hilos fueron configurados cada 1 segundo. Para facilitar el entendimiento de la **Tabla 3.1** se explican los parámetros que la componen:

- **Muestras:** cantidad de hilos utilizados para la URL.
- **Media:** tiempo promedio en milisegundos para un conjunto de resultados.
- **Min:** tiempo mínimo que demora un hilo en acceder a la página.
- **Max:** tiempo máximo que demora un hilo en acceder a la página.
- **Rendimiento:** rendimiento medido en los requerimientos por segundo/minuto/hora.
- **Kb/sec:** rendimiento medio en Kbytes por segundo.

Tabla 3.1: Resultados de prueba de carga y estrés

Muestras	Media	Min	Max	Error	Rendimiento	Kb/sec
1000	21898	71	72563	0.0	13.5/segundos	64.62
2000	59347	105	142880	0.106	16.2/segundos	55.37

Para un conjunto de 1000 usuarios, se observa un tiempo de respuesta promedio de 21898 milisegundos. La tasa de error detectada fue de 0.0 %, resultando en un rendimiento general de 13.5 peticiones respondidas por segundo. En otro escenario de 2000 usuarios el rendimiento mostrado fue de 16.2 peticiones respondidas por segundo con una tasa de error de 0.106 %. Estos resultados se toman como satisfactorios por el autor de la presente investigación ya que la aplicación se comporta de manera estable y esperada.

3.3.4. Validación de la hipótesis de la investigación

Para evaluar la confiabilidad de la hipótesis científica de la investigación se aplica el Criterio de expertos mediante el método *Delphi*. Este método es una técnica de obtención de información, basada en la consulta

de expertos en un área de estudio, con el fin de obtener la opinión de consenso más fiable proveniente del conocimiento y la experiencia de los participantes del grupo(García Valdés y col., 2013).
Variable independiente: el componente para la búsqueda por imágenes en la plataforma c.u.b.a. Esta variable es un componente del buscador que permite la búsqueda por imágenes en la web cubana. **Variable dependiente:** permitir encontrar imágenes similares, sus datos y localización en la web cubana.
Para aplicar el método *Delphi* se seguirán los siguientes pasos:

- Identificar y seleccionar los posibles expertos sobre el tema.
- Aplicar encuesta a los expertos seleccionados.
- Valorar la información obtenida.

Identificar y seleccionar los posibles expertos sobre el tema:
Se identifican 5 posibles expertos en materias relacionadas a la recuperación de información y procesamiento de imágenes. Con el objetivo de valorar el nivel de experiencia que poseen se aplica un cuestionario de autoevaluación sobre el tema en cuestión. En la **Tabla 3.2** se resumen los resultados obtenidos midiendo el nivel de estos en un rango del 1 al 10.

Tabla 3.2: Nivel de conocimiento de posibles expertos

Expertos	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**Kc**
Hubert Viltres Sala	-	-	-	-	-	-	-	X	-	-	0.8
Paúl Rodríguez Leyva	-	-	-	-	-	-	-	-	X	-	0.9
Eric Bárbaro Utrera Sust	-	-	-	-	-	-	-	X	-	-	0.8
Yuneldis Reyes Velázquez	-	-	-	-	-	-	-	X	-	-	0.8
Yordanka Fuentes Castillo	-	-	-	-	-	-	-	-	X	-	0.8

Para calcular el Coeficiente de Conocimiento o Información (Kc) del experto se utiliza la ecuación:
$K = n(0,1)$
Donde n es el nivel de conocimiento o información seleccionado por el experto. Para determinar, además, el coeficiente de argumentación o fundamentación de cada experto es necesario analizar los factores que aparecen en la **Tabla 3.3**.

Tabla 3.3: Coeficiente de Argumentación o Fundamentación

Fuentes de argumentación	**Alto**	**Medio**	**Bajo**
Análisis teóricos realizados	0.3	0.2	0.1
Experiencia obtenida	0.5	0.4	0.2
Trabajos nacionales	0.05	0.05	0.05
Trabajos internacionales	0.05	0.05	0.05
Conocimiento extranjero	0.05	0.05	0.05
Intuición	0.05	0.05	0.05

Para calcular el Coeficiente de Argumentación (Kc) del experto se utiliza la ecuación:
$\sum_{i=1}^{n} n_i$
Donde n_i es el valor obtenido en la fuente de argumentación i (de 1 hasta n) y n es la cantidad de fuentes de argumentación. Al aplicar la ecuación se obtienen los resultados siguientes:

- Hubert Viltres Sala. *Ka = 0.8*
- Paúl Rodríguez Leyva. *Ka = 0.9*
- Eric Bárbaro Utrera Sust. *Ka = 0.8*
- Yuneldis Reyes Velázquez. *Ka = 0.8*
- Yordanka Fuentes Castillo. *Ka = 0.9*

Con los respectivos coeficientes *Kc* y *Ka* se puede obtener el Coeficiente de Competencia (K) para determinar que expertos se toman en consideración para trabajar en la investigación. El cálculo de *K* se realiza con la ecuación:
$K = 0{,}5(Kc + Ka)$
El resultado obtenido se muestra en la **Tabla 3.4** y es valorado de la siguiente manera:

- El coeficiente es alto si $0{,}8 < K < 1$.
- El coeficiente es medio si $0{,}5 < K < 0{,}8$.
- El coeficiente es bajo si $K < 0{,}5$.

Tabla 3.4: Niveles de competencia

Expertos	**K**	**Valoración**
Hubert Viltres Sala	0.8	Alto
Paúl Rodríguez Leyva	0.9	Alto
Eric Bárbaro Utrera Sust	0.8	Alto
Yuneldis Reyes Velázquez	0.8	Alto
Yordanka Fuentes Castillo	0.9	Alto

Fueron seleccionados todos los expertos debido a que el nivel de competencia de todos fue alto. Finalmente, el panel de expertos quedó conformado de la siguiente manera:

Tabla 3.5: Expertos seleccionados

Nombre y apellidos	**Entidad**	**Años de experiencia**
Hubert Viltres Sala	DISW	15
Paúl Rodríguez Leyva	CIDI	9
Eric Bárbaro Utrera Sust	CIDI	8
Yuneldis Reyes Velázquez	DISW	13
Yordanka Fuentes Castillo	CIDI	8

Aplicar encuesta a los expertos seleccionados:
Seleccionado el panel de expertos que validará la hipótesis científica de la investigación, se aplica una encuesta para evaluar el Componente para la búsqueda por imágenes en la plataforma c.u.b.a. El juicio de expertos queda reflejado en la **Tabla 3.6**, atendiendo a las sentencias:

1. Facilidad de cargar la imagen.
2. Rapidez en la respuesta del sistema un vez introducida la imagen.
3. Precisión en la detección de las características de la imagen seleccionada.
4. Facilidad de visualizar las características propias de la imagen introducida y las similares encontradas.
5. Precisión en la similitud en las imágenes resultantes luego del proceso de búsqueda.

Para procesar y analizar esta información se clasificaron las respuestas en las categorías de: muy adecuado (MA), adecuado (A), poco adecuado (PA) e inadecuado (I). El resultado puede observarse en la **Tabla 3.6**.

Tabla 3.6: Resultado de la encuesta realizada

Sentencias	MA	A	PA	I	Total
1	5	0	0	0	5
2	4	1	0	0	5
3	2	3	0	0	5
4	3	2	0	0	5
5	5	0	0	0	5

Luego de analizar los criterios de los expertos en cada una de las categorías evaluativas para las diferentes ideas de la encuesta, se realizan los siguientes pasos para llegar a una conclusión y valoración de cada sentencia:

- Obtención de la tabla de frecuencia acumulada (**Tabla 3.7**).
- Obtención de la tabla de frecuencia relativa acumulativa (**Tabla 3.8**).
- Asignación de valor de la imagen que corresponde a cada frecuencia relativa acumulativa, por la inversa de la curva normal (**Tabla 3.9**).
- Obtención de puntos mediante el cálculo $N - P$ para el promedio relativo (**Tabla 3.9**).

Tabla 3.7: Frecuencia acumulada de los datos primarios

Sentencias	MA	A	PA	I
1	5	5	5	5
2	4	5	5	5
3	2	5	5	5
4	3	5	5	5
5	5	5	5	5

Tabla 3.8: Frecuencia relativa de los datos primarios

Sentencias	**MA**	**A**
1	1	1
2	0.8	1
3	0.4	1
4	0.6	1
5	1	1

Tabla 3.9: Imagen de la frecuencia relativa acumulativa

Sentencias	**MA**	**A**	**Suma**	**Promedio**	**Promedio relativo**
1	3.49	3.49	6.98	3.49	-0.96
2	0.85	3.49	0	0	5
3	2	3	0	0	5
4	3	2	0	0	5
5	5	0	0	0	5
Puntos de corte	1.57	3.49	25.27	-	-

Valorar la información obtenida:

Al analizar los promedios relativos (N-P) de las sentencias, se puede observar que ninguno excede el límite superior o punto de corte de la categoría evaluativa MA. Por lo tanto, con un nivel de concordancia de un 100 % , los expertos clasifican de nivel Muy adecuado las sentencias evaluadas, por lo que la hipótesis científica es apoyada por el juicio de los expertos. El alto valor y la calidad del componente para la búsqueda por imágenes en la plataforma c.u.b.a., queda evidenciado tras el análisis de los indicadores analizados.

Conclusiones

Completada la investigación se obtiene el Componente para la búsqueda por imágenes en la plataforma c.u.b.a., que permite la localización de estas en la web cubana, utilizando una imagen como criterio de búsqueda. Su uso agrega valor a la plataforma ya que la dota de una nueva funcionalidad. Otros aspectos significativos a destacar son:

- A partir del estudio realizado de los referentes teóricos relacionados con la recuperación de información y procesamiento de imágenes se determinó que existen una serie de Sistemas de Recuperación de Información que brindan funcionalidades que implican un procesamiento previo de las imágenes, se seleccionaron las funcionalidades para la propuesta de solución de acuerdo a las necesidades existentes ya que ninguno de estos es integrable a la plataforma c.u.b.a.
- El uso de las tecnologías y herramientas seleccionadas, permitió analizar y describir los subprocesos que se debían ejecutar para la implementación de las funcionalidades identificadas.
- El análisis de la descripción de la propuesta de solución, el modelo conceptual y el enfoque ágil propuesto por la metodología AUP en su variación para la UCI permitieron generar los artefactos correspondientes al escenario 2, haciendo uso de la herramienta Visual Paradigm empleando el lenguaje UML.
- Las pruebas de software permitieron solucionar los errores detectados en el componente y demuestran que es una solución funcional e integrable a la plataforma c.u.b.a.
- El criterio de expertos evidenció, con la valoración “muy adecuado", que el componente posee un alto nivel de valor y calidad.

Bibliografía

ALARCÓN, Vicenc Fernández, 2006. *Desarrollo de sistemas de información: una metodología basada en el modelado*. Univ. Politéc. de Catalunya.

AMATI, Gianni y VAN RIJSBERGEN, Cornelis Joost, 2002. Probabilistic models of information retrieval based on measuring the divergence from randomness. *ACM Transactions on Information Systems (TOIS)*. Vol. 20, n.º 4, págs. 357-389.

BATTAGLIA, Nicolás; MARTÍNEZ, Roxana; NEIL, Carlos y DE VINCENZI, Marcelo, 2017. Una propuesta de evaluación de herramientas CASE para la enseñanza. En: *Una propuesta de evaluación de herramientas CASE para la enseñanza*. *XXIII Congreso Argentino de Ciencias de la Computación (La Plata, 2017).*

CACHEDA, Fidel, 2008. Introducción a los modelos clásicos de Recuperación de Información/Introduction to the Classic Models of Information Retrieval. *Revista General de Información y Documentación*. Vol. 18, págs. 365.

CANÓS, José H y LETELIER, Mª Carmen Penadés Patricio, 2012. Metodologías ágiles en el desarrollo de software.

CASTELLS, Pablo; DÍEZ, Fernando y PULIDO, Estrella, 2011. Recuperación y almacenamiento de información en la web. *Escuela Politécnica Superior Universidad Autónoma de Madrid*.

CCJL. 2018 [`http://www.oracle.com/technetwork/java/codeconv-138413.html`]. [En línea] Accedido: 2018-02-27.

CHEN, Y, 2005. Clue: Cluster-based retriveral of images by unsupervised learning.

Ciberguerra contra Cuba: Mentiras en la red, 2011 [`http://www.cubadebate.cu/opinion/2011/03/22/ciberguerra-contra-cuba-mentiras-en-la-red`]. [En línea] Accedido: 2018-02-27.

CLEVERDON, Cyril, 1997. The Cranfield tests on index language devices. *Links*. Vol. 942, págs. 42.

Colocándonos en la web, 2012 [`https://periodismojosemarti.wordpress.com/2012/11/28/colocandonos-en-la-web/`]. [En línea] Accedido: 2018-02-18.

COMECHE, Juan Antonio Martínez, 2006. Los modelos clásicos de Recuperación de información y su vigencia. *Memoria del Tercer Seminario Hispano-Mexicano de investigación en bibliotecología y documentación 29 al 31 de marzo de 2006*, págs. 187.

CUNALATA MIRANDA, Jorge Luis y MORÁN EGUEZ, David Sebastián, 2016. *Levantamiento de los principales procesos para el Restaurante y Servicio de Cáterin Alexander; y automatizació del proceso de inventario y el proceso de gestión de reserva de mesas mediante una aplicación basada en Java aplicando la metodología de Programación Rational Unified Process (RUP)*. Tesis doctoral. PUCE.

DÁVILA LEGERÉN, Andrés, 2015. A la luz de la propia sombra. Incorporaciones de la fotografía a la sociología. *FOTOCINEMA. Revista científica de cine y fotografía*. N.º 10, págs. 306.

DEL CID, Alma; MÉNDEZ, Rosemary y SANDOVAL, Franco, 2011. Investigación: fundamentos y metodología.

DÍAZ, Raquel Gómez, 2002. *Estudio de la incidencia del conocimiento lingüístico en los sistemas de recuperación de la información para el español*. Universidad de Salamanca.

ELIOT, Simon y ROSE, Jonathan, 2009. *A Companion to the History of the Book*. John Wiley & Sons.

GARCÍA VALDÉS, Margarita y SUÁREZ MARÍN, Mario, 2013. El método Delphi para la consulta a expertos en la investigación científica. *Revista Cubana de Salud Publica*. Vol. 39, n.º 2, págs. 253-267.

HECHEVARRÍA-KINDELÁN, Ángela, 2002. Las consultorías de información en Cuba. Necesidad de su planeación mercadotécnia. *Revista Ciencias de la Información*. Vol. 33, n.º 1, págs. 45-53.

Image Recognition. 2018 [`https://www.tensorflow.org/tutorials/image_recognition`]. [En línea] Accedido: 2018-02-27.

ImageBrief Similar image search, 2018 [`http://www.imagebrief.com/about`]. [En línea] Accedido: 2018-02-27.

Internet Live Stats, 2018 [`http://www.internetlivestats.com`]. [En línea] Accedido: 2018-10-1.

JAIMES, Luis Gabriel y VEGA RIVEROS, Fernando, 2005. Modelos clásicos de recuperación de la información. *Revista Integración*. Vol. 23, n.º 1.

JAIMES, Óscar Aristizábal; MONTEALEGRE, Juan David Montoya y OROZCO, Jhony Óscar Salazar, 2015. Diseño e implementación de aplicación móvil para la gesti+on académica en la EAM. *IngEam*. Vol. 2, n.º 2.

JEZREEL, Mejia; MARCOS, González y MIRNA, Muñoz, 2017. Organization of the process areas of CMMI-Dev v1. 3 level 2 through of its dependencies. En: *Organization of the process areas of CMMI-Dev v1. 3 level 2 through of its dependencies. Information Systems and Technologies (CISTI), 2017 12th Iberian Conference on*, págs. 1-7.

JOHNSON, Rod y col., 2004. The spring framework–reference documentation. *Interface*. Vol. 21, págs. 27.

KOBAYASHI, Mei y TAKEDA, Koichi, 2000. Information retrieval on the web. *ACM Computing Surveys (CSUR)*. Vol. 32, n.º 2, págs. 144-173.

KUNA, Horacio; MARTIN, Rey; MARTINI, Esteban y SOLONEZEN, Lisandro, 2014. Desarrollo de un Sistema de Recuperación de Información para Publicaciones Científicas del Área de Ciencias de la Computación. *Revista Latinoamericana de Ingeniería de Software*. Vol. 2, n.º 2, págs. 107-114.

L., S. E. S. y STAROSTENKO, O., 2007. Modelo de indexación de formas basado en anotaciones ontológica. Primer encuentro de estudiantes en Ciencias de la Computación (=E2C2.

LARMAN, Craig, 2003. *UML y Patrones*. Pearson Educación eMadrid Madrid.

LEYVA, Paúl Rodríguez; SALA, Hubert Viltres y FLORES, Leiny Amel Pons, 2016. Componentes y funcionalidades de un sistema de recuperación de la información. *Revista Cubana de Ciencias Informáticas*. Vol. 10, págs. 150-162.

LÓPEZ CÓRDOVA, Verna; SANTILLANA FIGUEROA, Adela y VITTET GARCÍA, Lucía, 2015. Plan estratégico para Google Inc. Inc. 2015-2017.

LÓPEZ, Silvia Esther Sánchez, 2007. Modelo de indexación de formas en sistemas VIR basado en ontologías.

LUGO, Alma Jovita Domínguez; ÁVILA, Alicia Elena Silva; GUTIÉRREZ, Marco Polo Vázquez y MONTENEGRO, Eduardo Jesús Medina, 2016. Creación de un odontograma con aplicaciones Web/Creation of an odontogram with Web applications. *RECI Revista Iberoamericana de las Ciencias Computacionales e Informática*. Vol. 5, n.º 10, págs. 20-32.

MARCHIONINI, Gary, 1997. *Information seeking in electronic environments*. Cambridge university press. N.º 9.

MARTÍNEZ ESPADAS, Daniel, 2015. Herramienta para la generación de representaciones gráficas (ePanel).

NetBeans IDE. 2018 [`https://netbeans.org/features/`]. [En línea] Accedido: 2018-02-27.

Oracle's NetBeans IDE is the smarter and faster way to code. 2018 [`http://www.oracle.com/technetwork/developer-tools/netbeans/overview/index.html`]. [En línea] Accedido: 2018-02-27.

ORTEGA GONZALEZ, Erik Vladimir, 2008. *Una técnica para el análisis de similitud entre imágenes*. Tesis doctoral.

PALOMINO, Nora La Serna y CONCHA, Ulises Norberto Román, 2009. Técnicas de segmentación en procesamiento digital de imágenes. *Revista de investigación de Sistemas e Inform+atica*. Vol. 6, n.º 2, págs. 9-16.

PAREDES, Lourdes; PEÑAFIEL, Gonzalo Allauca; ARCOS, Gloria; GUERRA, José y ALLAUCA, Marcelo, 2015. Estudio del impacto del uso del sistema de control de versiones GitHub como herramienta de monitoreo y evaluaci+on acad+emica de trabajos colaborativos en instituciones de educación superior. *Revista Tecnológica-ESPOL*. Vol. 28, n.º 5.

PAUL, Oldfield, 2002. Domain Modelling. *Appropiate Process Movement*.

PORTILLO, José Antonio Pow Sang, 2002. La Especificación de Requisitos con Casos de Uso: Buenas y Malas Prácticas.

RAMOS, Luis Miguel Estrada, 2017. Apache Solr, un motor de búsqueda de código abierto. *Revista Digital Universitaria*. Vol. 13, n.º 11.

REYNOSO, Carlos, 2004. Introducción a la Arquitectura de Software. *Universidad de Buenos Aires*. Vol. 33.

ROBERTSON, Stephen E, 1977. The probability ranking principle in IR. *Journal of documentation*. Vol. 33, n.º 4, págs. 294-304.

ROSALES, Y. et al., 2013. Extensión de la herramienta Visual Paradigm para la generación de clases de acceso a datos con Doctrine 2.0. *Serie Científica de la Universidad de las Ciencias Informáticas*. Vol. 6, n.º 10.

SALTON, Gerard, 1989. Automatic text processing: The transformation, analysis, and retrieval of. *Reading: Addison-Wesley*.

SALTON, Gerard y MCGILL, Michael J, 1986. Introduction to modern information retrieval.

SÁNCHEZ, Tamara Rodríguez, 2015. Metodología de desarrollo para la Actividad productiva de la UCI v1.2.

SANDERSON, M. y CROFT, W. B., 2012. The History of Information Retrieval Research. *Proceedings of the IEEE*. Vol. 100, n.º Special Centennial Issue, págs. 1444-1451. ISSN 0018-9219. Disponible desde DOI: 10.1109/JPROC.2012.2189916.

SANTOVENIA DÍAZ, Javier y CAÑEDO ANDALIA, Rubén, 2007. 2x3: el primer buscador cubano en Internet. *Acimed*. Vol. 15, n.º 5.

SCHAEFER, Chris; HO, Clarence y HARROP, Rob, 2014. *Pro Spring*. Apress.

SOMMERVILLE, 2005. *Ingeniería de Software*. Pearson Educación.

TinEye Reverse image search, 2018 [https://www.tineye.com/how]. [En línea] Accedido: 2018-02-27.

VAN RIJSBERGEN, Cornelis J, 1986. A non-classical logic for information retrieval. *The computer journal*. Vol. 29, n.º 6, págs. 481-485.

VASQUEZ, Diana Silvino, 1997. La cultura de la imagen. *Ensayos: revista de la facultad de Educación de Albacete*. N.º 12, págs. 231.

YOO, Hun-Woo; JANG, Dong-Sik; JUNG, Seh-Hwan; PARK, Jin-Hyung y SONG, Kwang-Seop, 2002. Visual Information Retriveral System via Content-based Approach. Vol. 35, págs. 749-769.