Universidad de las Ciencias Informáticas

Facultad 10

"Agente Recuperador de Información en Tiempo Real para el Sistema Gestor de Contenidos Drupal"

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

Autores: Isleam Balceiro Rodríguez.
Orlando Merayo Maceda.

Tutor: Ing. José Ramón Fernández Pérez.

Ciudad de La Habana, Cuba.

Junio, 2010

"Año 52 de la Revolución"

DECLARACIÓN DE AUTORÍA

Declaramos ser autores de la presente tesis y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo. Para que así conste firmo la presente a los ____ días del mes de _____ del año _____.

_____ _____

Firma del Autor
Isleam Balceiro Rodríguez

Firma del Autor
Orlando Merayo Maceda

_____

Firma del Tutor
Ing. José Ramón Fernández Pérez

Resumen

Usuarios de todo el mundo publican y difunden constantemente información en Internet, tributando al crecimiento considerable del volumen de datos disponibles en la web. Escritores y periodistas integran a la red diariamente miles de artículos, en no pocas ocasiones estos carecen de enlaces que vinculen contenidos relacionados, afectando a lectores y redactores.

El Sistema Gestor de Contenidos Drupal constituye una herramienta muy empleada en la edición y publicación de artículos, aunque se evidencian rasgos de información desconectada en sitios concebidos con esta tecnología. Con el fin de disminuir este efecto perjudicial se hace necesario incorporar a Drupal un mecanismo que recupere información durante el proceso de redacción y permita vincularla.

Luego de un estudio profundo en aras de resolver la problemática mencionada se decide desarrollar un nuevo agente recuperador de información en tiempo real, IMAGENT. El mismo es capaz de sugerir documentos relacionados almacenados en bases de datos y vincularlos mientras se redacta un artículo a publicar.

Palabras clave: *IMAGENT, Agente Recuperador de Información en Tiempo Real, JITIR, Recuperación de Información, Sistemas Gestores de Contenidos (CMS), Sistema Gestor de Contenido Drupal.*

Índice de contenido

Introducción

Información desconectada, un efecto perjudicial

Ciertas situaciones perjudican a escritores, cronistas, reporteros, periodistas, lectores u otro personal que en señalado momento manipula información en la web. A continuación se ubican algunas de estas adversas cuestiones:

- El redactor ha concebido varios artículos abordando un tema general, cada artículo en particular argumenta un planteamiento específico y puntual, irrebatiblemente el contenido está relacionado, sin embargo, en ilimitados casos dichos artículos se publican dispersos, desenlazados y sin referencia alguna, quedando en considerable desuso o desechándose en el peor de los casos.
- El redactor incorpora en su artículo término(s) de significado confuso o desconocido para determinado lector, evita esclarecerlo para no descuidar o alejarse de la idea central, sin embargo, no se halla una referencia al significado de dicho término.
- El redactor, mientras escribe un artículo, pretende ubicar argumentos en aras de documentarse y enriquecer el contenido abarcado, para esta situación pospone su tarea e inicia un proceso de búsqueda, desvinculándose inevitablemente, y desplazando a un segundo plano la creación del artículo.
- Un lector, al tiempo que consulta determinado artículo, halla términos o frases que le sugiere información adicional y no existe un vínculo que conecte de manera directa al lector con algún contenido relacionado, aislando así la información asociada y obligando al lector a posponer la lectura e iniciar un proceso de búsqueda, desplazando a un segundo plano la lectura del artículo y desvinculándolo inevitablemente.

Al experimentar uno o varios de los efectos mencionados emerge el empleo desorientado de la información, la consulta de materiales carece de una línea que guíe el proceso de asimilación del conocimiento, se examinan fuentes dispersas, diseminadas en la red, atentando contra la concentración, posponiendo continuamente la lectura en búsqueda de argumentos relacionados. Esta situación problemática sugiere el término *“Información Desconectada”, que es la inutilización de referencias entre artículos e información con cierta relación.*

Evolución de las publicaciones digitales, influencia en los Sistemas Gestores de Contenidos

La segunda mitad de la década de los noventa reportó un auge significativo en cuanto a contenido publicado en Internet. Cada publicación requería escribir códigos para alcanzar el diseño y la funcionalidad apropiada, situación que se tornaba engorrosa. Numerosas organizaciones necesitadas de actualizar continuamente artículos, noticias y otros tipos de información digital, a favor de mejorar la forma tradicional de gestionar la información, desarrollaron los primeros sistemas de gestión de contenidos, en inglés: *content management system (CMS)*. Un CMS es básicamente, una herramienta que posibilita la creación y administración de contenidos, controlando mediante su interfaz, una o varias bases de datos donde se almacena el contenido del sitio web. Sus ventajas sobre lo que había sido "la forma tradicional de gestionar el contenido en la web" radican en manejar de manera independiente el diseño y el contenido; posibilitando en cualquier momento un cambio de diseño sin afectar el formato original del contenido, además de permitir la fácil y controlada publicación en el sitio a varios editores. (1) (2) (3)

Tecnología Drupal en la edición y publicación

Drupal es un CMS modular multipropósito, de gran potencia en la administración de usuarios y permisos, altamente configurable. Además, permite la edición y publicación de artículos, imágenes, documentos y otros tipos de archivos en la web e incluye servicios añadidos como foros, encuestas, blogs, entre otros. Drupal aglutina paquetes de códigos que brindan funcionalidades específicas y el usuario puede decidir aplicarlo o no al contenido. A estos paquetes se les denomina "módulos"; atendiendo a necesidades concretas puede utilizarse o no un determinado módulo. (4) (5)

La tendencia actual de Internet manifiesta un incremento de la información que se maneja, el contenido es considerable en cuanto a variedad y cantidad. La aceptación e interacción directa de los usuarios ha hecho de Drupal una herramienta eficiente para tratar distintos modelos de información. Aparejado a esta evolución crecen las exigencias de los usuarios y surgen ineficiencias que urge resolver a la comunidad de Drupal. La carencia de un módulo que proporcione información relacionada y posibilite vincularla con un artículo mientras este se redacta, es una dificultad latente que socava la eficiencia en la edición de contenido y contribuye con el incremento del efecto "Información Desconectada" en las publicaciones concebidas con este CMS.

Agentes Recuperadores de Información en Tiempo Real

Pueden hallarse sistemas de software que recuperen, considerando el entorno del usuario y de modo proactivo, información conveniente y afín con determinado tema. Es el caso de los agentes recuperadores de información en tiempo real, en inglés: *Just-in-time information retrieval agents ó JITIR agents*. (6) (7)

Un agente recuperador de información en tiempo real es capaz de ofrecer información potencialmente útil para el usuario, extraída de bases de datos de documentos (e-mail, notas, sistemas gestores de contenidos, etc.), y sin una petición concreta de la misma. El procedimiento se origina cuando el usuario estimula cierto evento sobre el contexto computacional, desencadenando así la alternativa de búsqueda; esta característica presente en los agentes JITIR es distintiva de otros métodos recuperadores de información como son los directorios, motores de búsqueda, entre otros (8). Tres condiciones elementales distinguen estos agentes recuperadores (9):

- La información debe ser presentada de manera accesible y no intrusiva, de manera que pueda ser ignorada por el usuario si ésta no le interesa.
- La información que se presenta debe referirse a eventos que tengan lugar en el contexto local del usuario.
- El agente busca y expone la información sin una solicitud explícita del usuario, este decide si le resulta valiosa o no. Esto se conoce como pro-actividad.

Directrices de la investigación (10) (11)

Examinando los efectos negativos de la información desconectada en publicaciones concebidas con el CMS Drupal se define el **problema de la investigación**: *Mientras se concibe un artículo, el déficit de referencias automáticas a documentos relacionados potencia la desvinculación de la información almacenada en bases de datos*.

Se orienta el **objeto de estudio** hacia los *Agentes Recuperadores de Información*, centrando el **campo de acción** en los *Agentes Recuperadores de Información para la web integrable con el sistema manejador de contenido Drupal 6.x*.

El **objetivo general** constituye un aspecto ineludible en la solución y se define como: *Desarrollar un subsistema para el CMS Drupal que permita devolver de manera autónoma, mientras se redacta un*

artículo, sugerencias de documentos relacionados con el tema que se aborda y la posibilidad de establecer vínculos. Originando los **objetivos específicos**:

- *Caracterizar el desarrollo de agentes recuperadores de información para la web.*
- *Desarrollar un agente recuperador de información para el CMS Drupal como un módulo más a instalar.*
- *Probar la eficiencia y funcionalidad operativa del agente desarrollado.*

Se **defiende la idea** de *que implementando un agente recuperador de información e integrándolo al CMS Drupal, se consigue disminuir el efecto perjudicial de la información desconectada en publicaciones concebidas con esta tecnología.*

Las **tareas** que se encaran son las siguientes:

- *Recopilar información acerca del desarrollo de sistemas recuperadores para la web.*
- *Describir el desarrollo alcanzado por los sistemas de recuperación de información para la web.*
- *Investigar sobre los componentes de implementación de módulos para el CMS Drupal.*
- *Describir el proceso de desarrollo de software correspondiente al subsistema del agente recuperador de información para Drupal.*
- *Implementar un módulo para Drupal que se comporte como un sistema pro-activo de obtención de información.*
- *Probar la funcionalidad operativa del módulo creado.*

Se emplean algunos **métodos científicos** en la presente investigación: La *entrevista* al Ing. José Ramón Fernández Pérez, tutor de este trabajo de diploma, establece el movimiento inicial para acometer la presente investigación científica y dicho método empírico persiste como alternativa de obtención de conocimientos. La *observación* posibilita detectar en sitios web de todo el mundo, información desconectada. También en sitios de la prensa cubana se halla de manera reiterada este efecto, Juventud Rebelde[1], Granma[2], Vanguardia[3], etc. son muestra de ello. El *análisis* de la

[1] Disponible en: http://www.juventudrebelde.co.cu.

[2] Disponible en: http://www.granma.co.cu.

[3] Disponible en: http://www.vanguardia.co.cu.

bibliografía referenciada proporciona, a través de la *síntesis*, elementos acerca del desarrollo de los agentes recuperadores de información. La aplicación del *método analítico – sintético* es de gran utilidad en la caracterización de elementos técnicos de interés. Mediante el análisis *histórico – lógico* se comprende la evolución de fenómenos que dan lugar al problema de la investigación. Para representar y comprender un problema de la vida real en algún lenguaje de programación se necesita modelarlo, representarlo de forma entendible para los implicados en la solución mediante un lenguaje visual. Con este fin se utiliza el método científico "*modelación*".

Estructuración del contenido

El presente trabajo investigativo se estructura en capítulos. A saber:

- Capítulo1: Orientado al fundamento teórico de la investigación, aborda temas referidos al estado del arte que envuelve al problema planteado. Profundiza acerca de la información desconectada, tecnologías de la web enmarcada en el uso de Drupal como CMS capacitado en la edición y publicación de artículos, estudio de tecnologías, incluyendo el uso de agentes recuperadores de información como vía a la solución del problema en cuestión.
- Capítulo 2: Establece las características del Agente JITIR a desarrollar, la información que se procesa, descripción general del sistema como elemento funcional y objeto de automatización, metodología de desarrollo aplicada, requisitos y descripción de los casos de uso.
- Capítulo 3: Análisis y diseño del sistema. Presenta cada artefacto perteneciente al flujo de trabajo análisis y diseño propuesto por la metodología de desarrollo empleada. Detalla con precisión los pasos a seguir en la implementación del sistema.
- Capítulo 4: Documenta la codificación del agente recuperador. Presenta algunas vistas de los componentes que integran el sistema y el despliegue del mismo. Confecciona y efectúa casos de pruebas para evaluar el desempeño funcional del sistema.
- Conclusiones.
- Bibliografía.
- Anexos.

Capítulo I: Estado actual de los Agentes Recuperadores de Información. Selección de herramientas, lenguajes y tecnologías

El presente capítulo establece una panorámica actual sobre los agentes recuperadores de información como mecanismo para aminorar los efectos perjudiciales de la "información desconectada". Se aborda además el uso de las tecnologías, metodologías y herramientas de apoyo que intervienen en el proceso de construcción del producto final.

1.1 Agentes recuperadores de información en tiempo real

Existen sistemas de software que recuperen información conveniente y afín con determinado tema. Es el caso de los agentes recuperadores de información en tiempo real, en inglés: *Just-in-time information retrieval agents ó JITIR agents*. (6) (7)

Un agente recuperador de información en tiempo real recupera y presenta información potencialmente útil para el usuario basada en el contexto local del mismo, de manera accesible y no intrusiva. El agente supervisa el ambiente del usuario y presenta información que puede ser o no de utilidad sin una acción explícita de la persona. Este ambiente es computacional, habitualmente un documento o área de una página web donde se escribe, aunque puede ser el entorno físico de la persona captado por cámaras, micrófonos, sistema de posicionamiento global (GPS) u otros sensores. No es objetivo del presente trabajo monitorizar el entorno físico del usuario, solo el área de una página web destinado a la redacción.

La información es recuperada de bases de datos de documentos pre-indexados (archivos de correo electrónico, notas y documentos, sistemas gestores de contenidos etc.) y sin una petición concreta de la misma. El procedimiento se origina cuando el usuario estimula cierto evento sobre el contexto computacional, desencadenando así la alternativa de búsqueda; esta característica presente en los agentes JITIR es distintiva de otros métodos recuperadores de información como son los directorios, motores de búsqueda, entre otros (8). Tres condiciones elementales distinguen estos agentes recuperadores (9):

- La información es presentada de manera accesible y no intrusiva, de manera que pueda ser ignorada por el usuario si ésta no le interesa.
- La información que se presenta debe referirse a eventos que tengan lugar en el contexto local del usuario.

- El agente busca y expone la información sin una solicitud explícita del usuario, este decide si le resulta valiosa o no. Esto se conoce como pro-actividad.

1.2 Agentes JITIR conocidos

Remembrance Agent (RA), Jimminy y *Margin Notes (MN)* son agentes JITIR conocidos en el mundo (9), este último centrado en la recuperación de información en la web. Todos fueron creados en el MIT *(Massachussets Institute of Technology)* como parte de un proyecto de desarrollo. Otro Agente Recuperador para la web es Plora, desarrollado en la Universidad de las Ciencias Informáticas (UCI) para el CMS Zope Plone (12).

RA comienza a utilizarse en 1996 en el editor de texto Emacs, del Sistema Operativo UNIX. RA presenta constantemente una lista de documentos que están relacionados con el material que se está leyendo o escribiendo. Es una herramienta para la memoria asociativa. Los documentos propuestos se muestran según el orden de relevancia y sin tener en cuenta el historial del usuario, en un búfer especial de la ventana de Emacs, estos se actualizan cada pocos segundos teniendo en cuenta las últimas cien o más palabras según la posición del cursor. Los documentos que propone son obtenidos de sus propios documentos de texto, facilita la configuración de las muestras según preferencias del usuario, puede analizar archivos de correo electrónico, HTML y documentos de texto sin formato. Se ejecuta en la mayoría de sistemas Unix, Mac y Windows. (6) (9) (13) (14) (15)

Jimminy, también llamado *Wearable R.A.* o "R.A. portátil", proporciona información basada en el ambiente físico de una persona, su localización, el entorno, la fecha y la hora, o el tema de la conversación en curso activan la búsqueda de información. Todo el proceso se lleva a cabo en un ordenador portátil que lleva consigo el usuario y las indicaciones son presentadas a través de una pantalla acoplada al ojo del mismo. Toda la información sobre el ambiente físico está disponible para Jimminy mediante sensores automáticos. Jimminy es una arquitectura general que usa plugins para cada sensor de forma que puede ser acoplado a un ordenador portátil. Se introduce en el sistema de forma manual toda aquella información o temas de conversación que no puedan ser recogidos a través de los sensores. La implementación del sistema es llevada a cabo por sensores pasivos que detectan la localización física de las personas presentes en una habitación y utiliza el reloj del sistema para determinar la hora. (6) (9) (16) (17)

Estos agentes utilizan un sistema de recuperación de información denominado *Savant*. Lo conforman dos programas fundamentales, *ra-retrieve* y *ra-index*. El programa ra-retrieve recupera información

basada en una consulta, ra-index crea el índice de los archivos donde se realiza la búsqueda, incluyendo recursos útiles como: colecciones de artículos o periódicos, documentos personales, notas, etc. Savant actúa como un artefacto de recuperación de información que realiza sus acciones en el interior del agente JITIR, creando un listado de documentos relacionados con la consulta planteada. (9) (12)

1.3 Agentes JITIR en la Web

Los agentes JITIR funcionan de forma análoga a los motores de búsqueda en lo que refiere a recuperación de información en la web. La monitorización del entorno computacional (página Web, área de redacción o editor de texto embebido en una página web) capta los eventos que estimulan la recuperación constante de información, generando así la pro-actividad. Como todo agente JITIR, el comportamiento no intrusivo y la recuperación de información basada en el contexto local del usuario son características presentes.

Margin Notes (MN) reescribe páginas Web automáticamente al ser cargadas, añadiendo vínculo para relacionar documentos. Una vez que la página Web es cargada por el navegador, MN añade un margen en el documento donde anota las sugerencias propuestas, compara cada sección del documento a pre-indexar, ya sean archivos de correo, campos de notas, u otros campos de texto basados en palabras clave concurrentes. Si se encuentra uno de esos campos indexados será relevante en la sección general de la página web y se incluye una pequeña “anotación” en el margen. La nota contiene una breve descripción del texto sugerido una serie de círculos que representan la relevancia de la sugerencia y un enlace que brinda más información. La anotación se compone por una materia, fecha y autor del texto sugerido, aunque los datos mostrados pueden ser personalizados por el usuario. (6) (9) (18)

Plone Remembrance Agent (PloRA) se integra al editor de texto kupu del CMS Zope Plone y su función fundamental es insertar enlaces a documentos relacionados a partir de las sugerencias devueltas. Es una herramienta de fácil manejo para usuarios inexpertos. Recomendable para sistemas dedicados a la edición y publicación de noticias. (6) (12)

1.4 Modelos de recuperación de información

Los modelos de recuperación asumen que el contenido de los documentos de las bases de datos y las necesidades de información de cada usuario pueden expresarse mediante un conjunto de

términos índice. Son usados como herramienta para comparar una consulta determinada y una serie de textos, imágenes, audio u otros tipos de información no textual sobre los cuales se realiza la consulta. Basan su funcionamiento en la creación de un índice en función del contenido de la información a recuperar, los índices proporcionan la relación de documentos de texto en los que aparece una palabra específica. Para la creación de índices de documentos se evalúan factores como la frecuencia de aparición de una palabra en el documento. El mayor problema es determinar qué documentos son los más relevantes. Según las premisas que se adopten se conocen varios modelos de recuperación. Entre los modelos clásicos de recuperación de información se encuentran el modelo booleano, modelo vectorial y modelo probabilístico. Existen otros modelos basados en el lenguaje, en redes de inferencia y lógica difusa pero estos habitan fuera de la frontera objetiva del presente trabajo.

1.4.1 Modelo booleano

Es un modelo de recuperación simple, basado en la teoría de conjuntos y el álgebra booleana. Su estrategia de recuperación se basa en un criterio de decisión binario sin ninguna noción de escala de medida. Para este modelo, las variables de peso de los términos índice son todas binarias. Con sus inconvenientes, el modelo booleano es el modelo dominante en los sistemas comerciales de bases de datos de documentos y proporciona un buen punto de partida en la recuperación de información. (12) (19) (20)

1.4.2 Modelo del Espacio Vectorial

Propone un marco en el que es posible el emparejamiento parcial a diferencia del modelo de recuperación booleano, asignando pesos no binarios a los términos índice de las preguntas y de los documentos. Estos pesos de los términos se usan para computar el grado de similitud entre cada documento guardado en el sistema y la pregunta del usuario. Ordenando los documentos recuperados en orden decreciente a este grado de similitud, el modelo de recuperación vectorial toma en consideración documentos que sólo se emparejan parcialmente con la pregunta, así el conjunto de la respuesta con los documentos alineados es mucho más preciso que el conjunto recuperado por el modelo booleano. Los rendimientos de alineación del conjunto de la respuesta son difíciles de mejorar. La mayoría de los motores de búsqueda lo implementan como estructura de datos. (12) (19) (20)

1.4.3 Modelo probabilístico

Su funcionamiento se basa en el cálculo de la probabilidad que posee un documento de ser relevante a una pregunta dada. Los modelos anteriores están basados en la equiparación en la forma más pura. En el booleano es o no coincidente, y en el vectorial el umbral de similitud es un conjunto, y si un documento no está no es similar y, por lo tanto, no recuperable. La equiparación probabilística se basa en que, dado un documento y una pregunta, existe la posibilidad de calcular la probabilidad de que ese documento sea relevante para esa pregunta. De esta forma, la idea de relevancia está relacionada con los términos de la pregunta que aparecen en el documento. Se ha demostrado que mediante los procedimientos del modelo de recuperación probabilístico se obtienen buenos resultados, de cualquier forma, los resultados no superan en gran medida a los obtenidos en el modelo booleano o el vectorial, y se hace más complejo el utilizarlo por la cantidad de cálculos y premisas que envuelve. (12) (19) (20)

1.4.4 Data Fusion

Es una técnica con varios algoritmos, métodos de indexado y búsqueda que son utilizados para producir juegos de documentos importantes. Estos resultados se combinan y se obtienen los documentos útiles. El sistema Savant es un ejemplo de sistema de recuperación de información Data Fusion. (6) (12)

1.5 Ley del mínimo esfuerzo en los agentes JITIR

Un individuo cualquiera valora las distintas opciones que posee para resolver una situación determinada y realizarla con el menor esfuerzo posible. Esta regla es conocida como la Ley del Mínimo Esfuerzo y plantea que una persona trata de minimizar su trabajo físico o mental y así optimizar el esfuerzo en la actividad que va a realizar (13). Los agentes JITIR proporcionan facilidades a la hora de encontrar contenido relevante para los usuarios. El tiempo de devolución de la información se reduce cuando la búsqueda pasa de ser manual a un proceso automatizado. Se obtiene el contenido evaluado rápidamente y el usuario ahorra tiempo, esfuerzo mental para encontrar, evaluar y acceder a la información, además, no se desvincula totalmente de su actividad primaria. Aunque un agente JITIR reduce los costos de acceder y evaluar información, no los elimina completamente. Existe un pequeño esfuerzo que no deja de ser importante: revisar las sugerencias que brinda y evaluar si el resultado es relevante o no. La información que devuelve un JITIR no siempre tiene la utilidad esperada para el usuario. Existen algunas categorías para valorar la utilidad

de los resultados:

- Falsa positiva (inútil): en este caso la información presentada no es útil. Puede estar dada por deficiencias en el sistema de búsquedas o baja calidad en la información.
- Falsa positiva (conocida): aquí la información es útil, pero ya es conocida y está siendo usada u obviada por el usuario.
- Costo disminuido: el usuario conoce que existe esa información, pero no cree que valga la pena realizar la búsqueda. Al presentar la información directamente se disminuye el costo de acceder a esta y por tanto se espera mayor beneficio para el usuario de acuerdo con el resultado obtenido.
- Incremento del beneficio esperado: la información no es útil, pero indica la existencia de otra que pueden ser de valor. Aquí el JITIR no disminuye el costo de las acciones para encontrar alguna información pero incrementa el beneficio esperado, que se fomenta con la obtención de otro contenido importante para otra búsqueda.
- Costo disminuido e incremento del beneficio esperado: la información que se proporciona es desconocida y útil. El JITIR facilita el acceso al documento completo disminuyendo el costo de accesibilidad a este e incrementa el beneficio esperado de recuperar dicho documento.

Estas evaluaciones de la información devuelta por un agente JITIR permiten identificar desde los resultados relevantes para el usuario en la tarea que esté realizando, hasta aquellos que carecen de importancia y que no tienen ninguna relación con los resultados que se esperan. Todos los resultados que devuelva un JITIR estarán condicionados por el(los) modelo(s) de recuperación de información que se incluyan en el agente JITIR.

1.6 La evaluación de la información en Sistemas Recuperadores de Información

La utilidad de la información se basa en la relevancia que pueda tener esta para cierta persona. El significado de relevancia es complejo, para simplificar, se definen dos tipos fundamentales.

- **Relevancia "formal":** los resultados de una búsqueda de información responden a la ecuación de búsqueda que se había planteado.
- **Relevancia "semántica":** los resultados obtenidos responden a las necesidades del usuario.

Es necesario que los agentes JITIR cumplan con la relevancia semántica de la información. No es

objetivo un agente JITIR que brinde gran cantidad de información si no satisface la necesidad de conocimiento del usuario. Un agente JITIR orientado a facilitar información semánticamente relevante contribuye, en mayor grado, a reducir el esfuerzo mental requerido para encontrar, evaluar y acceder a la información que se necesita para enriquecer cualquier tema. El corazón de un agente JITIR es su sistema de recuperación de información, conformado a su vez por los métodos de recuperación de información, estos se evalúan en términos de relevancia dada una consulta. La relevancia de un documento recuperado con respecto a una consulta es el grado de importancia que posee dicho documento y se calcula utilizando modelos matemáticos.

Dentro de los sistemas de recuperación de información se manejan dos características fundamentales que definen el resultado de las búsquedas de información relevante, estas son *precision* y *recall* (6). *Recall* es la medida de que todos los documentos relevantes a la consulta sean sugeridos al usuario, en este caso se muestran todos los documentos pero sin especificar cuáles sugerencias tienen mayor relevancia, consecuentemente el usuario desvía su atención de la tarea que está realizando para distinguir aquellos documentos que tienen la importancia requerida para ser enlazados con el contexto, entre todos los que fueron devueltos por el agente JITIR. Sin embargo, *precision* asegura que los documentos devueltos sean los que mayor relevancia tengan de acuerdo a la consulta obtenida del entorno de trabajo del usuario. Esta condición hace posible que el usuario mantenga su atención en la redacción del artículo, sólo cambiará la tarea que está realizando en el momento que haga una valoración de las sugerencias que se muestran en la interfaz del agente, para seleccionar cuál de ellas vinculará con la redacción. El desvío de atención que realiza el usuario es necesario, pues en ese instante él necesita escoger la sugerencia que considere más relevante. Los agentes JITIR normalmente devuelven pocas sugerencias, y estas con los mayores grados de relevancia. Si se ofrecen muchos documentos como sugerencias, conduce a demasiada distracción para el trabajo del usuario, esta es la razón fundamental por la que se escogen sistemas con mayor *precision* que *recall* en la implementación de los agentes JITIR.

1.7 Interfaz de usuario para un agente JITIR

Un agente JITIR se encarga de que el usuario mantenga su atención en la tarea primaria (redacción o lectura) que realiza y se distraiga en el menor grado posible, por ello el agente debe evitar el cúmulo excesivo de información y los mensajes innecesarios. La interfaz de un agente JITIR cumple con un comportamiento no intrusivo, así el usuario, si lo desea, puede ignorar el contenido mostrado y este no interfiere la tarea primaria. El chequeo constante del entorno de trabajo permite al agente obtener

palabras que ensamblan las consultas necesarias, luego se recuperan los documentos y posteriormente se muestran mediante la interfaz, erigiendo una conducta pro-activa. Analizar los documentos relevantes sugeridos debe constituir para el usuario una tarea secundaria, una interfaz poco discreta produce acciones que pueden desvincularlo. Si al generarse nuevas sugerencias, la interfaz varía su forma o estilo, ocasiona irregularidades a la vista del usuario y es posible que de manera involuntaria se pierda concentración. Ineludiblemente el usuario, al analizar la información que brinda el agente desplaza momentáneamente la tarea primaria a un segundo plano, este cambio de actividad no debe producirse como consecuencia de acciones producidas por la interfaz del agente, debe ocurrir de manera consciente y voluntaria para el usuario.

La interfaz debe ser moderada pero no invisible, así el usuario percibe que tiene un apoyo en el trabajo que realiza y puede alternar la tarea primaria y la secundaria sin demasiado esfuerzo. Un diseño de interfaz es apropiado para el agente JITIR cuando le permite al usuario atender su tarea principal y dividir su atención cuando lo desee. Tres aspectos fundamentales se enlazan a la información entregada por un agente JITIR:

- El contenido no es provechoso en todo momento, incluso con la recuperación perfecta de datos. Puede que el usuario no precise de más información y una sobrecarga lo distraiga, afectándolo así.
- El usuario decide que es apropiado determinado material cuando asume que brinda alguna información útil sobre el contexto del contenido que lee o redacta.
- Posponer la lectura o redacción para determinar si es apropiado señalado material constituye una distracción y una carga mental para el usuario.

Considerando los puntos anteriores, debe mostrarse la información recuperada en una interfaz moderada que actualice de forma progresiva el contenido, incrementando en cada fase la información y el nivel de especificidad, esto se conoce como interfaz gradual. Cada fase adiciona, respecto a la fase anterior, un pequeño costo al leer y valorar la relevancia del contenido. El propósito de una interfaz gradual es incrementar el alcance informativo en cada fase, reducir el costo de los falsos positivos, reducir el esfuerzo requerido para recuperar el cuerpo íntegro del documento sugerido y decidir con mayor simpleza la relevancia de la información.

1.8 Lenguajes y tecnologías

Luego del estudio profundo de los agentes recuperadores de información existentes en la actualidad y en interés de concebir un agente recuperador de información para la Web integrable al CMS Drupal se seleccionan un grupo de lenguajes y tecnologías.

1.8.1 Lenguaje de Marcado de Hipertexto (HTML)

HTML, siglas de ***HyperText Markup Language*** (Lenguaje de Marcas de Hipertexto), es el lenguaje de marcado predominante para la construcción de páginas web. Es usado para describir la estructura y el contenido en forma de texto, así como para complementar el texto con objetos tales como imágenes. HTML se escribe en forma de "etiquetas", rodeadas por corchetes angulares (<,>). HTML también puede describir, hasta un cierto punto, la apariencia de un documento, y puede incluir *scripts*, capaces de afectar el comportamiento de navegadores web y otros procesadores de HTML.

HTML también es usado para referirse al contenido del tipo de MIME text/html o todavía más ampliamente como un término genérico para el HTML, ya sea en forma descendida del XML (como XHTML 1.0 y posteriores) o en forma descendida directamente de SGML (como HTML 4.01 y anteriores).

Por convención, los archivos de formato HTML usan la extensión .htm o .html. (21) (22)

1.8.2 Lenguaje de Marcado Extensible (XML)

XML, sigla en inglés de *Extensible Markup Language* («lenguaje de marcas ampliable»), es un metalenguaje extensible de etiquetas desarrollado por el *World Wide Web Consortium* (W3C). Es una simplificación y adaptación del SGML y permite definir la gramática de lenguajes específicos (de la misma manera que HTML es a su vez un lenguaje definido por SGML). Por lo tanto XML no es realmente un lenguaje en particular, sino una manera de definir lenguajes para diferentes necesidades. Algunos de estos lenguajes que usan XML para su definición son XHTML, SVG, MathML.

XML no ha nacido sólo para su aplicación en Internet, sino que se propone como un estándar para el intercambio de información estructurada entre diferentes plataformas. Se puede usar en bases de datos, editores de texto, hojas de cálculo y casi cualquier cosa imaginable. (21)

XML es una tecnología sencilla que tiene a su alrededor otras que la complementan y la hacen mucho más grande y con unas posibilidades mucho mayores. Tiene un papel muy importante en la actualidad ya que permite la compatibilidad entre sistemas para compartir la información de una manera segura, fiable y fácil. (23)

1.8.3 Hojas de Estilo en Cascada (CSS)

CSS, Las hojas de estilo en cascada *(Cascading Style Sheets, CSS)* son un lenguaje formal usado para definir la presentación de un documento estructurado escrito en HTML o XML (y por extensión en XHTML). El W3C *(World Wide Web Consortium)* es el encargado de formular la especificación de las hojas de estilo que servirán de estándar para los agentes de usuario o navegadores. La idea que se encuentra detrás del desarrollo de CSS es separar la estructura de un documento de su presentación.

CSS permite crear páginas web de una manera más exacta. Gracias a las CSS se controlan mucho mejor los resultados finales de la página, lo que se dificultaba utilizando solamente HTML, incluir márgenes, tipos de letra, fondos, colores y otros. (24)

Por ejemplo, el elemento de HTML <H1> indica que un bloque de texto es un encabezamiento y que es más importante que un bloque etiquetado como <H2>. Versiones más antiguas de HTML permitían atributos extra dentro de la etiqueta abierta para darle formato (como el color o el tamaño de fuente). No obstante, cada etiqueta <H1> debía disponer de la información si se deseaba un diseño consistente para una página, y además, una persona que lea esa página con un navegador pierde totalmente el control sobre la visualización del texto.

Cuando se utiliza CSS la información de estilo puede ser adjuntada tanto como un documento separado o en el mismo documento HTML. En este último caso podrían definirse estilos generales en la cabecera del documento o en cada etiqueta particular mediante el atributo "style". (21)

1.8.4 Preprocesador de Hipertexto (PHP)

PHP es un lenguaje de programación interpretado, diseñado originalmente para la creación de páginas web dinámicas. Es usado principalmente en interpretación del lado del servidor (server-side scripting) pero actualmente puede ser utilizado desde una interfaz de línea de comandos o en la creación de otros tipos de programas incluyendo aplicaciones con interfaz gráfica usando las

bibliotecas Qt o GTK+.

PHP es un acrónimo recursivo que significa PHP *Hypertext Pre-processor* (inicialmente PHP Tools, o, Personal Home Page Tools). Fue creado originalmente por Rasmus Lerdof en 1994; sin embargo la implementación principal de PHP es producida ahora por *The PHP Group* y sirve como el estándar de facto para PHP al no haber una especificación formal. Publicado bajo la *PHP License*, la *Free Software Foundation* considera esta licencia como software libre.

PHP es un lenguaje interpretado de propósito general ampliamente usado y que está diseñado especialmente para desarrollo web y puede ser embebido dentro de código HTML. Generalmente se ejecuta en un servidor web, tomando el código en PHP como su entrada y creando páginas web como salida. Puede ser desplegado en la mayoría de los servidores web y en casi todos los sistemas operativos y plataformas sin costo alguno. PHP se encuentra instalado en más de 20 millones de sitios web y en un millón de servidores, aunque el número de sitios en PHP ha declinado desde agosto de 2005. (21)

Teniendo en cuenta la integración del Agente Recuperador con el CMS Drupal, se selecciona PHP como uno de los lenguajes a utilizar, por su protagonismo en la implementación de módulos para dicho CMS.

1.8.5 JavaScript

JavaScript fue creado por Brendan Eich en la empresa *Netscape Communications* (25). Es un lenguaje de programación interpretado, es decir, no requiere compilación, se utiliza principalmente en el desarrollo de páginas web y aunque su nombre puede llevar a equívocos, no tiene nada que ver con Java, puesto que a diferencia de éste, no está orientado a objetos, sino que es un lenguaje basado en prototipos, ya que las nuevas clases se generan clonando las clases base (prototipos) y extendiendo su funcionalidad (26). Se utiliza para acceder a objetos en las aplicaciones web. Se caracteriza por ser un lenguaje que puede aportar cierto dinamismo a las aplicaciones. Este lenguaje de script tiene como características fundamentales:

- Manejado por eventos: Puede responder a eventos como el movimiento del mouse, presionado de alguna tecla o algo tan simple como recargar una página Web.
- Independiente de cualquier plataforma: Los programas de JavaScript están diseñados para ejecutarse dentro del código de documentos HTML. Son independientes de cualquier

plataforma o sistema operativo.

- Permite desarrollo rápido: El navegador Web o el código HTML manejan la mayoría de las características como formas, cuadros y otros elementos de interfaces por lo que los desarrolladores solo tienen que usar el código como complemento a estos objetos.

El código JavaScript puede ser integrado dentro de las páginas Web. Para evitar incompatibilidades el W3C diseñó un estándar denominado *Document Object Model* (DOM), en su traducción al español Modelo de Objetos del Documento. (25)

1.8.6 AJAX

AJAX, acrónimo de *Asynchronous JavaScript And XML* (JavaScript asíncrono y XML), es una técnica de desarrollo web para crear aplicaciones interactivas o RIA *(Rich Internet Applications)*. Estas aplicaciones se ejecutan en el cliente, es decir, en el navegador de los usuarios mientras se mantiene la comunicación asíncrona con el servidor en segundo plano. De esta forma, es posible realizar cambios sobre las páginas sin necesidad de recargarlas, lo que significa aumentar la interactividad, velocidad y usabilidad en las aplicaciones.

AJAX es una tecnología asíncrona, en el sentido de que los datos adicionales se requieren al servidor y se cargan en segundo plano sin interferir con la visualización ni el comportamiento de la página. JavaScript es el lenguaje interpretado *(scripting language)* en el que normalmente se efectúan las funciones de llamada de AJAX mientras que el acceso a los datos se realiza mediante *XMLHttpRequest*, objeto disponible en los navegadores actuales. En cualquier caso, no es necesario que el contenido asíncrono esté formateado en XML.

AJAX es una técnica válida para múltiples plataformas y utilizable en muchos sistemas operativos y navegadores, dado que está basado en estándares abiertos como JavaScript y *Document Object Model* (DOM). (21) (27)

Por la necesidad de implementar un Agente Recuperador que cumpla con la recuperación de información en forma proactiva se selecciona AJAX.

1.8.7 Lenguaje Unificado de Modelado (UML)

UML, acrónimo en inglés de Unified Modeling Language, es un lenguaje para visualizar, especificar, construir y documentar los artefactos de un sistema que involucra una gran cantidad de software. El

UML está compuesto por diversos elementos gráficos que se combinan para conformar diagramas. Debido a que el UML es un lenguaje, cuenta con reglas para combinar tales elementos (28). Es importante recalcar que UML no es un método de desarrollo. No indicará cómo pasar del análisis al diseño y de éste al código. No son una serie de pasos que llevan a producir código a partir de especificaciones. UML no es una guía para realizar el análisis y diseño orientado a objetos, es decir, no es un proceso (29). UML es un lenguaje que permite la modelación de sistemas con tecnología orientada a objetos, como lo constituye el Agente Recuperador en desarrollo. UML:

- Puede conectarse con lenguajes de programación (Ingeniería directa e inversa).
- Permite modelar sistemas utilizando técnicas orientadas a objetos.
- Permite especificar todas las decisiones de análisis y diseño, construyéndose así modelos precisos, no ambiguos y completos.
- Permite documentar todos los artefactos de un proceso de desarrollo (requisitos, arquitectura, pruebas, versiones, etc.).
- Es un lenguaje muy expresivo que cubre todas las vistas necesarias para desarrollar y luego desplegar los sistemas.
- Existe un equilibrio entre expresividad y simplicidad, pues no es difícil de aprender ni de utilizar.
- Es utilizado por la mayoría de las metodologías.

1.9 Herramientas de modelado

Las herramientas de modelado son fundamentales para el análisis del sistema. Hay varias herramientas creadas para el desarrollo de la Ingeniería de Software, estas existen con el fin de desarrollar programas, utilizando técnicas de diseño y metodologías bien definidas, soportadas por herramientas automáticas.

Las herramientas CASE *(Computer Aided Software Engineering)*, y en español Ingeniería de Software Asistida por Computador, brindan un conjunto de ayudas para el desarrollo de programas informáticos, desde la planificación, pasando por el análisis y el diseño, hasta la generación del código fuente de los programas y la documentación. (30)

1.9.1 Visual Paradigm Enterprise Edition 5.3

Visual Paradigm para UML es una de las herramientas UML CASE más completa y fácil de usar, con soporte multiplataforma y que proporciona excelentes facilidades de interoperabilidad con otras aplicaciones. Fue creada para el ciclo vital completo del desarrollo del software que lo automatiza y acelera, permitiendo la captura de requisitos, análisis, diseño e implementación. Tiene disponible distintas versiones: *Enterprise*, *Professional, Standard, Modeler, Personal y Community.* Ayuda a construir aplicaciones de calidad más rápido, mejor y a más bajo costo. Se pueden dibujar todos los tipos de diagramas de clase, código inverso, generar el código de diagramas y generar la documentación. Visual Paradigm para UML apoya un conjunto de idiomas tanto en la generación del código como en la Ingeniería Inversa en Java, C + +, Ada, Python y PHP. (31)

1.10 Herramientas de desarrollo

Las herramientas de desarrollo son aquellos programas o aplicaciones que tengan cierta importancia en el desarrollo de un programa (programación). Elegir las herramientas adecuadas para acometer el desarrollo de programas es fundamental, ahorra tiempo y esfuerzo.

1.10.1 Zend Studio 6.1

Editor Web orientado a la programación en PHP, proporciona una serie de ayudas en la gestión y creación de proyectos, así como en la depuración de código y permite desarrollar aplicaciones web. Consta de dos partes en las que se dividen las funcionalidades: la parte del cliente y las del servidor. Las dos partes se instalan por separado, la del cliente contiene la interfaz de edición y la ayuda. Permite además hacer depuraciones simples de scripts, aunque para disfrutar de toda la potencia de la herramienta de depuración habrá que disponer de la parte del servidor, que instala Apache y el módulo PHP o, en caso de que estén instalados, los configura para trabajar juntos en depuración. Contiene una ayuda contextual con todas las librerías de funciones del lenguaje que asiste en todo momento ofreciendo nombres de las funciones y parámetros que deben recibir. Otras ayudas que ofrece a la hora de escribir son las típicas en editores avanzados, como permitir editar varios archivos, y moverse fácilmente entre ellos, marcar a qué elementos corresponden los inicios y cierres de las etiquetas, paréntesis o llaves, moverse al principio o al final de una función, identificación automática del código, etc. Zend Studio dispone de una herramienta muy interesante de depuración. Gracias a ella se puede ejecutar páginas y conocer en todo momento el contenido de las variables de

la aplicación y las variables del entorno como las cookies, las recibidas por formulario o en la sesión. Podemos colocar puntos de parada de los scripts y realizar las acciones típicas de depuración. (32)

1.10.2 SVN (SubVersion)

Software libre bajo una licencia de tipo Apache/BSD, sistema de control de versiones. Un sistema de control de versiones es un sistema de gestión de archivos y directorios, cuya principal característica es que mantiene la historia de los cambios y modificaciones que se han realizado sobre ellos a lo largo del tiempo. De esta forma, el sistema es capaz de "recordar" las versiones antiguas de los datos, lo que nos permite examinar el histórico de cambios o recuperar versiones anteriores de un fichero, incluso aunque haya sido borrado. Una característica importante de SVN es que los archivos versionados no tienen números de revisión independiente, en cambio, todo el repositorio tiene un único número de versión que identifica un estado común de todos los archivos del repositorio en cierto punto del tiempo. Empleado para el control de versiones a lo largo del desarrollo del sistema.

Las principales características de SVN y sus mejoras frente a CVS son (33):

- Mantiene versiones no sólo de archivos, sino también de directorios.
- También se mantienen versiones de los metadatos asociados a los directorios.
- Además de los cambios en el contenido de los documentos, se mantiene la historia de todas las operaciones de cada elemento, incluyendo la copia, cambio de directorio o de nombre.
- Atomicidad de las actualizaciones. Una lista de cambios constituye una única transacción o actualización del repositorio. Esta característica minimiza el riesgo de que aparezcan inconsistencias entre distintas partes del repositorio.
- Posibilidad de elegir el protocolo de red. Además de un protocolo propio (svn), puede trabajar sobre http (o https) mediante las extensiones WebDAV.
- Soporte tanto de ficheros de texto como de binarios.
- Mejor uso del ancho de banda, ya que en las transacciones se transmiten sólo las diferencias y no los archivos completos.
- Mayor eficiencia en la creación de ramas y etiquetas que en CVS.

1.10.3 Apache 2.4.0

Servidor HTTP de código abierto, software libre, presenta entre otras características, mensajes de error altamente configurables, bases de datos de autenticación y negociado de contenido con amplia aceptación en la red. Multiplataforma, lo que lo hace prácticamente universal. Es un servidor Web conforme al protocolo HTTP/IP. Modular, por lo que puede ser adaptado a diferentes entornos y necesidades, con los diferentes módulos de apoyo que proporciona. Incentiva la realimentación de los usuarios, obteniendo nuevas ideas, informes de fallos y parches para la solución de los mismos. Es una tecnología gratuita de código fuente abierto, esto le da una transparencia de manera que si queremos ver que es lo que estamos instalando como servidor, lo podemos saber, sin ningún secreto, sin ninguna puerta trasera. Extensible: gracias a ser modular se han desarrollado diversas extensiones entre las que destaca PHP (34). Apache permite mantener un servidor HTTP *open-source,* disponible para varios sistemas operativos de red, como las principales versiones de UNIX, *Windows NT y Mac.* (35)

1.11 Metodologías de desarrollo

Es un proceso de desarrollo donde se definen técnicas y procedimientos para llevar a cabo el software. No existe hasta el momento una metodología que sea utilizada de forma universal. Existen diversas y cada una con sus características propias, pero en todas, el propósito es el mismo y es que el proceso sea configurable. En este caso se utilizará el Proceso Unificado de Desarrollo (RUP), a continuación se justifica el porqué de su elección.

RUP, llamada así por sus siglas en inglés *Rational Unified Process* (36), es una metodología orientada a objetos basada en UML. Es una de las metodologías más utilizadas a nivel mundial. Tiene una forma organizada y disciplinada de asignar las tareas y responsabilidades. Además, posee un desarrollo iterativo e incremental, centrado en la arquitectura y dirigido por los casos de uso. Utiliza una arquitectura basada en componentes. También tiene un modelado visual del software, control de cambios y verificación de la calidad del software. Incluye además artefactos y roles. Permite la personalización según las necesidades del cliente. (37)

Otras metodologías como la programación extrema (XP) pueden ser muy buenas para algunos proyectos. XP se nutre del ancho de banda más grande que se puede obtener cuando existe algún tipo de comunicación, la comunicación directa entre personas (38), pero entre sus características está

el tener muy en cuenta e involucrar al cliente en el proceso de desarrollo. Este desarrollo en particular no tiene un cliente específico, y esto ha hecho descartar el empleo de XP. (39)

1.12 Sistemas de gestión de bases de datos

Un Sistema Gestor de base de datos (SGBD) es un conjunto de programas que permiten crear y mantener una Base de Datos, asegurando su integridad, confidencialidad y seguridad (40). Los SGBD constituyen un almacén de información organizada, pero además brindan las herramientas para manipular esta información. Uno de los sistemas de gestión de base datos fundamentales en el software libre son MYSQL y PostgreSQL.

1.12.1 MySQL

MySQL proporciona un servidor veloz de base de datos SQL (*Structured Query Language*), multi-hilo, multiusuario y robusto. El servidor está proyectado tanto para sistemas críticos en producción soportando intensas cargas de trabajo como para empotrarse en sistemas de desarrollo masivo de software.

El software MySQL tiene licencia dual, pudiéndose usar de forma gratuita bajo licencia GNU o bien adquiriendo licencias comerciales de MySQL AB en el caso de no desear estar sujeto a los términos de la licencia GPL. MySQL es una marca registrada de MySQLAB. (41)

MySQL posee un alto rendimiento y velocidad. Tiene utilidades de administración como respaldo y recuperación de errores, facilitándole la vida al desarrollador. Al ser integrado con PHP se obtienen excelentes resultados. Es ilimitado el tamaño de registros. Los usuarios tienen acceso a las tablas.

1.12.2 PostgreSQL

Funciona bajo licencia BSD *(Berkeley Software Distribution),* además es *Full ACID compliant (Atomicity, Consistency, Isolation and Durability),* en español propiedades como Atomicidad (Indivisible), Consistencia, Aislamiento, Durabilidad. La velocidad del código de su motor de datos ha sido incrementada aproximadamente en un 20-40%, y su tiempo de arranque ha bajado el 80% desde que la versión 6.0 fue lanzada, esto se debe a la arquitectura de su diseño. Soporta transacciones desde la versión 7.0. Posee características orientadas a objetos. Gran escalabilidad y rendimiento bajo grandes cargas de trabajo. (42)

A pesar de las ventajas y características muy particulares que presenta PostgreSQL, se selecciona como gestor de base de datos a MySQL por su gran compatibilidad con el lenguaje PHP y con el CMS Drupal. Debe agregarse que Drupal también puede funcionar con PostgreSQL.

1.13 Conclusiones del capítulo

Seleccionadas las herramientas, metodologías, tecnologías y conociendo el estado actual de los Agentes Recuperadores de Información, están creadas las condiciones iniciales para enfrentar el desarrollo del producto final. Este primer capítulo establece los cimientos de la investigación y el fundamento teórico a consultar para comprender el camino que se transita.

Capítulo II: Agente Recuperador de Información en Tiempo Real a desarrollar. Características del mismo como sistema

El presente capítulo, basado en la teoría expuesta con anterioridad, profundiza en cuanto a características y funcionamiento como sistema del Agente JITIR a desarrollar. La información que se procesa, descripción general del sistema como elemento funcional y objeto de automatización son aspectos tratados.

El software se fabrica siguiendo todo un proceso, por ello debe aplicarse alguna metodología de desarrollo, en este caso RUP. El modelado del negocio, la especificación de requisitos funcionales y no funcionales así como la definición de los casos de uso, se integran a las características del sistema.

2.1 Necesidad de un Agente Recuperador

Todo sistema orientado a la manipulación de información en la web tiende a incrementar el contenido manejable. La persona dedicada a operar con la información necesita alguna herramienta que le facilite la gestión de la misma en función de su trabajo, de otra forma tendría que establecer un mecanismo de recuperación manual.

En los sistemas de prensa, revistas digitales y todo sitio web que publique habitualmente contenido noticioso, es indispensable para los redactores recuperar artículos ya publicados y vincularlos con las nuevas redacciones. El agente a desarrollar, denominado ***IMAGENT***, pretende cubrir tales necesidades para el proceso de redacción en sitios web concebidos con cualquier versión del CMS DRUPAL 6. Entiéndase que el agente no será concebido para una organización específica ni un proceso de negocio determinado, cualquier institución que tenga como objetivo automatizar la recuperación y vinculación de la información podría usarlo.

2.2 Información que se maneja.

Mientras ocurre el proceso de redacción, IMAGENT recuperará constantemente el contenido y este podrá tratarse como documento. Serán filtradas fechas, siglas y palabras clave para construir consultas que aplicadas a las bases de datos del sitio web arrojen una colección de documentos, estos analizados, evaluados y ordenados en términos de relevancia antes de sugerirlos al usuario. El

conjunto de todos los documentos va a constituir la información manejable por el agente.

2.3 Estructura y Funcionalidad de IMAGENT

A continuación se abordan algunos aspectos considerados claves o fundamentales en la descripción general del sistema propuesto como vía de solución del problema a resolver.

2.3.1 Intercambio con el servidor, peticiones y respuestas.

Monitorizar el proceso de redacción, extraer y filtrar la información en tiempo real implica el chequeo de eventos y realización de múltiples peticiones y respuestas al y desde el servidor, el crecimiento de la información almacenada en bases de datos es directamente proporcional a las peticiones y respuestas. En ese sentido se busca un mecanismo que aporte fiabilidad, agilidad y optimice el flujo de datos evitando sobrecarga de peticiones con datos innecesarios, además de comunicación asincrónica con el servidor. Para cubrir estos requisitos se propone una pequeña librería con el uso de AJAX que medie entre el área de redacción y el mecanismo de búsqueda, encargada de controlar los distintos eventos y transportar datos útiles del cliente al servidor y viceversa.

2.3.2 Mecanismo de búsqueda y manejo de información

El mecanismo de búsqueda tendrá un diseño orientado a objetos y un conjunto de clases controlarán todo el proceso al que sea sometida la información, una clase controladora será gestora del flujo de trabajo entre las distintas clases, además del intercambio de información con la interfaz. A medida que el redactor escriba, serán extraídos el párrafo donde se encuentra el cursor, y si existieran se extraen además el párrafo superior y el inferior. Como parte del análisis de texto van a ser eliminadas palabras de poco aporte a la búsqueda como pronombres, artículos, preposiciones y palabras comunes del castellano, tomando solamente las consideradas “palabras clave” para convertirlas a minúsculas. Se filtrarán fechas y siglas otorgándole mayor valor que a las palabras clave. Se construirá un índice de búsqueda asignando un peso a cada elemento (palabra clave, sigla, fecha, número) según su longitud, tipo y cantidad de repeticiones, este índice refleja las necesidades de información del usuario. Se conformará una consulta con los elementos del índice que recuperará los documentos de la base de datos. Cada documento recuperado será evaluado respecto a la consulta a través del Modelo del Espacio Vectorial que, aplicando una fórmula matemática al dúo consulta-documento para la frecuencia de un término y frecuencia inversa de un documento (7) otorga un peso

al documento en término de relevancia. Los documentos de mayor relevancia serán estructurados en formato XML antes de devolverlo como respuesta, será etiquetado el título, texto íntegro, URL, autor, información del autor y un resumen del documento.

Se almacenará en una memoria caché la información de los siete documentos de mayor relevancia en cada proceso completo de recuperación y la consulta que lo recuperó, si en determinado momento se repitiera la consulta no sería necesario explorar la base de datos y tratar los documentos recuperados porque estarían disponibles en la caché, este mecanismo va a contribuir con la velocidad de respuesta del agente, especialmente cuando se recupere gran volumen de información. En cada proceso nuevo de redacción la caché de sugerencias será limpiada totalmente.

Es preciso señalar que los agentes recuperadores estudiados, exceptuando a Plora, utilizan Savant como sistema de recuperación, IMAGENT tomará de esta herramienta la idea de crear el índice de búsqueda y el índice de los archivos recuperados para evaluar la información pero no empleará tal herramienta. IMAGENT tendrá sus propias estrategias de recuperación y evaluación del contenido, cada algoritmo a implementar será diseñado por los autores con el objetivo de engranar el proceso de recuperación y evaluación de documentos e integrarle a los mismos aspectos y particularidades que no consideran otros agentes, ejemplo: la diferenciación de los pesos entre siglas, fechas y palabras clave, el empleo de memoria caché para evitar explorar la base de datos con consultas repetidas, etc.

2.3.3 Presentación de la información relevante

La información relevante será presentada al usuario en forma de sugerencias y para no saturar el área de resultados se mostrarán de manera ordenada las siete sugerencias más relevantes en un elemento de bloque HTML (43). La clase control va a encargarse de la estructuración de la información a mostrar. La intrusión en el proceso de redacción es imprescindible para no distraer al redactor de su tarea principal, por ello será evitada la saturación de colores, cambio de formas y tamaño del contenido, así como la sobrecarga de información.

2.3.4 Modelo del Espacio Vectorial en la evaluación de la información

Al aplicar el Modelo del espacio Vectorial mediante el algoritmo Okapi en su versión TF/iDF (7) Frecuencia de un Término y Frecuencia inversa de un Documento, en inglés: *Term Frequency and Inverse Document Frequency*, y adaptándolo al agente IMAGENT, cada documento será representado mediante un vector de n elementos, siendo n igual al número de términos del índice de

búsqueda. Habrá entonces, un vector para cada documento, y en cada vector, un elemento para cada término o palabra de posible aparición en el documento. Cada uno de esos elementos será cubierto u ocupado con un valor numérico. Si la palabra no está presente en el documento, ese valor es igual a 0. En caso contrario, ese valor será calculado teniendo en cuenta el tipo de elemento y la cantidad de ocurrencias, dado que una palabra puede ser más o menos significativa, este valor se conoce con el nombre de peso del término en el documento.

Una consulta (considera un documento) es representada también mediante un vector de las mismas características que las de los documentos, variando los valores numéricos de cada elemento en función de las palabras que forman parte de la consulta y la cantidad de ocurrencias en el índice de búsqueda. Esto permite calcular fácilmente una función de semejanza entre el vector de una consulta y el de un documento, de manera que, aquellos documentos que, en teoría, se ajustan más a la consulta formulada, producen un índice más alto de semejanza. En la siguiente figura podemos observar la representación gráfica de la semejanza entre dos documentos.

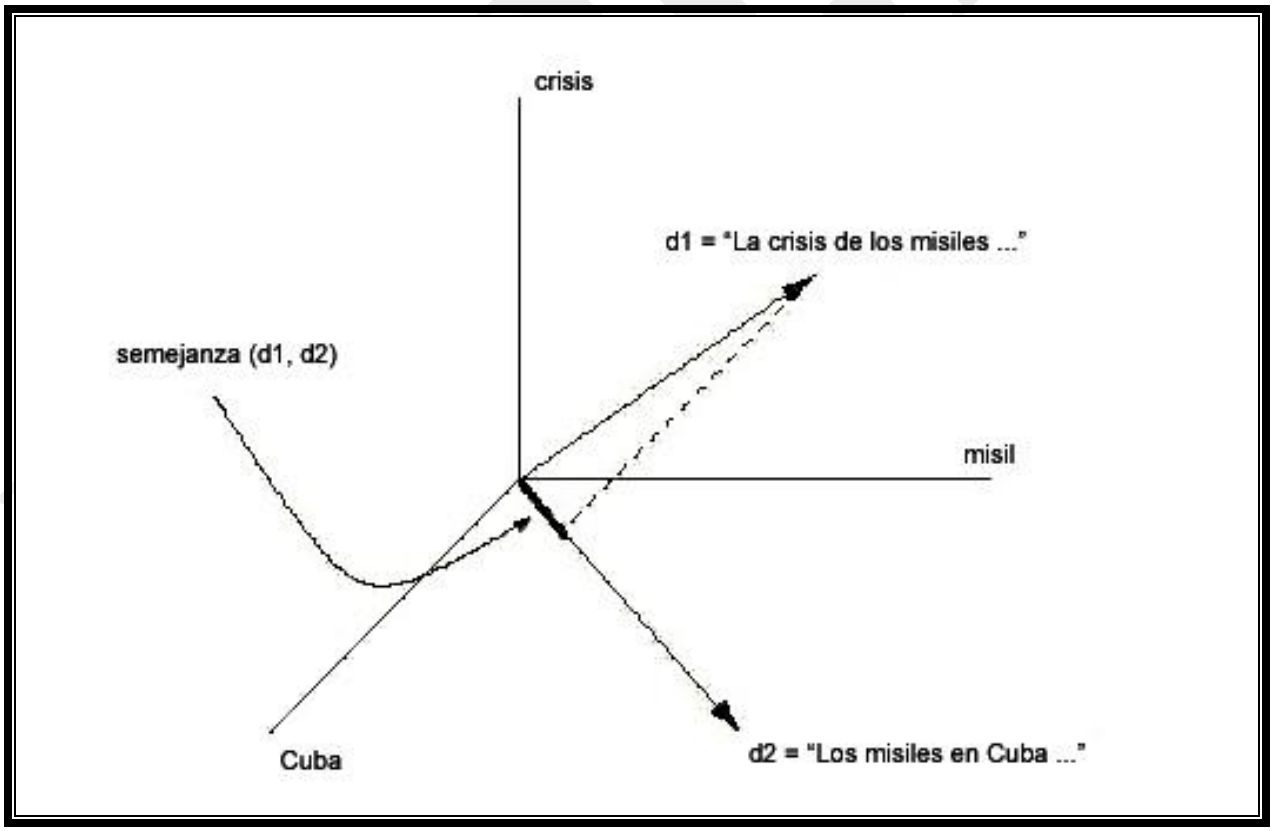

Figura.1: Representación gráfica de la semejanza entre dos documentos.

2.3.5 Control de eventos y vinculación con editores de texto

El agente IMAGENT no estará vinculado a un editor de texto específico, al ser instalado como módulo se va a adherir al espacio de redacción de cada tipo de contenido. Que un usuario utilice o no un editor de texto será una cuestión inherente al uso del agente recuperador. El agente chequeará y controlará los eventos del área de redacción de cada tipo de contenido, si el redactor se mantiene escribiendo o leyendo sin seleccionar una porción del texto el evento desencadenará una búsqueda completa, actualizando el índice y el búfer con documentos nuevos. Si el redactor selecciona una porción del texto el evento activará los botones que permiten establecer vínculos, presionando el botón de establecer vínculo sobre alguna sugerencia serán añadidas al texto seleccionado las etiquetas correspondientes para consumar la acción una vez concluido el proceso de redacción y publicado el documento. Si el redactor posiciona la flecha del ratón sobre alguna sugerencia mostrada, el evento desencadenará una acción para mostrar en un globo superpuesto la información completa de la sugerencia para posibilitar la lectura.

2.4 Modelo de dominio

Cuando no se logra establecer trabajadores, actores y procesos del negocio, se identifican y definen conceptos tratando de relacionarlos mediante clases en un modelo distinto que ofrezca una visión del funcionamiento del sistema, este modelo es el de dominio. Las clases se obtienen del conocimiento de unos pocos expertos del dominio o posiblemente del conocimiento asociado con sistemas similares al que se desarrolla.

El agente recuperador IMAGENT no estará concebido para una organización específica ni un proceso de negocio determinado, cualquier usuario con la intención de automatizar la recuperación y vinculación de la información podría usarlo. No existe un proceso de negocio establecido, solo el proceso de recuperación y vinculación de contenido dentro del dominio de la redacción.

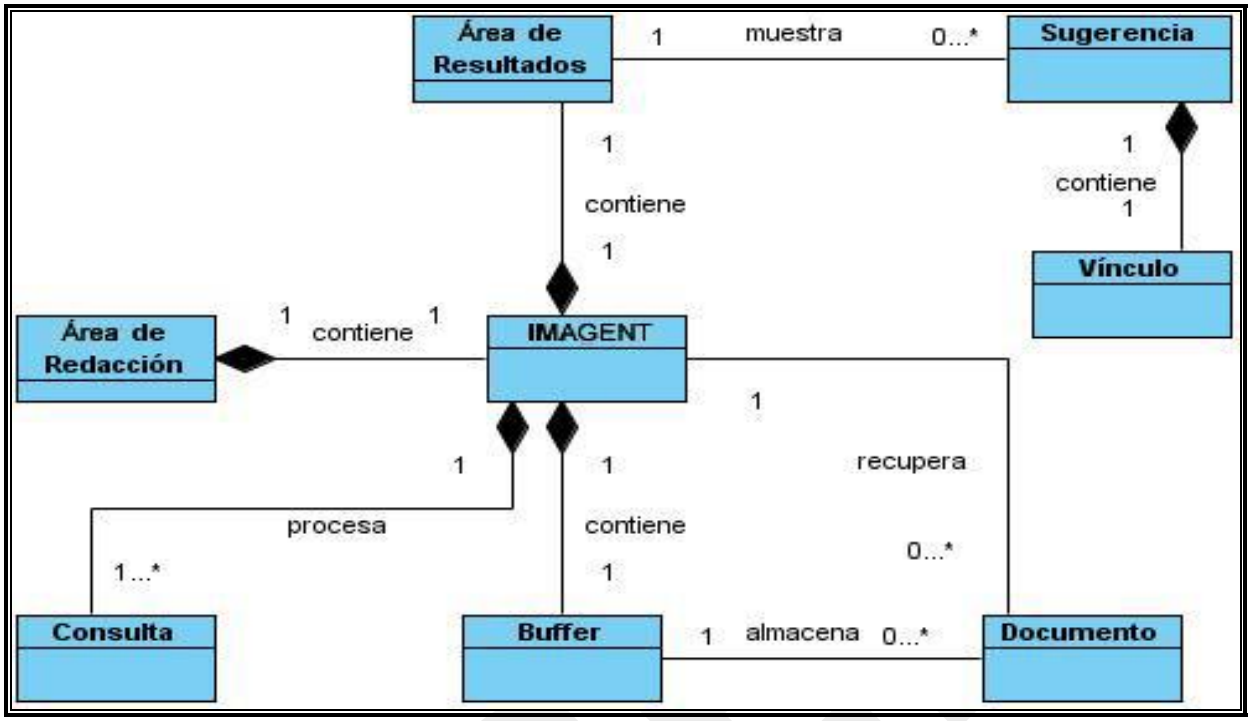

Figura 2: Modelo de Dominio.

2.5 Entidades y conceptos principales

Concepto	Descripción
Área de Redacción	Espacio de una página web donde el redactor escribe texto a publicar.
Consulta	Pregunta aplicada a una base de datos que devuelve contenido almacenado en ésta como respuesta.
IMAGENT	Agente Recuperador de Información que ofrece documentos relacionados en forma de sugerencias y la posibilidad de vincularlos con la redacción.
Área de Resultados	Espacio contenedor de sugerencias con botones de vínculos que se actualiza paulatinamente mientras se redacta.
Vínculo	Posible relación entre el documento sugerido y determinada porción de la redacción. Se consuma la acción por medio de un botón.

Sugerencia	Pequeña descripción de un documento relevante recuperado por IMAGENT a través del cual el usuario puede consultar el contenido del documento y vincularlo con la redacción.
Búfer	Espacio donde se almacena información completa y formateada de documentos relevantes recuperados. Se actualiza tomando como referencia la relevancia de los nuevos documentos que se recuperan y los ya almacenados.
Documento	Artículo, noticia u otro contenido texto-informativo almacenado en la(s) Base(s) de Datos del sitio web. Posee título, autor, cuerpo, identificador, etc.

Tabla 1: Entidades y conceptos principales.

2.6 Especificación de los requisitos de software

En la especificación de requisitos se capta y muestra de modo apropiado las funcionalidades y cualidades que debe poseer el producto con el objetivo de lograr un mejor entendimiento entre usuarios y desarrolladores. Además, una buena elección de los requerimientos asegura que el mantenimiento futuro de la aplicación conste de un orden y seguimiento detallado.

2.6.1 Requisitos funcionales

Los requisitos funcionales (RF) son las características requeridas por el sistema que expresan capacidades de acción del mismo, definidas a continuación:

- **RF 1:** Mostrar la información recuperada en forma de sugerencias.
- **RF 2:** Posibilitar la lectura completa de la información recuperada sin abandonar el ambiente de redacción.
- **RF 3:** Vincular la información recuperada con la redacción.

2.6.2 Requisitos no funcionales

Los requisitos no funcionales, más que comportamientos detallados, especifican los criterios que se pueden utilizar para juzgar la operación de un sistema, aseguran que se disponga de un sistema manejable y gestionable que ofrezca la funcionalidad requerida de manera fiable, ininterrumpida o con el tiempo mínimo de interrupción, incluso ante situaciones inusuales.

- **Apariencia o interfaz externa:** La interfaz del módulo debe ser amigable y sencilla, de modo que sea inconsecuente para el trabajo del usuario.
- **Usabilidad:** El módulo puede ser utilizado por cualquier persona que tenga conocimientos básicos de computación.
- **Rendimiento:** Mostrar las sugerencias no debe exceder los 6 segundos.
- **Soporte:**
    - Intérprete de aplicaciones Web (Navegador) capaz de interpretar JavaScript.
    - Sitio Web creado con el CMS Drupal 6.X.
    - Servidor apache 1.5 o superior.
    - Sistema Gestor de Bases de Datos MySQL 4.0 o superior.
    - Versión de PHP 5.0 o superior.
- **Portabilidad:** El módulo funciona sobre Linux o Windows, es multiplataforma.
- **Seguridad:** Un administrador debe controlar los permisos según el rol de cada usuario. Todo usuario con permisos para crear contenidos puede hacer uso del módulo.
- **Políticos-culturales:** El módulo debe respetar los términos empleados normalmente por los especialistas en el tema de la esfera que se automatiza. El agente podrá ser utilizado por usuarios, entidades o empresas que lo necesiten, tanto a nivel nacional como internacional.
- **Confiabilidad:** Para que el módulo se considere confiable, debe cumplir con las normas de seguridad y buenas prácticas de programación establecidas por el equipo de desarrollo del sitio oficial de la comunidad de Drupal[4].
- **Ayuda y documentación en línea:** Se debe incluir una ayuda de fácil entendimiento, para satisfacer las necesidades del usuario en cuanto a funcionalidad, comprensión e información del módulo se refiere.
- **Legales:** La plataforma y herramientas utilizadas para el desarrollo del agente recuperador deben poseer licencia GNU/GPL.

[4] Disponible en: http://drupal.org/writing-secure-code.

2.7 Diagrama de Casos de Uso del Sistema

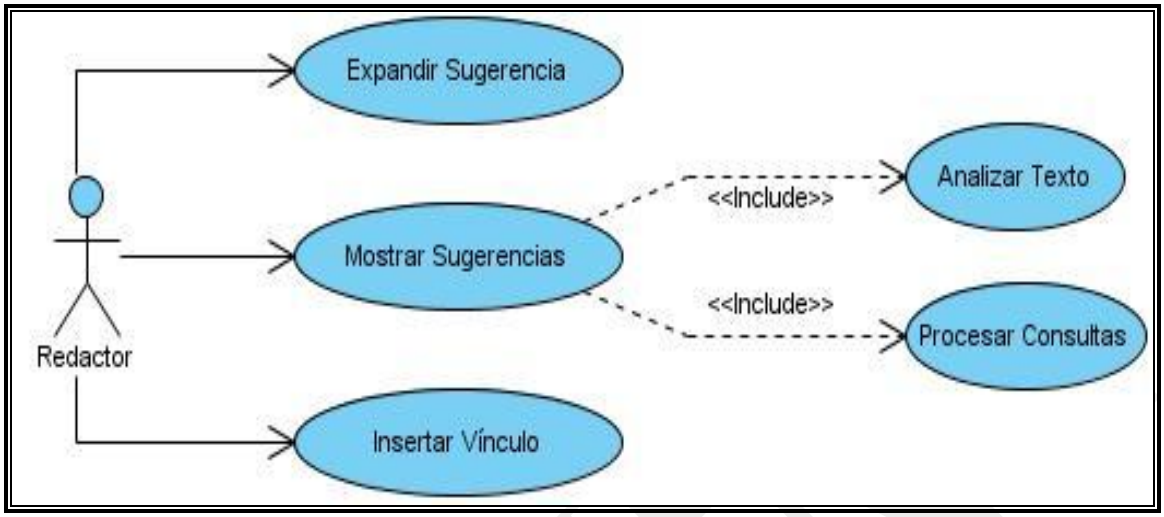

Figura 3: Diagrama de Casos de Uso del Sistema.

2.8 Descripción de los Casos de Uso del Sistema

CU –1	Mostrar Sugerencias
Propósito	Mostrar sugerencias de documentos que tengan la mayor relevancia respecto al contenido redactado por el usuario.
Actor	Redactor
Resumen	Este caso de uso inicia cuando el redactor escribe en el área de redacción, luego de analizar el texto, procesar las consultas y construir las sugerencias, estas son mostradas al usuario.
Referencias	R1
Precondiciones	El actor haya escrito en el área de redacción
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**

1. El redactor escribe en el área de redacción.	**2.** El sistema extrae el párrafo donde se encuentra el cursor, de existir se extraen además el párrafo superior y el inferior. **3.** El sistema analiza el texto extraído. **4.** El sistema procesa las consultas. **5.** El sistema recoge 7 de los documentos de mayor relevancia almacenados en el búfer, construye las sugerencias y las muestra al usuario en el área de resultados.
Flujo Alternativo	
Acción del actor	**Respuesta del sistema**
1. El redactor no escribe en el área de redacción.	**2.** El sistema muestra un mensaje en el área de resultados en espera del texto. **5.** El sistema no muestra ninguna sugerencia, por no haber almacenados documentos en el búfer del sistema.
Prioridad	Crítico.
Puntos de extensión	Línea 3: Ver Caso de Uso Analizar Texto. Línea 4: Ver Caso de Uso Procesar Consultas.

Tabla 4: Descripción del Caso de Uso Mostrar Sugerencias.

CU –2	Analizar Texto
Propósito	Conformar un índice de palabras eliminando las menos significativas y dejando las de mayor relevancia, con las que se realizarán las consultas.
Actor	Redactor

Resumen	Se toma el texto del área de redacción, extrayendo siglas, fechas y palabras clave por la importancia de dichos términos para la búsqueda, descartando preposiciones, conjunciones, artículos, palabras vacías del castellano, etc. Luego se conforma un índice de palabras con un peso dado su valor, tamaño, y tipo.
Referencias	R1, Caso de Uso Mostrar Sugerencias
Precondiciones	El actor haya escrito en el área de redacción.
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El redactor escribe en el área de redacción.	**2.** El sistema elimina las palabras de poco aporte a la búsqueda como pronombres, artículos, preposiciones y palabras comunes del castellano y toma solamente las palabras clave. **4.** El sistema convierte las palabras clave a minúsculas. **5.** El sistema filtra fechas y siglas otorgándole mayor valor en la búsqueda. **6.** El sistema construye un índice de palabras otorgándole un peso a cada una de ellas según su longitud y cantidad de repeticiones.
Flujo Alternativo	
Acción del actor	**Respuesta del sistema**
1. El redactor no escribe en el área de redacción.	**2.** El sistema muestra un mensaje en el área de redacción en espera del texto.
Prioridad	Crítico.

Tabla 5: Descripción del Caso de Uso Analizar Texto.

CU –3	Procesar Consultas
Propósito	Recuperar documentos utilizando consultas, analizarlos según su relevancia y almacenarlos en el búfer del sistema.
Actor	Redactor
Resumen	Dado un índice de palabras se toma cada una o pequeño grupo de ellas y construyen consultas que devuelven un conjunto de documentos, estos son analizados en término de relevancia, formateado en XML y almacenados en el búfer de IMAGENT.
Referencias	R1, Caso de Uso Mostrar Sugerencias
Precondiciones	El actor haya escrito en el área de redacción.
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El redactor escribe en el área de redacción.	**2.** El sistema construye las consultas que recuperan los documentos con cada palabra o conjunto de ellas. **3.** El sistema evalúa los documentos recuperados según la consulta. **3.1** Se le otorga un peso empleando el modelo del espacio vectorial y aplicando una fórmula matemática al par consulta-documento para la frecuencia de un término y frecuencia inversa de un documento. **3.2** El sistema otorga un peso adicional por la ocurrencia de palabras del índice. **4.** El sistema almacena en el búfer los documentos de mayor relevancia estructurados en formato XML para mejorar en su momento el acceso a sus detalles.

Flujo Alternativo	
Acción del actor	**Respuesta del sistema**
1. El redactor no escribe en el área de redacción.	**2.** El sistema muestra un mensaje en el área de redacción en espera del texto. **3.** El sistema no realiza la evaluación por no haber encontrado documentos como resultado de recuperación.
Prioridad	Crítico.

Tabla 6: Descripción del Caso de Uso Procesar Consultas.

CU –4	Expandir Sugerencia
Propósito	Permitir al usuario la lectura de la sugerencia en su totalidad.
Actor	Redactor
Resumen	Se construye un bloque contenedor de la información completa de una sugerencia determinada, mostrándola al usuario sin abandonar el ambiente de redacción.
Referencias	R2
Precondiciones	El actor haya escrito en el área de redacción.
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El redactor coloca el cursor encima de una de	**2.** El sistema muestra un globo de texto sobre la sugerencia, mostrando toda su información con el objetivo de facilitar la lectura.

las sugerencias expuestas en el área de resultados.	
Prioridad	Crítico.

Tabla 7: Descripción del Caso de Uso Expandir Sugerencia.

CU –5	Insertar Vínculo
Propósito	Permitir establecer vínculos entre las sugerencias mostradas y un texto seleccionado por el usuario.
Actor	Redactor
Resumen	El actor selecciona determinada porción del texto redactado, se activan los botones de vínculos en las sugerencias. Accionando uno de ellos se adicionan al texto las etiquetas correspondientes para que el contenido se publique vinculado al documento referenciado.
Referencias	R3
Precondiciones	El actor haya escrito en el área de redacción.
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**

1. Una porción del texto es seleccionada por el redactor. **3.** El redactor presiona el botón de la sugerencia elegida para vincular.	**2.** El sistema habilita un botón por cada sugerencia para realizar el vínculo. **4.** El sistema coloca las etiquetas de HTML necesarias para vincular el texto seleccionado con la sugerencia elegida.
Prioridad	Crítico.

Tabla 8: Descripción del Caso de Uso Insertar Vínculo.

2.9 Conclusiones del Capítulo

El empleo de AJAX en la confección de una pequeña librería que sirve de enlace entre el mecanismo de búsqueda y la presentación de resultados a través de componentes HTML, tributa a la velocidad de respuesta de las páginas servidoras del agente. Con el empleo del formato XML y de un búfer para almacenar resultados relevantes se logra la característica deseable de separar los procesos de búsqueda y presentación

Capítulo III: Análisis y diseño del Agente Recuperador de Información IMAGENT

El presente capítulo dirige su atención a la presentación de los artefactos correspondientes al flujo de trabajo análisis y diseño. Este capítulo tiene el propósito de transformar los requerimientos en un diseño de lo que será el sistema, adaptado al ambiente de implementación.

3.1 Análisis

El análisis tiene lugar al obtener una visión del funcionamiento del sistema, sobre la base de los requisitos funcionales. Lo componen clases y paquetes que establecen la estructura interna del sistema. Tiene como meta la comprensión de los requisitos definidos para el software, sin tener en cuenta el modo de solución. Las clases que se identifican están asociadas al dominio del problema, por ello representan conceptos y relaciones.

En el agente recuperador “IMAGENT” se ubica una clase interfaz por cada interacción actor-caso de uso, sin tomar en cuenta el empleo de más de una ventana en la solución. Dos clases se encargan del control de los objetos que invocan las funcionalidades para cumplir el trabajo de los casos de uso. Dos clases entidades fundamentales representan el manejo de la información persistente. A continuación se muestran los diagramas del modelo de clases del análisis para los casos de uso identificados.

3.1.1 Diagrama de clases del Análisis

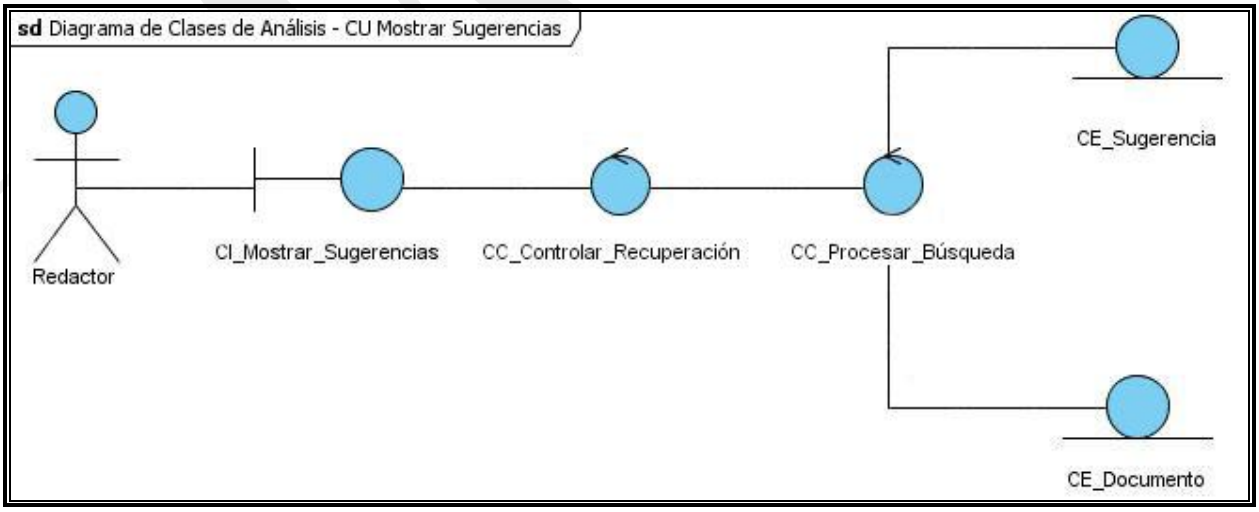

Figura 4: Diagrama de Clases del Análisis del CU Mostrar Sugerencias.

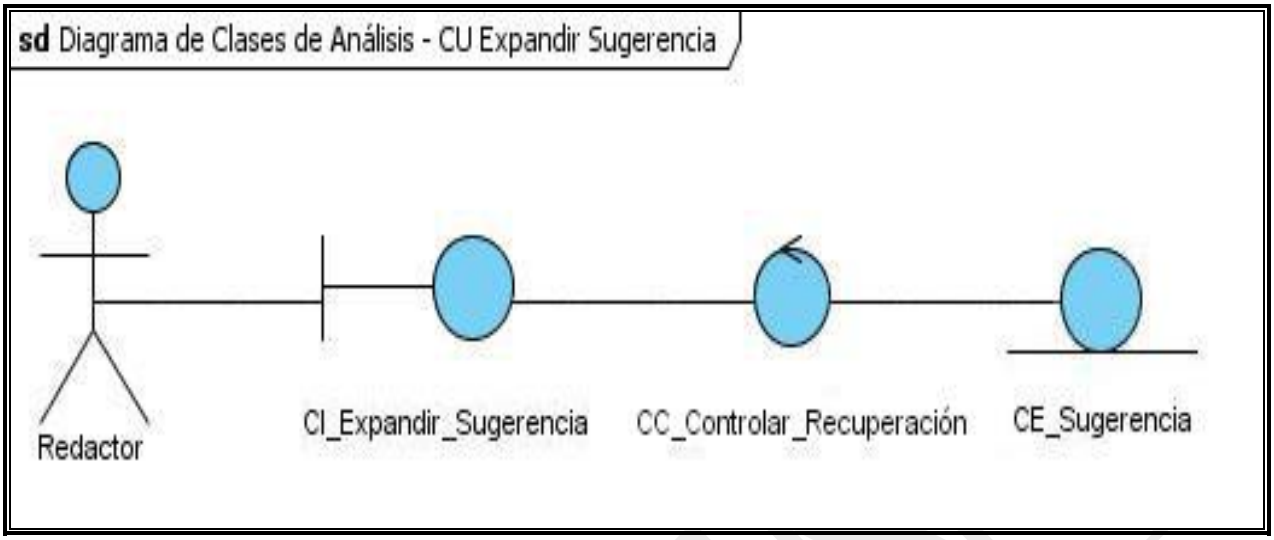

Figura 5: Diagrama de Clases del Análisis del CU Expandir Sugerencias.

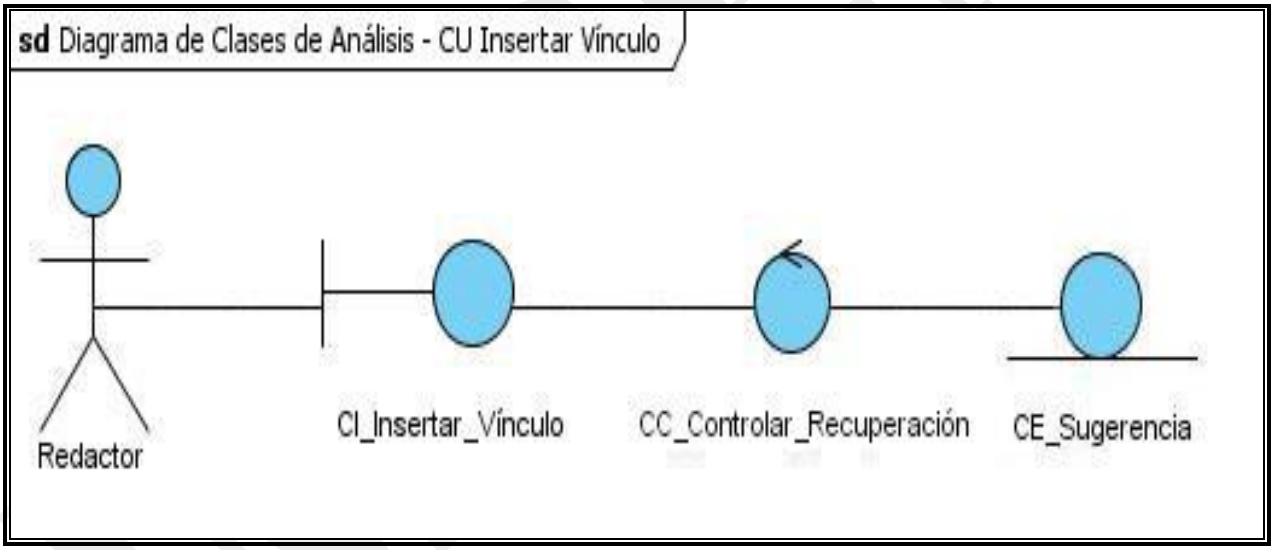

Figura 6: Diagrama de Clases del Análisis del CU Insertar Vínculo.

3.2 Diseño

El diseño enriquece y refina el resultado del análisis, se consideran los requisitos no funcionales, la forma en que el sistema cumple sus objetivos y se agregan nuevas clases que aportan a la estructura del software. El diseño debe garantizar que el sistema pueda ser implementado sin contratiempos y especifica el lenguaje de programación a utilizar en el flujo de trabajo de implementación.

IMAGENT posee un diseño orientado a objetos, potenciando la reutilización de código y aportando a las funciones descritas en los casos de uso flexibilidad ante los cambios. Para el manejo de la información que se envía al servidor y se recibe del mismo se emplean un conjunto de lenguajes, conformando lo que se conoce como AJAX que utiliza HTML y CSS para crear una presentación basada en estándares, DOM para la interacción y manipulación dinámica de la presentación, XML para el intercambio y la manipulación de información, el objeto XMLHttpRequest para el intercambio asíncrono con el servidor y JavaScript como lenguaje integrador. Del lado del servidor el lenguaje empleado es PHP.

3.2.1 Diagramas de Interacción

En esta etapa, las clases ya tienen definidas las operaciones que en el análisis estaban siendo comprendidas. Un diagrama de secuencia muestra la interacción entre un conjunto de objetos de la aplicación, mediante mensajes que se envían entre sí, en una secuencia de tiempo. A continuación se muestran las representaciones de los diagramas de colaboración correspondientes a cada uno de los casos de usos.

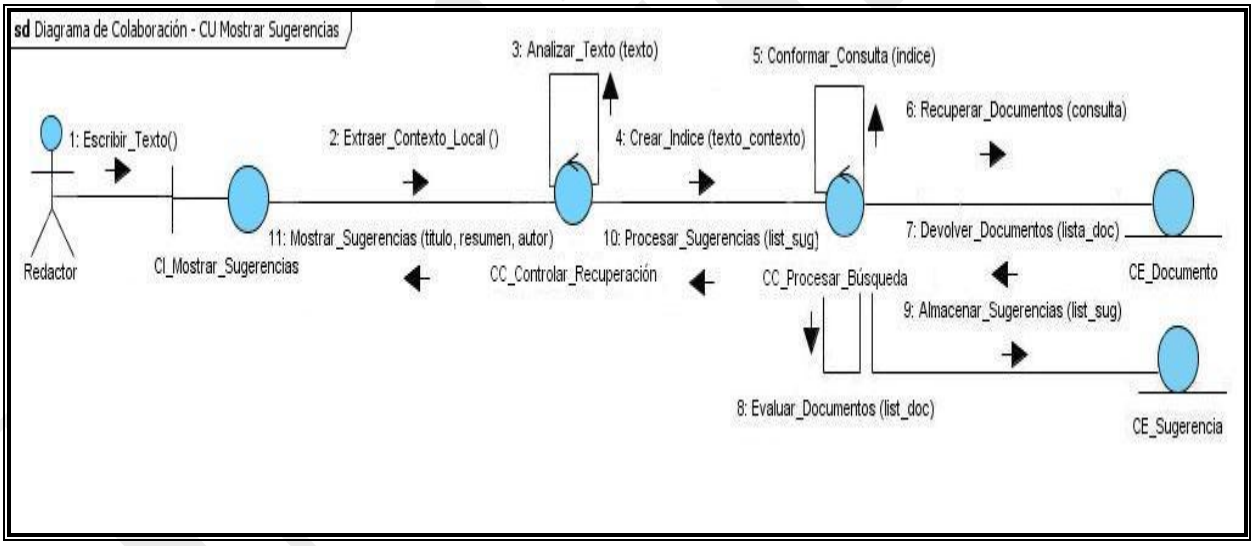

Figura 7: Diagrama de interacción del CU Mostrar Sugerencias.

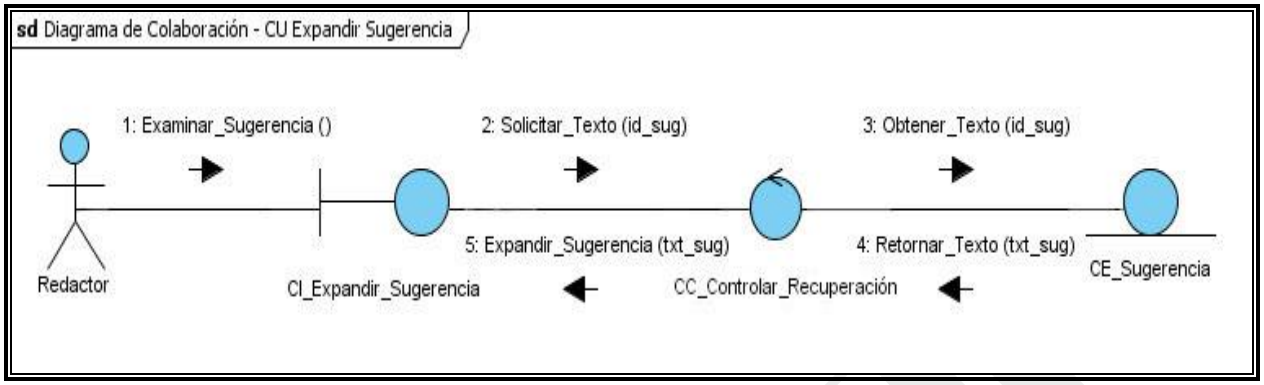

Figura 8: Diagrama de interacción del CU Expandir Sugerencia.

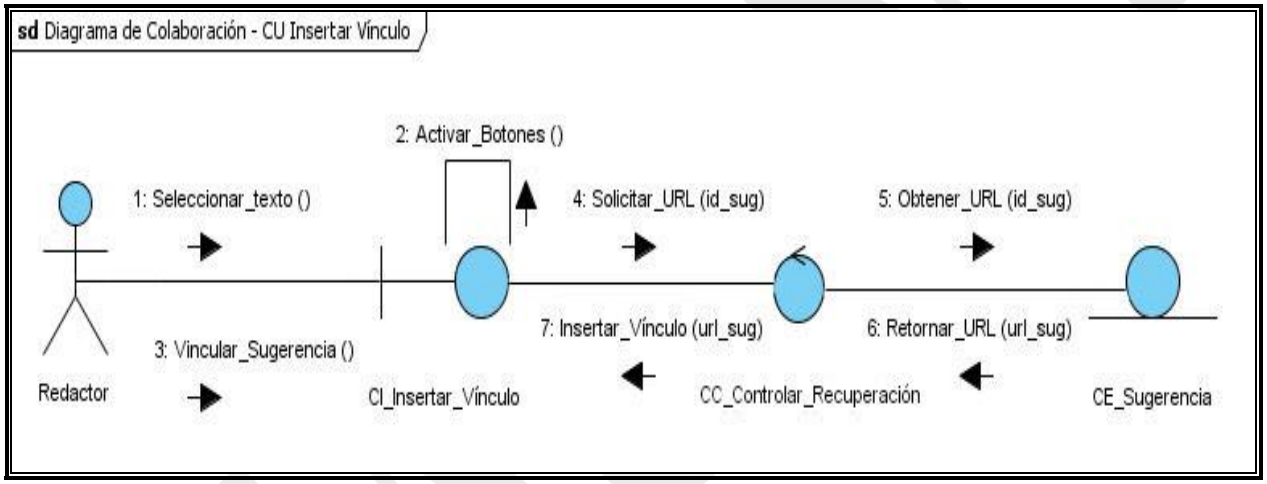

Figura 9: Diagrama de interacción del CU Insertar Vínculo.

3.2.2 Diagrama de Clases del Diseño

A continuación se presenta el diagrama de clases de diseño web del sistema, considerando su diseño orientado a objetos, obtenido como resultado del análisis y basado esencialmente en los diagramas de interacción.

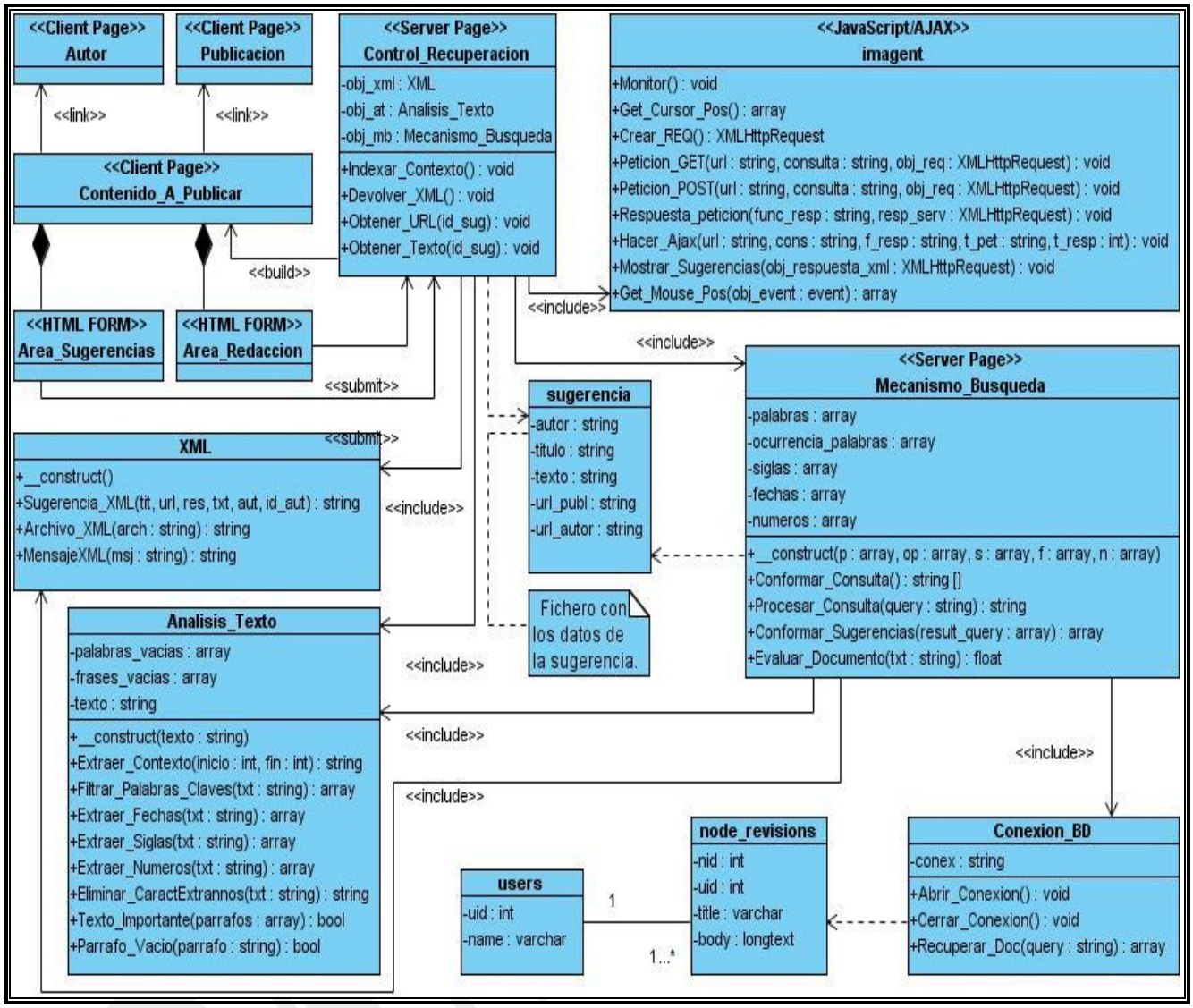

Figura 10: Diagrama de estereotipos WEB.

3.2.3 Descripciones de las clases del diseño

Nombre: imagent	
Tipo de clase: interfaz	
Atributo	**Tipo**

Para cada responsabilidad:	
Nombre:	**Crear_REQ ()**
Descripción:	Identifica el navegador en uso y crea el objeto indicado para la comunicación asincrónica con el servidor, retorna dicho objeto.
Nombre:	**Peticion_GET (url, consulta, obj_req)**
Descripción:	Envía peticiones de tipo GET al servidor.
Nombre:	**Peticion_POST (url, consulta, obj_req)**
Descripción:	Envía peticiones de tipo GET al servidor.
Nombre:	**Respuesta_peticion (func_resp, resp_serv)**
Descripción:	Ejecuta la función pasada por parámetro con el objeto de respuesta entregado por el servidor.
Nombre:	**Hacer_AJAX (url, consulta, funct_resp, tipo_pet, tipo_resp)**
Descripción:	Utiliza las funciones anteriores para la comunicación asincrónica con el servidor y llamada a funciones de procesamiento de respuestas.
Nombre:	**Monitor ()**
Descripción:	Monitorea el área de redacción, gestiona eventos y controla la comunicación con el servidor y el procesamiento de respuestas. Hace uso directo de la función Hacer_AJAX.
Nombre:	**Mostrar_Sugerencias (obj_respuesta_xml)**
Descripción:	Procesa respuestas XML y construye de forma dinámica los elementos XHTML necesarios para la presentación de sugerencias y confección de eventos asociados a las mismas.

Nombre:	**Get_Cursor_Pos ()**
Descripción:	Devuelve la posición inicial y final del cursor dentro del área de redacción.
Nombre:	**Get_Mouse_Pos (obj_event)**
Descripción:	Devuelve las coordenadas de posición del puntero del mouse para ubicar de forma correcta, cuando se requiera, el globo de información de cada sugerencia.

Tabla 9: Clase imagent.

Nombre: Control_Recuperacion	
Tipo de clase: controladora	
Atributo	**Tipo**
obj_xml obj_at obj_mb	XML Analisis_Texto Mecanismo_Busqueda
Para cada responsabilidad:	
Nombre:	**Indexar_Contexto ()**
Descripción:	Emplea obj_at para extraer siglas, fechas, números, palabras claves y cantidad de ocurrencias, con estos datos se instancia la clase Mecanismo_Busqueda a través del objeto obj_mb conformando así el índice de búsqueda.
Nombre:	**Devolver_XML ()**
Descripción:	Emplea obj_mb para conformar, efectuar y procesar consultas partiendo del índice de búsqueda, además de conseguir las sugerencias y/o

	mensajes informativos con que responde el servidor. Si la consulta generada está almacenada en el búfer o caché de sugerencias se extrae de aquí la respuesta evitando efectuar y procesar consultas. Utiliza el objeto obj_xml para estructurar y enviar las respuestas del servidor en archivos XML.
Nombre:	**Obtener_URL (id_sugerencia)**
Descripción:	Para construir vínculos entre la redacción y un documento sugerido se necesita conseguir la url del mencionado documento, esta función es la encargada de localizar en el búfer o caché de sugerencias tal url a través del id recibido por parámetro. Utiliza el objeto obj_xml para estructurar y enviar las respuestas del servidor en archivos XML.
Nombre:	**Obtener_Texto (id_sugerencia)**
Descripción:	Para construir el globo de información con el texto íntegro de un documento sugerido se necesita conseguir el texto del mencionado documento, esta función es la encargada de localizar en el búfer o caché de sugerencias esta información a través del id recibido por parámetro. Utiliza el objeto obj_xml para estructurar y enviar las respuestas del servidor en archivos XML.

Tabla 10: Clase Control_Recuperacion.

Nombre: Mecanismo_Busqueda	
Tipo de clase: controladora	
Atributo	**Tipo**
palabras ocurrencia_palabras siglas fechas	array array array array

numeros	array
Para cada responsabilidad:	
Nombre:	**Conformar_Consulta ()**
Descripción:	Construye de forma dinámica la consulta a efectuar basándose en los atributos de la clase (índice de búsqueda) y teniendo en cuenta si poseen prefijos las tablas de la Base de Datos. Retorna en forma de cadena la consulta SQL si esta se genera, en el caso de índice vacío retorna -1.
Nombre:	**Procesar_Consulta (consulta)**
Descripción:	Emplea una instancia de la clase Conexion_BD para abrir una conexión a la Base de datos, conseguir la respuesta a la consulta recibida por parámetro y cerrar la conexión. Si la respuesta es distinta de -1 procede a conformar las sugerencias, almacenar en el búfer o caché de sugerencias el conjunto de respuestas en estructura de archivo y retornar el resultado en forma de cadena, en caso contrario informa de la inexistencia de documentos asociados retornando -1.
Nombre:	**Conformar_Sugerencias (resultado_consulta)**
Descripción:	Cada documento recibido por parámetro como resultado satisfactorio de la consulta es evaluado por su texto y teniendo en cuenta los atributos de la clase (índice de búsqueda), si el peso del documento clasifica entre los siete más relevantes se procede a conformar la sugerencia empleando una instancia de la clase XML, se almacena la sugerencia (titulo, URL, resumen, texto, autor, id del autor) ordenándola según su peso. Retorna una cadena con el contenido de la sugerencia estructurado en etiquetas XML.
Nombre:	**Evaluar_Documento (texto)**
Descripción:	Implementa en forma de algoritmo el modelo matemático del espacio

	vectorial para calcular la relevancia de un documento respecto a una consulta. Emplea una instancia de la clase Analisis_Texto para extraer del texto recibido por parámetro las siglas, fechas, números, palabras claves y cantidad de ocurrencias, conformando así el índice del documento. Utilizando los atributos de la clase (índice de búsqueda) que representa la consulta y el índice del documento, calcula a través del modelo matemático la relevancia en forma de valor numérico, este indicador de peso es retornado.

Tabla 11: Clase Mecanismo_Busqueda.

Nombre: Analisis_Texto	
Tipo de clase:	
Atributo	**Tipo**
palabras_vacias frases_vacias texto	array array string
Para cada responsabilidad:	
Nombre:	**Extraer_Contexto (inicio)**
Descripción:	Teniendo en cuenta el valor pasado por parámetros que indica la posición del cursor en el área de texto, se extrae del atributo texto el contexto local del usuario, este se considera como el párrafo donde se ubica el cursor y el párrafo inmediato superior e inmediato inferior si existiesen. El valor extraído es retornado en forma de cadena.
Nombre:	**Filtrar_Palabras_Claves (contexto_local_usuario)**
Descripción:	Extrae del texto recibido por parámetro las palabras clave y número de ocurrencias de cada una retornando ambos indicadores en forma de arreglo. En el proceso omite siglas, fechas, números, palabras y frases

	del idioma español que no aportan información al índice de búsqueda, elimina caracteres extraños y corrige acentos. No almacena palabras repetidas y asegura que las palabras clave sean relevantes en el proceso de recuperación de documentos.
Nombre:	**Extraer_Fechas (contexto_local_usuario)**
Descripción:	Extrae del texto recibido por parámetro las fechas para retornarlas en forma de arreglo, se considera que las mismas poseen un valor adicional en el proceso de recuperación de documentos.
Nombre:	**Extraer_Siglas (contexto_local_usuario)**
Descripción:	Extrae del texto recibido por parámetro las siglas y acrónimos para retornar en forma de arreglo, se considera que poseen un valor adicional en el proceso de recuperación de documentos.
Nombre:	**Extraer_Numeros (contexto_local_usuario)**
Descripción:	Extrae del texto recibido por parámetro los números para retornarlos en forma de arreglo, se considera que poseen un valor adicional en el proceso de recuperación de documentos.
Nombre:	**Eliminar_CaractExtrannos (texto)**
Descripción:	Retorna la cadena recibida por parámetro luego de extraer sus caracteres extraños.
Nombre:	**Parrafo_Vacio (parrafo)**
Descripción:	Analiza el texto recibido por parámetro y determina si existe alguna palabra que aporte al proceso de recuperación de documentos, retornando falso, en caso de no encontrar palabra alguna el párrafo se considera vacío por lo que la función retorna verdadero.

Tabla 12: Clase Analisis_Texto.

Nombre: XML	
Atributo	**Tipo**
Para cada responsabilidad:	
Nombre:	**Sugerencia_XML (titulo, url_publicacion, resumen, texto, autor , url_autor)**
Descripción:	Codifica la información recibida por parámetro y la estructura en etiquetas XML, retornando el contenido XML correspondiente a una sugerencia.
Nombre:	**Archivo_XML (contenido)**
Descripción:	Recibe por parámetro la estructura XML de un conjunto de sugerencias (las siete de mayor relevancia), enviándola como respuesta del servidor con estructura de archivo.
Nombre:	**Mensaje_XML (mensaje)**
Descripción:	Recibe por parámetro una cadena con la información de un mensaje, esta se codifica y la estructura en etiquetas XML, enviándola como respuesta del servidor con estructura de mensaje.

Tabla 13: Clase XML.

Nombre: Conexion_BD	
Atributo	**Tipo**
conex	string
Para cada responsabilidad:	

Nombre:	**Abrir_Conexion ()**
Descripción:	Establece conexión con la Base de Datos y almacena el identificador de la conexión en el atributo conex. El proceso de obtención de nombre, ubicación, usuario y contraseña de la Base de datos se realiza de forma automática, explorando los archivos de configuración de Drupal.
Nombre:	**Cerrar_Conexion ()**
Descripción:	Cierra la conexión con la Base de Datos cuyo identificador se almacena en el atributo conex.
Nombre:	**Recuperar_Doc (consulta)**
Descripción:	Efectúa la consulta recibida por parámetro y almacena en un arreglo el texto, titulo, identificador del documento, autor e identificador del autor para cada documento recuperado. Retorna un arreglo donde cada posición corresponde al arreglo con la información de un documento, retorna -1 en caso de no recuperar documento alguno.

Tabla 14: Clase Conexion_BD.

3.2.4 Descripción de las tablas

Nombre:	node_revisions	
Descripción:	Tabla donde se almacenan los registros de cada nodo creado.	
Atributo	**Tipo**	**Descripción**
nid	int	Identificador del nodo.
uid	int	Identificador del usuario que creó el nodo.
title	varchar	Título del nodo.

body	longtext	Cuerpo del nodo.

Tabla 15: Descripción de la tabla node_revisions.

Nombre:	users	
Descripción:	Tabla donde se almacenan los registros de los usuarios.	
Atributo	**Tipo**	**Descripción**
uid	int	Identificador del usuario.
name	varchar	Nick del usuario.

Tabla 16: Descripción de la tabla users.

3.2.5 Registros de caché

El empleo de registros de caché o búfer constituye un mecanismo para evitar la excesiva conformación, ejecución y procesamiento de consultas, tributando a la velocidad de respuesta de IMAGENT, especialmente para el trabajo con grandes volúmenes de información. A continuación se muestra la estructura de un fichero del registro de caché.

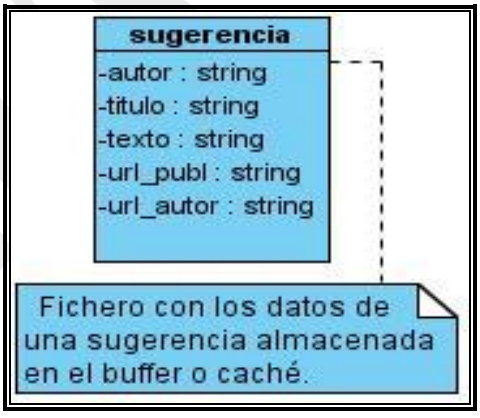

Figura 12: Estructura de un registro de caché.

Nombre:	sugerencia	
Descripción:	Fichero que almacena información de una sugerencia.	
Atributo	**Tipo**	**Descripción**
autor	string	Autor del documento sugerido.
titulo	string	Título del documento sugerido.
texto	string	Texto íntegro del documento sugerido.
url_publ	string	URL del documento sugerido.
url_autor	string	URL de la información del usuario autor del documento sugerido.

Tabla 17: Descripción del registro.

3.2.6 Arquitectura

IMAGENT es un módulo para el CMS Drupal, por lo que la arquitectura y los patrones usados se heredan de este. La arquitectura del CMS Drupal utiliza el patrón Modelo Vista Controlador (MVC). MVC es un patrón de arquitectura de software que separa la interfaz de usuario, los datos de una aplicación, y la lógica de control en tres componentes distintos. Este patrón se identifica con frecuencia en aplicaciones web, donde la vista es la página HTML y el código que provee de datos dinámicos a la página; el modelo es el Sistema de Gestión de Base de Datos y la Lógica de negocio; y el controlador es el responsable de recibir los eventos de entrada desde la vista. A continuación se presenta la estructura arquitectónica del CMS Drupal.

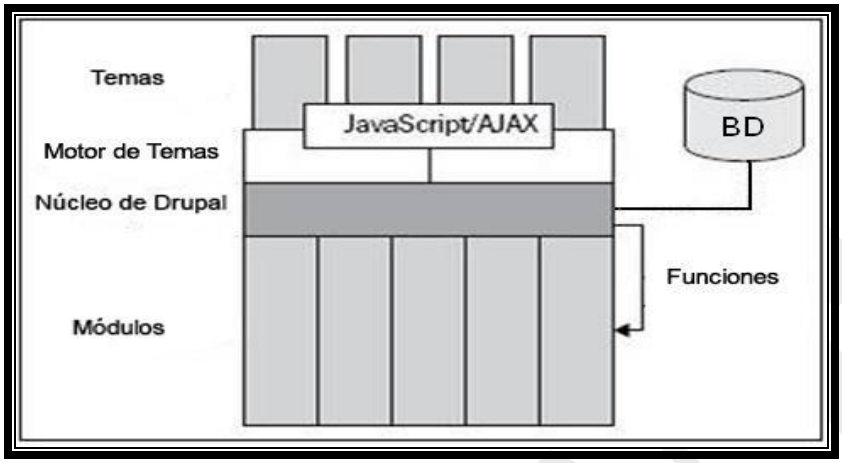

Figura 13: Patrón Modelo Vista Controlador del CMS Drupal.

Donde el modelo es la **Base de Datos (BD)**, la vista son los **Temas**, **Motor de Temas** y **JavaScript/AJAX**, el controlador lo componen el **Núcleo de Drupal** y los **Módulos**, y las **Funciones** son funciones que facilitan la comunicación entre el **Núcleo de Drupal** y los **Módulos**.

Drupal hace uso de técnicas como la herencia, el polimorfismo, el encapsulamiento, entre otras, todas características de la Programación Orientada a Objetos (POO). Por ende Drupal se diseña considerando patrones presentes en la mayoría de los sistemas orientados a objetos, los patrones GOF *(Gang Of Four).* Los patrones GOF están categorizados en creacionales, estructurales y de comportamiento. Dentro de los patrones que hacen de Drupal un sistema mucho más extensible y flexible se encuentran:

- *Singleton* (instancia única): dirigido a garantizar que una clase tenga una sola instancia y proporcionar un punto de acceso global a ella.
- *Decorator* (envoltorio): permite añadir dinámicamente funcionalidades a un objeto, posibilitando no tener que crear sucesivas clases que hereden de la primera incorporando la nueva funcionalidad, sino otras que la implementan y se asocian a la primera. Drupal hace un uso extensivo de este patrón, pudiendo citar el uso de /hook_nodeapi/ que permite a cualquier nodo extender el comportamiento de todos los nodos.
- *Observer* (observador): define una dependencia del tipo uno-a-muchos entre objetos, de manera que cuando uno de los objetos cambie su estado, el observador se encarga de notificar este cambio a todos los otros dependientes. Este también es muy utilizado en Drupal,

por ejemplo, implementando hook, estos se definen como observadores del objeto vocabulario, cualquier cambio que se realice es notificado y actualizado en los dependientes.

- *Bridge* (puente): que desacopla una abstracción de su implementación
- *Chain of Responsability* (cadena de responsabilidad): permite establecer la línea que deben llevar los mensajes para que los objetos realicen la actividad indicada.

3.2.7 Tratamiento de errores

El tratamiento de errores en las aplicaciones, de manera general ayuda a validar las posibles respuestas que pueden darse en determinado momento por el sistema. IMAGENT se limita a operar sobre las áreas de redacción y sugerencia, cualquier error que se produzca fuera de este ámbito es ignorado por el agente. La única entrada de datos es mediante el área de redacción, el redactor puede introducir cualquier tipo de texto, incluso caracteres extraños o sentencias con intenciones destructivas, el agente no chequea algún modelo de escritura correcta, solo filtra las palabras que considera importantes para el índice de búsqueda y desecha lo demás, garantizando que no se introduzcan errores al sistema o se produzcan otros por parte del agente.

3.2.8 Seguridad

En todo sistema la revelación de datos sensibles, modificación o eliminación de información que se maneja y protección ante ataques directos es determinante para garantizar la integridad, confidencialidad y disponibilidad de los datos. IMAGENT solo maneja la información que se considera publicada, generalmente noticias o artículos informativos. Además no envía a la base de datos consultas de alteración o borrado de la información, solo selecciones de la misma. Dichas consultas son construidas únicamente con palabras o términos previamente analizados, garantizando así la protección ante inyecciones SQL. Los datos transportados del cliente al servidor y viceversa viajan codificados, aportando también a la seguridad del sistema.

3.2.9 Interfaz

Parte del capítulo I aborda en profundidad el tema de las características para la interfaz de usuario de un agente JITIR, la interfaz de IMAGENT se subordina totalmente a tales planteamientos. Como parte del análisis y diseño debe agregarse que la funcionalidad operativa del agente a desarrollar, en especial la presentación de resultados, debe respetar los estilos CSS definidos por el desarrollador del sitio. Los elementos XHTML que construye IMAGENT heredan los tamaños, colores y todo tipo de

estilos del tema activo del sitio, evitando así que el desarrollador tenga que explorar y modificar de forma manual los estilos del módulo.

3.2.10 Concepción de la Ayuda

El módulo cuenta con un manual de usuario en formato .pdf destinado a orientar al usuario en la utilización del mismo. Siguiendo los pasos explicados y apoyándose visualmente en las imágenes expuestas en el manual se garantiza una correcta instalación y uso efectivo del producto. El nivel con que se detallan las operaciones asegura un fácil entendimiento y la explotación efectiva de IMAGENT por un usuario inexperto.

3.3 Conclusiones del capítulo

El presente capítulo arroja los artefactos correspondientes al flujo de trabajo análisis y diseño que propone RUP, definiendo así la estructura sobre la cual se llevará a cabo la implementación del sistema. La modelación y descripción de las clases que integran la estructura funcional del producto facilitan en gran medida la etapa de codificación de la aplicación.

Capítulo IV: Implementación y Pruebas del Agente Recuperador de Información en Tiempo Real IMAGENT

El presente capítulo documenta la implementación en términos de componentes del sistema analizado y diseñado en el capítulo III. Ofrece una vista del despliegue del sistema organizando sus partes en nodos. El sistema es ejecutado bajo aspectos particulares y circunstancias específicas, los resultados son observados, registrados y finalmente evaluados.

4.1 Implementación

El objetivo principal del flujo de trabajo implementación es desarrollar la arquitectura y el sistema como un todo. De forma más específica, los propósitos de la Implementación son:

- Definir la organización del código.
- Implementar clases y objetos en forma de componentes (código fuente, ejecutables, etc.).
- Probar los componentes desarrollados.
- Integrar los componentes como un sistema.

Conformando un modelo de implementación están los artefactos diagrama de componentes y diagrama de despliegue, pues describen los componentes a construir e indican la organización y dependencia entre nodos físicos en los que funcionará el sistema.

4.1.1 Diagrama de Despliegue

El Diagrama de Despliegue se utiliza para modelar la topología del hardware utilizado en la implementación y sobre el que se ejecuta el sistema. Los elementos usados en este diagrama son nodos, componentes y asociaciones. Los nodos son elementos físicos, que existen en tiempo de ejecución y representan un recurso computacional que generalmente tiene memoria y capacidad de procesamiento. Un nodo representa un procesador o un dispositivo sobre el que se pueden desplegar los componentes, por lo que el diagrama de despliegue sitúa el software en el hardware que lo contiene. A continuación se muestra el despliegue del sistema sobre tres nodos fundamentales.

- **PC Cliente:** Es el ordenador que permite a los usuarios acceder al sitio web y usar el agente IMAGENT durante el proceso de redacción, emplea el protocolo HTTP para la comunicación con el servidor web.
- **Servidor Web:** Almacena información en forma de páginas web y a través del protocolo HTTP

la presentan a petición de los clientes (navegadores web) en formato HTML.

- **Servidor de BD:** Es el nodo que almacena la información del sistema.

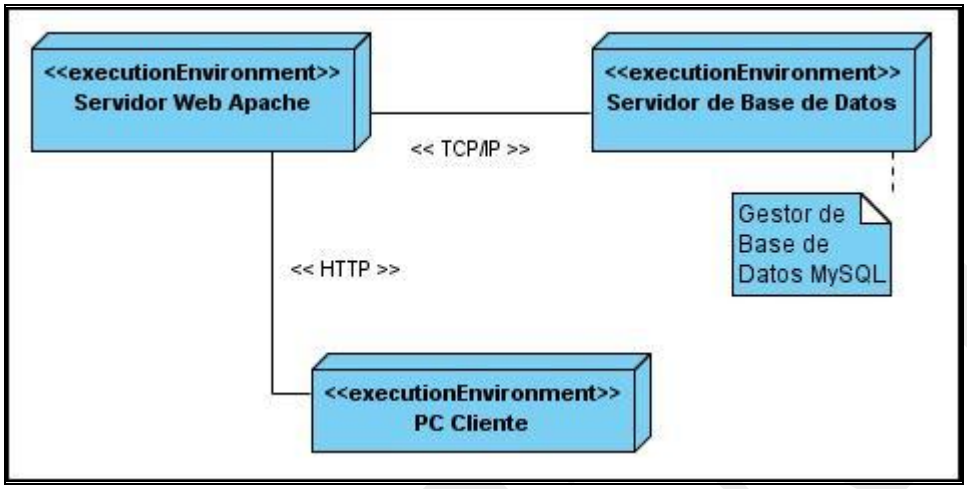

Figura 14: Diagrama de Despliegue.

4.1.2 Diagrama de Componentes

Un diagrama de componentes representa cómo un sistema es dividido en componentes que son utilizados en tiempo de compilación, mostrando la organización y las dependencias entre estos. Los componentes físicos incluyen archivos, cabeceras, bibliotecas compartidas, documentos, módulos, ejecutables, o paquetes que formen parte del sistema. Considerando los requisitos relacionados con la facilidad de desarrollo, la reutilización, las características de los lenguajes de programación y las herramientas utilizadas, se muestra a continuación los diagramas de componentes correspondientes al sistema.

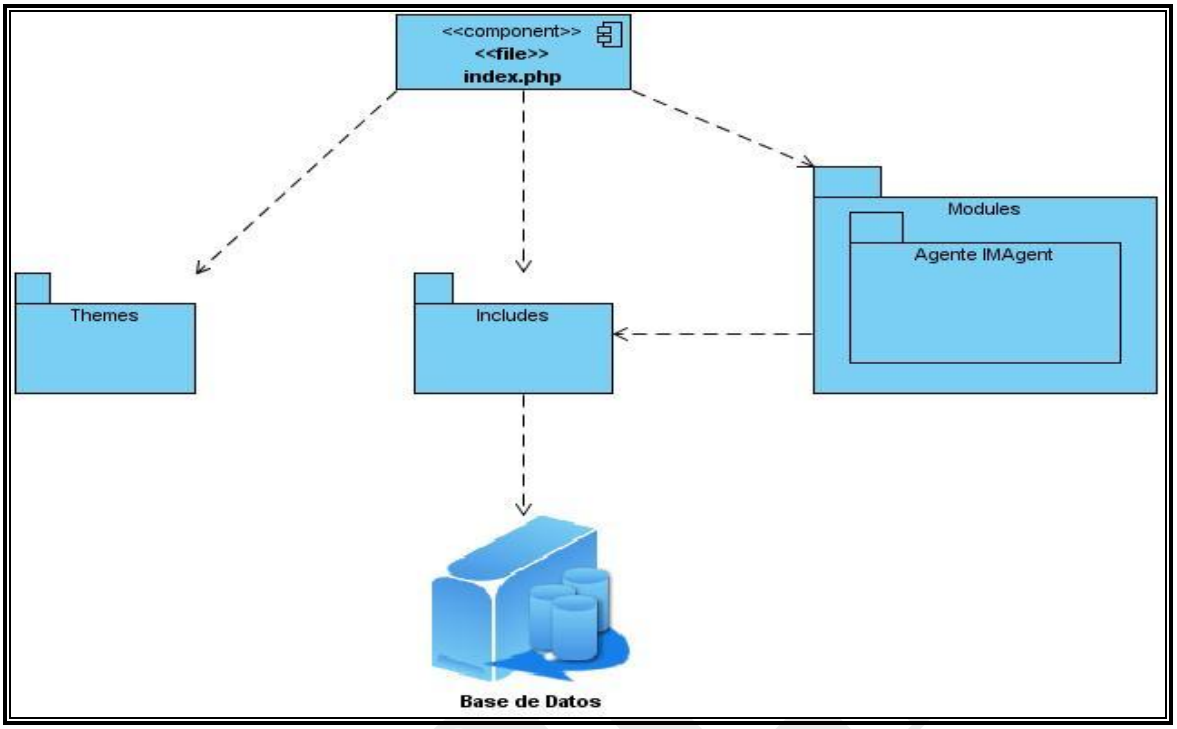

Figura 15: Diagrama de Componentes para el CMS Drupal.

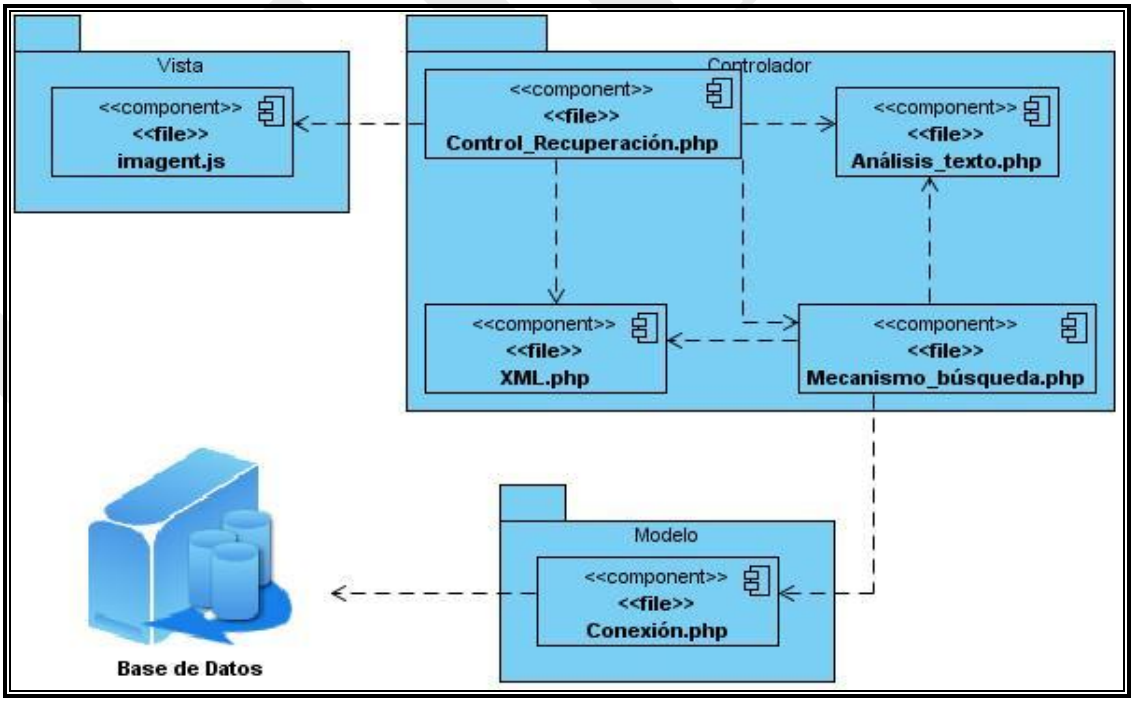

Figura 16: Diagrama de Componentes para el Agente IMAGENT.

4.2 Prueba

Desarrollar determinado software debe ir acompañado de una actividad que garantice su calidad, para asegurarse de ello la prueba de software es una etapa vital, pues representa una revisión final de las especificaciones del diseño y de la codificación, entre sus principales objetivos se encuentran:

- Verificar la interacción entre los objetos.
- Verificar la integración apropiada de componentes.
- Validar que se satisfacen los requerimientos.
- Identificar los defectos y corregirlos antes de dar por terminado el software.

Los casos de prueba intentan demostrar que el software es funcionalmente operativo, que las entradas son aceptadas y que se produce un resultado correcto, además de asegurar que se mantiene la integridad de la información. Para conformar los casos de prueba para IMAGENT se considera que su funcionalidad está orientada a la pro-actividad y que el redactor no posee restricción alguna en cuanto a la información que introduce al sistema y consecuentemente sirve de entrada al agente.

IMAGENT consta de dos áreas fundamentales para su funcionamiento, un área de redacción de donde se extrae el texto a considerar en el proceso de búsqueda y el área de resultados para mostrar las sugerencias de documentos asociados que hayan sido recuperados. De esta forma, se descartan las posibles entradas de datos a través de formularios, campos obligatorios y otros, que puedan generar ventanas emergentes, errores o dificultar la recuperación de información. Los casos de prueba están dirigidos a comprobar que el agente es capaz de cumplir con sus principales funcionalidades: la recuperación y presentación de información al usuario, más que a verificar el comportamiento del agente ante acciones indebidas sobre la interfaz.

4.2.1 Modelo de Prueba

Entrada	Respuesta del Sistema	Resultado
Área de redacción vacía.	El agente emite un mensaje en el área de resultados indicando que las sugerencias serán mostradas	Satisfactorio.

	durante el proceso de redacción.	
El redactor introduce en el área de redacción palabras de poco aporte al criterio de búsqueda.	El agente emite un mensaje en el área de resultados indicando que las palabras no aportan al criterio de búsqueda.	Satisfactorio.
El usuario escribe palabras que guardan relación con documentos almacenados.	El agente recupera y muestra en el área de resultados, un máximo de siete sugerencias, ordenadas de acuerdo con la relevancia de los documentos asociados al tema.	Satisfactorio.

Tabla 18: Casos de Prueba al CU Mostrar Sugerencias.

Entrada	Respuesta del Sistema	Resultado
El redactor coloca el cursor encima de una sugerencia expuesta en el área de resultados.	El agente muestra un globo de texto con el texto íntegro del documento correspondiente a la sugerencia.	Satisfactorio.

Tabla 19: Casos de Prueba al CU Expandir Sugerencias.

Entrada	Respuesta del Sistema	Resultado
El redactor selecciona una porción del texto redactado.	El agente habilita los botones de vínculos para cada una de las sugerencias.	Satisfactorio.

El redactor, una vez seleccionado el texto y activados los botones de vínculos en las sugerencias, acciona con clic sobre uno de estos.	El agente añade al texto seleccionado las etiquetas correspondientes para que el contenido se publique vinculado al documento referenciado.	Satisfactorio.

Tabla 20: Casos de Prueba al CU Insertar Vínculo.

4.3 Evaluación del Agente IMAGENT

A favor de evaluar la eficiencia del Agente IMAGENT en un ambiente real, se somete el mismo a un conjunto de pruebas adicionales sobre una base de datos de más de 500 publicaciones. A continuación se describen los resultados:

- **Prueba 1:** El tiempo de respuesta de IMAGENT para mostrar 7 sugerencias asociadas, con índices de búsqueda relativamente extensos si se compara con la longitud media de los párrafos en publicaciones reales, no alcanzó los 4 segundos, superando con creces el tiempo estimado en los requisitos no funcionales, este propone no más de 6 segundos.
- **Prueba 2:** Con el objetivo de comprobar la efectividad del algoritmo implementado para el cálculo de relevancia, se introducen a la base de datos 10 documentos con información relacionada únicamente con la temática “La Gran Muralla China”, antes se explora la base de datos garantizando que ningún otro documento almacenado aborda el tema. Tomando como referencia un texto intencionalmente relacionado con la temática, se aplica manualmente a cada documento el algoritmo para el cálculo de relevancia ya implementado, se obtienen los siguientes resultados (redondeados):

Documentos Asociado	Por ciento de Relevancia (%)
Documento 1	68
Documento 2	62
Documento 3	81

Documento 4	30
Documento 5	26
Documento 6	34
Documento 7	75
Documento 8	47
Documento 9	59
Documento 10	28

Tabla 21: Relevancia calculada manualmente.

Se introduce al área de redacción el texto empleado como referencia para el cálculo de la relevancia a los documentos de forma manual. IMAGENT presenta 7 sugerencias ordenadas descendentemente de acuerdo con el por ciento de relevancia de los documentos recuperados. Se muestra a continuación, los resultados arrojados por el agente:

Sugerencia	**Documento Asociado**	**Relevancia (%)**
Sugerencia 1	Documento 3	80,6
Sugerencia 2	Documento 7	74,5
Sugerencia 3	Documento 1	68,3
Sugerencia 4	Documento 2	62,0
Sugerencia 5	Documento 9	58,5
Sugerencia 6	Documento 8	47,4
Sugerencia 7	Documento 6	34,1

Tabla 22: Relevancia arrojada por IMAGENT.

De este modo y considerando la inexactitud del método manual, se observan solo pocas variaciones respecto a los por cientos obtenidos con anterioridad, garantizando la efectividad del cálculo de relevancia por parte de IMAGENT.

- **Prueba 3:** El agente desarrollado es probado en todas las versiones de Drupal 6.x existentes hasta la fecha, comprobando en cada caso todas las funcionalidades. Los resultados permiten afirmar que IMAGENT es totalmente funcional para todas las versiones de Drupal 6.x hasta la fecha.

4.4 Conclusiones del capítulo

Tomando por entrada principal el modelo de diseño, en este capítulo se obtuvieron los artefactos generados durante las actividades de los flujos de trabajo implementación y pruebas propuestas por la metodología de desarrollo empleada (RUP). Concluido este flujo de trabajo se obtuvo la primera versión del Agente Recuperador de Información en Tiempo Real, ***IMAGENT 1.0***, probado funcionalmente y de forma satisfactoria en simulaciones de entornos de trabajo reales y con gran cúmulo de información.

Conclusiones Generales

Considerando el auge del CMS Drupal en la edición y publicación de artículos y luego de identificar los efectos de la información desconectada en la web, particularmente en sitios concebidos con este CMS, se identificó la carencia de un módulo capaz de disminuir el efecto perjudicial de la información desconectada en las publicaciones.

Para acometer el desarrollo del Agente Recuperador de Información que solucionaría esta situación se estudiaron a fondo los agentes JITIR existentes, fundamentalmente los orientados a la web, caracterizándolos en cuanto a funcionamiento, evaluación y presentación de la información, extrayendo de cada uno las particularidades deseables para el nuevo agente. Una vez seleccionadas las herramientas, lenguajes, tecnologías a emplear se transitó por los flujos de trabajos propuestos por la metodología de desarrollo RUP. Así nació el Agente Recuperador de Información IMAGENT como un módulo más a instalar para el CMS Drupal. Desarrollado bajo el estilo arquitectónico MVC, el agente disminuye el efecto perjudicial de la Información desconectada.

Las pruebas acometidas al agente desarrollado prueban la calidad y eficiencia de las funcionalidades en simulaciones de ambientes reales de trabajo, obteniendo resultados satisfactorios. De esta forma, se cumplieron los objetivos específicos y la idea planteada al inicio de la investigación y defendida por los autores en el transcurso del trabajo.

Recomendaciones

Vencidos los objetivos del presente trabajo y teniendo presente las experiencias adquiridas durante el desarrollo del mismo, se recomienda:

- Proseguir el estudio de la Recuperación de Información como rama en constante evolución para garantizar la continuidad de la idea materializada en el agente desarrollado.
- Desarrollar agentes semejantes para otros entornos y herramientas centradas en la edición y publicación de contenido texto-noticioso en la web.
- Emplear el agente desarrollado en sitios web de la Universidad orientados a la edición y publicación de contenido texto-noticioso en la web, extender la propuesta a centros de estudios, diarios digitales e instituciones de ámbito nacional.

Bibliografía

1. **Alvarez, Miguel Angel.** desarrolloweb.com. [En línea] 2008. [Citado el: 20 de Diciembre de 2009.] http://www.desarrolloweb.com/articulos/que-es-un-cms.html.

2. Joomla! [En línea] 2010. [Citado el: 12 de Enero de 2010.] http://www.joomla.cl/que-es-un-cms.html.

3. **Merelo Guervos, Juan Julián.** Introducción a los sistemas de gestión de contenidos. [En línea] 2005. [Citado el: 12 de Enero de 2010.] http://geneura.ugr.es/~jmerelo/tutoriales/cms/.

4. Drupal. [En línea] 2010. [Citado el: 14 de Enero de 2010.] http://drupal.org/about.

5. PILOS. [En línea] Junio de 2009. [Citado el: 16 de Enero de 2010.] http://www.pilos.com.co/drupal/27-caracteristicas-de-drupal/.

6. **Rhodes, Bradley James.** *Just-In-Time Information Retrieval.* 2000.

7. **Rhodes, Bradley James y Maes, P.** *Just-In-Time Information Retrieval agents.* 2000.

8. **Navarra, Pablo Lara y Martínez Usero, José Angel.** *Agentes inteligentes en la búsqueda y recuperación de información.* Barcelona : s.n., 2006.

9. **Claver, Raquel Blanch, y otros.** *Just-in-Time Information Retrieval Agents (JITIR'S).* 2005.

10. **Hernández León, Rolando Alfredo y Coello González, Sayda.** *El Paradigma Cuantitativo de la Investigación Científica.* Ciudad de la Habana : s.n., 2002.

11. **Hernández Sampieri, Roberto, Fernández Collado, Carlos y Baptista Lucio, Pilar.** *Metodología de la Investigación.*

12. **Estrada Marchena, Dionisio y Fernández Pérez, José Ramón.** *Agente de recuperación de información Just-in-time (JITIR Agent) para el Proyecto de Informatización de la Prensa.* La Habana : s.n., 2007.

13. **James Rhodes, Bradley.** *The Wearable Remembrance Agent: A System for Augmented Memory.* 2005.

14. —. Free Software Directory. [En línea] 30 de Abril de 2003. [Citado el: 2 de Febrero de 2010.] http://directory.fsf.org/project/RemembranceAgent/.

15. **Haas, Juergen.** About.com Guide to Linux. [En línea] 2010. [Citado el: 9 de Febrero de 2010.]

http://linux.about.com/cs/linux101/g/remembranceagen.htm.

16. **College of Computing Georgia Institute of Technology.** *Wearable Computers as Intelligent Agents.* Atlanta : s.n.

17. **Barfield, Woodrow y Caudell, Thomas.** *Fundamentals of Wearable Computers and Augmented Reality.* 2005.

18. **James Rhodes, Bradley.** *Margin Notes. Building a Contextually Aware Associative Memory.* New Orleans : s.n., 2005.

19. **García Broncano, Rubén.** *RECUPERACIÓN Y ACCESO A LA INFORMACIÓN.* 2006.

20. **López Herrera, Antonio Gabriel.** *Modelos de Sistemas de Recuperación de Información Documental Basados en Información Linguística Difusa.* Granada : s.n., 2006.

21. medioscorp.com. [En línea] 2009. [Citado el: 15 de Febrero de 2010.] http://www.medioscorp.com/v4/index.php?option=com_content&view=article&id=26&Itemid=73.

22. **Alvarez, Miguel Angel.** desarrolloweb.com. [En línea] 2004. [Citado el: 16 de Febrero de 2010.] http://www.desarrolloweb.com/articulos/que-es-html.html.

23. —. desarrolloweb.com. [En línea] 13 de Junio de 2001. [Citado el: 20 de Febrero de 2010.] http://www.desarrolloweb.com/articulos/449.php.

24. **Monteiro Lazaro, Juliana.** desarrolloweb.com. [En línea] 2008. [Citado el: 20 de Febrero de 2010.] http://www.desarrolloweb.com/articulos/26.php.

25. **PHPBB Group.** ivemfinity.com. [En línea] 16 de Abril de 2010. [Citado el: 1 de Marzo de 2010.] http://www.ivemfinity.com/diferentes-lenguajes-de-programacion-para-web-t1908.html.

26. **Sierra Calderón, Ana.** Desarrollo de una aplicación Web para representación de datos de posicionamiento. [En línea] Junio de 2009. [Citado el: 27 de Febrero de 2010.] http://ddd.uab.cat/pub/trerecpro/2009/hdl_2072_48077/PFC_AnaSierraCalderon.pdf.

27. **Talleres y consultas de la materia modulo de software de la Universidad de Boyaca.** AJAX. [En línea] 21 de 7 de 2008. [Citado el: 2 de Marzo de 2010.] http://albertcm.wordpress.com/AJAX/.

28. **G. Allegue, Facundo y Bugaletto, Guillermo.** Lenguaje Unificado de Modelamiento. [En línea] Septiembre de 2001. [Citado el: 5 de Marzo de 2010.] http://www.neuronsrl.com.ar/training/uml/uml_intro.html.

29. **Guerrero, Jesús.** EL RAM. [En línea] 28 de Abril de 2009. [Citado el: 5 de Marzo de 2010.] http://www.fiec.up.ac.pa/elram/index.php?option=58&cont=7.

30. **Jiménez Ramos, Claudia.** *Herramientas CASE en desarrollo de Sistemas.* 2003.

31. **Sierra, Daniel.** Visual Paradigm For Uml. [En línea] 2007. [Citado el: 7 de Marzo de 2010.] http://www.slideshare.net/vanquishdarkenigma/visual-paradigm-for-uml.

32. **The PHP Company.** Zend. [En línea] 2010. [Citado el: 9 de Marzo de 2010.] http://www.zend.com/en/products/studio/.

33. **García, Luis.** OBSERVATORIO TECNOLÓGICO. [En línea] 17 de Enero de 2008. [Citado el: 11 de Marzo de 2010.] http://observatorio.cnice.mec.es/modules.php?op=modload&name=News&file=article&sid=548.

34. **San Félix, Alvaro del Castillo.** El servidor de web Apache: Introducción práctica. [En línea] 2000. [Citado el: 15 de Marzo de 2010.] http://acsblog.es/articulos/trunk/LinuxActual/Apache/html/x31.html.

35. Apache HTTP Server . [En línea] Febrero de 2009. [Citado el: 18 de Marzo de 2010.] http://www.quebajar.com/detallar-apache-http-server.html.

36. **Mendoza Sanchez, María A.** Metodologías De Desarrollo De Software. [En línea] 7 de Junio de 2004. [Citado el: 19 de Marzo de 2010.] http://www.informatizate.net/articulos/metodologias_de_desarrollo_de_software_07062004.html.

37. **Jacobson, I.** El Proceso Unificado de Desarrollo de Software. [En línea] 2000. [Citado el: 20 de Marzo de 2010.] http://bibliodoc.uci.cu/pdf/reg00060.pdf.

38. **Sánchez González, Carlos.** ONess. [En línea] 28 de Septiembre de 2004. [Citado el: 22 de Marzo de 2010.] http://oness.sourceforge.net/proyecto/html/ch05.html.

39. **Calero Solís, Manuel.** Una explicación de la programación extrema (XP). [En línea] 2003. [Citado el: 21 de Marzo de 2010.] http://www.willydev.net/descargas/prev/ExplicaXp.pdf.

40. Garbage Collector. [En línea] 1 de Noviembre de 2004. [Citado el: 23 de Marzo de 2010.] http://www.error500.net/garbagecollector/archives/categorias/bases_de_datos/sistema_gestor_de_base_de_datos_sgbd.php.

41. Mastermagazine.Definición de MySQL. [En línea] 15 de Febrero de 2005. [Citado el: 24 de Marzo de 2010.] http://www.mastermagazine.info/termino/6051.php.

42. **Quiñones A, Ernesto.** postgresql.org.pe. [En línea] 2006. [Citado el: 26 de Marzo de 2010.] http://postgresql.org.pe/articles/introduccion_a_postgresql.pdf.

43. **POZO, J. R.** HTML con Clase. La estructura de un documento HTML. [En línea] 21 de Junio de 2001. [Citado el: 5 de Abril de 2010.] http://html.conclase.net/tutorial/html/2/4.

Glosario de términos

IMAGENT: Agente Recuperador de Información para el CMS Drupal.

Cache: Tipo de memoria temporal utilizada para almacenar consultas y respuestas relacionadas con un contexto determinado.

CMS: Content Management Systems, Sistema de Gestión de Contenidos. Una herramienta que posibilita la creación y administración de contenidos, controlando mediante su interfaz, una o varias bases de datos donde se almacena el contenido del sitio web.

Información Desconectada: Se refiere a la desvinculación existente entre publicaciones digitales que tratan sobre una temática común.

Information Retrieval: Recuperación de información. Trata de obtener el contenido de la información que puede ser relevante para los usuarios.

JITIR Agent: Agente Recuperador de Información Just-in-time. Estos agentes muestran información relacionada con determinado tema, teniendo en cuenta el contexto local del usuario, información que puede estar almacenada en bases de datos, e-mails, carpetas personales y otros.

GNU/Linux: Es el término empleado para referirse al sistema operativo similar a Unix que utiliza como base las herramientas de sistema de GNU y el núcleo Linux.

GPL: La Licencia Pública General de GNU o más conocida por su nombre en inglés GNU General *Public License* o simplemente su acrónimo del inglés GNU GPL. Su propósito es declarar que el software cubierto por esta licencia es software libre y protegerlo de intentos de apropiación que restrinjan esas libertades a los usuarios.

Programación basada en prototipos: es un estilo de programación orientada a objetos en l cual, las "clases" no están presentes, y la re-utilización de procesos se obtiene a través de la clonación de objetos ya existentes, que sirven de prototipos, extendiendo sus funcionalidades.

RUP: Metodología de desarrollo de software basada en UML. Organiza el desarrollo de software en 4 fases.

WebDAV: es un protocolo que amplía las posibilidades del HTTP/1.1 añadiendo nuevos métodos y cabeceras.

Anexos

Anexo 1

Figura 17: Vista del sistema, funcionalidad Mostrar Sugerencias.

Anexo 2

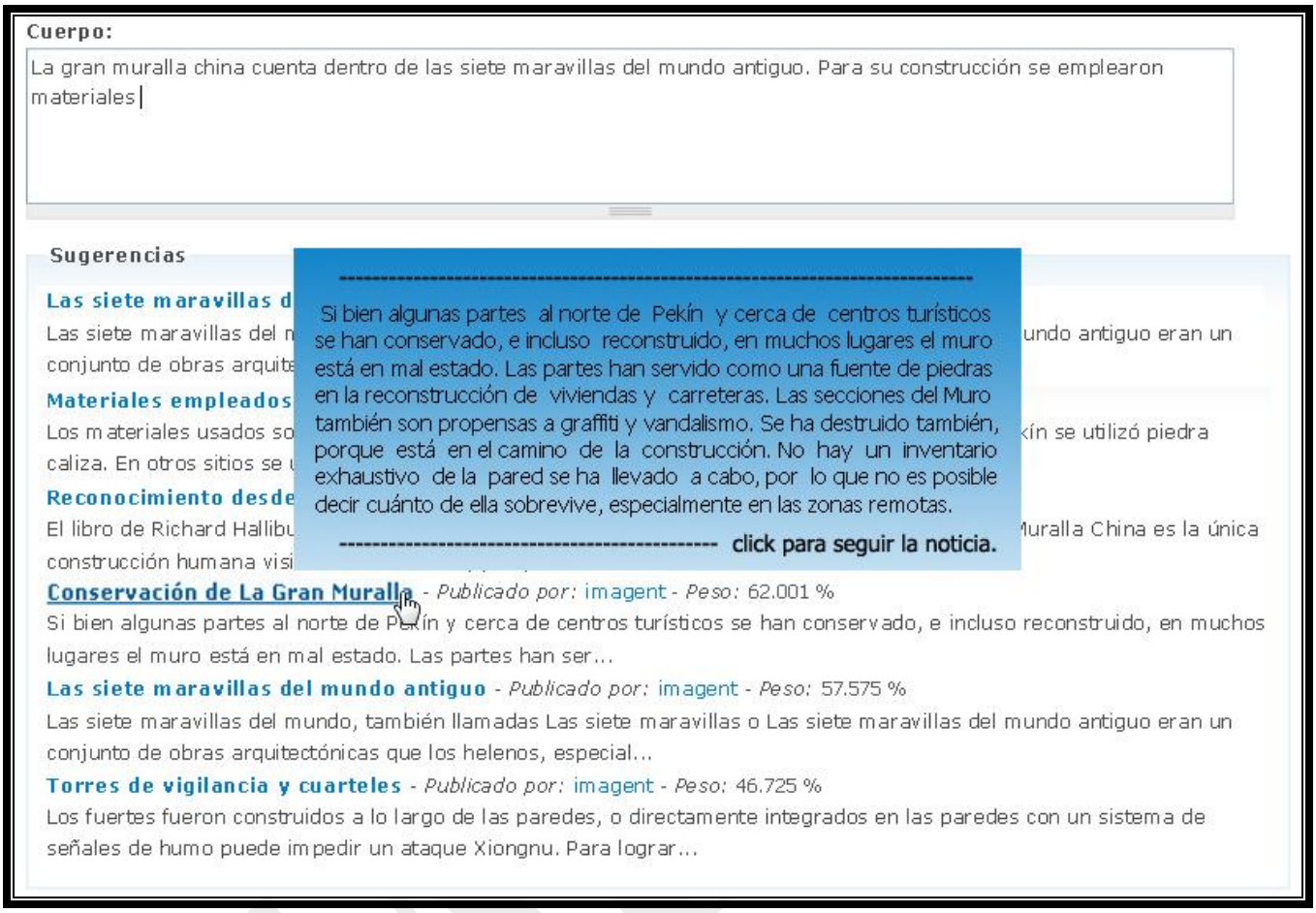

Figura 18: Vista del sistema, funcionalidad Expandir Sugerencia.

Anexo 2

Figura 19: Vista del sistema, funcionalidad Insertar Vínculo.