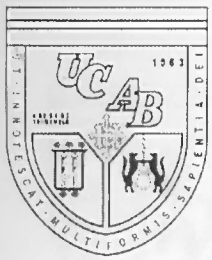

UNIVERSIDAD CATÓLICA ANDRÉS BELLO

FACULTAD DE INGENIERÍA

ESCUELA DE INGENIERÍA INFORMÁTICA

Sistema de búsquedas históricas actualizado en tiempo real a través de tecnología RSS

TRABAJO ESPECIAL DE GRADO

presentado ante la

UNIVERSIDAD CATÓLICA ANDRÉS BELLO

como parte de los requisitos para optar al título de

INGENIERO EN INFORMÁTICA

REALIZADO POR Toussaint, Fernando

PROFESOR GUÍA Bonalde, Gustavo

FECHA Caracas, Octubre de 2007.

Índice General

DEDICATORIA I

AGRADECIMIENTOS II

ÍNDICE DE TABLAS VIII

ÍNDICE DE FIGURAS IX

SINOPSIS 1

INTRODUCCIÓN 2

CAPITULO I. PROBLEMA 3

I.1. Planteamiento del problema 3
I.2. Objetivo general 6
I.3. Objetivos Específicos 6
I.4. Limitaciones 7
I.5. Alcance 8
I.6. Justificación 8

CAPITULO II. MARCO TEÓRICO 11

II.1 World Wide Web 11
II.1.1 Hypertext Markup Language (HTML) 12
II.1.2 JavaScript 12
II.1.3 Dynamic HTML (DHTML) 12
II.1.4 Cascading Style Sheets (CSS) 12
II.1.5 Python 13
II.1.6 Asynchronous JavaScript and XML (Ajax) 13
II.1.7 Extensible Markup Language (XML) 13
II.1.7.1 XML-RPC 14
II.2 Motores de búsqueda 15
II.2.1 Funcionamiento 15
II.2.2 Evolución de los motores de búsqueda 16
II.2.3 Motores de búsqueda en Venezuela 18
II.2.4 Motores de búsqueda en medios de comunicación 18
II.3 Algoritmos de búsqueda 19
II.3.1 Algoritmo *Fish-Search* 19
II.3.2 Algoritmo *Shark-Search* 20

II.3.3 ALGORITMO *PAGERANK* 20
II.3.4 ALGORITMO *TRUSTRANK* 21
II.4 WEB SEMÁNTICA 22
II.5 BLOGS 23
II.5.1 DIRECTORIOS Y EDITORES DE BLOGS 23
II.5.2 BLOGS EN VENEZUELA 24
II.7 REDIFUSIÓN DE CONTENIDO WEB 25
II.7.1 RSS (REALLY SIMPLE SYNDICATION) 25
II.7.2 ATOM 26

CAPÍTULO III. MARCO METODOLÓGICO 27

III.1 EL PROCESO UNIFICADO RACIONAL 27
III.1.1 CARACTERÍSTICAS DEL PROCESO UNIFICADO RACIONAL 27
III.2 CICLO DE VIDA 30

CAPÍTULO IV. DESARROLLO 31

IV.1 CONFIGURACIÓN DEL PROCESO UNIFICADO RACIONAL 31
IV.2 INICIO 32
IV.2.1 CAPTURA DE REQUERIMIENTOS 32
IV.2.1.1 REQUERIMIENTOS NO FUNCIONALES 33
IV.2.1.2 REQUERIMIENTOS FUNCIONALES 34
IV.2.1.2.1 Casos de Uso 36
IV.3 ELABORACIÓN 38
IV.3.1 ANÁLISIS DE REQUERIMIENTOS 38
IV.3.1.1 ARQUITECTURA DEL SISTEMA 39
IV.3.1.2 ORGANIZACIÓN DEL SISTEMA 40
IV.3.1.3 ANÁLISIS DEL MÓDULO DE INDEXACIÓN 40
IV.3.1.3.1 Análisis Caso de Uso "iniciar indexación" 40
IV.3.1.3.2 Análisis Caso de Uso "procesar indexación" 41
IV.3.1.3.3 Análisis Caso de Uso "sincronizar indexación" 42
IV.3.1.4 ANÁLISIS DEL MÓDULO DE BÚSQUEDA 43
IV.3.1.4.1 Análisis Caso de Uso "iniciar búsqueda" 44
IV.3.1.4.2 Análisis Caso de Uso "realizar búsqueda" 44
IV.3.1.4.3 Análisis Caso de Uso "guardar histórico" 45
IV.3.1.5 ANÁLISIS DEL MÓDULO DE MANTENIMIENTO 45
IV.3.1.5.1 Análisis Casos de Uso del módulo mantenimiento 45
IV.3.2 DISEÑO DE COMPONENTES 46
IV.3.2.1 DISEÑO DE LA ORGANIZACIÓN DEL SISTEMA 46
IV.3.2.2 DISEÑO DEL MÓDULO DE INDEXACIÓN 48
IV.3.2.2.1 Diseño de componentes para "iniciar indexación" 48
IV.3.2.2.2 Diseño de componentes para "procesar indexación" 50
IV.3.2.2.3 Diseño de componentes para "sincronizar indexación" 50
IV.3.2.3 DISEÑO DEL MÓDULO DE BÚSQUEDA 51
IV.3.2.3.1 Diseño de componentes para "iniciar búsqueda" 51
IV.3.2.3.2 Diseño de componentes para "realizar búsqueda" 52

IV.3.2.3.3 Diseño de componentes para "guardar histórico" 54
IV.3.2.4 Diseño del módulo de mantenimiento 54
IV.3.2.4.1 Administración del contenido 54
IV.4 Construcción 55
IV.4.1 Implementación 55
IV.4.1.1 Implementación del módulo de indexación 56
IV.4.1.1.1 Objetos del modelo 56
IV.4.1.1.2 Métodos de la vista y el controlador 59
IV.4.1.2 Implementación del módulo de búsqueda 60
IV.4.1.2.1 Objetos del modelo 60
IV.4.1.2.2 Métodos de la vista y el controlador 62
IV.4.1.3 Implementación del módulo de mantenimiento 62
IV.4.1.3.1 Objetos del modelo 63
IV.4.1.3.2 Métodos de la vista y del controlador 63
IV.4.2 Validación, verificación y Pruebas 64
IV.4.3 Estudio de casos 65

RESULTADOS 66

CONCLUSIONES 69

RECOMENDACIONES 70

BIBLIOGRAFÍA 71

GLOSARIO 75

APÉNDICE A – COMPARATIVA DE BÚSQUEDAS 77

Comparación de búsqueda global para palabra "3D" 77
Comparación de búsqueda en Venezuela para palabra "3D" 77
Resultados de 2da búsqueda en Venezuela para palabra "3D" 78

APÉNDICE B – EVOLUCIÓN DE LOS BUSCADORES 79

APÉNDICE C – CARACTERÍSTICAS DE XML 80

C.1 Fortalezas de XML 80
C.2 Estructura de comunicación vía XML 81

APÉNDICE D – CONTENIDO XML EN EL NAVEGADOR 82

APÉNDICE E – EL PROCESO UNIFICADO RACIONAL 83

E.1 LENGUAJE UNIFICADO DE MODELADO (UML) 83
E.2 FLUJOS DE TRABAJO DEL PROCESO UNIFICADO RACIONAL (RUP) 83
E.3 FASES DEL PROCESO UNIFICADO RACIONAL (RUP) 84
E.4 DURACIÓN DE CADA FASE DEL PROYECTO 85

APÉNDICE F – CASOS DE USO DEL SISTEMA 86

F.1 DIAGRAMA GENERAL DE CASOS DE USO DEL SISTEMA 86
F.2 CASOS DE USO DEL MÓDULO DE INDEXACIÓN 87
F.3 CASOS DE USO DEL MÓDULO DE BÚSQUEDA 92
F.4 CASOS DE USO DEL MÓDULO DE MANTENIMIENTO 97

APÉNDICE G – DIAGRAMAS DE ACTIVIDADES 102

G.1 DIAGRAMA DE ACTIVIDADES DEL MÓDULO DE INDEXACIÓN 102
G.2 DIAGRAMA DE ACTIVIDADES DEL MÓDULO DE BÚSQUEDA 105
G.3 DIAGRAMA DE ACTIVIDADES DEL MÓDULO DE MANTENIMIENTO 108

APÉNDICE H – DIAGRAMAS DE SECUENCIA 113

H.1 MÓDULO DE INDEXACIÓN 113
H.2 MÓDULO DE BÚSQUEDA 117

APÉNDICE I – VERSIONES PRELIMINARES 121

APÉNDICE J – DIAGRAMA ENTIDAD-RELACIÓN 122

APÉNDICE K – FORMATOS UTILIZADOS EN EL SISTEMA 123

K.1 FORMATO DE DOCUMENTO XML A INTERPRETAR 123
K.2 FORMATO DE FECHA Y HORA DEL SISTEMA 123

APÉNDICE L – CRITERIOS DE VALORACIÓN DE RESULTADOS 124

L.1 MANEJO DE PUNTUACIÓN 124

APÉNDICE M – IMPLEMENTACIÓN DEL SISTEMA 125

APÉNDICE N – FUNCIONAMIENTO XML—RPC 126

APÉNDICE O – ESTRUCTURAS DE ALMACENAMIENTO 127

APÉNDICE P – CASOS DE PRUEBA 128

APÉNDICE Q – ESTUDIO DE CASOS 133

APÉNDICE R – DOCUMENTO VISIÓN 134

Índice de Tablas

TABLA 1. CARACTERÍSTICAS DEL MÓDULO DE INDEXACIÓN 35
TABLA 2. CARACTERÍSTICAS DEL MÓDULO DE BÚSQUEDA (AVANZADA Y SIMPLE) 35
TABLA 3. CARACTERÍSTICA DEL MÓDULO DE MANTENIMIENTO 35
TABLA 4. CASOS DE USO DEL MÓDULO DE INDEXACIÓN 36
TABLA 5. CASOS DE USO DEL MÓDULO DE BÚSQUEDA 36
TABLA 6. CASOS DE USO DEL MÓDULO DE MANTENIMIENTO 37
TABLA A.1. COMPARACIÓN ENTRE 2 BUSCADORES GLOBALES 37
TABLA A.2. COMPARACIÓN ENTRE 2 BUSCADORES EN VENEZUELA 37
TABLA A.3. 2DA COMPARACIÓN ENTRE 2 BUSCADORES EN VENEZUELA 78
TABLA B.1. EVOLUCIÓN DE LOS BUSCADORES 37
TABLA E.1. TABLA RESUMEN DE FASES EN RUP 84
TABLA E.2. DURACIÓN DE LAS FASES DEL PROYECTO 85
TABLA F.1. FLUJO PRINCIPAL DEL CASO DE USO INICIAR INDEXACIÓN 87
TABLA F.2. FLUJO PRINCIPAL DEL CASO DE USO PROCESAR INDEXACIÓN 89
TABLA F.3. FLUJO PRINCIPAL DEL CASO DE USO SINCRONIZAR INDEXACIÓN 90
TABLA F.4. FLUJO PRINCIPAL DEL CASO DE USO INICIAR BÚSQUEDA 92
TABLA F.5. FLUJO PRINCIPAL DEL CASO DE USO REALIZAR BÚSQUEDA 94
TABLA F.6. FLUJO PRINCIPAL DEL CASO DE USO GUARDAR HISTÓRICO 95
TABLA F.7. FLUJO PRINCIPAL DEL CASO DE USO INICIAR MANTENIMIENTO 97
TABLA F.8. FLUJO PRINCIPAL DEL CASO DE USO SELECCIONAR PARÁMETRO 98
TABLA F.9. FLUJO PRINCIPAL DEL CASO DE USO ADMINISTRAR SITIO 99
TABLA F.10. FLUJO PRINCIPAL DEL CASO DE USO ADMINISTRAR ARTÍCULOS 100
TABLA F.11. FLUJO PRINCIPAL DEL CASO DE USO ADMINISTRAR USUARIOS 101
TABLA F.12. FLUJO PR ADMINISTRAR ARTÍCULOS 100
TABLA L.1. CRITERIOS DE ASIGNACIÓN DE PUNTOS 118
TABLA M.1. HERRAMIENTAS UTILIZADSA EN IMPLEMENTACIÓN 119

Índice de Figuras

FIGURA 1. FUNCIONAMIENTO DE XML – RPC 14
FIGURA 2. FUNCIONAMIENTO DE PAGERANK 21
FIGURA 3. CONFIGURACIÓN DE RUP PARA EL PROYECTO 31
FIGURA 4. 1ERA ITERACIÓN FASE DE INICIO 33
FIGURA 5. DIAGRAMA GENERAL DE CASOS DE USO DEL SISTEMA 37
FIGURA 6. FIN DE LA 2DA ITERACIÓN (FASE DE INICIO) 38
FIGURA 7. FIN DE LA 1ERA ITERACIÓN (FASE DE ELABORACIÓN) 46
FIGURA 8. FIN DE LA 2DA ITERACIÓN (FASE DE ELABORACIÓN) 55
FIGURA 9. FIN DE LA 1ERA ITERACIÓN (FASE DE CONSTRUCCIÓN) 59
FIGURA 10. FIN DE LA 2DA ITERACIÓN (FASE DE CONSTRUCCIÓN) 62
FIGURA 11. FIN DE LA 3ERA ITERACIÓN (FASE DE CONSTRUCCIÓN) 64
FIGURA 12. FIN DE LA 1ERA ITERACIÓN (FASE DE TRANSICIÓN) 65
FIGURA C.1. EJEMPLO DE UNA PETICIÓN XML-RPC 81
FIGURA C.2. EJEMPLO DE UNA RESPUESTA XML-RPC 81
FIGURA D.1. DOCUMENTO XML VISTO DESDE IE 7.0 82
FIGURA E.1. FASES Y FLUJOS DE TRABAJO DE RUP 83
FIGURA F.1. DIAGRAMA DE CASO DE USO GENERAL DEL SISTEMA 86
FIGURA F.2. DIAGRAMA DE CASO DE USO MÓDULO DE INDEXACIÓN 91
FIGURA F.3. DIAGRAMA DE CASO DE USO MÓDULO DE BÚSQUEDA 96
FIGURA F.4. DIAGRAMA DE CASO DE USO MÓDULO DE MANTENIMIENTO 101
FIGURA G.1. DIAGRAMA DE ACTIVIDADES "INICIAR INDEXACIÓN" 102
FIGURA G.2. DIAGRAMA DE ACTIVIDADES "PROCESAR INDEXACIÓN" 103
FIGURA G.3. DIAGRAMA DE ACTIVIDADES "SINCRONIZAR INDEXACIÓN" 104
FIGURA G.4. DIAGRAMA DE ACTIVIDADES "INICIAR BÚSQUEDA" 105
FIGURA G.5. DIAGRAMA DE ACTIVIDADES "REALIZAR BÚSQUEDA" 106
FIGURA G.6. DIAGRAMA DE ACTIVIDADES "GUARDAR HISTÓRICO" 107
FIGURA G.7. DIAGRAMA DE ACTIVIDADES "INICIAR MANTENIMIENTO" 108
FIGURA G.8. DIAGRAMA DE ACTIVIDADES "SELECCIONAR TIPO PARAMETRO" 109
FIGURA G.9. DIAGRAMA DE ACTIVIDADES "ADMINISTRAR SITIO" 110
FIGURA G.10. DIAGRAMA DE ACTIVIDADES "ADMINISTRAR ARTÍCULOS" 111
FIGURA G.11. DIAGRAMA DE ACTIVIDADES "ADMINISTRAR USUARIOS" 112
FIGURA H.1. DIAGRAMA DE SECUENCIA "MÓDULO DE INDEXACIÓN" 113
FIGURA H.1.1. DIAGRAMA DE SECUENCIA "INICIAR INDEXACIÓN" 114
FIGURA H.1.2. DIAGRAMA DE SECUENCIA "PROCESAR INDEXACIÓN" 115
FIGURA H.1.3. DIAGRAMA DE SECUENCIA "SINCRONIZAR INDEXACIÓN" 116
FIGURA H.2. DIAGRAMA DE SECUENCIA "MÓDULO DE BÚSQUEDA" 117
FIGURA H.2.1. DIAGRAMA DE SECUENCIA "INICIAR BÚSQUEDA" 118
FIGURA H.2.2. DIAGRAMA DE SECUENCIA "REALIZAR BÚSQUEDA" 119
FIGURA H.2.3. DIAGRAMA DE SECUENCIA "GUARDAR HISTÓRICO" 120
FIGURA I.1. VERSIÓN PRELIMINARES DEL SISTEMA 121
FIGURA J.1. DIAGRAMA ENTIDAD-RELACIÓN 122
FIGURA K.1. DOCUMENTO XML INTERPRETADO POR EL SISTEMA 123
FIGURA N.1. COMUNICACIÓN ENTRE SISTEMA LOCAL Y SISTEMA REMOTO 126
FIGURA O.1. VECTOR DE REGISTROS ENCONTRADOS EN LA BÚSQUEDA 127
FIGURA P.1. SOLICITUD DE DIRECCIÓN A INDEXAR 128
FIGURA P.2. RESPUESTA AL INTENTAR INDEXAR 128
FIGURA P.3. INICIO DE BÚSQUEDA CON FRASE 129
FIGURA P.4. SOLICITUD DE PALABRA CLAVE AL USUARIO 129

FIGURA P.5. RESPUESTA DEL SISTEMA AL NO DETECTAR CADENA ESCRITA 130
FIGURA P.6. SOLICITUD DE USUARIO Y CONTRASEÑA 131
FIGURA P.7. VISTA PARA ADMINISTRAR SITIOS INDEXADOS 132
FIGURA P.8. VISTA PARA ADMINISTRAR ARTÍCULOS INDEXADOS 132

Sinopsis

El presente trabajo se titula "*Sistema de búsquedas históricas actualizado en tiempo real con tecnología RSS*" y describe el desarrollo de una aplicación web, en lenguaje Python, capaz de indexar y realizar, en tiempo real, la búsqueda de noticias y eventos registrados en diversos medios de comunicación, y *blogs* previamente seleccionados, de acuerdo a un Estado y/o ciudad específica de Venezuela.

A través de esta aplicación el usuario puede obtener rápidamente un reporte de los últimos eventos (registrados por el sistema), ocurridos en una localidad determinada, realizar búsquedas en un rango de fechas específico, y de igual forma, mostrar, desde una misma interfaz de usuario, el número de ocurrencias de artículos relacionados a un mismo tema, dentro de los distintos estados del territorio nacional, a fin de facilitar: la comparación, análisis e investigación del mismo.

Para el desarrollo de esta aplicación web se utilizó la metodología de Proceso Racional Unificado (RUP por sus siglas en inglés), la cual al poseer un ciclo de vida iterativo, permite organizar el trabajo en mini-proyectos que contribuyen a garantizar la calidad del producto final con cada iteración.

Como resultado se obtuvo una aplicación web apoyada en el patrón de diseño modelo vista controlador (MVC) que ofrece, al administrador, un panel de control que facilita la administración de los artículos indexados en la base de datos, y al usuario, una interfaz gráfica con soporte AJAX que permite realizar búsquedas avanzadas, desde un sistema con manejo de pestañas, y búsquedas simples capaces de emitir reportes relacionados a los últimos eventos ocurridos en un lugar (Estado y/o ciudad) específico del país.

Introducción

Realizar búsquedas de noticias o eventos ocurridos en una región específica del país, visualizar el punto de vista de los diversos medios de comunicación, y de particulares, en un mismo sitio resulta complicado en la actualidad debido a que la mayoría de los sitios en Internet no ofrecen la posibilidad de clasificar noticias o eventos y de ésta manera facilitar la comparación de los mismos en distintas zonas del país.

En el presente documento se describe el proceso de desarrollo de un sistema de búsquedas históricas de noticias y eventos, siguiendo la estructura que se muestra a continuación:

- **Capítulo I**: presenta el planteamiento del problema, el objetivo general y los objetivos específicos del trabajo especial de grado, de igual forma describe la justificación, alcance y limitaciones del sistema desarrollado.
- **Capítulo II**: describe en detalle la documentación teórica que avala el presente trabajo especial de grado.
- **Capítulo III**: explica de forma amplia la metodología utilizada para el desarrollo del sistema así como las razones que justifican su uso.
- **Capítulo IV**: describe la forma en la cual fueron cumplidos los objetivos planteados, para el desarrollo del sistema, durante las distintas iteraciones en cada una de las fases de la metodología.
- **Capítulo V**: presenta el producto obtenido como resultado del desarrollo de cada una de las fases del presente trabajo.

Finalmente, se muestran las conclusiones, recomendaciones y bibliografía del material consultado.

Capítulo I. Problema

I.1. Planteamiento del problema

Actualmente Venezuela cuenta con 2 tipos de herramientas para encontrar información a través de Internet: la primera, se trata de directorios de páginas web venezolanas que colocan información relacionada a Venezuela en un mismo rubro (lo cual dificulta la investigación, comparación y seguimiento de eventos ocurridos en el pasado y, en ocasiones en el presente). En segundo lugar, cuenta con motores de búsqueda que permiten encontrar contenidos de variada índole de acuerdo a diversos criterios de búsqueda. Esto hace posible encontrar información a partir de la selección de una categoría relacionada al tema deseado o simplemente escribiendo una palabra clave que finalmente mostrará al usuario, en muchos casos, un número significativo de contenido relacionado al tema.

Sin embargo, este contenido, en pocos casos guarda una relación directa y de forma simultánea con un lugar, región y momento en el cual se realiza la búsqueda, ignorando de esta manera la posibilidad de establecer prioridad sobre los contenidos mas recientes que guardan relación a eventos de actualidad en Venezuela o inclusive detallar como ha ocurrido un evento asociado a una misma palabra clave durante el pasado. De tal forma que un usuario que realiza una búsqueda utilizando, por ejemplo, la palabra clave "3D" durante las elecciones en Venezuela obtendría distintos enfoques dependiendo de cada motor de búsqueda (véase Apéndice A para ampliar información).

Se desea realizar un motor de búsqueda (exclusivo para contenido en Venezuela) que sea capaz de clasificar las búsquedas por estado y ciudades (las más comunes) teniendo en cuenta los eventos de actualidad del país,

pero, ¿Qué o quién define "los eventos de actualidad"?, ¿Cómo se mantendría actualizada esta información sin consumir gran cantidad de recursos del sistema?.

De acuerdo a investigaciones preliminares se pudo observar que los *eventos de actualidad* en Internet, dentro de Venezuela, se definen por aquellos contenidos publicados por los medios de comunicación social y artículos escritos por particulares a través de ciertos *blogs* (bitácoras personales) que se encuentran indexadas sin clasificación por fecha o región específica tanto en los directorios de blogs como en los motores de búsqueda mas populares.

Mientras los sitios web de los medios de comunicación ofrecen un punto de vista acerca de los temas de actualidad, los particulares ofrecen, en ocasiones, otro ángulo de la situación, sin embargo, no existe un lugar que permita realizar investigaciones en el tiempo con los datos obtenidos de los medios de comunicación y las bitácoras personales.

Asimismo surge la necesidad, en el presente, de realizar búsquedas apoyándose en el uso de la sindicación (RSS) como herramienta para encontrar contenido de actualidad *en el instante exacto* en que es publicado por los medios de comunicación o particulares, hecho que permitiría mostrar: información reciente relacionada a eventos significativos en el país, actualizada en tiempo real y relacionada a una región específica de Venezuela. Algunos medios de comunicación que tienen soporte para esta tecnología son: cadenaglobal.com, eluniversal.com.

Un motor de búsqueda, con una visión distinta a los motores tradicionales, se podría actualizar en tiempo real a través del uso de RSS, clasificar la información de acuerdo a la zona del país y darle relevancia de acuerdo al tema más popular de cada región permitiendo buscar en el presente o en el pasado, a través de un caché que ofrezca la posibilidad de realizar la búsqueda entre un rango de fechas. Estas fechas serían aquellas en las que se produzcan cambios en las noticias publicadas, con el objetivo principal de facilitar la realización de investigaciones y comparaciones de eventos ocurridos tanto en el pasado como en el presente, a lo largo de cierto rango de fechas y de acuerdo a cada zona del país.

I.2. Objetivo general

Desarrollar un motor de búsqueda que permita buscar contenido relacionado a Venezuela con la particularidad de utilizar las características de sindicación, *Really Simple Syndication* (RSS), de las páginas indexadas a fin de actualizarse en tiempo real cuando un nuevo registro es publicado y así darle relevancia a los temas más populares.

I.3. Objetivos Específicos

- Realizar levantamiento de información referente a directorios de *blogs*, motores de búsqueda de medios de comunicación y motores de búsqueda generales que permiten encontrar contenido relacionado a Venezuela.
- Realizar levantamiento de información con respecto a los algoritmos de búsqueda utilizados actualmente y su evolución.
- Definir políticas de seguridad para el almacenamiento de información, ingreso al sistema y mantenimiento de la integridad de la información indexada.
- Definir e implementar el algoritmo de búsqueda y ordenamiento a utilizar de acuerdo a las tareas que se ejecutarán en el servidor.
- Diseñar e implementar un módulo *backend* que permita indexar páginas existentes.

- Diseñar e implementar un módulo *frontend* que permita al administrador depurar la información y a los usuarios indexar páginas existentes.

- Evaluar el desempeño del algoritmo de búsqueda a través de la medición del tiempo de respuesta de diversas peticiones. Analizar resultados obtenidos y determinar que aspectos se podrían optimizar.

I.4. Limitaciones

- Debido al costo, el motor de búsqueda estará corriendo sobre un servidor compartido. Para guardar el caché de todas las páginas indexadas se utilizará un ambiente de servidores compartidos distribuidos donde la información será almacenada en varios *hosts*.
- Solo se podrán realizar consultas sobre páginas venezolanas que tengan soporte para la tecnología RSS.
- Las páginas venezolanas serán clasificadas como tales de acuerdo al criterio de los editores o administradores a fin de garantizar la calidad de la información que será indexada. Se puede indexar cualquier página pero no será válida hasta que sea autorizada por los editores. Esta autorización se podrá revocar o extender a todo un dominio teniendo en cuenta la ruta principal que permite identificar un sitio como *página venezolana*.
- Las pruebas de desempeño del algoritmo se realizarán dentro del mismo servidor y permitirán conocer cuanto tiempo se tarda el motor de búsqueda en obtener un resultado.

I.5. Alcance

Al finalizar el desarrollo de este proyecto se obtendrá un motor de búsqueda que se actualiza en tiempo real y es capaz de mostrar registros históricos de los sitios indexados de acuerdo a una región de Venezuela, una fecha buscada, una frase o palabra escrita y tomando en cuenta la relevancia del momento en que se realiza la búsqueda. Ofrece una forma rápida de encontrar información histórica de un tema de interés para los venezolanos.

Adicionalmente, ofrecerá la posibilidad de realizar investigaciones y seguimientos a eventos ocurridos en el acontecer nacional, de acuerdo a las páginas venezolanas que han sido previamente agregadas al buscador. De esta manera, aunque el servicio del servidor web original no esté disponible se podrá acceder a su información sin costo alguno con la capacidad de comparar el contenido antiguo. El conocimiento otorgado por los sitios web indexados es centralizado en un solo lugar que se va actualizando automáticamente al instante en que se produce un cambio en las páginas indexadas.

I.6. Justificación

El uso de Internet para realizar investigaciones y conseguir información es cada vez más frecuente. En la actualidad existen herramientas que ayudan en este proceso, sin embargo, buscar información en el pasado es por lo general una tarea difícil porque los sitios web dejan de funcionar o los buscadores eliminan esta información por considerarla obsoleta, de igual forma, tener información al momento exacto en que se produce un evento a la vez de poder compararla con el pasado o el presente entre distintas fuentes (de acuerdo a las regiones del país) tampoco es habitual.

Por tal razón, surge la necesidad de diseñar un sistema que, basado en la adaptación de los algoritmos *PageRank* y *TrustRank*, y mezclando contenido tanto de *blogs* como de páginas oficiales de noticias, facilite, no solo la búsqueda de información en Venezuela, sino también la posibilidad de investigar o realizar seguimiento a noticias, eventos o temas de interés para los venezolanos.

De esta manera, se podrían realizar consultas teniendo una hemeroteca virtual y gratuita en la red, y en caso de necesitarse el soporte físico para una investigación, el usuario podrá conocer si la noticia proviene de un medio impreso y en que momento fue publicada.

De igual forma, se contaría con una herramienta capaz de realizar búsquedas relacionadas al territorio nacional y que se alimenta en tiempo real de páginas en Internet que son actualizadas con frecuencia y desde distintas zonas de Venezuela a través del uso de *Really Simple Syndication* (RSS).

Por lo tanto, se tendrían las siguientes ventajas:

- Búsqueda en una base de datos que se actualiza en tiempo real.
- Búsqueda de acuerdo a los estados y algunas regiones de Venezuela.
- Capacidad de buscar temas que han ocurrido a través de la historia de Venezuela (desde que el motor de búsqueda comience a indexar).
- Una herramienta para realizar investigaciones y obtener distintos puntos de vista, tanto de parte de medios de comunicación como de bitácoras personales.
- Posibilidad de comparar los registros encontrados entre regiones del país o fechas en que fueron publicados.
- Posibilidad de hallar registros de páginas que han sido borradas pero fueron almacenadas por el caché.

Hoy mas que nunca es importante almacenar información que pueda ser consultada en un futuro para realizar investigaciones, un ejemplo de esto, ha sido la ventaja que ofrece el caché temporal del buscador *Google*, a través del cual fue posible conseguir páginas de veedores nacionales que durante las elecciones del 3 de Diciembre de 2006 en Venezuela habían publicado opiniones que luego fueron eliminadas del *site* original. Sin embargo, a pesar de esta ventaja, el caché de *Google* solo dura algunos días después de que la página original es eliminada. De igual forma, es necesario desarrollar el proyecto con un lenguaje de programación estratégicamente diseñado para ambientes web y con posibilidad de optimización de los tiempos de respuesta mediante el uso de herramientas adicionales que permitan precompilar parte del código fuente antes que sea ejecutado desde el navegador del usuario.

Capítulo II. Marco Teórico

El presente trabajo se circunscribe en el concepto de las consideraciones y premisas que a continuación se plantean.

II.1 World Wide Web

También conocida con el acrónimo WWW nace en el laboratorio europeo de física de partículas (CERN por sus siglas en inglés) y se refiere a un conjunto de servidores que utiliza el hipertexto para enlazar y acceder a archivos que pueden estar localizados en un mismo lugar (servidor) o en un lugar remoto.

El hipertexto se refiere a un conjunto de archivos que contienen frases destacadas de una manera particular y permite enlazar un documento a otro, aunque este último se encuentre en un país distinto. Cada documento de hipertexto posee hipervínculos que son los que permiten realizar esta acción.

No existe una ruta predeterminada que sea capaz de establecer el final de los enlaces (hipervínculos) contenidos en un documento, éstos siempre dependerán del autor.

Los documentos de hipertexto se acceden a través de un navegador, en la actualidad, se encuentran disponibles: Internet Explorer (Windows), Mozilla Firefox, Konqueror (Linux), Safari (Apple), entre otros. Una vez dentro del navegador, cuando se encuentre abierto un documento o página en la WWW, el usuario simplemente debe posicionar el cursor del ratón sobre las palabras o frases resaltadas y luego apretar el botón. El resultado obtenido puede variar desde un archivo en texto plano, una imagen, un archivo de sonido u otro enlace a un nuevo hipervínculo.

II.1.1 Hypertext Markup Language (HTML)

Es un lenguaje utilizado para realizar documentos (páginas) en la World Wide Web que contiene un conjunto de instrucciones en texto plano, que son interpretadas posteriormente por el navegador, con el objetivo principal de presentar al usuario: texto, imágenes, hipervínculos y contenido multimedia.

II.1.2 Javascript

Es un lenguaje utilizado en las páginas web que no requiere compilación, se ejecuta directamente en la computadora del usuario (cliente), y tiene como ventaja principal la capacidad de realizar validaciones en formularios o ejecutar pequeños cálculos sin sobrecargar el servidor.

II.1.3 Dynamic HTML (Dhtml)

Dynamic HTML (dhtml) es la mezcla de un conjunto de tecnologías capaces de ofrecer al usuario, a través del navegador, páginas web con un nivel adicional de interactividad mediante decoraciones visuales en el contenido. El uso de DHTML puede variar dependiendo del navegador que se está utilizando y, a diferencia de javascript, Dhtml tiene como único objetivo contribuir a mejorar el aspecto de la interfaz gráfica.

II.1.4 Cascading Style Sheets (CSS)

Cascading Style Sheets (CSS) es un lenguaje utilizado para describir de forma estructurada y facilitar el manejo de los elementos de una página web.

II.1.5 Python

Es un lenguaje de programación multiparadigma (programación estructurada ó "funcional" y orientado a objetos) de alto nivel que posee grandes optimizaciones para el desarrollo en ambientes web a través del compilador pyc (Python compiler) que permite mantener código precompilado al momento de ejecutar la aplicación lo cual se traduce en un mejor desempeño a otros lenguajes de ambiente web como PHP.

II.1.6 Asynchronous JavaScript and XML (Ajax)

Asynchronous JavaScript and XML (Ajax) es una técnica utilizada en el desarrollo de aplicaciones web que mejora considerablemente la capacidad de respuesta e interactividad con el usuario. Se caracteriza por representar una capa intermedia entre código que se ejecuta del lado cliente (javascript) y el código que se ejecuta del lado del servidor (python) esto permite que el usuario esté informado con gran velocidad acerca de los resultados de los eventos que se están ejecutando en el servidor.

II.1.7 Extensible Markup Language (XML)

Extensible Markup Language (XML) es un metalenguaje extensible basado en etiquetas que fue definido por la W3C (World Wide Web Consortium) que permite definir la gramática de un lenguaje de acuerdo a unos requerimientos específicos.

Esta tecnología resulta, en la actualidad, una excelente forma de comunicar distintas plataformas o distintos sistemas dentro de una misma plataforma. Véase Apéndice C para conocer las fortalezas y la estructura de de los archivos utilizados para establecer comunicación utilizando XML.

II.1.7.1 XML-RPC

Es un protocolo de llamadas a procedimientos remotos que utiliza Internet para la transmisión de mensajes a través del protocolo TCP/IP.

Los mensajes son peticiones HTTP enviadas con el método POST. El cuerpo de la solicitud va escrito en XML y el procedimiento se ejecuta en el servidor que a su vez devuelve un documento en formato XML.

A continuación (Ver Figura 1) se muestra un diagrama de funcionamiento general de la tecnología XML–RPC:

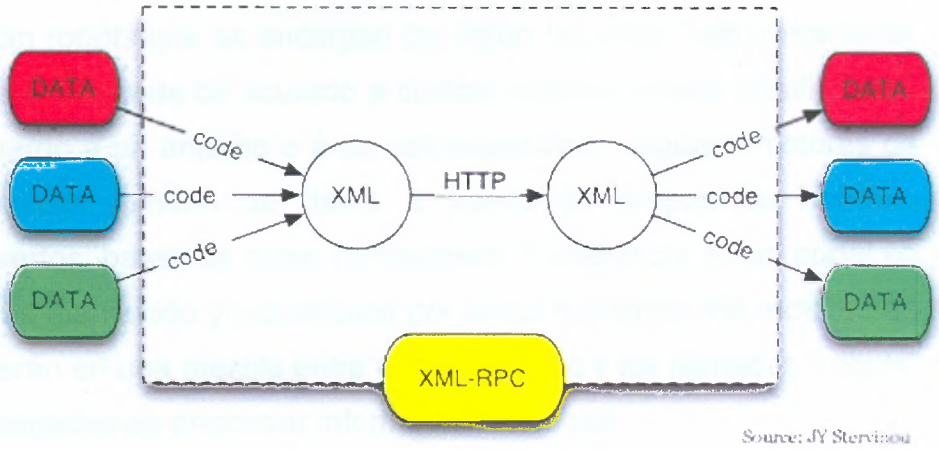

Figura 1. Funcionamiento de XML – RPC
Fuente: [USE03]

En el Apéndice C (tabla C.1) es posible observar la petición que es enviada al servidor en un documento XML identificado principalmente por una etiqueta padre llamada "*Methodcall*", este documento XML es interpretado por el servidor y posteriormente es retornada una respuesta (tabla C.2) que muestra un código de respuesta identificado a través de la etiqueta raíz "*methodResponse*".

II.2 Motores de búsqueda

De acuerdo a [WIK05], el concepto *de motor de búsqueda* para sitios web, aparece por primera vez en 1990. Un motor de búsqueda es un sistema de recuperación de información destinado a ayudar a encontrar datos almacenados en una computadora, tanto en Internet como en una empresa, red interna o inclusive una computadora personal. El motor de búsqueda permite al usuario buscar un contenido a partir de un criterio específico, por lo general una frase o palabra, y devuelve una lista de elementos que coinciden con ese criterio.

Esta lista de elementos también se encuentra ordenada con respecto a la relevancia de los resultados y es precisamente allí, en la relevancia de los resultados, es donde se diferencian los distintos buscadores existentes. Algunos utilizan *robots* que se encargan de visitar los sitios web y mantener la información actualizada de acuerdo a ciertos criterios y otros clasifican los temas de acuerdo a un entorno o a un uso específico. Algunos motores de búsqueda realizan minería de datos a través de grupos de noticias, directorios abiertos, bases de datos particulares. A diferencia de un portal en Internet, que es mantenido y actualizado por seres humanos, los motores de búsqueda operan en una mezcla entre el ser humano y los llamados "robots" o *spiders* encargados de encontrar información en la red.

II.2.1 Funcionamiento

El motor de búsqueda para funcionar requiere realizar los siguientes pasos:

1. *Web Crawling:* consiste en recuperar páginas de Internet y almacenarlas para su futura clasificación. Esta recuperación se hace a través de spiders, también conocidos como crawlers o robots. Los *spiders*: "Recorren las páginas recopilando información sobre los contenidos de las páginas. Cuando

se busca una información en los motores, ellos consultan su base de datos, y la presentan clasificados por su relevancia. De las webs, los buscadores pueden almacenar desde la página de entrada, a todas las páginas de la web" [WIK05].

2. *Indexación:* se refiere a como los datos son almacenados, procesados y colocados en la base de datos del buscador para facilitar la posterior búsqueda. El diseño del índice incorpora conceptos interdisciplinarios de Lingüística, psicología Cognoscitiva, Matemáticas, Informática, Física, y Ciencias de la computación.

3. *Búsqueda:* es el proceso mediante el cual se accede a la información previamente indexada, esta búsqueda es la que marca la diferencia en la mayoría de los buscadores utilizando herramientas como el *Pagerank* o "popularidad" de una página web. En la actualidad aparece un concepto denominado "Metabuscadores", en donde durante este proceso de búsqueda se muestran varios resultados de otros motores permitiendo así mostrar resultados más completos.

II.2.2 Evolución de los motores de búsqueda

En 1990 aparece "Archie" un pequeño buscador de archivos en servidores FTP, sin embargo, con la masificación de la información en Internet comienza también a aparecer la necesidad de buscar contenido específico dentro de una gran base de datos de información, por tal razón, surgieron los primeros motores de búsqueda basados en Web Crawlers (también conocidos como *robots* o *spiders*).

En 1994 aparece *WebCrawler®*, el primer buscador que utilizaba robots para indexar páginas a su base de datos y permitía buscar a través de palabras clave. Inicialmente utilizaba su propia base de datos pero en la actualidad se nutre de la información que pueden proporcionar otros motores de búsqueda como: Google, Yahoo!, Windows *Live Search*.

A partir de 1995 aparecen otros buscadores muy conocidos en la actualidad como: AltaVista, Excite, Dogpile, Inktomi, Ask Jeeves y Northern Light, cada uno con un elemento particular que le permitiera marcar la diferencia entre los demás buscadores, sin embargo, no es hasta 1998 con la aparición de Google que ocurre una gran diferencia entre la forma de buscar. Esta nueva forma de buscar consiste en incluir una estrategia conocida como "Link Popularity" que consiste en una medida de la calidad y cantidad de sitios web que mantienen enlaces a un sitio específico en la World Wide Web, de modo tal que, si hay muchos enlaces de un site en distintas páginas web significa que este sitio no es solo popular sino que también tiene material de calidad. Además de esto *Google* incorpora esta estrategia en un algoritmo conocido como PageRank que permite asignar de forma numérica la relevancia de los sitios web. Mas adelante y durante los siguientes años *Google* va incorporando otras tecnologías como AJAX, van mejorando su hardware y ofreciendo nuevos servicios basados en web y finalmente el 10 de Abril de 2006 logran ganar los derechos para utilizar un algoritmo australiano de búsqueda [HUT06] que pretende optimizar aun más los resultados.

Véase Apéndice B para conocer detalles acerca de la evolución de los motores de búsqueda.

II.2.3 Motores de búsqueda en Venezuela

En 1995 aparece en Venezuela el primer "portal" venezolano, realizado por la empresa Cyberven (www.cyberven.com), más adelante aparecería un catálogo de páginas venezolanos (aunque no un buscador) llamado VenWeb, sin embargo, desaparece en 1998 debido a que era mantenido de forma manual y no contaba con un método automatizado para ingresar y actualizar la información existente.

El 9 de Abril de 1996 aparece Auyantepui.com como el primer motor de búsqueda de Venezuela, sin embargo, en octubre de ese mismo año surge *Chevere.com* y más adelante en diciembre *Yuada*, convirtiéndose estos últimos en los mas populares entre los venezolanos.

En 1999 *Google* comienza a ganar terreno ofreciendo la posibilidad de buscar contenido en Venezuela y, un poco mas tarde, en marzo del año 2000, *Terra Networks* adquiere el portal venezolano *Chevere.com* lo cual ocasiona una reducción considerable en el número de usuarios de este portal entre los venezolanos y, por su facilidad de uso, *Google* y auyantepui.com permanecen hasta la fecha como las herramientas primordiales de búsqueda de los venezolanos mientras que la tendencia del resto de buscadores venezolanos existentes ha sido especializarse en un área específica, y así lo hizo: Infoguiaweb, paginasamarillas, caveguias, entre otros.

II.2.4 Motores de búsqueda en medios de comunicación

En la actualidad, los principales medios de comunicación venezolanos en Internet ofrecen una herramienta que permite la búsqueda de artículos de fechas anteriores, sin embargo, no todos ofrecen la posibilidad de buscar entre un rango de fechas específico (cadenaglobal.com) y otros requieren de registro (El-Nacional.com) o pagos para permitir visualizar contenido histórico

y dentro de los principales medios escritos: el-nacional.com, eluniversal.com, ultimasnoticias.com.ve, cadenaglobal.com (Cadena Capriles) no es posible realizar una comparación entre las noticias publicadas por cada medio de comunicación.

II.3 Algoritmos de búsqueda

En este punto se describen algunos algoritmos utilizados actualmente por los motores de búsqueda para mejorar los resultados.

II.3.1 Algoritmo *Fish-Search*

Es un algoritmo de búsqueda dinámico que permite que usuarios busquen dinámicamente subáreas de la Web predefinida por 'un salto' (palabra clave) dado. El algoritmo parte de una metáfora donde se comparan los agentes de búsqueda que exploran la web (spiders, robots, crawlers) a una escuela de peces en el mar. Cuando el alimento (información relevante) es encontrado, los peces (agentes de búsqueda) se reproducen y siguen buscando alimento, en ausencia del alimento (ninguna información relevante) o cuando el agua es contaminada (ancho de banda escaso), ellos mueren.

El principio clave del algoritmo es el siguiente: en primer lugar toma como semilla una URL previamente escrita y una pregunta de búsqueda o palabra clave, y dinámicamente construye una lista de prioridades (comenzando en la URL semilla) de las siguientes URLs (a continuación llamados nodos) para ser exploradas. En cada paso el primer nodo es sacado de la lista y procesado. Cuando el texto de cada documento está disponible, es analizado y evaluado por un componente de tanteo, si resulta relevante o irrelevante a la pregunta de búsqueda (valor 1 ó 0) y, basado en ese resultado, se decide si hay que continuar la exploración en esa dirección

o no: siempre que una fuente del documento sea traída, es explorada por los eslabones. Los nodos señalados por estos eslabones (hijos de los nodos) reciben cada uno un valor de profundidad. Si el padre es relevante, la profundidad de los hijos recibe automáticamente un valor predefinido. Por otra parte, se establece que la profundidad de los hijos sea menor que la profundidad del padre. Cuando la profundidad alcanza cero, la dirección es abandonada y ninguno de sus hijos es insertado en la lista.

II.3.2 Algoritmo *Shark-Search*

Es un algoritmo de búsqueda dinámico propuesto por Michael Hersovici, Yoelle S. Maarek1, Dan Pelleg, Menachem Shtalhaim y Sigalit Ur donde se ofrecen varias mejoras al algoritmo de búsqueda de pescado original a fin de vencer ciertas limitaciones. Este algoritmo utiliza la misma metáfora simple, que conduce al descubrimiento de más información relevante a la vez que se va realizando la exploración. Una mejora es utilizar en lugar de evaluación binaria (relevante/irrelevante) de un documento, lo que se define como un "motor de semejanzas" a fin de evaluar la relevancia de los documentos dada una petición especial. Este motor analiza dos documentos dinámicamente y devuelve un resultado "intermedio" entre 0 y 1 (0 para ninguna semejanza en absoluto y 1 para una coincidencia "conceptual" perfecta) en lugar de un valor binario.

II.3.3 Algoritmo *PageRank*

Es un conjunto de algoritmos, desarrollados por los fundadores de Google: Larry Page y Sergey Brin, en la Universidad de Stanford en 1998, que se encargan de asignar de forma numérica la relevancia de los documentos que han sido indexados por un motor de búsqueda.

El *ranking* de una página es el resultado de una "votación" entre todas las demás páginas de la World Wide Web acerca del nivel de importancia que tiene esa página.

A continuación se muestra gráficamente como se relacionan los distintos sitios entre sí mediante *PageRank*:

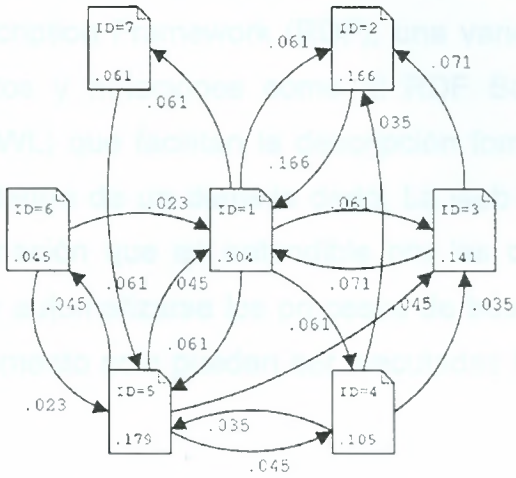

Figura 2. Relación de sitios mediante PageRank
Fuente: [WIK06]

II.3.4 Algoritmo *TrustRank*

Es un algoritmo que funciona de manera análoga al *PageRank* con la particularidad de evitar enlaces de páginas que son consideradas *spam*. De esta forma, en un documento publicado por la Universidad de Stanford [11] se establece que las webs semilla van a transmitir Trust Rank, las webs consideradas *spam* por los humanos que evalúan las webs, van a transmitir Trust Rank negativo. Según este documento ciertas webs van a restar Trust Rank y cuando una página web situe su nivel de Trust Rank por debajo de una determinada cifra, va a convertirse automáticamente en una web que transmite Trust Rank negativo en lugar de positivo.

II.4 Web semántica

Es una evolución de la *World Wide Web* en la cual la información puede ser procesada por las computadoras permitiendo a navegadores u otros agentes de software: buscar, compartir y combinar información más fácilmente.

El corazón de la web semántica consiste en un modelo de datos llamado Resource Description Framework (RDF), una variedad de formatos de intercambio de datos y notaciones como el RDF Schema y el Web Ontology Language (OWL) que facilitan la descripción forma de conceptos, términos y relaciones dentro de un dominio dado. La web semántica ofrece una visión de la información que es entendible por las computadoras, de manera tal que puedan automatizarse los procesos de búsqueda en la web, tareas que hasta el momento solo pueden ser ejecutadas fácilmente por los seres humanos.

II.4.1 Transición a web semántica

Gran cantidad de empresas se encuentran en una constante competencia por el desarrollo de tecnologías que permitan darle significado a los documentos y páginas web actualmente indexadas en Internet, sin embargo, la evolución de un software capaz de interpretar las necesidades o peticiones del usuario varía por idiomas y por países, pues muchas frases o palabras tienden a representar pequeñas diferencias en el interés del usuario dependiendo del país desde el cual se solicite el documento o inclusive el contexto bajo el cual se realiza la petición. Por tal razón, desde 1998, por su facilidad de uso, se ha establecido como estándar de facto el metalenguaje XML, capaz de crear árboles que contienen palabras clave que pueden ayudar al programador a diseñar estrategias que contribuyan a la interpretación, mejor aproximada, de las solicitudes del usuario.

II.5 Blogs

Los *blogs*, también conocidos como *web logs,* son sitios web o bitácoras personales que se actualizan con frecuencia y van recopilando, de manera cronológica, textos o artículos de un tema en particular escrito por uno o varios autores en donde se muestra al usuario primero las informaciones más recientes.

A diferencia de un sitio web, un *blog* es actualizado a través de una interfaz gráfica que muestra una o más cajas de texto que permiten al autor editar la información sin necesidad de conocer el lenguaje HTML.

El término fue empleado como es conocido actualmente, por primera vez en 1997 por Jorn Barrer, pero no es sino hasta Agosto de 1999 cuado Evan Williams crea una comunidad llamada *blogger* en donde ofrecía a los usuarios una manera sencilla de editar páginas web y compartir el contenido con otros usuarios de manera instantánea. Antes de esta fecha la forma de enviar contenido a la World Wide Web era a través de aplicaciones externas o servicios de transferencia de archivos (FTP por sus siglas en inglés).

II.5.1 Directorios y editores de blogs

Un Directorio de *blogs* se refiere a un sitio de Internet en el cual se encuentran almacenados un conjunto de bitácoras (o *blogs*) clasificados de acuerdo a algún tema o categoría específica. Uno de los primeros directorios en aparecer en la atmósfera de Internet fue "blogger.com" (1999). Este sitio ofrecía al usuario un sistema manejador de contenidos (CMS[1] por sus siglas en inglés) para la actualización de un blog personal. Mas tarde aparece "b2"

[1]Content Managament System: en la actualidad el uso de manejadores de contenido es de uso habitual para la actualización de cualquier blog o bitácora personal debido a su facilidad de uso.

que luego en Mayo de 2003 se convertiría en lo que en la actualidad se conoce como *Wordpress*, un directorio de blogs.

Wordpress además de almacenar blogs ofrece una aplicación en PHP, del mismo nombre, que permite la administración de una bitácora personal, bien sea que ésta se encuentre hospedada dentro o fuera del dominio *wordpress.com*.

II.5.2 Blogs en Venezuela

El 18 de Mayo de 2003 fue creado el primer directorio de *blogs* en Venezuela llamado Veneblogs. En este directorio es posible conseguir varias bitácoras personales ordenadas de acuerdo a la fecha, sin embargo, no establece criterios de búsqueda avanzados que permitan encontrar contenidos específicos. Mas adelante, en Septiembre de 2006 fue lanzado otro directorio de blogs (*to2blogs*) que se diferencia principalmente del primero (*Veneblogs*) en la capacidad de aceptar *blogs* escritos en otros idiomas por venezolanos y utiliza la API de *Google* para realizar búsquedas en los blogs que a su vez han sido indexados por el famoso buscador.

Actualmente, los blogs toman cada día mayor relevancia y poco a poco son incorporados como parte fundamental de los portales de noticias, tal es el caso de: el-nacional.com, cadenaglobal.com y noticierodigital.com que incluyen secciones especiales donde un grupo selecto de periodistas tiene un espacio abierto para escribir en una bitácora un artículo inclusive antes que aparezca en los medios impresos.

II.7 Redifusión de contenido web

Conocido en inglés como *web syndication*, se refiere a un procedimiento mediante el cual varias secciones o la totalidad del contenido de un sitio web se hacen disponibles a otros sitios en Internet para su uso a través de la publicación de sumarios de contenido que se encuentran en archivos XML. Esto beneficia, tanto a los sitios que publican la información como a los que la reciben. Los autores obtienen una mayor difusión del contenido de su sitio y quienes lo retransmiten pueden mostrar variedad informativa que puede resultar de interés para sus usuarios. Los formatos por excelencia para transmitir este tipo de información son RSS y ATOM, los cuales se han popularizado con la inclusión de soporte para XML-RPC en las comunidades de blogs y manejadores de blogs tipo *Wordpress*.

Los navegadores web más recientes permiten al usuario leer contenido a través de este formato mediante la interpretación directa de los documentos XML. En el Apéndice D se muestra como Internet Explorer 7.0 ofrece al usuario la posibilidad de acceder al contenido de un sitio web sin necesidad de cargar todas las imágenes y contenido pesado.

II.7.1 RSS (Really Simple Syndication)

Es un formato de la familia XML que se utiliza para la redifusión de contenido de un sitio web y está orientado principalmente a todos los sitios que se actualicen con frecuencia. Por medio de este formato es posible compartir la información y usarla en otros sitios web o programas. Con el uso de RSS no es necesario abrir el navegador y visitar miles de webs debido a que éste guarda un resumen del contenido que no posee las imágenes o el código HTML que hace pesado el proceso de descargar la información.

Actualmente el formato se encuentra en su versión 2.0 y permite al usuario tener acceso a imágenes, documentos PDF, video y animaciones flash a partir del contenido en un solo archivo. Los documentos RSS también son conocidos como *feeds*.

II.7.2 ATOM

Es un formato de la familia XML utilizado para redistribuir contenido de un sitio web, pero a diferencia de RSS, ATOM pretende estandarizar ciertos procesos y el formato en que deben ser escritos los datos.

ATOM además de ser una "Extensión de Correo Internet Multipropósito" (MIME-type[2] por sus siglas en inglés) registrado ante la *Internet Assigned Numbers Authority* (IANA[3]), establece que para las fechas debe utilizarse el formato descrito en la RFC 3339[4] que forma parte de los estándares internacionales para el manejo y representación de la fecha y la hora (ISO 8601) publicados por la *International Organization for Standardization*. De igual forma, este formato ofrece soporte para firmar documentos XML, sin embargo, debido al nivel de detalle requerido para su implementación el formato RSS 2.0 se ha mantenido con mayor popularidad.

[2] Son un conjunto de convenciones dirigidas a facilitar el intercambio de distintos tipos de archivo a través de Internet.
[3] La Agencia de Asignación de Números de Internet (IANA por sus siglas en inglés) fue antes de 1998 el organismo central en el cual se registraban los protocolos de Internet, puertos, empresas y códigos, en la actualidad se conoce con el nombre de ICANN.
[4] Se refiere al documento oficial que establece los parámetros del manejo de los formatos de fecha y hora en Internet (*timestamps*). Se puede encontrar en http://www.ietf.org/rfc/rfc3339.txt

Capítulo III. Marco Metodológico

En el capítulo se explicará la metodología usada en el desarrollo del proyecto, sus características y su ciclo de vida.

III.1 El Proceso Unificado Racional

Tiene sus orígenes en la metodología *Ericsson Approach* diseñada por Ivar Jacobson en el año 1967 donde se introduce por primera vez el concepto de "caso de uso" a través del desarrollo mediante componentes, mas tarde, en 1987 es fundada la compañía *Objectory AB* y se publica como proceso de desarrollo el *Object Factory*. Luego, en 1995 la empresa de Ivar Jacobson es adquirida por *Rational Software Corporation* quien desarrolla el Rational Objectory Process (ROP) utilizando como base el *Object Factory 3.8* (*Objectory 3.8*) y adoptando UML como lenguaje de modelado.

A partir de ese momento Ivar Jacobson, Grady Booch y James Rumbaugh van incorporando distintos elementos a fin de expandir el ROP, haciendo especial énfasis en el flujo de trabajo (modelado del negocio) que posteriormente, en 1998, daría origen al *Rational Unified Process* (RUP) [JBR00].

III.1.1 Características del Proceso Unificado Racional

El Proceso Unificado Racional se ajusta al desarrollo de software orientado a objetos mediante la creación de componentes y a través del establecimiento de una estructura capaz de analizar los riesgos en fases tempranas.

De igual forma puede apoyarse en el modelo vista controlador (MVC) a fin de facilitar el control de las configuraciones y garantizar la calidad del producto final.

Su característica primordial es que se trata de un proceso: dirigido principalmente por los casos de uso, centrado en la arquitectura, iterativo e incremental [JBR00].

Dirigido por casos de uso: Los casos de uso son parte fundamental del sistema debido a que capturan los requerimientos funcionales, tanto del usuario, como los elementos que interactúan con el sistema. Al estar dirigido por casos de uso, no se eligen de manera aislada, se desarrollan a la par con la arquitectura del sistema y ésta a su vez determina la elección de los casos de uso por lo que, tanto la arquitectura como los casos de uso, van evolucionando a medida que avanza el ciclo de vida estableciendo la posibilidad de llevar a cabo un seguimiento desde su concepción y posterior realización de pruebas funcionales y pruebas unitarias hasta llegar al caso de prueba.

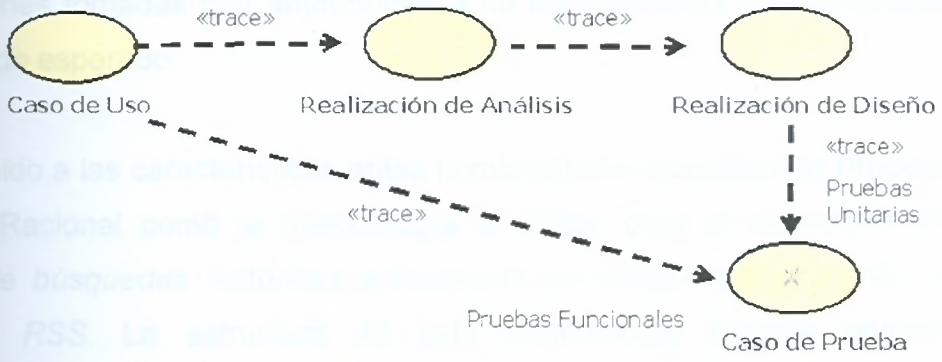

Figura 3. Seguimiento a partir de los casos de uso

Fuente: [JBR99]

Centrado en la arquitectura: se refiere a incorporar una visión global del diseño completo del sistema teniendo en cuenta los aspectos estáticos y

dinámicos más significativos. Estos aspectos se ven afectados por la plataforma de software sobre la cual se ejecutará el sistema, los requerimientos no funcionales (como el desempeño o la confiabilidad), la disponibilidad de componentes reutilizables, entre otros factores [JBR00].

Iterativo e incremental: debido a que desarrollar un software es una tarea compleja, RUP toma en consideración la división del desarrollo en mini-proyectos, que van a representar una iteración, y ésta iteración culmina en un incremento.

Cada mini-proyecto se inicia con los casos de uso teniendo en cuenta aquellos que implican los riesgos más importantes, las siguientes iteraciones construyen nuevos componentes basados en el estado en el que fueron dejados en la iteración anterior. En cada iteración los desarrolladores especifican y determinan los casos de uso más importantes, se crea el *diseño (basado en la arquitectura), éste se implementa en componentes y luego se verifica que dichos componentes cumplan los requisitos de los casos de uso, al cumplirlos, se avanza a la siguiente iteración. Si no se* cumplieran los requisitos de una iteración los desarrolladores deben verificar las decisiones tomadas con anterioridad a fin de orientarlas a la obtención del resultado esperado.

Debido a las características antes nombradas se consideró el Proceso Unificado Racional como la metodología a utilizar para el desarrollo del Sistema de búsquedas históricas actualizado en tiempo real a través de *tecnología RSS*. La estructura de esta metodología permite obtener componentes que facilitan la evolución del proyecto durante el desarrollo y contribuyen a la detección temprana de riesgos sin que sea necesario culminar el desarrollo de todos los componentes para visualizar la funcionalidad de alguno de ellos.

De igual forma, la utilización del patrón de diseño modelo-vista-controlador, para el manejo de las interfaces de usuario, permite realizar cambios rápidamente a la vista sin afectar componentes previamente operativos, esta ventaja, aunada a la utilización de la metodología permite la traza o seguimiento de los cambios en los requisitos funcionales y sus restricciones durante las iteraciones.

III.2 Ciclo de vida

El Proceso Unificado Racional (RUP por sus siglas en inglés) establece 4 fases: inicio, elaboración, construcción y transición.

Inicio: esta fase hace especial énfasis en la comprensión del problema, la tecnología que será utilizada, la delimitación del ámbito del proyecto así como la eliminación de riesgos críticos. Se priorizan las actividades de modelado del negocio y de requisitos. [KRU00]

Elaboración: las iteraciones son orientadas a los flujos de trabajo de requerimientos, refinamiento del modelo de negocio, análisis y diseño.

Construcción: se realiza la construcción de los componentes a través de un conjunto de iteraciones. Para cada iteración se seleccionan algunos casos de uso, se analiza el diseño y posteriormente se realiza su implementación y correspondientes pruebas.

Transición: en esta etapa el producto se entrega al usuario final en su versión final, de igual forma se realizan las pruebas beta ante los usuarios finales. Si el producto cumple con los objetivos planteados en la fase "inicio", el ciclo termina, si no cumple los objetivos el ciclo en esta fase debe comenzar nuevamente (Véase Apéndice E para detalles del ciclo de vida).

Capítulo IV. Desarrollo

En éste capítulo se describe la realización del sistema, desglosando las distintas actividades y fases involucradas de acuerdo a la metodología propuesta.

IV.1 Configuración del Proceso Unificado Racional

Antes de iniciar la captura de requerimientos la metodología exige la realización de un cronograma que especifique el número de iteraciones y las actividades a realizar en cada una de las distintas fases.

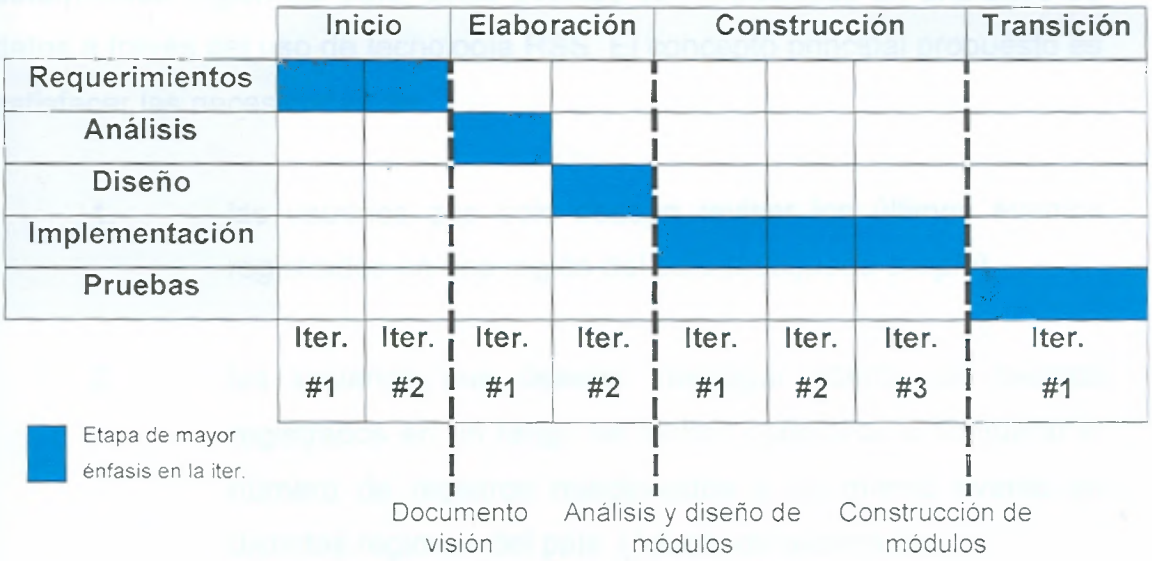

Figura 3. Configuración de RUP para el proyecto
Fuente: elaborado por el autor

Se establece un período de 4 semanas para cada una de las iteraciones y se define como hito: el documento de visión durante la fase de inicio, el análisis y diseño de los módulos del proyecto, la construcción de los módulos y en la fase de transición la ayuda en línea una vez realizado el sistema (véase la tabla E.2 del Apéndice E para planificación detallada).

IV.2 Inicio

En esta primera fase se da inicio a la obtención de los requerimientos funcionales y no funcionales de la aplicación durante un período de 8 semanas.

IV.2.1 Captura de requerimientos

Durante la primera iteración en la fase de Inicio (primeras 4 semanas del desarrollo del proyecto) se plantea un motor de búsqueda que tome en cuenta, tanto a medios de comunicación reconocidos, como a un conjunto de *blogs* previamente seleccionados que reflejen noticias y eventos en una determinada región del país, estos eventos son registrados en una base de datos a través del uso de tecnología RSS. El concepto principal propuesto es satisfacer las necesidades de:

1. los usuarios que solo desean revisar los últimos eventos registrados en una región del país (*búsqueda simple*).
2. los usuarios que desean investigar acerca de eventos registrados en un rango de fechas específico o comparar el número de registros relacionados a un mismo evento en distintas regiones del país (*búsqueda avanzada*).

Para ambos modelos el usuario puede escribir una palabra clave o frase que permita refinar la búsqueda, esta frase es analizada y posteriormente son devueltos los resultados a través de una interfaz gráfica en ambiente web.

Debido a que la aplicación se expone al acceso a través de Internet, se propone *Python* como lenguaje a utilizar y se establecen requerimientos generales de: seguridad, interfaz de usuario y casos de uso a fin de delimitar el alcance del proyecto, posteriormente se realiza una actualización al cronograma o configuración de RUP, como se muestra a continuación en la Figura 4:

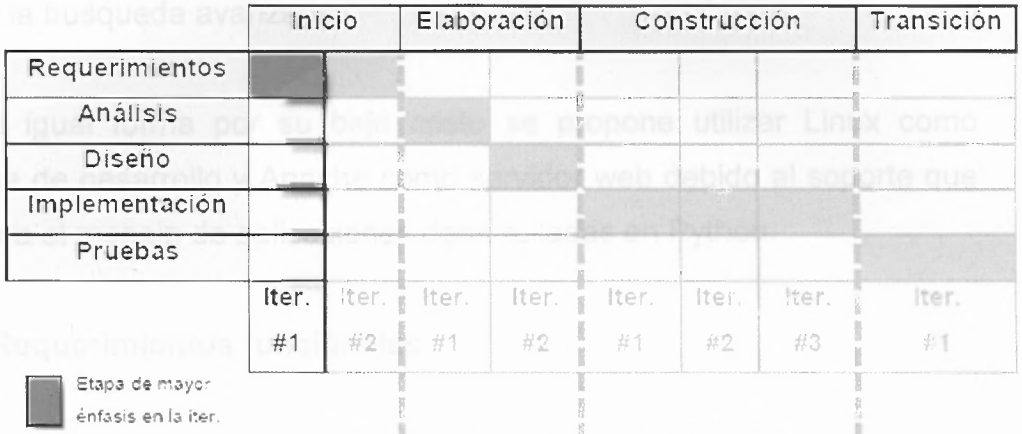

Figura 4. 1era iteración fase de inicio
Fuente: elaborado por el autor

IV.2.1.1 Requerimientos no funcionales

Se especifican a continuación algunas restricciones y elementos del ambiente de desarrollo que fueron encontrados y refinados a partir de la 5ta semana de la fase de inicio.

Seguridad: se propone utilizar un panel de control que permita controlar el acceso a las áreas administrativas del sistema, de igual forma, se plantea la creación de componentes que contribuyan a garantizar de forma independiente la integridad de la información indexada.

Interfaz de usuario: a través de ella los distintos tipos de usuario podrán realizar sus peticiones y visualizar los resultados. Se propone establecer un tipo de usuario "regular" que no requiere autenticación a través de una contraseña y un usuario "administrador" que sí requiere autenticación. Asimismo, se plantea una interfaz que haga uso de las nuevas tecnologías disponibles (AJAX, CSS, DHTML) a fin de ofrecer facilidades a los usuarios regulares que utilizan la búsqueda simple y aquellos que desean investigar a través de la búsqueda avanzada.

De igual forma por su bajo costo se propone utilizar Linux como plataforma de desarrollo y Apache como servidor web debido al soporte que ofrece para el manejo de aplicaciones desarrolladas en Python.

IV.2.1.2 Requerimientos funcionales

Al finalizar la captura de requerimientos básicos del sistema, y conocer los requerimientos no funcionales, se continúa en la 2da iteración de esta fase, y se comienzan a refinar los casos de uso, obteniendo de esta manera 3 módulos para el manejo de los datos en el sistema: indexación, búsqueda y mantenimiento.

A continuación se muestra en las tablas siguientes (Véase Tabla 1, 2 y 3) las funcionalidades o características principales, de cada uno de los módulos antes citados. Asimismo, el tipo de proceso al cual responde cada funcionalidad asociada, éste puede ser: interno o externo. Los procesos internos son ejecutados por el sistema, sin requerir la interacción con el usuario, mientras que los externos deben recibir información del usuario para ejecutar una tarea específica.

Módulo de indexación: este módulo se encarga de leer, interpretar y almacenar la información proveniente de los sitios a indexar. Estos sitios incluyen tanto bitácoras personales (*blogs*) como medios de comunicación reconocidos (Cadena Capriles, Agencia Bolivariana de Noticias, El Universal, entre otros).

Característica	Tipo de proceso
Solicitar parámetros de ejecución	Externo
Recibir y almacenar en la base de datos información relacionada al sitio a través de RSS	Interno

Tabla 1. Características del módulo de indexación

Fuente: Elaborado por el autor

Módulo de búsqueda: este módulo toma en consideración los 2 modelos de búsqueda (avanzada y simple). Ambos modelos dan origen a dos clases de procesos: internos y externos.

Característica	Tipo de proceso
Solicitar parámetros de ejecución	Externo
Recibir e interpretar data asociada a la solicitud	Interno
Almacenar parámetros de búsqueda a histórico de la base de datos	Interno

Tabla 2. Características del módulo de búsqueda (avanzada y simple)

Fuente: elaborado por el autor

Módulo de mantenimiento: este módulo se encarga de administrar los sitios indexados y sus artículos asociados así como los usuarios que pueden ingresar al sistema para realizar actividades de depuración o mantenimiento de la data (usuario administrador).

Característica	Tipo de proceso
Solicitar parámetros de ejecución	Externo
Recibir y procesar data asociada a la solicitud	Interno

Tabla 3. Característica del módulo de mantenimiento

Fuente: elaborado por el autor

IV.2.1.2.1 Casos de Uso

Para la sexta semana del desarrollo del proyecto es clara la necesidad de establecer de forma permanente 3 módulos principales a fin de satisfacer los requerimientos funcionales del sistema, por tal razón, a partir de ese momento, y durante las siguientes semanas de las primeras 2 fases (Inicio y Elaboración), se obtiene el diagrama general de casos de uso del sistema (Véase figura F.1. del Apéndice F).

En las siguientes tablas (Véase Tabla 4, 5 y 6) se especifica la descripción y el nombre de los casos de uso correspondientes a cada uno de los módulos.

Módulo de indexación:

Id.	Nombre del Caso de Uso	Descripción
1	Iniciar indexación	Se solicita al usuario administrador los parámetros de entrada para indexar un nuevo sitio.
2	Procesar indexación	Se interpreta la información suministrada por el usuario a través de un documento en formato RSS y posteriormente, de ocurrir cambios, se almacenan en la Base de Datos.
3	Sincronizar indexación	Se realiza la indexación a partir del servidor XML-RPC.

Tabla 4. Casos de uso del módulo de indexación

Fuente: elaborado por el autor

Módulo de búsqueda:

Id.	Nombre del Caso de Uso	Descripción
1	Iniciar búsqueda	Se solicita al usuario regular los parámetros de entrada para realizar una búsqueda (avanzada o simple) en la base de datos.
2	Realizar búsqueda	Se interpreta la información suministrada por el usuario a través del formulario y se procesa la búsqueda en el estado y región seleccionada.
3	Guardar histórico	Se almacenan las palabras clave utilizadas en un histórico de búsqueda para posterior análisis.

Tabla 5. Casos de uso del módulo de búsqueda

Fuente: elaborado por el autor

Módulo de mantenimiento:

Id.	Nombre del Caso de Uso	Descripción
1	Iniciar mantenimiento	Se solicita al usuario nombre de usuario (login) y contraseña (password).
2	Seleccionar tipo de parámetro	Se solicita al usuario administrador los parámetros de entrada para administrar un sitio, artículo o usuario.
3	Administrar sitio	Se recibe y procesa la data asociada al sitio seleccionado por el usuario administrador.
4	Administrar artículos	Se recibe y procesa la data asociada a los artículos seleccionados por el usuario administrador.
5	Administrar usuarios	Se recibe y procesa la data asociada al usuario seleccionado por el administrador autenticado actualmente en el sistema.

Tabla 6. Casos de uso del módulo de mantenimiento

Fuente: elaborado por el autor

A continuación se muestra, en la Figura 5, el diagrama general de los Casos de Uso del sistema:

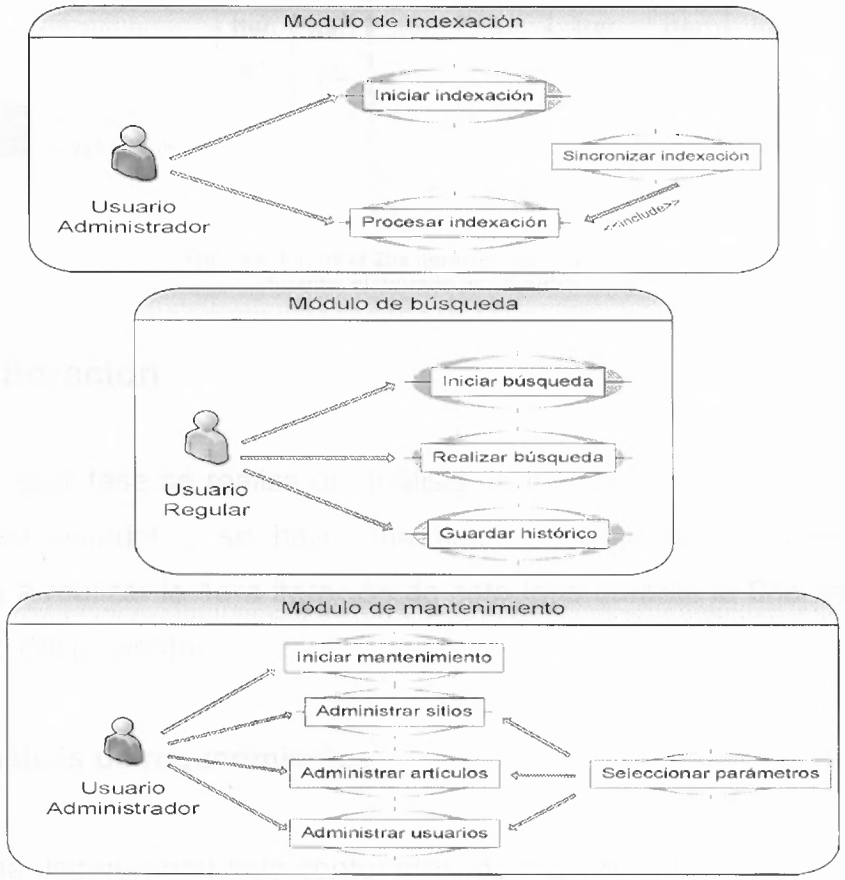

Figura 5. Diagrama general de Casos de Uso del sistema
Fuente: elaborado por el autor

La explicación detallada de los casos de uso correspondientes al módulo de indexación, módulo de búsqueda y módulo de mantenimiento del sistema puede ser encontrada en el Apéndice F.

Una vez finalizada la 2da iteración de la fase de inicio se han obtenido los primeros entregables (casos de uso) requeridos para la realización del documento visión. En la Figura 6 se muestra la actualización del cronograma una vez finalizada esta fase del desarrollo.

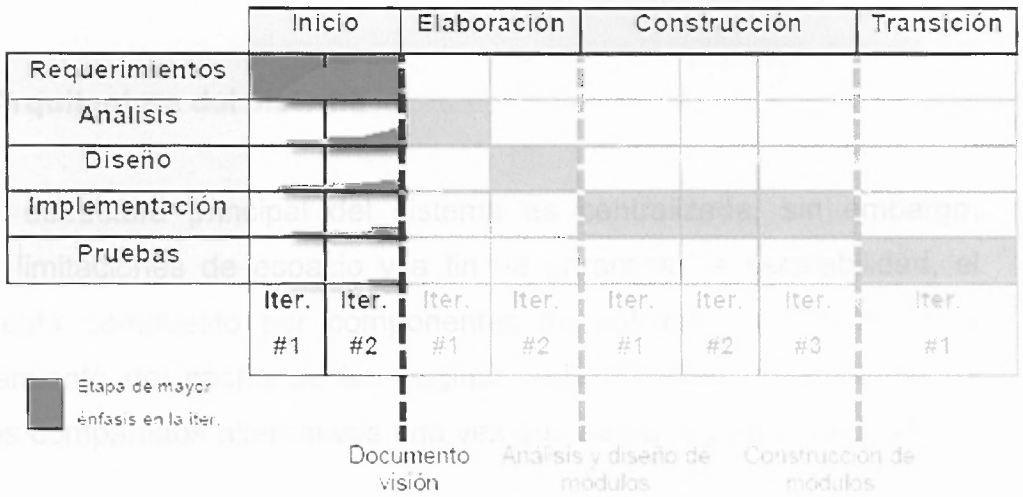

Figura 6. Fin de la 2da iteración (fase de inicio)
Fuente: elaborado por el autor

IV.3 Elaboración

En esta fase se realiza un análisis de los requerimientos encontrados en la fase anterior y se halla una solución al problema planteado, se comienza a realizar la 1era iteración de esta fase durante la 9na semana del desarrollo del proyecto.

IV.3.1 Análisis de requerimientos

Una vez que han sido capturados los requerimientos funcionales y no funcionales del sistema se procede a realizar un análisis de cada uno de los

módulos (indexación, búsqueda y mantenimiento), obteniendo de esta manera los diagramas de actividades asociados a los casos de uso correspondientes. Dichos diagramas representan de manera gráfica los pasos que deben seguirse para que los casos de uso en cada módulo cumplan con su propósito.

El detalle de los diagramas de actividades de los módulos de indexación, búsqueda y mantenimiento puede ser encontrado en el Apéndice G.

IV.3.1.1 Arquitectura del sistema

La estructura principal del sistema es centralizada, sin embargo, debido a limitaciones de espacio y a fin de garantizar la escalabilidad, el sistema está compuesto por componentes de software que permiten el almacenamiento del caché de las páginas indexadas en un ambiente de servidores compartidos alternativos una vez que se ha llegado a un límite de espacio.

Tanto los servidores adicionales para el almacenamiento del caché como el servidor principal (en el cual se realizan todos los procesos que solicita el usuario a través del navegador) se ejecutan bajo el sistema operativo *Red Hat Enterprise Linux (GNU)* y poseen como servidor web Apache, soporte para el lenguaje de la aplicación (Python 2.1) y MySQL 5 para el almacenamiento de los datos. Asimismo fueron instalados los siguientes componentes a fin de optimizar el manejo de la data: imagemagick (manejo de imágenes), Python Compiler (optimización de scripts en python).

IV.3.1.2 Organización del sistema

Desarrollar los componentes del sistema, utilizando el patrón de modelo-vista-controlador (MVC) como guía, resulta fundamental cuando se desea realizar un sistema en ambiente web que permita ser escalable y modificable en el tiempo. Estructurar el desarrollo del sistema de esta manera reduce el riesgo de afectar el correcto funcionamiento de toda la aplicación cuando se realizan actualizaciones sobre un componente específico porque MVC se encarga de mantener independiente: los datos, la interfaz de usuario y la lógica de control del sistema, razón por la cual, el desarrollador puede modificar un componente específico sin necesidad de modificar otros componentes previamente desarrollados. La aplicación de estos principios se traduce en una manera rápida y efectiva de aislar y resolver problemas detectados mientras el resto de los componentes continúan su normal funcionamiento.

IV.3.1.3 Análisis del módulo de indexación

Como se pudo conocer mediante la captura de requerimientos funcionales del sistema (fase de inicio), este módulo constituye el vínculo principal entre los sitios a indexar y el sistema, a continuación se analiza cada uno de los elementos involucrados en este módulo.

IV.3.1.3.1 Análisis Caso de Uso "iniciar indexación"

Antes de iniciar el proceso de indexación es necesario garantizar que la petición se realiza con la información requerida y los aspectos de seguridad validados, por tal razón, en esta etapa se realiza una verificación de sesión del usuario administrador y posteriormente se valida la existencia de la dirección escrita por el usuario y la validez de los campos seleccionados (ciudad y Estado). El proceso detallado se puede observar

gráficamente mediante el diagrama de actividades mostrado en la figura G.1 del Apéndice G.

IV.3.1.3.2 Análisis Caso de Uso "procesar indexación"

Análisis del manejo de documentos RSS

Los documentos con formato RSS son el vínculo principal con los sitios a indexar debido a que aportan información básica requerida para indexar un evento o noticia.

El documento RSS (ver Apéndice K) interpretado por el sistema está formado principalmente por etiquetas (*tags*) en lenguaje XML que guardan la siguiente información:

Título: es almacenado bajo la etiqueta *title* en el documento RSS y aporta información relacionada al nombre del sitio web, cuando se encuentra dentro de la etiqueta *channel*, y se refiere al título de un artículo cuando aparece dentro de la etiqueta *item*.

Enlace al documento: se almacena bajo la etiqueta *link* y contiene un hipervínculo al sitio principal cuando se encuentra dentro de la etiqueta *channel* y posee un enlace al artículo exacto de la noticia cuando se encuentra en la etiqueta *item*.

Categoría: se almacena bajo la etiqueta *category* y contiene uno o más elementos que guardan información relacionada a la categoría bajo la cual el autor del sitio web clasificó un artículo o noticia, ésta puede pertenecer a una o más categorías simultáneamente.

Descripción: se almacena bajo la etiqueta *description* y contiene un breve sumario que describe de forma general el artículo al cual hace referencia la etiqueta *link* (cuando ésta se encuentra dentro de la etiqueta *item).*

Fecha de publicación: se almacena bajo la etiqueta *pubDate* y contiene la fecha y la hora en la cual fue actualizado un artículo. Si bien se ha establecido como estándar de facto la utilización del formato de hora GMT +0000 no todos los sitios web de noticias en Venezuela cumplen con esta condición[5].

Esta información es clasificada de acuerdo a ciertos criterios de auto-clasificación que se obtienen mediante la interpretación de la información proporcionada por el documento RSS (categorías) y por el usuario (estado y ciudad al cual pertenece el sitio indexado).

IV.3.1.3.3 Análisis Caso de Uso "sincronizar indexación"

Análisis de sincronización del sistema con los sitios a indexar

Debido a la naturaleza del sistema es necesario mantener una sincronía entre: la base de datos, la interfaz presentada al usuario y la información que se encuentra disponible en los sitios remotos que son indexados por la aplicación.

[5] En investigaciones preliminares para el presente trabajo se pudo observar que la versión digital del diario "El Universal" y de la "Agencia Bolivariana de Noticias" utilizan la hora local de Venezuela (-04:00 GMT) en lugar de GMT+0000 y, adicionalmente, el primer sitio identifica la hora de publicación de un artículo en formato de 24 horas mientras que el segundo utiliza el formato de 12 horas (AM / PM). De igual forma el primer sitio presenta la fecha de acuerdo al formato: MM/DD/AAAA y el segundo DD/MM/AAAA donde DD: se refiere al día actual en formato numérico (2 dígitos), MM: mes en formato numérico (2 dígitos), AAAA: año en formato numérico (4 dígitos).

Asimismo, debe considerarse la utilización de un servidor que soporte XML-RPC a fin de mantener una comunicación en tiempo real entre el sitio a indexar y el sistema. A través de este servidor, el sitio a indexar se comunica con el sistema y notifica cuando ha ocurrido un nuevo evento o cambio en su base de datos lo cual permite actualizar la base de datos del sistema local en el instante en que se produzca una noticia en el sitio remoto.

Sin embargo, no todos los sitios a indexar poseen soporte completo para un pleno aprovechamiento de la tecnología RSS por lo que esta comunicación a través del servidor XML—RPC debe ser complementada con una interfaz gráfica que soporte AJAX y permita la transferencia, en tiempo real, de documentos tipo resumen (en formato RSS y ATOM) de aquellos sitios previamente indexados que no soporten comunicación directa con el servidor XML—RPC.

Una vez que el sitio remoto es validado se deben aplicar los criterios de auto-clasificación y posteriormente, teniendo en cuenta estas condiciones, es posible incrementar al máximo la usabilidad, interactividad y velocidad de carga de la aplicación web que, de igual forma, ofrecerá la posibilidad de almacenar en la base de datos local o en alguno de los servidores alternativos una copia en texto plano del documento (caché) que se ha indexado a fin de ofrecer al usuario la posibilidad de realizar una consulta en el futuro en caso que el documento sea eliminado del sitio original.

IV.3.1.4 Análisis del módulo de búsqueda

Como se pudo conocer mediante la captura de requerimientos funcionales del sistema, este módulo se encarga de administrar la búsqueda simple y la búsqueda avanzada que realiza el usuario regular desde la interfaz. A continuación se analiza cada uno de los elementos involucrados en este módulo.

IV.3.1.4.1 Análisis Caso de Uso "iniciar búsqueda"

Análisis de algoritmo para semántica en la búsqueda

Para interpretar el significado de la palabra clave o frase escrita por el usuario es necesario apoyarse en herramientas adicionales como diccionarios y una tabla de equivalencia que sea capaz de reconocer una misma palabra con o sin vocales acentuadas.

El uso de diccionarios permite determinar si una palabra fue o no escrita de manera correcta y así ofrecer sugerencias al usuario que permita encontrar el tema deseado. De igual forma el sistema debe utilizar codificación de tipo utf-8 a fin de ser capaz de realizar búsquedas sobre frases que poseen caracteres especiales, vocales acentuadas o la letra "ñ".

IV.3.1.4.2 Análisis Caso de Uso "realizar búsqueda"

Análisis de algoritmo de búsqueda

Para mejorar los resultados de la búsqueda es necesario tomar en cuenta las palabras individuales de la cadena escrita por el usuario, la frase completa y la posición del texto buscado dentro de la descripción así como el título del artículo.

La búsqueda debería manejar un sistema de puntos en el cual se asignen valores altos a los artículos que coinciden en el título, contienen parte de la frase, o la frase exacta. Por tal razón, debe analizarse tanto los títulos de los artículos como las descripciones.

IV.3.1.4.3 Análisis Caso de Uso "guardar histórico"

Después de realizarse una búsqueda resulta ideal almacenar la petición del usuario a fin de analizar en un futuro que nuevas palabras pueden ser agregadas al diccionario.

IV.3.1.5 Análisis del módulo de mantenimiento

Este módulo es el encargado de administrar los sitios y artículos contenidos en la base de datos del sistema. A continuación se analiza cada uno de sus elementos.

IV.3.1.5.1 Análisis Casos de Uso del módulo mantenimiento

A fin de mejorar la calidad de la búsqueda y la clasificación del contenido es necesario incluir un módulo que, mediante el manejo de sesiones de acceso restringido, permita al usuario administrador tener control sobre la ubicación de los artículos y su disponibilidad así como la disponibilidad del sitio que ha sido indexado.

Poseer una interfaz gráfica que permita, de manera sencilla, reclasificar aquellos artículos que no fueron catalogados de manera correcta por el autor o bloquear de forma temporal (o permanente) tanto artículos como sitios completos que en un momento dado publiquen contenido de baja relevancia o que disperse las búsquedas contribuye a garantizar al usuario final resultados óptimos.

Al finalizar el análisis de cada uno de los módulos se ha completado la primera iteración de la fase de elaboración, actualizando el cronograma como se muestra en la Figura 7.

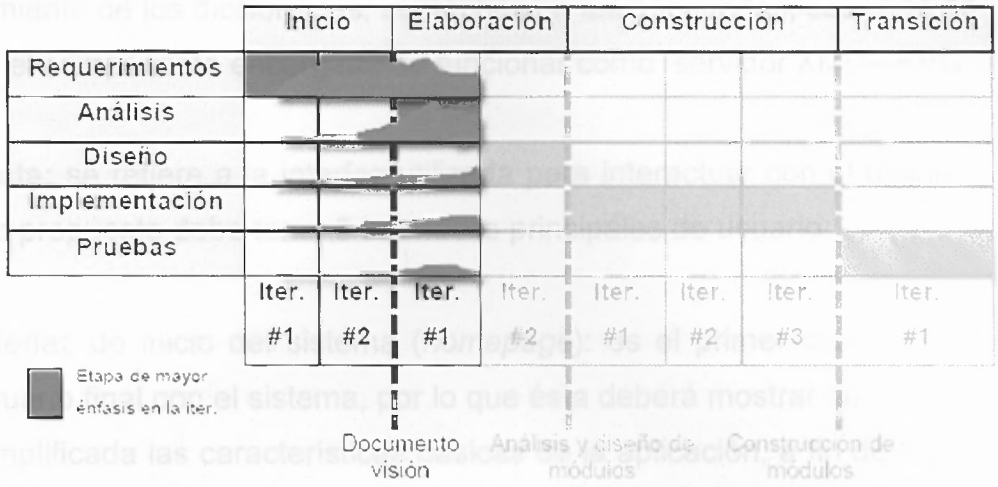

Figura 7. Fin de la 1era iteración (fase de elaboración)
Fuente: elaborado por el autor

IV.3.2 Diseño de componentes

Una vez que ha sido recaudada información fundamental del sistema, a través de la captura y análisis de requerimientos, se da inicio a la segunda iteración de la fase de elaboración debido a que se cuenta con elementos suficientes para el diseño de los componentes del sistema cuyas funcionalidades son explicadas a través de los diagramas de secuencia mostrados en el Apéndice H.

IV.3.2.1 Diseño de la organización del sistema

Utilizando como base el análisis de la estructura del sistema, se especifican a continuación 3 componentes fundamentales:

Modelo: representa el dominio de los datos sobre los cuales opera el sistema. Contiene las funciones encargadas del manejo de la información suministrada por los sitios a indexar: artículos, fechas de publicación, descripciones, manejo del caché, así como: la base de datos del sistema,

funcionamiento de los diccionarios, semántica en las búsquedas, sesiones de usuario y el componente encargado de funcionar como servidor XML—RPC

Vista: se refiere a la interfaz utilizada para interactuar con el usuario, el modelo propuesto debe tener 3 interfaces principales de usuario:

- Interfaz de inicio del sistema (*homepage*): es el primer contacto del usuario final con el sistema, por lo que ésta deberá mostrar de manera simplificada las características básicas de la aplicación, a fin de lograr captar la atención y confianza del usuario. Para lograr estos objetivos se debe mostrar un contenido que posea un rango mínimo de calidad (a través de la auto-clasificación del contenido) y que de forma simultánea pueda representar un tema de interés para el usuario que lo visualiza.

- Interfaz de búsqueda en el sistema: es la encargada de mostrar al usuario los resultados de la búsqueda de acuerdo a una palabra clave o frase escrita previamente. Se debe ofrecer la posibilidad de realizar una búsqueda simple, para aquellos usuarios que solo desean visualizar los últimos eventos registrados en todo el país o en una región específica, y una avanzada para los usuarios que desean realizar una investigación mas profunda de acuerdo a varias regiones del país y fechas específicas.

- Interfaz de administración del sistema: es la única interfaz que posee control a través del manejo de sesiones de usuario y permite al usuario administrador realizar el mantenimiento del sistema. La interfaz deberá ofrecer al usuario administrador un conjunto de opciones que permitan editar de manera sencilla los sitios indexados y su contenido.

Controlador: es el componente encargado de responder a los eventos generados por el usuario, el controlador accede al modelo, actualizándolo de acuerdo a la acción solicitada por el usuario. Se encarga de decidir cuando deben ser llamadas las funciones que soportan tecnología AJAX.

Estos 3 componentes están en constante comunicación de la manera siguiente:

- El *modelo* se comunica con la *vista* a través de funciones previamente programadas.
- La *vista* se comunica con el *controlador* apoyándose en la utilización de tecnología AJAX desde el navegador que utiliza el usuario.
- El *controlador* establece un enlace entre la *vista* y el *modelo*.

IV.3.2.2 Diseño del módulo de indexación

Como se ha especificado durante el análisis, los documentos con formato RSS son el medio fundamental de la comunicación entre, los sitios remotos y el sistema local, éstos, son administrados mediante los componentes que se muestra a continuación.

IV.3.2.2.1 Diseño de componentes para "iniciar indexación"

Validar sitio: en primer lugar verifica la existencia de un documento RSS en la dirección suministrada por el usuario, de no existir, el sistema modifica la dirección del usuario automáticamente intentando buscar el documento RSS de acuerdo a un conjunto de criterios que permiten conocer las ubicaciones más frecuentes de estos documentos dada una

dirección web. Si la dirección no es encontrada, se genera un mensaje de error.

Validar categorías del sitio: se encarga de cargar en el sistema el conjunto de categorías autorizadas para un sitio en particular, todos los artículos pertenecientes a categorías prohibidas son bloqueados automáticamente. El administrador del sistema debe configurar previamente que categorías son, por defecto, prohibidas y cuales son, por defecto, aceptadas.

Conversión utf8: utf-8 es un formato de codificación de caracteres ampliamente utilizado que ofrece soporte para una amplia cantidad de caracteres especiales y símbolos. Resulta de vital importancia para el desarrollo de esta aplicación debido a que ofrece la posibilidad de manejar textos que contengan: vocales acentuadas, letra ñ (mayúscula y minúscula), los símbolos de apertura de exclamación o interrogación, entre otros valores. El convertidor utf-8 se encarga de llevar toda la información recabada de los sitios remotos a un solo formato estándar en el cual se puedan realizar búsquedas de palabras acentuadas, búsquedas que contengan símbolos o ciertos caracteres especiales.

Conversión de fecha: para evitar inconsistencias en el manejo de los formatos de fecha y hora, el sistema debe leer el formato de fecha que utiliza el sitio remoto, interpretarlo, transformarlo a GMT+0000 y convertirlo posteriormente al formato unix *timestamp* para ser almacenado en la base de datos. Véase Apéndice K.2 para más detalles de este formato para el manejo fechas.

IV.3.2.2.2 Diseño de componentes para "procesar indexación"

Actualizar sitio: verifica los cambios ocurridos en los artículos de un sitio existente en caso que el autor modifique alguno de los artículos.

Insertar sitio: verifica el sitio, luego sus categorías y posteriormente realiza la inserción o la actualización de los artículos correspondientes en la base de datos del sistema.

IV.3.2.2.3 Diseño de componentes para "sincronizar indexación"

Sincronización del sistema con los sitios a indexar

A fin de mantener sincronizada la información disponible en los sitios indexados y por indexar con el sistema es necesario que se cumplan las características que se muestran a continuación.

Servidor XML-RPC: recibe peticiones de los sitios remotos a fin de mantener la información actualizada en tiempo real. Cuando un sitio es actualizado éste envía una petición al servidor XML-RPC del sistema y el servidor se encarga de realizar una llamada al componente de manejo de documento RSS quien realiza una actualización o inserción del sitio y de sus respectivos artículos asociados.

AJAX-RSS: en caso que el cliente (sitio remoto) no posea soporte completo para la tecnología RSS, una interfaz con soporte de AJAX extraerá en, tiempo real, el archivo resumen en formato RSS del sitio remoto en el momento en que el usuario realiza la petición. Dadas las características de este archivo, la transferencia resulta muy rápida y la información es mostrada de manera sincronizada con el sitio remoto.

Indexación del caché: una vez que ha sido validado el sitio a indexar se identifica cual de los métodos fue utilizado y posteriormente, de acuerdo, al caso se realiza una copia del caché inmediata o diferida. Si el sitio indexado utilizó directamente el servidor XML-RPC, se realiza una copia del caché de forma inmediata debido a que éste servidor se está ejecutando de manera independiente y no reduce el tiempo de respuesta, sin embargo, si los sitios recientemente indexados corresponden a una cantidad significativamente alta, es posible que se reduzca el tiempo de respuesta, si esto ocurriera se difiere la indexación hasta que el usuario administrador la autorice desde el módulo de mantenimiento.

IV.3.2.3 Diseño del módulo de búsqueda

IV.3.2.3.1 Diseño de componentes para "iniciar búsqueda"

Algoritmo para semántica en la búsqueda

De acuerdo al análisis realizado en la etapa anterior, para el manejo de semántica en la búsqueda se debe contar con un componente que cumpla con las características descritas a continuación:

Depuración de la entrada: debe eliminar, de la frase o palabra clave escrita por el usuario, código malicioso o caracteres especiales no permitidos. De igual forma debe interpretar el uso de comillas dobles como una petición para realizar una búsqueda exacta en la base de datos del sistema.

Sugerencia de escritura: tiene como objetivo principal conseguir un resultado aproximado a una palabra conocida cuando la cadena de búsqueda escrita por el usuario no ofrece resultados satisfactorios. Para interpretar la palabra se debe hacer uso de un diccionario, una base de

datos que contenga una amplia lista de palabras y sugerencias asociadas a ellas.

Diccionario de equivalencias: está orientado a conseguir distintas formas en las que puede estar conjugado un mismo verbo.

Cadenas sin acentos: en muchas oportunidades el usuario que realiza la búsqueda, o los autores de los sitios indexados, omiten la acentuación bien sea al buscar un tema específico o al escribir un artículo, por tal razón, es necesario contar con una funcionalidad que permita eliminar los acentos de una palabra para mejorar los resultados de la búsqueda, teniendo siempre en cuenta las sugerencias del diccionario.

IV.3.2.3.2 Diseño de componentes para "realizar búsqueda"

Algoritmo de búsqueda

Teniendo en cuenta los algoritmos estudiados en las fases previas se realiza un diseño que se adapte a las necesidades del sistema. Para dar inicio al algoritmo, una vez que se tiene la palabra o frase escrita por el usuario, la ciudad y el estado sobre el cual se desea realizar la búsqueda se deben cumplir los siguientes criterios mientras se examina la base de datos del sistema:

- La búsqueda debe realizarse al menos en el título de los artículos indexados y en el contenido de los mismos.

- Durante la búsqueda se establece un sistema de puntos a fin de darle relevancia a los artículos que reflejan de mejor manera la petición realizada por el usuario.

- La primera búsqueda se realiza buscando coincidencia exacta entre la palabra o palabras escritas por el usuario y las existentes en la base de datos. De encontrarse coincidencia esta etapa se realiza la asignación más alta de puntos de todo el algoritmo siempre y cuando existan artículos en los cuales la palabra clave o frase buscada por el usuario coincide parcial o totalmente con el título del artículo.

- Se realiza una segunda búsqueda utilizando palabras que se escriben de manera similar a la palabra o palabras escritas por el usuario, cuando se realiza esta segunda fase de exploración, se asigna menor puntuación a los artículos que coinciden con la búsqueda debido a que las palabras utilizadas representan una aproximación y no son necesariamente las palabras exactas escritas por el usuario.

- Si el orden de las palabras escritas coincide exactamente con el orden en que se encuentran las palabras escritas en el título de un artículo se asigna una puntuación igualmente alta aunque un poco menor a los puntos otorgados cuando el título coincide exactamente con el título del artículo.

- Si las palabras escritas por el usuario se encuentran contenidas en el título de un artículo y de igual forma se encuentran contenidas en la descripción se asignará una puntuación.

- Se realiza un último análisis de los resultados obtenidos, determinando cuales artículos han sido publicados de manera reciente y asignando puntuación a los mismos si la búsqueda que se realiza es una búsqueda simple, si la búsqueda es avanzada y

el usuario ha especificado un rango de fechas, no se realiza la asignación de puntos para darle relevancia a la coincidencia y no el momento en que han ocurrido los eventos reportados por los artículos que forman parte de los resultados de la búsqueda.

- Una vez encontrado los artículos que coinciden con los criterios de búsqueda, se dividen en páginas de *n* elementos, se ordenan de acuerdo a la puntuación obtenida y se descartan aquellos elementos con puntuación irrelevante dentro de los resultados de la búsqueda (Véase Apéndice L para detalle del sistema de manejo de puntos).

IV.3.2.3.3 Diseño de componentes para "guardar histórico"

Una vez que se ha realizado una búsqueda se almacena un histórico que contiene las palabras clave utilizada en la búsqueda y el orden en que fueron escritas por el usuario, de esta forma, el usuario administrador puede observar en la base de datos que patrones de búsqueda siguen los usuarios y de esa manera ir mejorando, en el tiempo, la construcción de los diccionarios existentes.

IV.3.2.4 Diseño del módulo de mantenimiento

IV.3.2.4.1 Administración del contenido

Se debe incorporar en la interfaz gráfica inicial del sitio una ventana que ofrezca al usuario la posibilidad de autenticarse. Una vez autenticado el usuario puede visualizar la interfaz de administración de contenido del sistema que cumple con las especificaciones que se indican a continuación:

- Interfaz de bienvenida donde el usuario administrador pueda seleccionar: sitios, usuarios y artículos.

- Capacidad de moderación de sitios y artículos. A través de esta interfaz del usuario administrador es posible asociar nuevas ciudades o estados a un mismo artículo, bloquear ciertos contenidos o sitios y actualizar las copias en caché de un artículo específico.

Al finalizar el diseño del módulo de mantenimiento culmina también la 2da iteración planificada para la fase de elaboración y se realiza la actualización en el cronograma de RUP (Véase Figura 8).

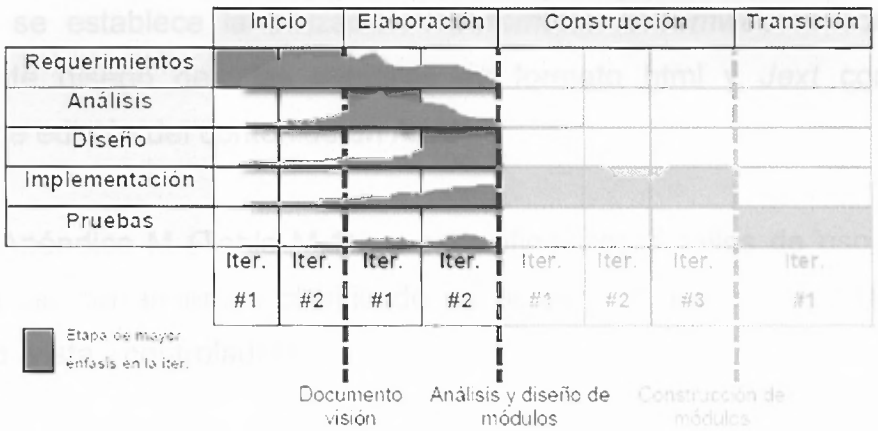

Figura 8. Fin de la 2da iteración (fase de elaboración)
Fuente: elaborado por el autor

IV.4 Construcción

Para la semana 17 se da inicio a la primera iteración de la fase de construcción. A continuación se muestra el proceso mediante el cual se traduce a código fuente los conceptos obtenidos durante las etapas previas de análisis y diseño.

IV.4.1 Implementación

Para realizar la conversión a código fuente es necesario utilizar un conjunto de herramientas que permitan el desarrollo de los 3 componentes

fundamentales del sistema: modelo, vista y controlador. A continuación se presenta una lista de las herramientas y su uso de acuerdo a los componentes citados anteriormente.

Modelo y Controlador: para la realización de los distintos módulos se define Python como lenguaje y Jext como entorno para el desarrollo de la aplicación debido a que ofrece facilidades de edición y manejo de archivos tipo .py (Python) y PHP.

Vista: se establece la utilización *Macromedia Dreamweaver* como herramienta de diseño de las plantillas en formato html y *Jext* como herramienta de edición del contenido en AJAX.

En el Apéndice M (Tabla M.1) se especifican los detalles de uso de cada una de las herramientas clasificado de acuerdo al patrón de diseño MVC (modelo -vista - controlador).

IV.4.1.1 Implementación del módulo de indexación

De acuerdo a los parámetros indicados anteriormente se utiliza el lenguaje Python para definir un conjunto de clases y métodos que permitan la realización de las operaciones del módulo a fin de lograr el objetivo principal de indexar un nuevo sitio y sus artículos asociados. Este procedimiento se realiza durante la primera iteración de la fase de construcción.

IV.4.1.1.1 Objetos del modelo

Indexar sitio: este objeto es invocado para validar todos los aspectos asociados a la indexación o actualización de un sitio existente en la base de datos del sistema. Para ello, el constructor del objeto, identifica en primer

lugar el tipo de sitio que se desea indexar, mediante un conjunto de patrones predefinidos, que permiten obtener la ruta exacta del documento RSS.

Al recibir un sitio el sistema intenta encontrar automáticamente el documento RSS o su equivalente, ATOM, en la dirección escrita por el usuario, si ocurre un error, se procede a verificar la coincidencia con los patrones predefinidos, mediante la identificación de los siguientes elementos:

- Existencia del subdominio "blogspot.com". De encontrarse se agrega automáticamente la cadena "rss.xml" o "atom.xml", que corresponde al documento con contenido RSS para blogs almacenados bajo este dominio.

- Si la dirección escrita por el usuario no finaliza con la cadena "feed" y es un dominio de primer nivel o un subdomnio de wordpress.com, se agrega la cadena al final de la dirección para intentar verificar si efectivamente se trata de un blog realizado en *wordpress*.

Si la dirección escrita no cumple ninguno de los patrones el objeto "indexar sitio" selecciona la vista correspondiente a un mensaje de error y es enviada una señal de "finalización con errores" al controlador quien se encarga de notificar al usuario. Si el sitio fue identificado por alguno de los patrones o se suministró la dirección del documento RSS directamente, el sistema extrae de allí los datos, los convierte al formato utf-8 y los almacena en un vector para compararlo posteriormente con los sitios indexados en la base de datos.

Una vez detectada la existencia o no del sitio en la base de dato, se marca el sitio como una actualización o como un nuevo ingreso y se procede a extraer, y a almacenar en otro vector, los artículos asociados al sitio.

Posteriormente, se almacena el sitio en la base de datos y se establece una comunicación con el objeto "indexar artículos" a fin de incluir o actualizar en la base de datos los artículos asociados.

Indexar artículos: se recibe un vector que contiene los artículos asociados al sitio y se procede a verificar cada una de las categorías asociadas a los artículos a través del método *categbloq(string categoria)* que se encarga de determinar cuales categorías de las que fueron asignadas por el autor del sitio remoto están bloqueadas o autorizadas para su indexación en el sistema local. Las categorías bloqueadas o autorizadas están preestablecidas en la base de datos y pueden ser modificadas por el administrador desde el módulo de mantenimiento. A fin de mantener la base de datos con la mayor cantidad de objetividad posible y ofrecer contenido de interés para los venezolanos la regla principal es mantener bloqueado, por defecto, los artículos clasificados bajo categorías de: opinión, farándula, internacional, *english version*, entre otros. Por defecto, son aceptados los artículos que pertenezcan a categorías con el nombre de: Venezuela, Caracas o cualquiera de las ciudades del país que han sido indexadas en el sistema.

Culminada la verificación de los artículos se procede a realizar la actualización en la base de datos.

Servidor XML-RPC: es un servidor que funciona como un componente independiente y se encarga de recibir peticiones enviadas al sistema local desde los sitios indexados. Las peticiones interpretadas por el servidor son documentos XML que poseen información acerca de los últimos artículos actualizados por el sitio que envía la petición. Véase el Apéndice N para mayor detalle del proceso de comunicación entre el sistema local y el sitio remoto.

Indexar caché: verifica el tipo de petición, si ésta proviene del servidor XML—RPC se realiza una actualización inmediata del caché asociado a un artículo específico, de lo contrario, se marca como diferida en la base de datos para posterior actualización mediante el módulo de mantenimiento.

IV.4.1.1.2 Métodos de la vista y el controlador

La vista principal que permite enlazar con el módulo de indexación consiste en una interfaz mediante la cual se solicita: la dirección del documento RSS del sitio remoto, el estado (al cual están asociado la mayoría de artículos que posee) y la ciudad. Es posible seleccionar como punto de interés: todo el país, una ciudad en un estado o todas las ciudades dentro de un estado.

El controlador se encarga de establecer las condiciones que muestren al usuario la ventana principal para indexar documentos, el mensaje de éxito o el mensaje de fracaso. Una vez implementada la vista, el controlador y el modelo, del módulo de indexación, ha culminado la primera iteración de la fase de construcción y se actualiza el cronograma del proyecto como se muestra a continuación:

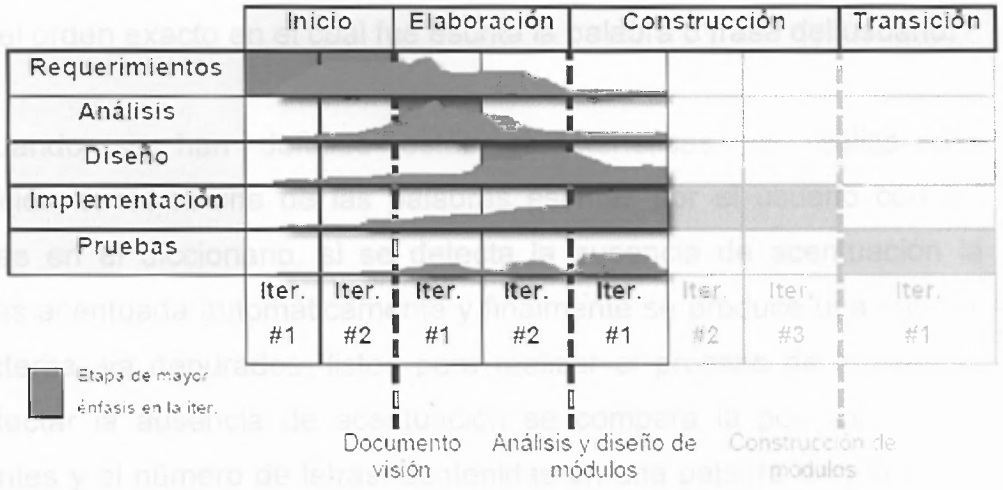

Figura 9. Fin de la 1era iteración (fase de construcción)
Fuente: elaborado por el autor

IV.4.1.2 Implementación del módulo de búsqueda

Se da inicio a la segunda iteración de la fase de construcción y es utilizado el lenguaje Python para definir las clases y métodos que permitan obtener resultados relevantes a partir de una palabra o frase escrita por el usuario. La utilización de la herramienta Python Compiler (PYC), propia del lenguaje, resulta fundamental en esta etapa del desarrollo debido a que contribuye a optimizar la velocidad de la aplicación.

IV.4.1.2.1 Objetos del modelo

Análisis semántico: este objeto está orientado a depurar y preparar una cadena de caracteres para su posterior búsqueda en la base de datos.

Una vez que se recibe la palabra o frase escrita por el usuario éste módulo procede a eliminar código malicioso que pueda ser interpretado como una secuencia SQL, posteriormente se identifica si en la cadena existe el uso de comillas dobles, de ocurrir esto se asigna a la variable *tipobus* un valor de 1 para indicar que la búsqueda será de tipo "exacta", la búsqueda exacta busca la cadena de caracteres como fue escrita literalmente por el usuario, la búsqueda regular intenta buscar variaciones de la frase sin priorizar el orden exacto en el cual fue escrita la palabra o frase del usuario.

Cuando se han definido estas características se realiza una comparación de cada una de las palabras escritas por el usuario con las contenidas en el diccionario, si se detecta la ausencia de acentuación la palabra es acentuada automáticamente y finalmente se produce una cadena de caracteres, ya depurados, listos para realizar el proceso de búsqueda. Para detectar la ausencia de acentuación se compara la posición de las consonantes y el número de letras, contenidas en una palabra escrita por el usuario, con las que se encuentran en el diccionario, de producirse

coincidencia, entre las consonantes y el número total de caracteres de la cadena, se procede a realizar el reemplazo automático, de no haber coincidencia se emite, de forma adicional a la cadena ya depurada, una cadena sugerida solo si se detectó coincidencia parcial entre el número de consonantes.

El diccionario del sistema se maneja a través de una clase que posee una lista y una estructura de tipo "diccionarios de datos" (por ahora, presente solo en *Python*).

Realizar búsqueda: una vez que se realiza el análisis semántico se procede a buscar de forma textual la cadena escrita por el usuario (si la búsqueda es del tipo exacta), o las palabras por separado (si la búsqueda no es exacta). El proceso de búsqueda de la palabra o frase escrita por el usuario se realiza a través de una conexión a la base de datos mediante la cual se analiza la descripción de los artículos y el título de los mismos, si se trata de una búsqueda avanzada, se toman en cuenta los rangos de fecha o los autores de los artículos. Este análisis consiste en manejar un sistema de puntos similar al algoritmo *PageRank* mediante el cual se realiza una asignación de puntos de acuerdo a la relevancia de la frase buscada en los artículos indexados (Véase Apéndice L para visualizar los criterios).

A medida que se va realizando la búsqueda la información es almacenada en una lista, de manera temporal, y esta lista es agregada a una estructura de tipo "diccionario" (Véase Apéndice O para mas detalles) que es la encargada de llevar el control de la puntuación obtenida por un artículo. Cuando se indexa un artículo, se almacenan los atributos mostrados en la Figura O.1 del Apéndice O.

IV.4.1.2.2 Métodos de la vista y el controlador

Para el manejo de la vista se utiliza un objeto definido como "plantilla" encargado de cargar la vista asociada a la búsqueda avanzada o a la búsqueda simple[6]. El controlador se encarga de interpretar el tipo de búsqueda realizada por el usuario y los parámetros escritos.

Al finalizar la implementación, de los objetos del modelo y los métodos asociados a la vista y al controlador, también culmina la segunda iteración de la fase de construcción, por lo cual se realiza la actualización del cronograma, como se muestra en la Figura 10:

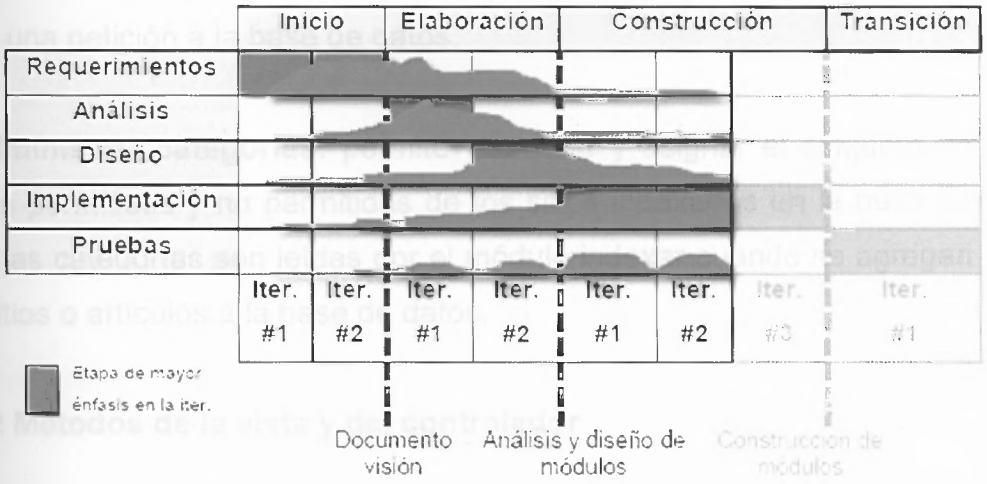

Figura 10. Fin de la 2da iteración (fase de construcción)
Fuente: elaborado por el autor

IV.4.1.3 Implementación del módulo de mantenimiento

Comienza en la tercera iteración de la fase de construcción y se establece administrar el conjunto de almacenados en el sistema utilizando el lenguaje Python.

[6] Ambos tipos de búsqueda se diferencian solo a nivel de la vista. La avanzada solicita al usuario un rango de fechas, y otros campos que permiten una especificación mas clara, mientras que la búsqueda simple, solicita el mínimo de datos requeridos.

IV.4.1.3.1 Objetos del modelo

Administrar sitio: posee un conjunto de métodos asociados que permiten la comunicación directa con el controlador a fin de bloquear o desbloquear un sitio de la base de datos del sistema local. El bloqueo permite evitar que todos los artículos asociados a un sitio puedan ser encontrados cuando se realiza la búsqueda, para ello se hace una petición a la base de datos y se cambia el atributo "bloquear" al valor 1 para indicar que el sitio está bloqueado.

Administrar artículos: permite asociar nuevas ciudades o estados a artículos existentes, de igual forma permite bloquearlos o desbloquearlos mediante una petición a la base de datos.

Administrar categorías: permite visualizar y asignar el conjunto de categorías permitidas y no permitidas de los sitios indexados en la base de datos. Estas categorías son leídas por el módulo indexar cuando se agregan nuevos sitios o artículos a la base de datos.

IV.4.1.3.2 Métodos de la vista y del controlador

Para el manejo de la vista se utiliza un objeto definido como "plantilla" encargado de cargar la vista asociada a la administración de los sitios, artículos y categorías. El controlador se encarga de interpretar el tipo de elemento a administrar.

Al finalizar la implementación de los objetos asociados al modelo, los métodos pertenecientes a la vista y al controlador, se marca la culminación de la tercera iteración en el cronograma tal como se puede apreciar en la Figura 11.

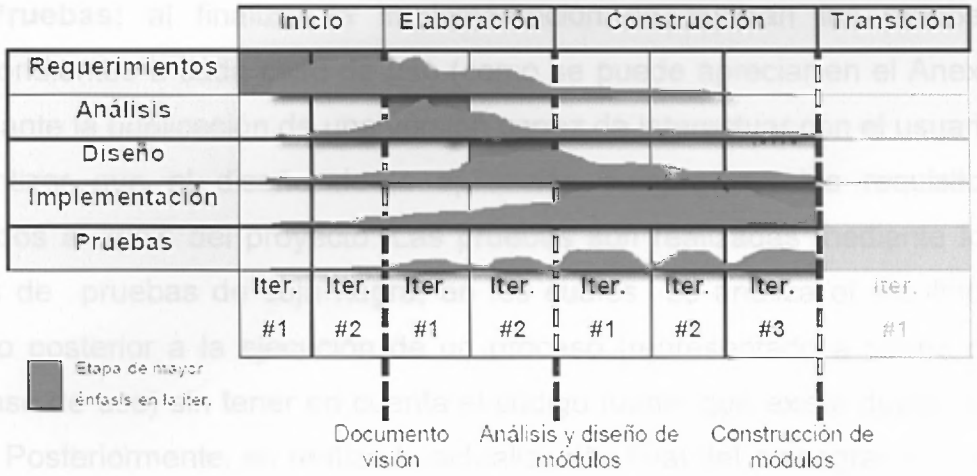

Figura 11. Fin de la 3era iteración (fase de construcción)
Fuente: elaborado por el autor

IV.4.2 Validación, verificación y Pruebas

Validación: En esta etapa se da inicio a la fase de transición mediante la cual se puede encontrar una correspondencia cercana entre los módulos planteados y los módulos implementados frente a las expectativas del usuario debido a que se establece una clasificación que permite encontrar un artículo de acuerdo a una región específica del país teniendo en cuenta el tema buscado. Se presenta una relación directa entre los lugares descritos por el artículo y la clasificación del mismo dentro del sistema local.

Verificación: A fin de comprobar que el código implementado se ajusta a los módulos realizados se procedieron a realizar las siguientes verificaciones:

- ☑ Los artículos se encuentran clasificados de acuerdo a una región en particular.
- ☑ La selección de un estado en particular permite obtener un listado de artículos asociados a ese estado.
- ☑ Los sitios remotos envían peticiones al sistema local para mantener actualizada la información en tiempo real.

Pruebas: al finalizar la implementación se realizan las pruebas correspondientes a cada caso de uso (como se puede apreciar en el Anexo P) mediante la publicación de una versión capaz de interactuar con el usuario y garantizar que el diseño de la aplicación cumple con los requisitos planteados al inicio del proyecto. Las pruebas son realizadas mediante los criterios de pruebas de caja negra, en los cuales se analiza el resultado obtenido posterior a la ejecución de un proceso (representado a través de cada caso de uso) sin tener en cuenta el código fuente que existe detrás del mismo. Posteriormente, se realiza la actualización final del cronograma RUP como se muestra en la Figura 12:

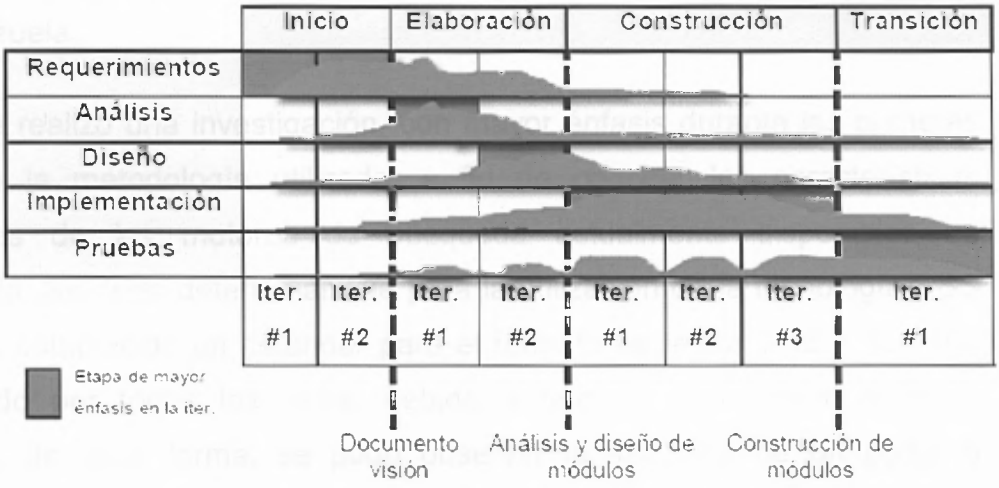

Figura 12. Fin de la 1era iteración (fase de transición)
Fuente: elaborado por el autor

IV.4.3 Estudio de casos

Se realizó una verificación del tiempo de respuesta para la búsqueda de palabras claves relevantes durante el mes de septiembre, para ellos se evaluó el desempeño del algoritmo utilizando distintas palabras clave como: "maletín", "maletin" (sin acentuar), "inundación" e "inundacion" (sin acentuar), "indulto", pudiéndose determinar una mejora considerable después de utilizar la herramienta PYC para la optimización del código. Véase Apéndice Q para detalle de los casos estudiados.

Resultados

Siguiendo la metodología del Proceso Racional Unificado de desarrollo de software, se cumplieron satisfactoriamente los objetivos requeridos al inicio de la realización del proyecto. Se logró diseñar e implementar con éxito el sistema. Los resultados del presente trabajo especial de grado, son presentados de acuerdo a los objetivos alcanzados:

1. Realizar levantamiento de información referente a directorios de *blogs*, motores de búsqueda de medios de comunicación y motores de búsqueda generales que permiten encontrar contenido relacionado a Venezuela.

Se realizó una investigación, con mayor énfasis durante las primeras fases de la metodología utilizada, a fin de conocer las características principales de los motores de búsqueda actualmente disponibles en Venezuela. Se pudo determinar que para la utilización de la tecnología RSS no se ha establecido un estándar para el formato de fecha y hora que sea compartido por todos los sitios, debido a que se encontraron diversos formatos, de igual forma, se pudo observar la ausencia de un portal o buscador global de noticias y eventos del país que involucre el punto de vista de editores de *blogs* y medios de comunicación reconocidos (versiones en línea de periódicos reconocidos).

2. Realizar levantamiento de información con respecto a los algoritmos de búsqueda utilizados actualmente y su evolución.

De acuerdo a la investigación realizada se pudo encontrar gran una variedad de algoritmos que parten desde aquellos donde la prioridad es conseguir documentos relacionados a una palabra clave, sin valorar la forma en como está escrita, hasta los recientes motores de búsqueda que priorizan

establecer parámetros semánticos que contribuyan a darle un significado a la palabra o frase escrita por el usuario y de esa forma encontrar registros mas relevantes y precisos.

3. Definir políticas de seguridad para el almacenamiento de información, ingreso al sistema y mantenimiento de la integridad de la información indexada.

Para el cumplimiento de este objetivo se diseñó un módulo con la capacidad de manejar sesiones y tipos de usuario a fin de restringir el acceso al sistema y así evitar modificaciones por parte de terceros no autorizados, de igual forma, se establecieron criterios de auto-clasificación en los cuales ciertos tipos de sitios remotos, y artículos asociados no relevantes (farándula opinión, versiones en inglés), son automáticamente ignorados o bloqueados por el sistema a fin de garantizar un mínimo de calidad en la información indexada.

4. Definir e implementar el algoritmo de búsqueda y ordenamiento a utilizar de acuerdo a las tareas que se ejecutarán en el servidor.

Posterior a la investigación, de los algoritmos de búsqueda, se realizó un análisis de aquellos elementos compatibles con el sistema a desarrollar, por tal razón, se implementó un algoritmo de búsqueda y ordenamiento basado en un sistema de puntuación de acuerdo a la relevancia de cada artículo, de igual forma se estableció el uso de un diccionario capaz de darle un significado semántico a la frase escrita por el usuario y de esta manera realizar búsquedas aunque la cadena escrita por el usuario contenga ligeros errores ortográficos que puedan ser detectados y corregidos por el sistema.

5. Diseñar e implementar un módulo *backend* que permita indexar páginas existentes.

Se realizó un módulo de indexación capaz de indexar sitios remotos y sus artículos asociados de 2 formas: por demanda del servidor local y por demanda del servidor remoto (utilizando un servidor local XML—RPC). Ambas maneras permiten indexar artículos asociados a sitios previamente registrados y autorizados por el usuario administrador desde el módulo de mantenimiento.

6. Diseñar e implementar un módulo *frontend* que permita al administrador depurar la información y a los usuarios indexar páginas existentes.

Para logro de este objetivo se implementó una interfaz (utilizando el criterio MVC) bajo el lenguaje Python, con soporte de AJAX y dHTML, que permite al usuario administrador depurar sitios y artículos indexados. De igual forma se realizó otra interfaz que permite a los usuarios, previamente autorizados, agregar la dirección del sitio para realizar la indexación de artículos asociados.

7. Evaluar el desempeño del algoritmo de búsqueda a través de la medición del tiempo de respuesta de diversas peticiones. Analizar resultados obtenidos y determinar que aspectos se podrían optimizar.

Una vez implementados los 3 módulos principales (indexación, búsqueda y mantenimiento) se procedió a realizar distintas búsquedas desde la consola del sistema operativo Linux a fin de medir los tiempos de respuesta de acuerdo a cada petición (Véase Apéndice Q), pudiendo detectar que, el tiempo de respuesta de las peticiones realizadas a la base de datos, podría ser optimizado al utilizar la herramienta PYC y al realizar corrección automática de una palabra antes de ejecutar el *query* en la base de datos en lugar de buscar directamente con la cadena con posibles errores ortográficos escrita por el usuario.

Conclusiones

Con base a los resultados obtenidos durante el desarrollo del presente trabajo de investigación y de conformidad a los objetivos del mismo se concluye que:

- Indexar en tiempo real artículos provenientes de *blogs*, previamente seleccionados, y diversos medios de comunicación en un mismo lugar permite realizar seguimiento a los eventos registrados desde distintos puntos de vista, y de esta manera, realizar una investigación con un número variado de fuentes.

- Aplicar los criterios de auto-clasificación de sitios y artículos asociados, de acuerdo a una región del país, facilita la ubicación e investigación de noticias y eventos ocurridos en un área específica de Venezuela, tanto al escribir una palabra clave o frase como al listar los últimos eventos ocurridos por región.

- El reconocimiento y auto-corrección de ciertas palabras a través del diccionario permite obtener resultados más acertados en la búsqueda sin solicitar al usuario confirmación de una palabra (dentro del diccionario) que no ha sido acentuada.

- La utilización de un panel de control para administrar los sitios y artículos indexados permite un fácil control de aquellas informaciones o eventos registrados que carecen de relevancia para el buscador pero han sido indexados debido a una clasificación errónea por parte del autor original del artículo. Esto es común cuando ciertos artículos de opinión son clasificados bajo la categoría de "noticias nacionales".

Recomendaciones

A continuación se presentan posibles mejoras y recomendaciones para el presente trabajo:

- Crear un módulo capaz de extraer noticias o eventos registrados en un sitio que no soporte RSS ó ATOM y crear un archivo XML con formato *Really Simple Syndication* para ser interpretado por el sistema.

- Incorporar una opción que permita filtrar los eventos registrados en 3 grandes categorías dentro del sistema local: política, deportes o sucesos.

- Contribuir a la difusión de información, acerca de las ventajas que ofrece el uso de tecnología RSS ó ATOM, a fin de incrementar el número de medios de comunicación que utilizan dicha tecnología en su contenido.

- Expandir el sistema de administración de contenido a los respectivos autores de los artículos indexados, de acuerdo a un sistema que maneje niveles de confianza. Cada sitio remoto y autor deberían estar previamente autorizados para realizar ciertas modificaciones dentro de los lugares asociados a sus artículos dentro del sistema local. Esta tendencia es observada en sitios web como wikipedia.org, y de forma análoga, es común en *blogs* editados por múltiples autores donde cada uno se encarga de la administración del contenido que publica.

Bibliografía

Artículos electrónicos

- [BEL96] Beltrán, Carlos (1996). Gestor de bases de datos documentales. Universidad Complutense de Madrid. [En línea]. Obtenido el 5 de Enero de 2007, en: http://www.ucm.es/info/multidoc/multidoc/revista/cuadern5/ceseda.htm
- [BUR05] Burden, Peter (2005). Javascript Features - the DOM. University of Wolverhampton. [En línea]. Obtenido el 19 de Julio de 2007, en: http://www.scit.wlv.ac.uk/~jphb/javascript/dom2.html
- [GGP04] Gyongyi, Zoltan; Garcia-Molina, Hector; Pedersen, Jan (2004). Combating Web Spam with TrustRank. Stanford University. [En línea]. Obtenido el 5 de Enero de 2007, en: http://dbpubs.stanford.edu:8090/pub/2004-17
- [HER01] Herman, Ivan (2001). Semantic Web. The World Wide Web Consortium. [En línea]. Obtenido el 5 de Enero de 2007, en: http://www.w3.org/2001/sw/
- [HMS00] Hersovici, Michael; Maarek, Yoelle; Shtalhaim, Menachem (2000). The Shark-Search Algorithm. Carnegie Mellon. School of Computer Science. [En línea]. Obtenido el 5 de Enero de 2007, en: http://www.cs.cmu.edu/~dpelleg/bin/360.html
- [HUT06] Hutcheon, Stephen (2006). Google wins rights to Aussie algorithm. The Sydney Morning Herald. [En línea]. Obtenido el 5 de Febrero de 2007, en: http://www.smh.com.au/news/breaking/aussies-formula-for-a-fortune/2006/04/10/1144521239582.html

- [LOP00] Lopez, María José (2000). Eficiencia de los algoritmos. Universitat Politècnica de Catalunya. [En línea]. Obtenido el 5 de Enero de 2007, en: http://www.lsi.upc.es/~euetit/tp/efi.pdf
- [MCI07] McIngvale, Frank (2007). All About Python and Unicode. Boodebr. [En línea]. Obtenido el 17 de Abril de 2007, en: http://boodebr.org/main/python/all-about-python-and-unicode
- [MON00] Monroy, Fran (2000). La "tropezada" historia de Internet en Venezuela. Diario "El Mundo". [versión en línea]. Obtenido el 5 de Febrero de 2007, en: http://www.elmundo.com.ve/ediciones/2000/07/25/p1-10s1.htm
- [RIG06] Rigo, Armin (2006). The Ultimate Psyco Guide. Source Forge. [En línea]. Obtenido el 19 de Julio de 2007, en: http://psyco.sourceforge.net/psycoguide/
- [SAL07] Salnikov, Andy (2007). How to convert OdcNtupleMaker timestamps into human-readable format. Stanford Linear Accelerator Center (SLAC). [En línea]. Obtenido el 25 de Abril de 2007, en: http://www.slac.stanford.edu/BFROOT/www/Computing/Online/Databases/Ambient/ONTM-timestamps.html
- [SHE05] Shekhar, Raj (2005). Rasmus' 30 second AJAX Tutorial. Lunatechian. [En línea]. Obtenido el 1 de Marzo de 2007, en: http://rajshekhar.net/blog/archives/85-Rasmus-30-second-AJAX-Tutorial.html
- [SHI06] Shinyama, Yusuke (2006). Preemptive Information Extraction using Unrestricted Relation Discovery. New York University. [En línea]. Obtenido el 19 de Julio de 2007, en: http://nlp.cs.nyu.edu/publication/papers/shinyama-naacl06.pdf

- [USE03] UserLand Software (2003). XML-RPC. [En línea]. Obtenido el 9 de Enero de 2007, en: http://www.xmlrpc.com
- [VAN00] Van Rossum, Guido (2000). Guía de aprendizaje de Python Release 2.0. The Linux Documentation Project. [En línea]. Obtenido el 5 de Abril de 2007, en: http:// es.tldp.org/Tutoriales/Python/Tutorial-Python/
- [VAN06] Van Rossum, Guido (2006). Python Tutorial Release 2.5. Python Software Foundation. [En línea]. Obtenido el 5 de Abril de 2007, en: http://docs.python.org/tut/
- [WIK05] Wikipedia, La Enciclopedia Libre (2005). Motor de búsqueda. [En línea]. Obtenido el 5 de Enero de 2007, en: http://es.wikipedia.org/wiki/Motor_de_b%C3%BAsqueda
- [WIK06] Wikipedia, La Enciclopedia Libre (2006). PageRank. [En línea]. Obtenido el 5 de Enero de 2007, en: http://es.wikipedia.org/wiki/PageRank
- [WIK07] Wikipedia, La Enciclopedia Libre (2007). XML. [En línea]. Obtenido el 5 de Enero de 2007, en: http://es.wikipedia.org/wiki/XML
- [WIN03] Winer, Dave (2003). RSS 2.0 Specification. Berkman Center for Internet & Society. [En línea]. Obtenido el 5 de Enero de 2007, en: http://cyber.law.harvard.edu/rss/rss.html

Libros

- [CON00] Conallen, Jim (2000). Building Web Applications with UML. Addison Wesley Professional. Pearson Education.
- [JBR99] Jacobson, I., Booch, G., Rumbaugh, J. (1999). The Unified Software Development Process. Addison Wesley
- [JBR00] Jacaboson, I., Booch, G., Rumbaugh J. (2000), El Proceso Unificado de Desarrollo de Software. Addison Wesley
- [KRU00] Kruchten, Philippe (2000). The Rational Unified Process: An Introduction (Second Edition). Addison Wesley.
- [RSC98] Rational Software Corporation, (1998). Rational Unified Process. Pearson Education.

Glosario

A

ATOM: formato XML análogo a RSS.

B

Base de Datos: conjunto de datos relacionados que almacenados de forma tal que luego pueden ser accedidos de manera sencilla, con la posibilidad de relacionarlos y ordenarlos basado en distintos criterios.

Blog: es una página web, con formato de bitácora personal, que posee un conjunto de anotaciones que pueden ser leídas desde el navegador.

C

Cliente: aplicación informática utilizada para acceder a los servicios que ofrece un servidor, normalmente a través de una red de telecomunicaciones.

I

Interfaz de usuario: es la parte del programa informático que permite la interacción entre el usuario, los distintos programas y el hardware.

Internet: conjunto de interconexiones formadas por varias redes de computadora.

J

Jext: Editor diseñado en java con soporte para distintos lenguajes de programación como: PHP, Python, HTML. Compatible con Linux y Windows.

M

Macromedia Dreamweaver: Editor que, no requiere conocimientos de programación, destinado a la creación y edición de páginas web a partir de una interfaz gráfica amigable que ayuda a previsualizar el contenido a medida que se va creando. También se le conoce como editor WYSIWYG (lo que ve es lo que obtiene).

MVC: Es un patrón de diseño que separa un programa en: Modelo (reglas del negocio), Vista (interfaz de usuario) y Controlador (lógica de control).

P

PHP: Lenguaje de programación orientado al desarrollo de aplicaciones web.

S

Servidor: es un programa informático que realiza algunas tareas en beneficio de otras aplicaciones llamadas clientes.

Sindicación: se refiere a la redifusión de contenido.

Software: son todos los componentes intangibles de una computadora, es decir, un conjunto de programas y procedimientos necesarios para hacer posible la realización de una tarea específica.

U

UML (Unified Modeling Language): es un lenguaje para la especificación, visualización, construcción y documentación de cada una de las partes que comprende el desarrollo de software.

Apéndice A – Comparativa de búsquedas

Comparación de búsqueda global para palabra "3D"

Para una búsqueda realizada el día 10 de Diciembre de 2006 se obtuvieron los resultados presentados en la tabla A.1

Motor de búsqueda global	1er resultado obtenido
Google.com	Portal sobre el mundo del **3D** y la animación, donde se incluyen tutoriales, galerías, foros, artículos y las últimas noticias sobre el **3d**.
Yahoo.com	Featuring samples of 3D artists from around the world...

Tabla A.1. Comparación entre 2 buscadores globales
Fuente: elaborado por el autor

Comparación de búsqueda en Venezuela para palabra "3D"

Para una petición realizada el día 10 de Diciembre de 2006 en motores de búsqueda orientados a mostrar contenido relacionado solo a Venezuela se obtuvieron los resultados presentados en la tabla A.2

Motor de búsqueda en Venezuela	1er resultado obtenido
Google.com (Marcando casilla de búsqueda "Venezuela")	**3D** Warehouse Results, Results 1 - 12 of about 32 (0.0 seconds) ... Modelo **3D** desarrollado por... Download to Google SketchUp ...
Auyantepui.com	Esta búsqueda arroja muchos resultados, sea más específico.

Tabla A.2. Comparación entre 2 buscadores en Venezuela
Fuente: elaborado por el autor

Resultados de 2da búsqueda en Venezuela para palabra "3D"

Para una segunda petición realizada el día 10 de Diciembre de 2006 en motores de búsqueda orientados a mostrar contenido relacionado solo a Venezuela se obtuvieron los resultados presentados en la tabla A.3

Motor de búsqueda en Venezuela	2do resultado obtenido
Google.com (Marcando casilla de búsqueda "Venezuela")	El juego final del **3D**. Publicado 29/11/2006 12:58 - Política ... Pobre mi Venezuela, en este **3D**, tener que escoger entre Chavez y Rosales, Yo, ...
Auyantepui.com	Esta búsqueda arroja muchos resultados, sea más específico.

Tabla A.3. Comparación entre 2 buscadores en Venezuela
Fuente: elaborado por el autor

Apéndice B – Evolución de los buscadores

Año	Motor de Búsqueda	Evento
1993	**Aliweb**	Lanzamiento
1994	**WebCrawler**	Lanzamiento
	Infoseek	Lanzamiento
	Lycos	Lanzamiento
1995	**AltaVista**	Lanzamiento (parte del **DEC**)
	Excite	Lanzamiento
1996	**Dogpile**	Lanzamiento
	Inktomi	Fundación
	Ask Jeeves	Fundación
1997	**Northern Light**	Lanzamiento
1998	**Google**	Lanzamiento
1999	**AlltheWeb**	Lanzamiento
	Baidu	Fundación
2000	**Singingfish**	Lanzamiento
	Teoma	Fundación
	Vivisimo	Fundación
2003	**Info.com**	Lanzamiento
2004	**Yahoo! Search**	Lanzamiento final
	MSN Search	Lanzamiento beta
2005	**MSN Search**	Lanzamiento final
2006	**Quaero**	Fundación
	Ask.com	Lanzamiento
	Windows Live Search	Lanzamiento
	Quintura	Lanzamiento beta

Tabla B.1. Evolución de los buscadores
Fuente: http://en.wikipedia.org/wiki/Web_search_engine

Apéndice C – Características de XML

C.1 Fortalezas de XML

El metalenguaje XML presenta un conjunto de fortalezas que, de acuerdo a [WIK07], son las siguientes:

- *Es extensible, lo que quiere decir que una vez diseñado un lenguaje y puesto en producción, igual es posible extenderlo con la adición de nuevas etiquetas de manera de que los antiguos consumidores de la vieja versión todavía puedan entender el nuevo formato.*

- *El analizador es un componente estándar, no es necesario crear un analizador específico para cada lenguaje. Esto posibilita el empleo de uno de los tantos disponibles. De esta manera se evitan bugs y se acelera el desarrollo de la aplicación.*

- *Si un tercero decide usar un documento creado en XML, es sencillo entender su estructura y procesarlo. Mejora la compatibilidad entre aplicaciones.*

C.2 Estructura de comunicación vía XML

```
POST /RPC2 HTTP/1.0
User-Agent: MiCliente/0.1.1
Host: fernando.coolfata.com
Content-Type: text/xml
Content-length: 194

<?xml version="1.0"?>

<methodCall>

   <methodName>nombredelmetodo</methodName>
   <params>

      <param>

         <value><i1>1904</i1></value>

         </param>

      </params>

   </methodCall>
```

Figura C.1. Ejemplo de una petición XML--RPC
Fuente: elaborado por el autor

```
HTTP/1.1 200 OK
Connection: close
Content-Length: 183
Content-Type: text/xml
Date: Thu, 19 Apr 2007 19:04:07 GMT
Server: Apache/1.3.37

<?xml version="1.0"?>
<methodResponse>
   <params>
      <param>
         <value><string>Recibido
correctamente</string></value>
         </param>
      </params>

   </methodResponse>
```

Figura C.2. Ejemplo de una respuesta XML--RPC
Fuente: elaborado por el autor

Apéndice D – Contenido XML en el navegador

En la Figura D.1 se puede observar la vista desde el navegador Internet Explorer 7.0 del contenido en formato XML del sitio web *con-cafe.com* correspondiente a los eventos registrados el día domingo, 12 de agosto de 2007.

La mayoría de las versiones, recientes, de los navegadores, ofrecen la posibilidad de leer contenido XML de manera sencilla para aquellos sitios que tengan soporte para ello. Esta forma de leer el contenido de una página ofrece a los usuarios la posibilidad de visualizar las noticias de un sitio sin tener que cargar todas las imágenes y secciones del sitio.

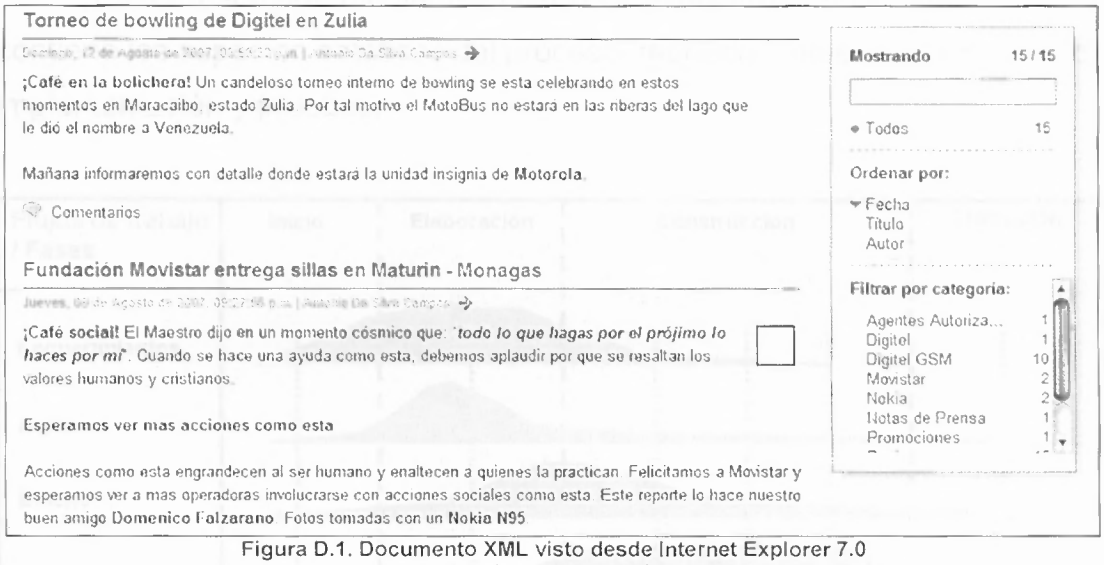

Figura D.1. Documento XML visto desde Internet Explorer 7.0
Fuente: elaborado por el autor

Apéndice E – El Proceso Unificado Racional

E.1 Lenguaje Unificado de Modelado (UML)

Es un lenguaje de propósito general, utilizado por la metodología de desarrollo RUP, que aparece como necesidad de establecer un estándar para el modelado orientado a objetos. Permite visualizar de manera sencilla el sistema y sus componentes, y de igual forma, está dirigido a unificar la experiencia anterior sobre técnicas de modelado e incorporar las mejores prácticas utilizadas en la actualidad.

E.2 Flujos de trabajo del proceso unificado racional (RUP)

Es representado mediante el eje vertical mostrado en la Figura E.1 y contiene los aspectos estáticos del proceso: requerimientos, análisis, diseño, implementación y pruebas.

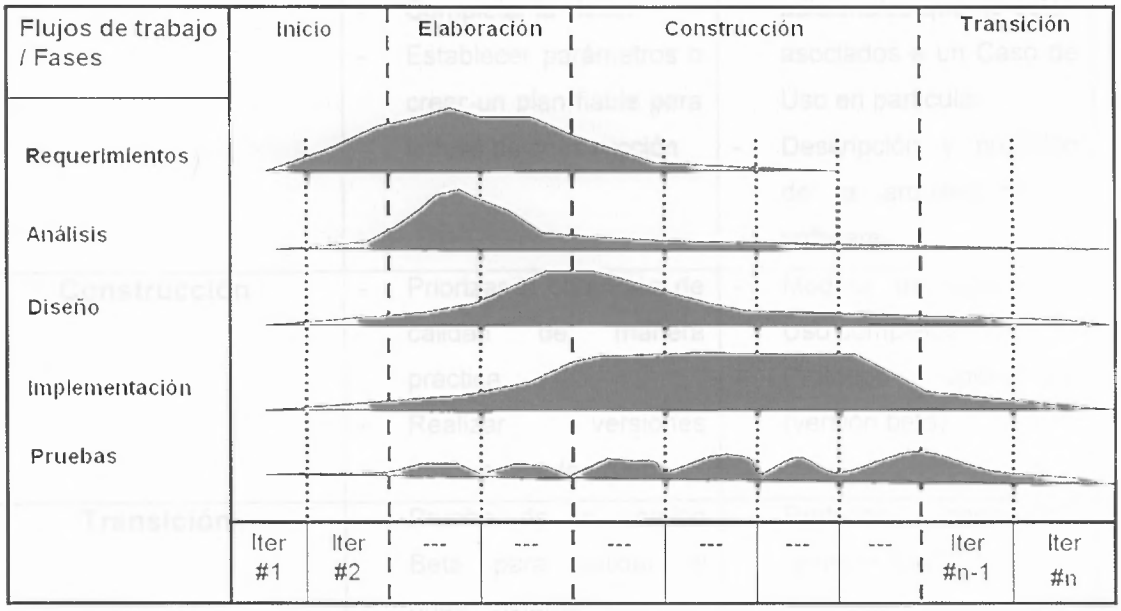

Figura E.1. Fases y flujos de trabajo de RUP
Fuente: [JBR00]

E.3 Fases del proceso unificado racional (RUP)

A continuación se presenta un cuadro resumen que contiene objetivos y resultados que, de acuerdo a [KRU00], se deben obtener de cada una de las fases representadas en el eje horizontal de la figura E.1:

Fase	Objetivos	Resultados
Inicio	- Fijar ámbito y límites del proyecto. - Encontrar los casos de uso críticos del sistema y establecer los escenarios básicos que definen sus características o funcionalidades.	- Documento visión: contiene una visión general de los requerimientos del proyecto, características clave y restricciones principales. - Modelo inicial de los casos de uso (10 a 20% completos)
Elaboración	- Definir y validar la arquitectura a utilizar - Completar la visión - Establecer parámetros o crear un plan fiable para la fase de construcción	- Modelo de casos de uso completo al menos hasta el 80% - Obtener requisitos adicionales que no estén asociados a un Caso de Uso en particular. - Descripción y prototipo de la arquitectura de software.
Construcción	- Priorizar la obtención de calidad de manera práctica. - Realizar versiones funcionales de prueba	- Modelo de Casos de Uso completos - Prototipo operacional (versión beta)
Transición	- Prueba de la versión Beta para validar el nuevo sistema.	- Prototipo operacional (versión final)

Tabla E.1. Tabla resumen de fases en RUP
Fuente: elaborado por el autor

E.4 Duración de cada fase del proyecto

A continuación se presenta el detalle de la distribución de semanas, fases y flujos de trabajo de la instancia de RUP durante el desarrollo del proyecto:

Fase	Duración total	# iteraciones	Etapa de mayor énfasis	
			Nombre	**Iteración involucrada**
Inicio	8 semanas	2	- Captura de requerimientos	1
			- Casos de Uso y requerimientos no funcionales	1 y 2
Elaboración	8 semanas	2	- Análisis de requerimientos	1
			- Diseño de componentes	1 y 2
Construcción	12 semanas	3	- Implementación del módulo de indexación	1
			- Implementación del módulo de indexación	2
			- Implementación del módulo de mantenimiento	3
Transición	4 semanas	1	- Prueba de la versión final	1
			- Ayuda en linea	1

Tabla E.2. Duración de las fases del proyecto
Fuente: elaborado por el autor

Apéndice F – Casos de Uso del Sistema

F.1 Diagrama general de casos de uso del sistema

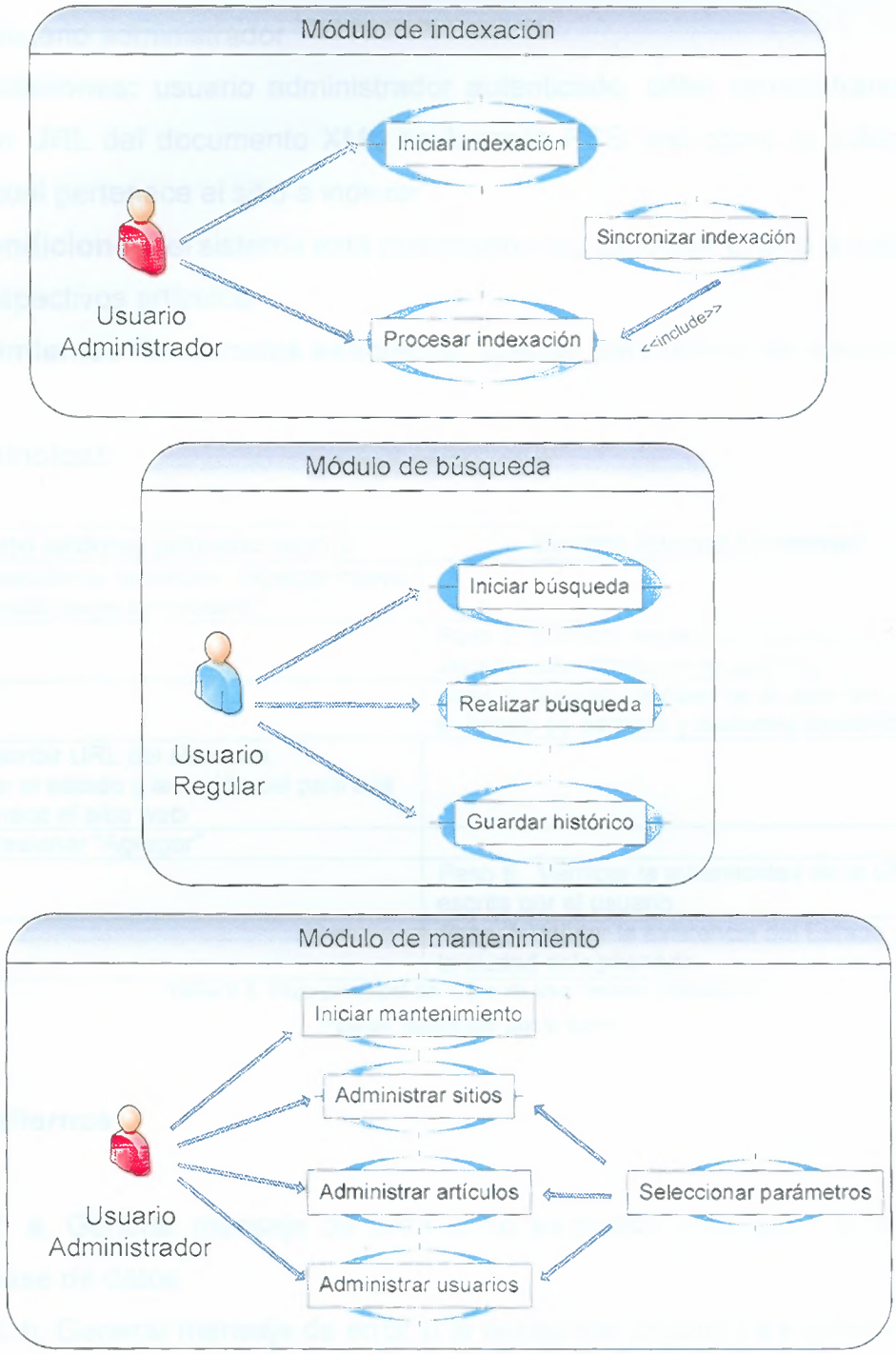

Figura F.1. Diagrama de Caso de Uso General del sistema
Fuente: elaborado por el autor

F.2 Casos de uso del módulo de indexación

Caso de Uso "*Iniciar indexación*"

Actor: usuario administrador

Pre-condiciones: usuario administrador autenticado, debe suministrarse la dirección URL del documento XML en formato RSS, así como el estado y país al cual pertenece el sitio a indexar.

Post-condiciones: el sistema está preparado para procesar el sitio a indexar y sus respectivos artículos.

Requerimientos funcionales asociados: solicitar parámetros de ejecución.

Flujo Principal:

Evento externo (usuario adm.)	Evento interno (sistema)
Paso 1. Seleccionar la opción: "agregar nuevo sitio" (inicializa carga del módulo).	
	Paso 2. Verificar validez de la sesión del usuario autenticado en el sistema.
	Paso 3. Buscar y extraer de la base de datos el listado de estados y ciudades disponibles.
Paso 4. Escribir URL del sitio web, seleccionar el estado y la región del país a la cual pertenece el sitio web.	
Paso 5. Presionar "Agregar".	
	Paso 6. Verificar la autenticidad de la URL escrita por el usuario.
	Paso 7. Validar la existencia del Estado y de la ciudad seleccionada.

Tabla F.1. Flujo principal del Caso de Uso "iniciar indexación"

Fuente: elaborado por el autor

Flujos alternos:

2. a. Generar mensaje de error si no se puede establecer conexión con la base de datos.

2. b. Generar mensaje de error si la sesión del usuario ha expirado.

3. a. Generar mensaje de error si no se puede establecer conexión con la base de datos.

3. b. Generar mensaje de error si no se encuentran ciudades o estados registrados en la base de datos.

4. a. Generar mensaje de error si no se han escrito los datos correctamente.

6. a. Generar mensaje de error si no se puede establecer una conexión al sitio que se desea indexar.

6. b. Generar mensaje de error si no se puede establecer conexión con la base de datos.

7. a. Generar mensaje de error si el código de ciudad o país es inválido.

Caso de Uso "*Procesar indexación*"

Actor: usuario administrador

Pre-condiciones: usuario administrador autenticado y haber presionado "agregar", y suministrar código de ciudad, de estado y dirección URL válida.

Post-condiciones: el sistema ha ingresado y actualizado en la base de datos el sitio web y sus artículos asociados, de igual forma ha asignado un estado y región del país al cual pertenecerán por defecto todos los futuros artículos.

Requerimientos funcionales asociados: solicitar parámetros de ejecución y recibir y almacenar en la base de datos información relacionada al sitio a través de RSS.

Flujo Principal:

Evento externo (usuario adm.)	Evento interno (sistema)
	Paso 1. Validar documento RSS.
	Paso 2. Agregar o actualizar en la base de datos la información del sitio indexado.
	Paso 3. Agregar los artículos asociados al sitio indexado

Tabla F.2. Flujo principal del Caso de Uso "procesar indexación"

Fuente: elaborado por el autor

Flujos alternos:

1. a. Generar mensaje de error si no se puede establecer conexión con la base de datos.

1. b. Generar mensaje de error si no es posible validar el documento RSS.

2. a. Generar mensaje de error si no se puede establecer conexión con la base de datos.

3. a. Generar mensaje de error si no se puede establecer una conexión con la URL del artículo asociado.

Caso de Uso "*Sincronizar indexación*"

Actor: usuario regular

Pre-condiciones: solicitud enviada por el usuario o el sitio remoto.

Post-condiciones: el sistema está preparado para procesar la indexación.

Requerimientos funcionales asociados: solicitar parámetros de ejecución.

Flujo Principal:

Evento externo (usuario regular.)	Evento interno (sistema)
Paso 1. Usuario realiza petición	
	Paso 2. Identificar tipo de petición: por demanda del servidor local. Se realiza mediante AJAX a petición del usuario regular. Se Marca actualización del caché como diferida
	Paso 3. Verificar que el sitio se encuentra en el listado de sitios previamente autorizados. No se permite la indexación de sitios nuevos mediante este método debido a que el sitio debe ser autorizado previamente a través del caso de uso "iniciar indexación" iniciado por el usuario administrador.

Tabla F.3 Flujo principal del Caso de Uso "sincronizar indexación"

Fuente: elaborado por el autor

Flujos alternos:

2. a. Se realiza petición por demanda del sitio remoto directamente a través del servidor XML—RPC.

2. a. 1 Se verifica el espacio disponible en el servidor de caché actual.

2. a. 2 Se selecciona servidor para almacenar copias de caché.

2. a. 3 Marcar actualización del caché como inmediata.

3. a. Grabar en un registro con un mensaje de error si no se puede establecer conexión con la base de datos.

3. b. Generar mensaje de error si no se puede establecer una conexión con el sitio remoto.

3. c. Grabar en un registro con un mensaje de error si no es posible validar el documento RSS.

En la Figura F.2 se muestra el diagrama de casos de uso general para el módulo de indexación.

Figura F.2. Diagrama de Caso de Uso General del módulo de indexación
Fuente: elaborado por el autor

F.3 Casos de uso del módulo de búsqueda

Caso de Uso *"Iniciar búsqueda"*

Actor: usuario regular

Pre-condiciones: usuario regular debe haber ingresado a la página principal del sitio, debe suministrar una palabra clave o frase, así como el estado y país en el cual se desea realizar la búsqueda.

Post-condiciones: el sistema está preparado para realizar la búsqueda.

Requerimientos funcionales asociados: solicitar parámetros de ejecución.

Flujo Principal:

Evento externo (usuario reg.)	Evento interno (sistema)
Paso 1. Seleccionar la sección de "búsqueda"	
	Paso 2. Buscar y extraer de la base de datos el listado de estados y ciudades disponibles.
Paso 3. Escribir palabra clave o frase a buscar y seleccionar el estado y la región del país a la cual pertenece el sitio web. Si la búsqueda es avanzada seleccionar rango de fechas.	
Paso 4. Presionar la opción: "buscar" o "búsqueda avanzada".	
	Paso 5. Validar la existencia del país y del estado seleccionado.
	Paso 6. Depurar cadena escrita por el usuario.

Tabla F.4. Flujo principal del Caso de Uso "iniciar búsqueda"

Fuente: elaborado por el autor

Flujos alternos:

2. a. Generar mensaje de error si no se puede establecer conexión con la base de datos.

2. b. Generar mensaje de error si no se encuentran ciudades o estados registrados en la base de datos.

4. a. Generar mensaje de error si no se han escrito los datos correctamente o faltan datos por seleccionar.

5. a. Generar mensaje de error si no se puede establecer conexión con la base de datos.

5. b. Generar mensaje de error si el código de ciudad o país es inválido

6. a. Si es detectado el uso de comillas dobles se tomará la búsqueda como coincidencia exacta de la frase o palabra a buscar.

Caso de Uso "*Realizar búsqueda*"

Actor: usuario regular

Pre-condiciones: haber presionado "buscar" o "búsqueda avanzada", suministrar código de ciudad, de estado y rango de fechas en caso de haber seleccionado búsqueda avanzada.

Post-condiciones: el sistema ha realizado la búsqueda en la base de datos y ha mostrado una lista de los sitios web y sus artículos asociados.

Requerimientos funcionales asociados: solicitar parámetros de ejecución y Recibir e interpretar data asociada a la solicitud.

Flujo Principal:

Evento externo (usuario reg.)	Evento interno (sistema)
	Paso 1. Verificar precisión de la búsqueda: exacta o regular.
	Paso 2. Extraer de la base de datos los artículos que guarden relación con la palabra clave o frase escrita por el usuario. (Aplicación del algoritmo de búsqueda)

Tabla F.5. Flujo principal del Caso de Uso "realizar búsqueda"

Fuente: elaborado por el autor

Flujos alternos:

2. a. Generar mensaje de error si no se puede establecer conexión con la base de datos.

2. b. Generar mensaje de error si no se encuentran artículos relacionados.

Caso de Uso "*Guardar histórico*"

Actor: usuario regular

Pre-condiciones: haber realizado la búsqueda y poseer el código de ciudad, de estado y rango de fechas en caso de haber seleccionado búsqueda avanzada.

Post-condiciones: el sistema guardado en un histórico la búsqueda realizada por el usuario para analizar en el futuro y mejorar las búsquedas de la aplicación.

Requerimientos funcionales asociados: almacenar parámetros de búsqueda a histórico de la base de datos, solicitar parámetros de ejecución y recibir e interpretar data asociada a la solicitud.

Flujo Principal:

Evento externo (usuario reg.)	Evento interno (sistema)
	Paso 1. Verificar número máximo permitido de registros históricos previos
	Paso 2. Extraer de la base de datos los artículos que guarden relación con la palabra clave o frase escrita por el usuario y almacenarlo para llevar un registro de búsquedas realizadas.

Tabla F.6. Flujo principal del Caso de Uso "guardar histórico"

Fuente: elaborado por el autor

Flujos alternos:

1. a. Generar mensaje de error si no se puede establecer conexión con la base de datos.

1. b. Limpiar histórico si supera el numero máximo de registros históricos previos permitidos.

2. a. Generar mensaje de error si no se puede establecer conexión con la base de datos.

A continuación (ver Figura F.3.) se muestra el diagrama de caso de uso general para el módulo de búsqueda.

Diagrama general (módulo de búsqueda):

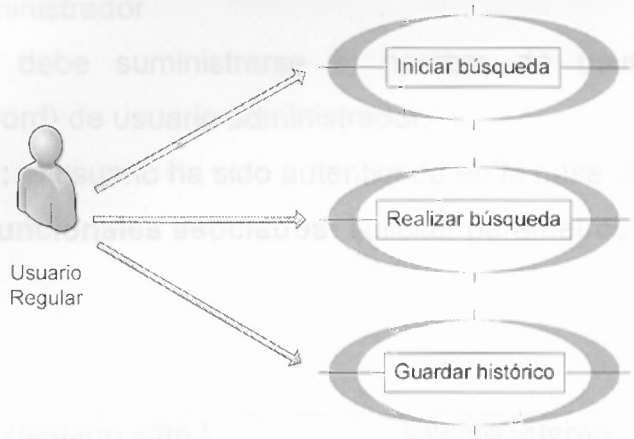

Figura F.3. Diagrama de Caso de Uso General del módulo de búsqueda
Fuente: elaborado por el autor

F.4 Casos de uso del módulo de mantenimiento

Caso de Uso "*Iniciar Mantenimiento*"

Actor: usuario administrador

Pre-condiciones: debe suministrarse el nombre de usuario (login) y contraseña (password) de usuario administrador.

Post-condiciones: el usuario ha sido autenticado en la base de datos.

Requerimientos funcionales asociados: solicitar parámetros de ejecución.

Flujo Principal:

Evento externo (usuario adm.)	Evento interno (sistema)
Paso 1. Apretar el botón: "Acceder".	
	Paso 2. Verificar la sintaxis de los datos escritos por el usuario.
	Paso 3. Verificar el usuario en la base de datos.

Tabla F.7. Flujo principal del Caso de Uso "iniciar mantenimiento"

Fuente: elaborado por el autor

Flujos alternos:

1. a. Generar mensaje de error si no se han escrito los datos correctamente o se han dejado las casillas en blanco.

2. a. Generar mensaje de error si el *login* o *password* contienen caracteres inválidos o código malicioso.

3. a. Generar mensaje de error si no se puede establecer conexión con la base de datos.

3. b. Generar mensaje de error si el nombre de usuario o contraseña no son correctos.

Caso de Uso "*Seleccionar tipo de parámetro*"

Actor: usuario administrador

Pre-condiciones: sesión de usuario administrador iniciada correctamente.

Post-condiciones: el sistema está preparado para administrar sitios, artículos o usuarios.

Requerimientos funcionales asociados: solicitar parámetros de ejecución, recibir y procesar data asociada a la solicitud.

Flujo Principal:

Evento externo (usuario adm.)	Evento interno (sistema)
Paso 1. Seleccionar opción a administrar: sitios, artículos o usuarios.	
	Paso 2. Extraer de la base de datos los datos correspondientes a la solicitud del usuario.

Tabla F.8. Flujo principal del Caso de Uso "seleccionar tipo de parámetro"

Fuente: elaborado por el autor

Flujos alternos:

2. a. Generar mensaje de error si no se puede establecer conexión con la base de datos.

Caso de Uso "*Administrar sitio*"

Actor: usuario administrador

Pre-condiciones: el usuario ha seleccionado la opción de administrar sitio.

Post-condiciones: el sistema está preparado para editar un sitio indexado.

Requerimientos funcionales asociados: solicitar parámetros de ejecución, recibir y procesar data asociada a la solicitud.

Flujo Principal:

Evento externo (usuario adm.)	Evento interno (sistema)
Paso 1. Seleccionar acción a realizar: bloquear ó editar artículos asociados a sitios seleccionados.	
	Paso 2. Realizar los cambios en la base de datos correspondientes a la solicitud del usuario.

Tabla F.9. Flujo principal del Caso de Uso "administrar sitio"

Fuente: elaborado por el autor

Flujos alternos:

2. a. Generar mensaje de error si no se puede establecer conexión con la base de datos.

2. b. Cargar lista de artículos asociados al sitio si el usuario selecciona la opción de editar artículos seleccionados.

Caso de Uso "*Administrar artículos*"

Actor: usuario administrador

Pre-condiciones: el usuario ha seleccionado la opción de administrar artículos.

Post-condiciones: el sistema está preparado para administrar otro artículo asociado a un sitio indexado.

Requerimientos funcionales asociados: solicitar parámetros de ejecución, recibir y procesar data asociada a la solicitud.

Flujo Principal:

Evento externo (usuario adm.)	Evento interno (sistema)
Paso 1. Seleccionar acción a realizar: bloquear ó asociar un lugar al artículo seleccionado.	
	Paso 2. Realizar los cambios en la base de datos correspondientes a la solicitud del usuario.

Tabla F.10. Flujo principal del Caso de Uso "administrar artículos"

Fuente: elaborado por el autor

Flujos alternos:

2. a. Generar mensaje de error si no se puede establecer conexión con la base de datos.

Caso de Uso "*Administrar usuarios*"

Actor: usuario administrador

Pre-condiciones: el usuario ha seleccionado la opción de administrar usuarios.

Post-condiciones: el sistema está preparado para administrar otro usuario.

Requerimientos funcionales asociados: solicitar parámetros de ejecución, recibir y procesar data asociada a la solicitud.

Flujo Principal:

Evento externo (usuario adm.)	Evento interno (sistema)
Paso 1. Seleccionar acción a realizar: bloquear o administrar atributos asociados (contraseña, creación de nuevos usuarios).	
	Paso 2. Realizar los cambios en la base de datos correspondientes a la solicitud del usuario.

Tabla F.11. Flujo principal del Caso de Uso "administrar usuarios"

Fuente: elaborado por el autor

Flujos alternos:

2. a. Generar mensaje de error si no se puede establecer conexión con la base de datos.

Diagrama general (módulo de mantenimiento):

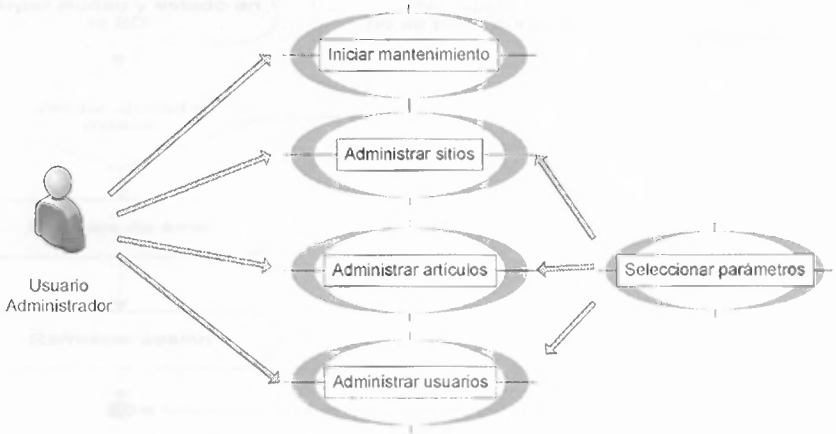

Figura F.4. Diagrama de Caso de Uso General del módulo de mantenimiento
Fuente: elaborado por el autor

Apéndice G – Diagramas de actividades

G.1 Diagrama de actividades del módulo de indexación

G.1.1 Caso de Uso "iniciar indexación"

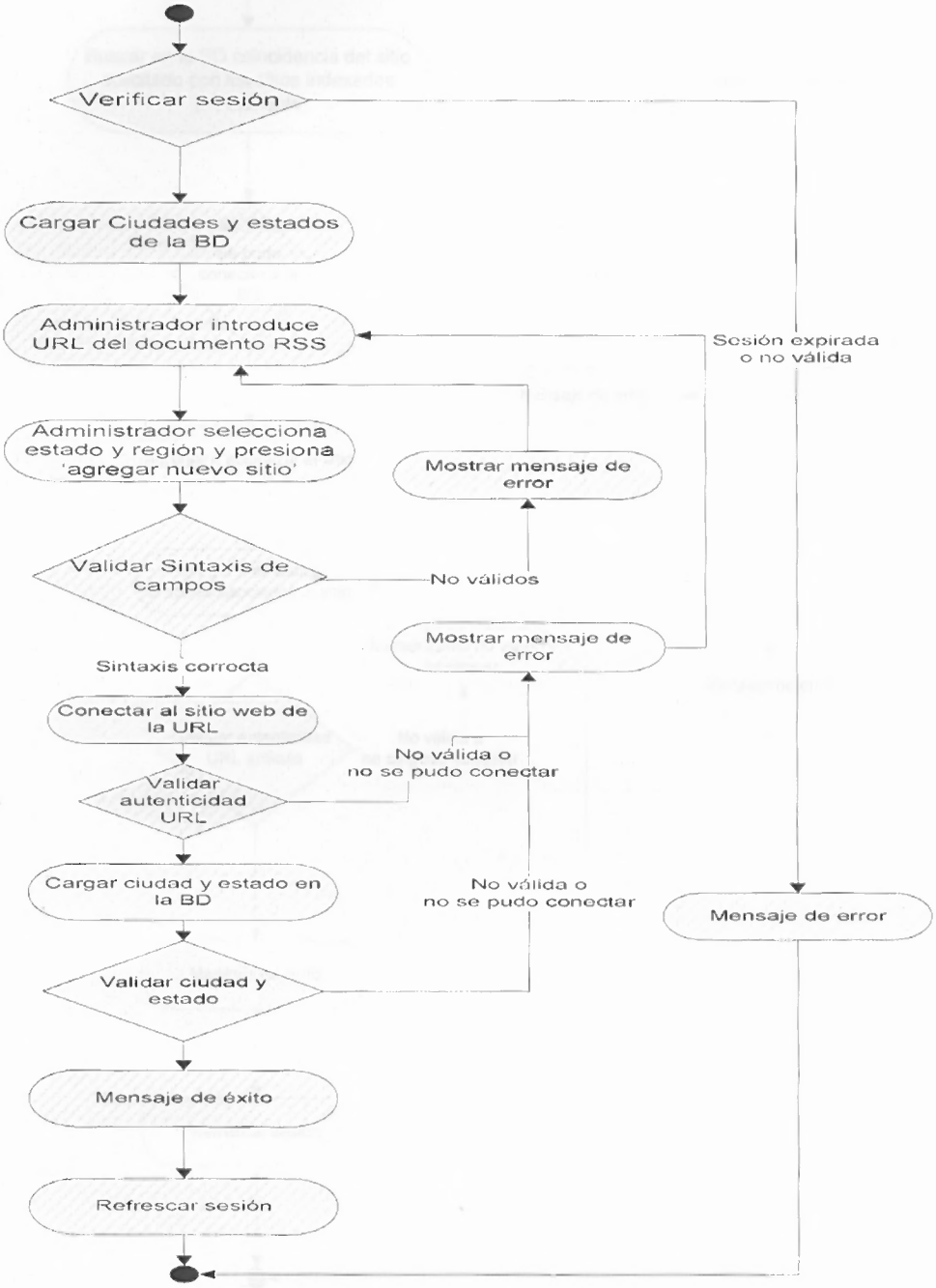

Figura G.1. Diagrama de actividades "iniciar indexación"
Fuente: elaborado por el autor

G.1.2 Caso de Uso "procesar indexación"

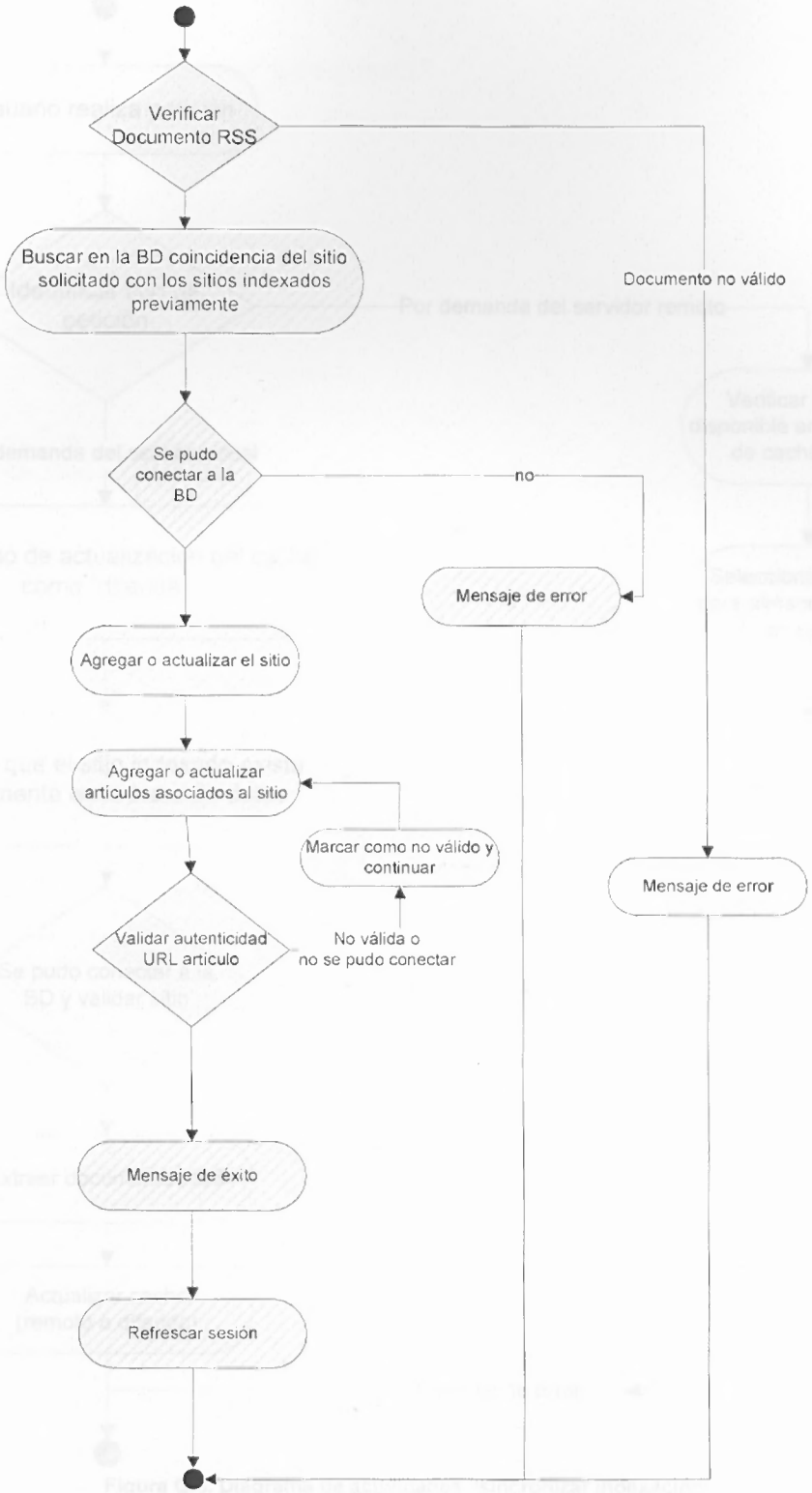

Figura G.2. Diagrama de actividades "procesar indexación"
Fuente: elaborado por el autor

G.1.3 Caso de Uso "sincronizar indexación"

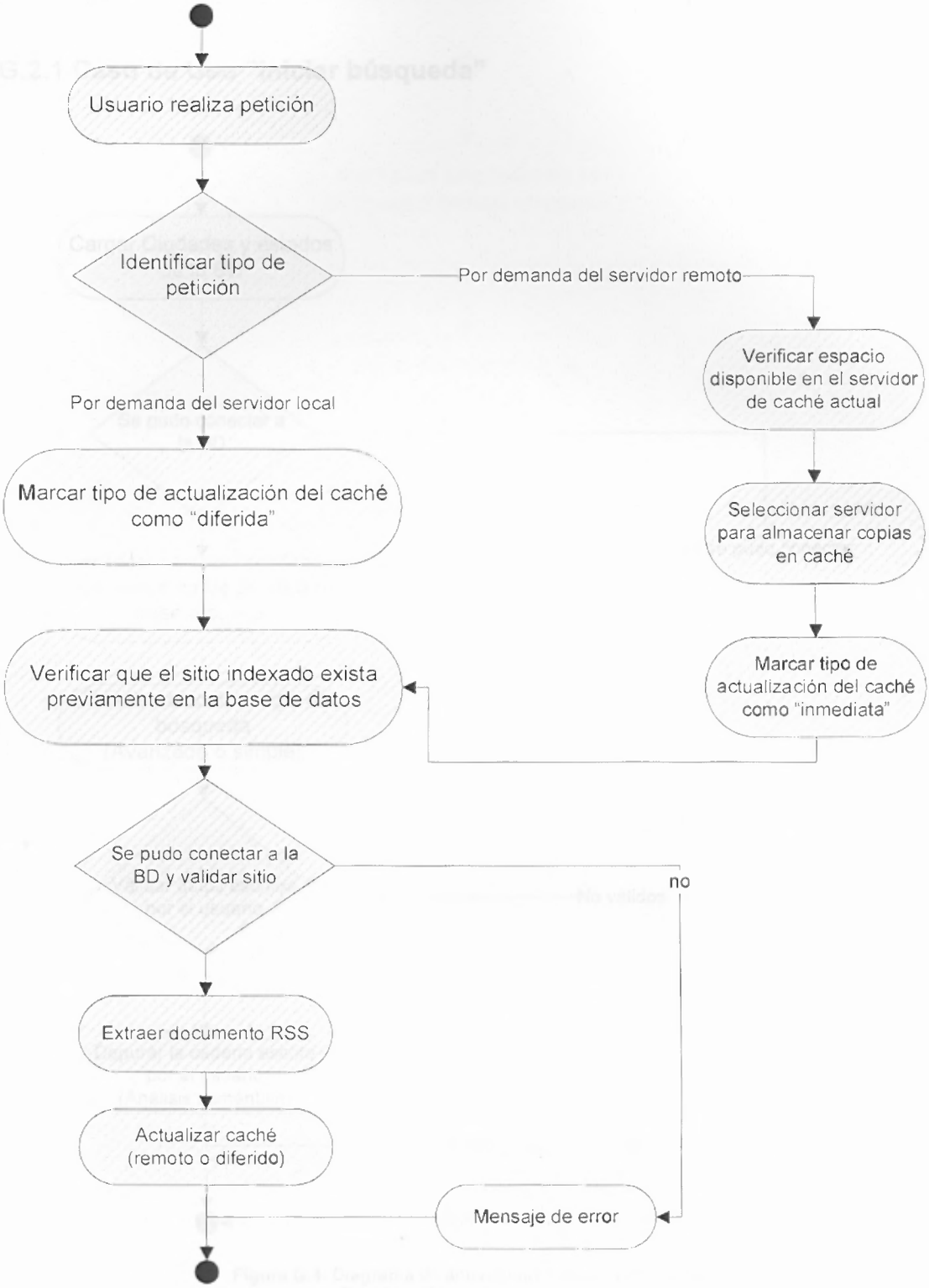

Figura G.3. Diagrama de actividades "sincronizar indexación"
Fuente: elaborado por el autor

G.2 Diagrama de actividades del módulo de búsqueda

G.2.1 Caso de Uso "iniciar búsqueda"

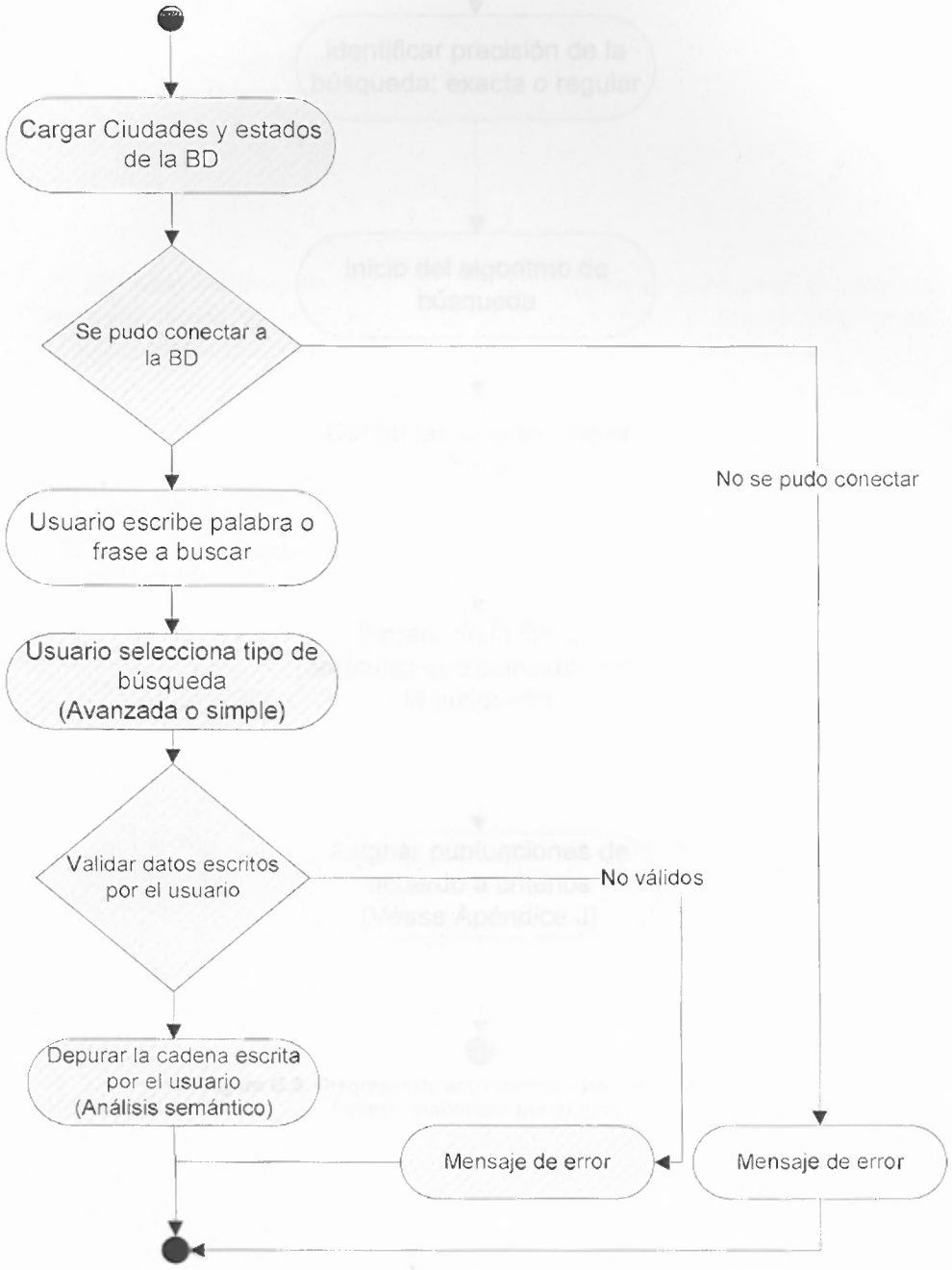

Figura G.4. Diagrama de actividades "iniciar búsqueda"
Fuente: elaborado por el autor

G.2.2 Caso de Uso "realizar búsqueda"

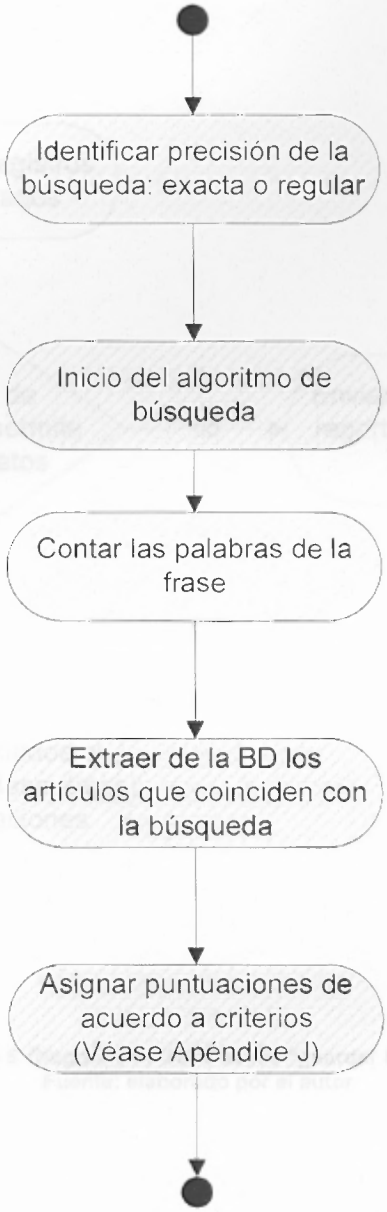

Figura G.5. Diagrama de actividades "realizar búsqueda"
Fuente: elaborado por el autor

G.2.3 Caso de Uso "guardar histórico"

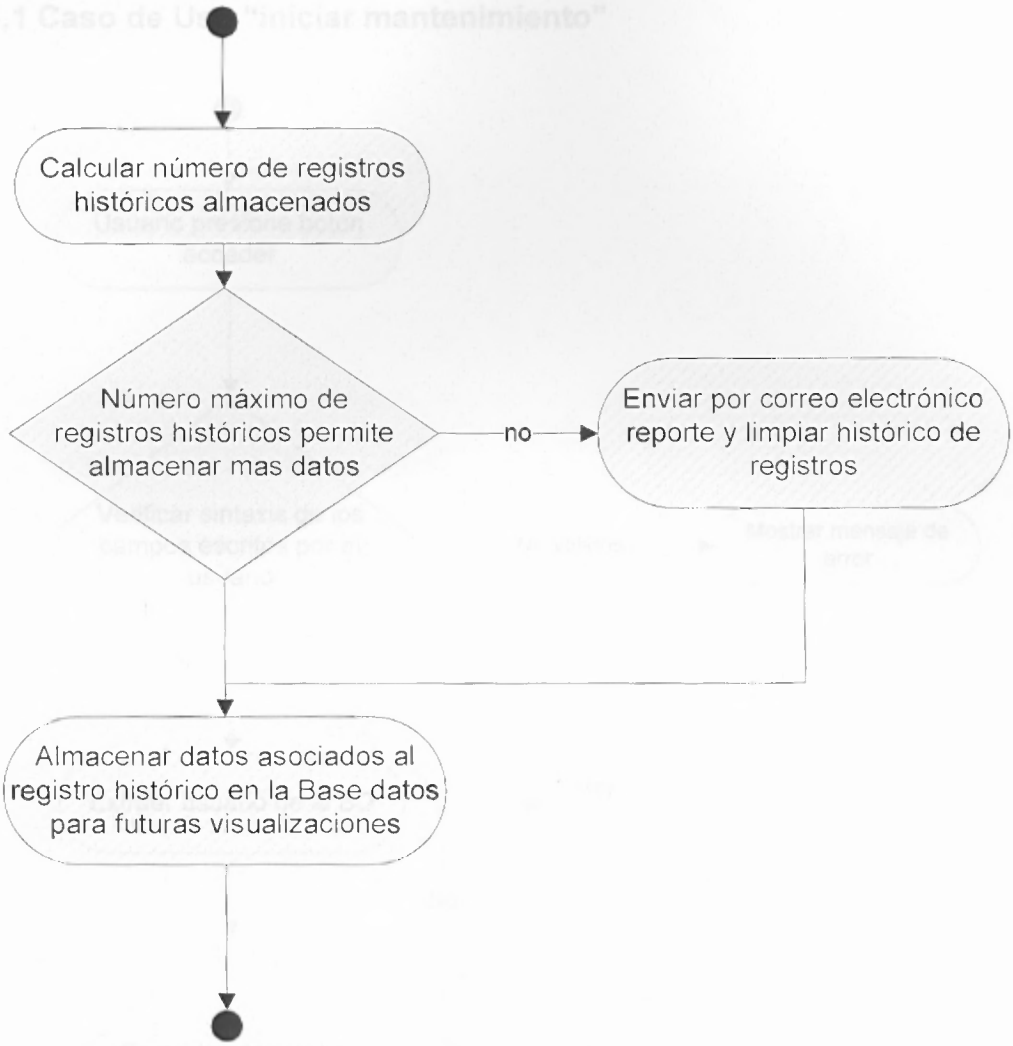

Figura G.6. Diagrama de actividades "guardar histórico"
Fuente: elaborado por el autor

G.3 Diagrama de actividades del módulo de mantenimiento

G.3.1 Caso de Uso "iniciar mantenimiento"

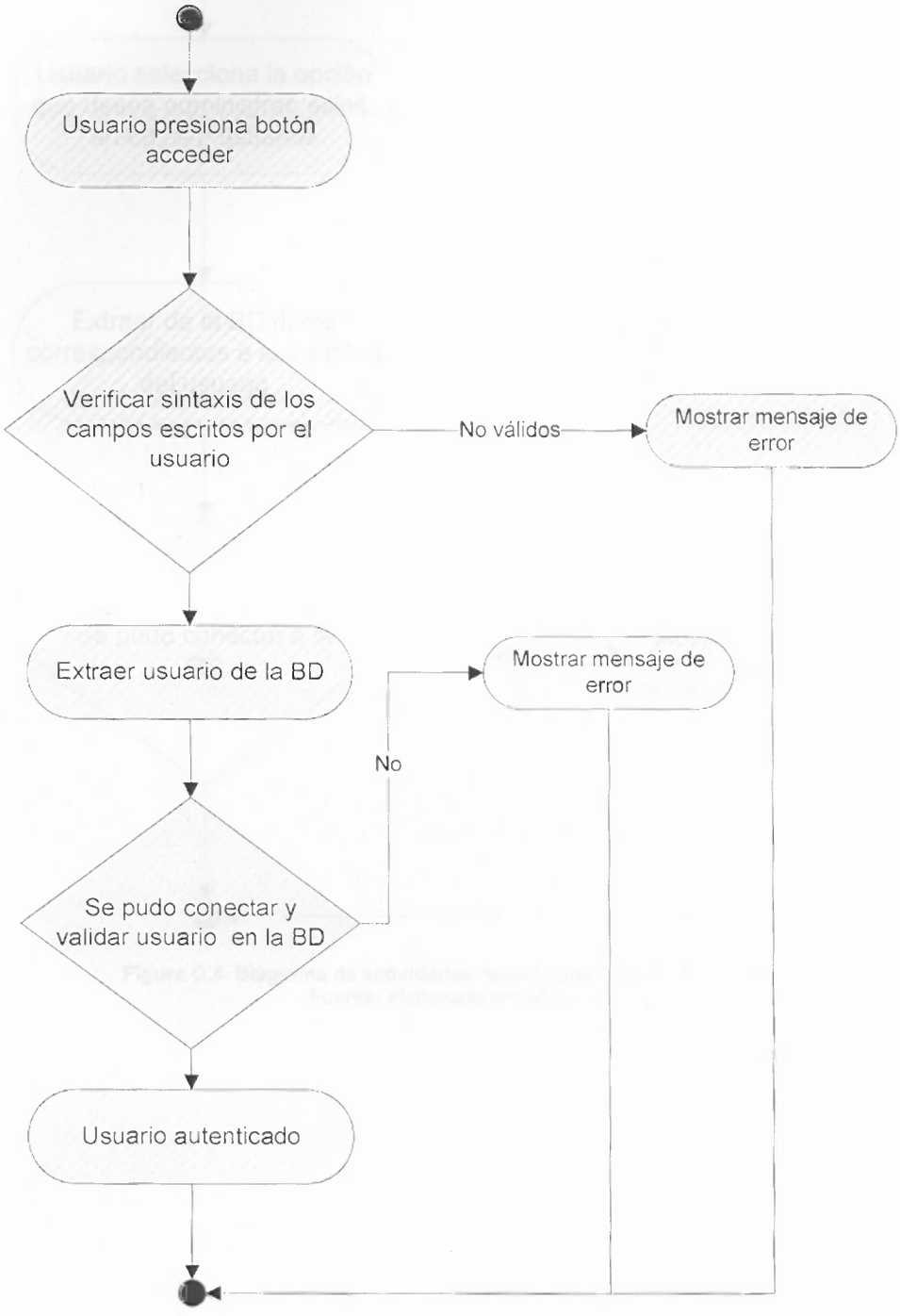

Figura G.7. Diagrama de actividades "iniciar mantenimiento"
Fuente: elaborado por el autor

G.3.2 Caso de Uso "seleccionar tipo de parámetro"

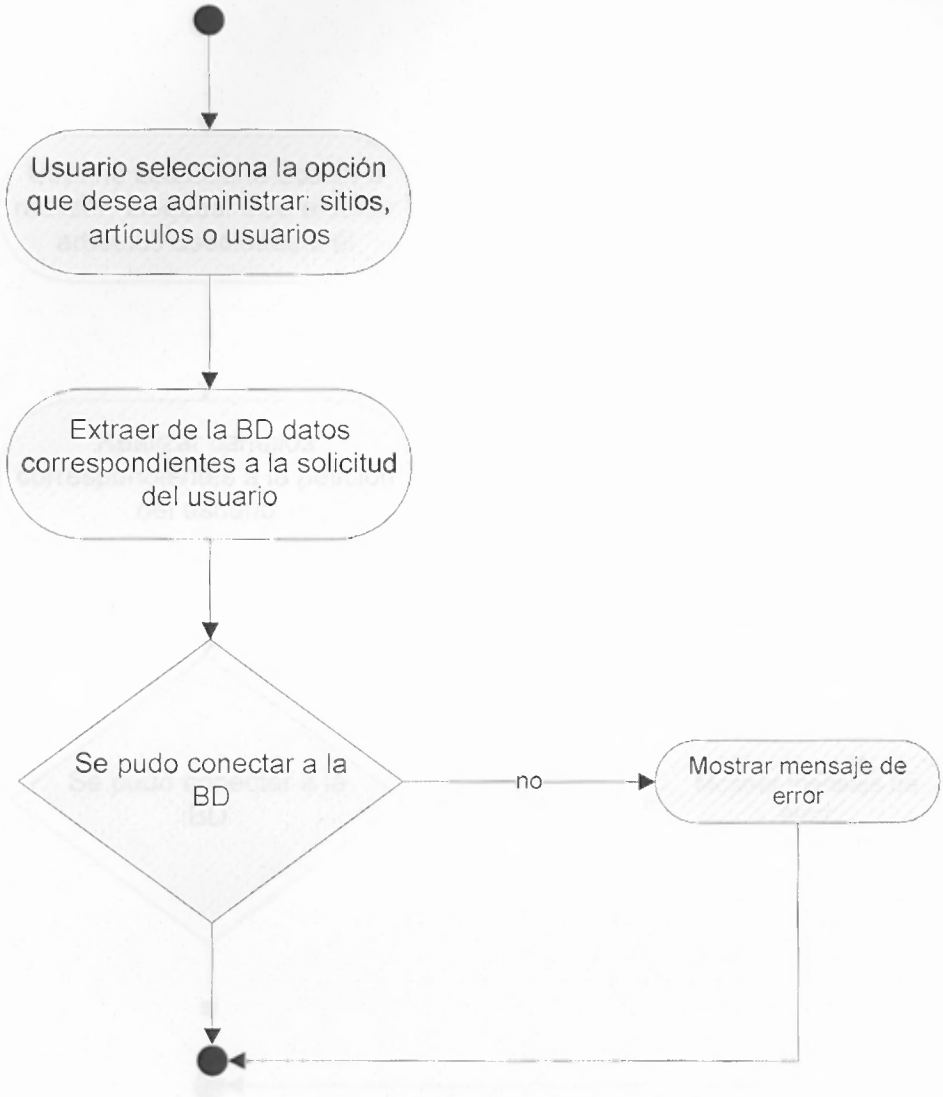

Figura G.8. Diagrama de actividades "seleccionar tipo de parámetro"
Fuente: elaborado por el autor

G.3.3 Caso de Uso "administrar sitio"

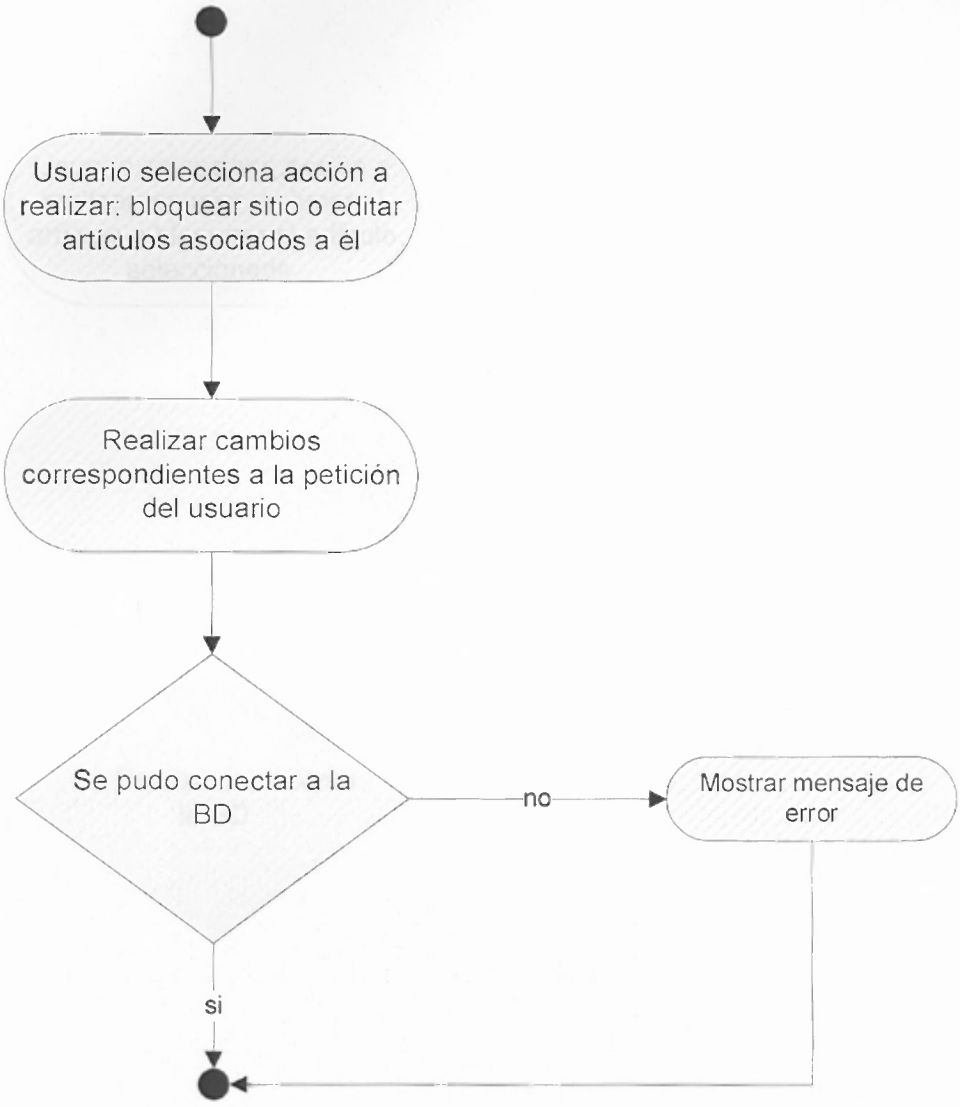

Figura G.9. Diagrama de actividades "administrar sitio"
Fuente: elaborado por el autor

G.3.4 Caso de Uso "administrar artículos"

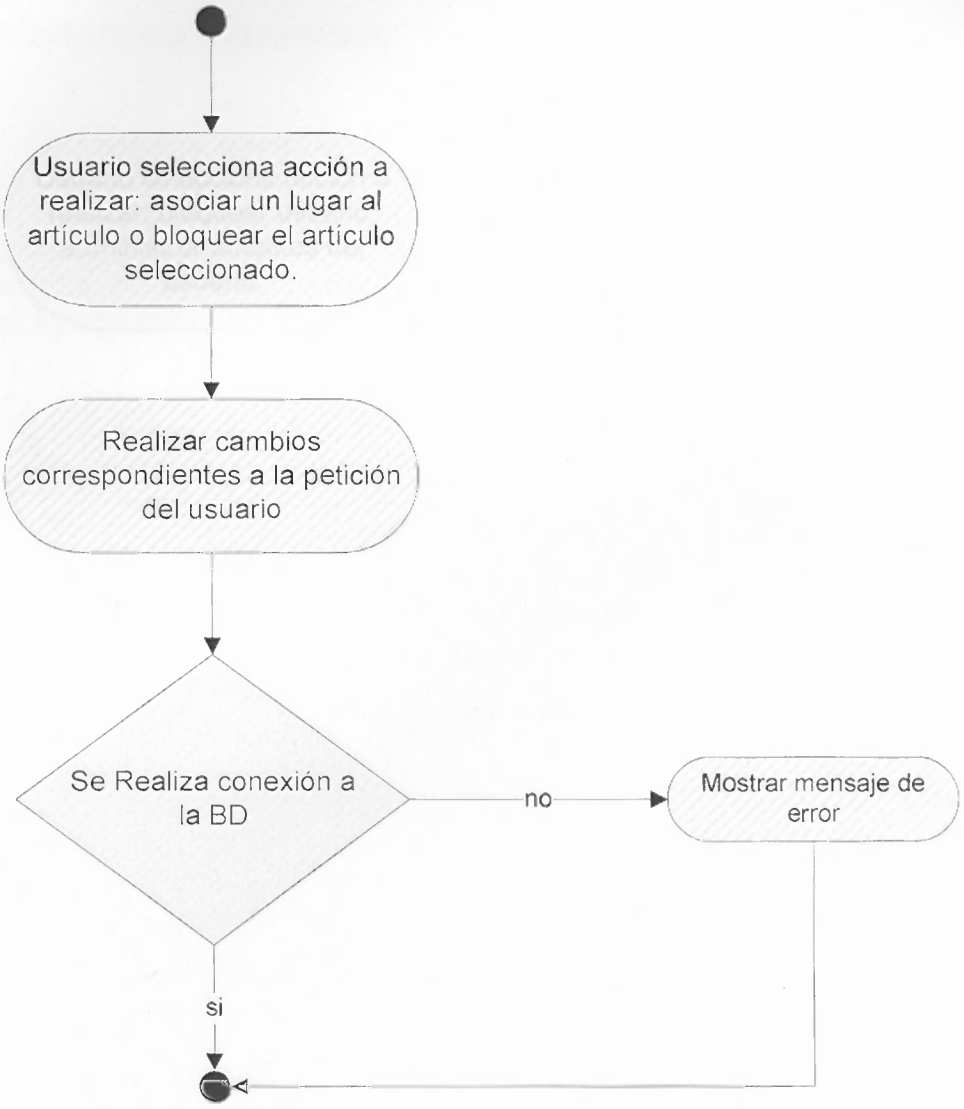

Figura G.10. Diagrama de actividades "administrar artículos"
Fuente: elaborado por el autor

G.3.5 Caso de Uso "administrar usuarios"

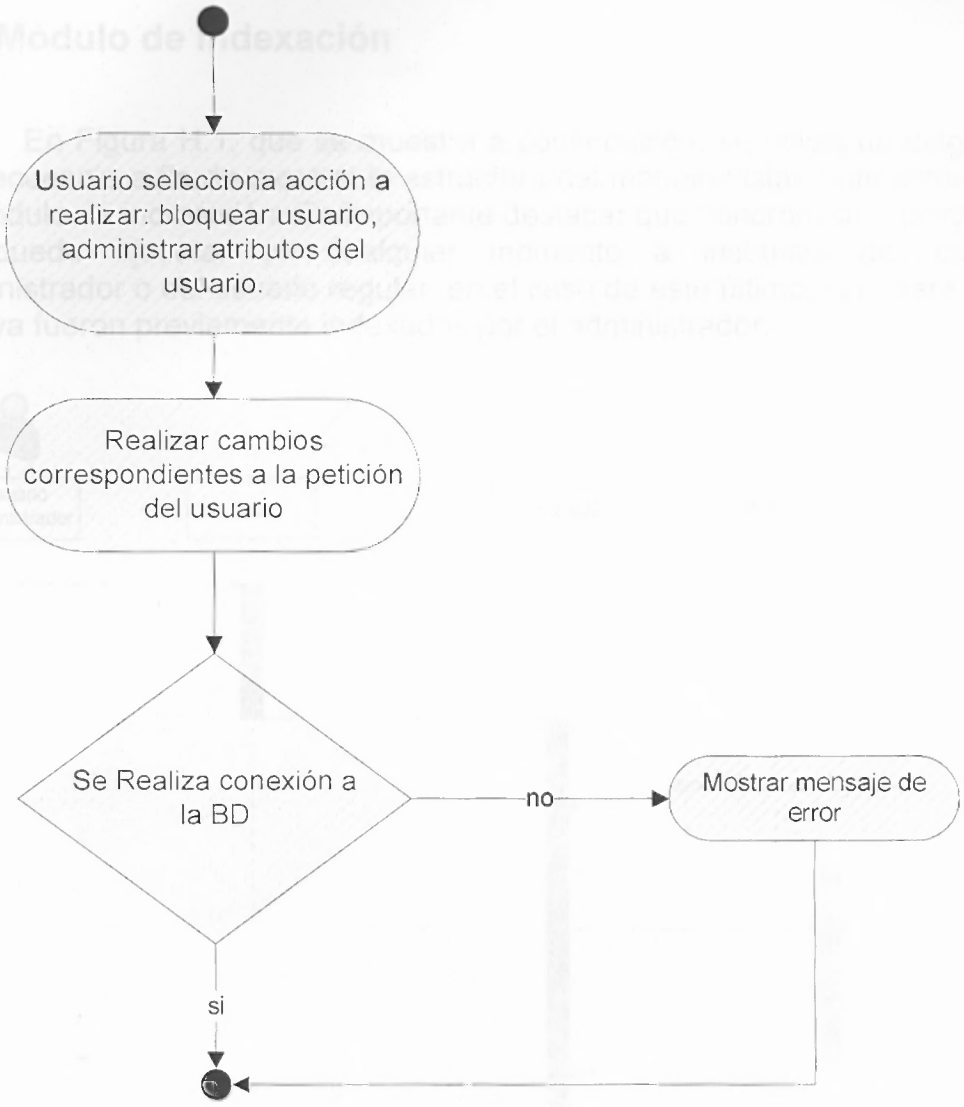

Figura G.11. Diagrama de actividades "administrar usuarios"
Fuente: elaborado por el autor

Apéndice H – Diagramas de secuencia

H.1 Módulo de indexación

En Figura H.1, que se muestra a continuación, se utiliza un diagrama de secuencia a fin de mostrar la estructura del modelo-vista-controlador para el módulo de indexación. Es importante destacar que "sincronizar indexación" se puede ejecutar en cualquier momento a instancia del usuario administrador o del usuario regular, en el caso de este último, solo para sitios que ya fueron previamente indexados por el administrador.

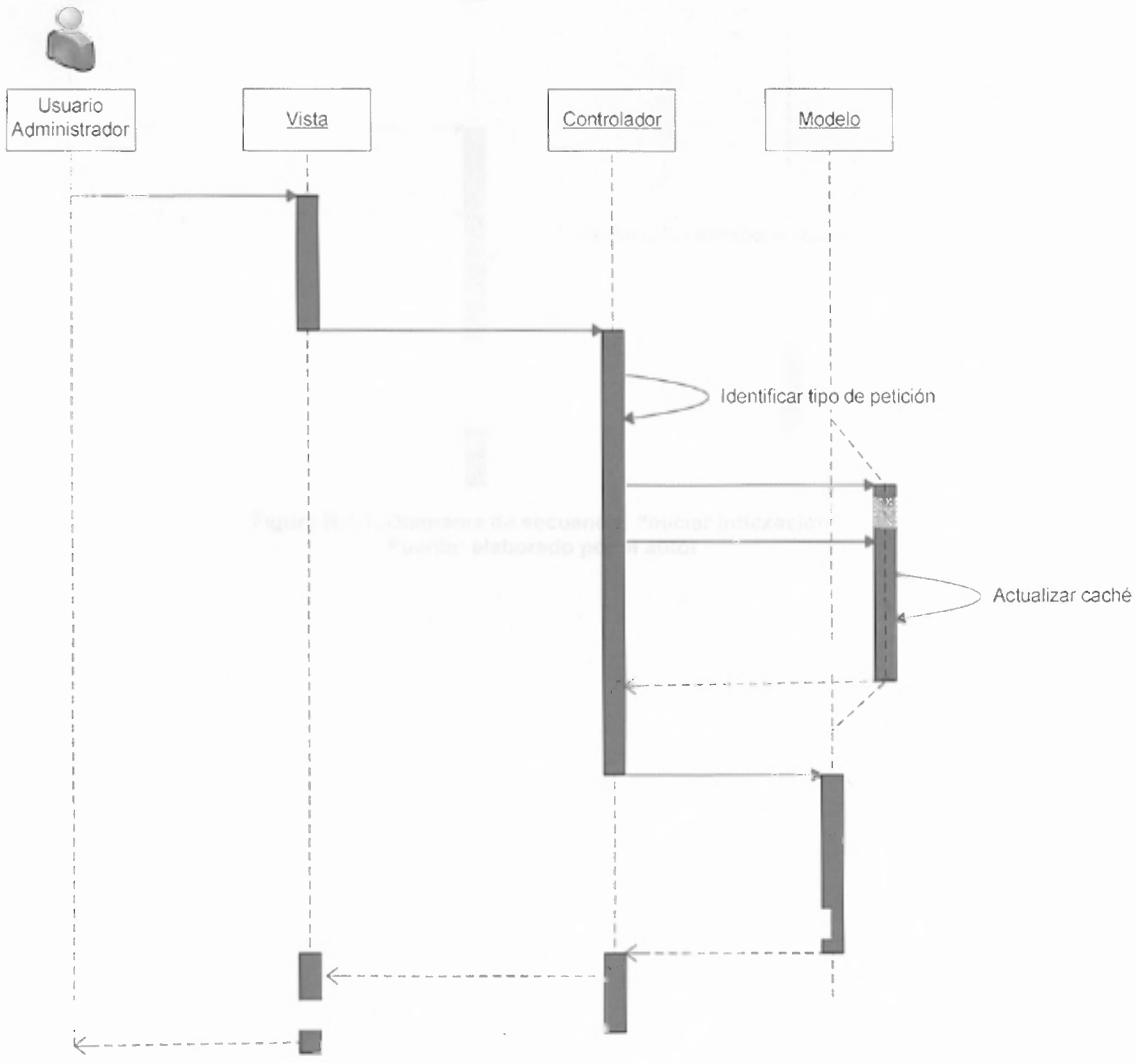

Figura H.1. Diagrama de secuencia "módulo de indexación"
Fuente: elaborado por el autor

H.1.1 Caso de Uso "Iniciar indexación"

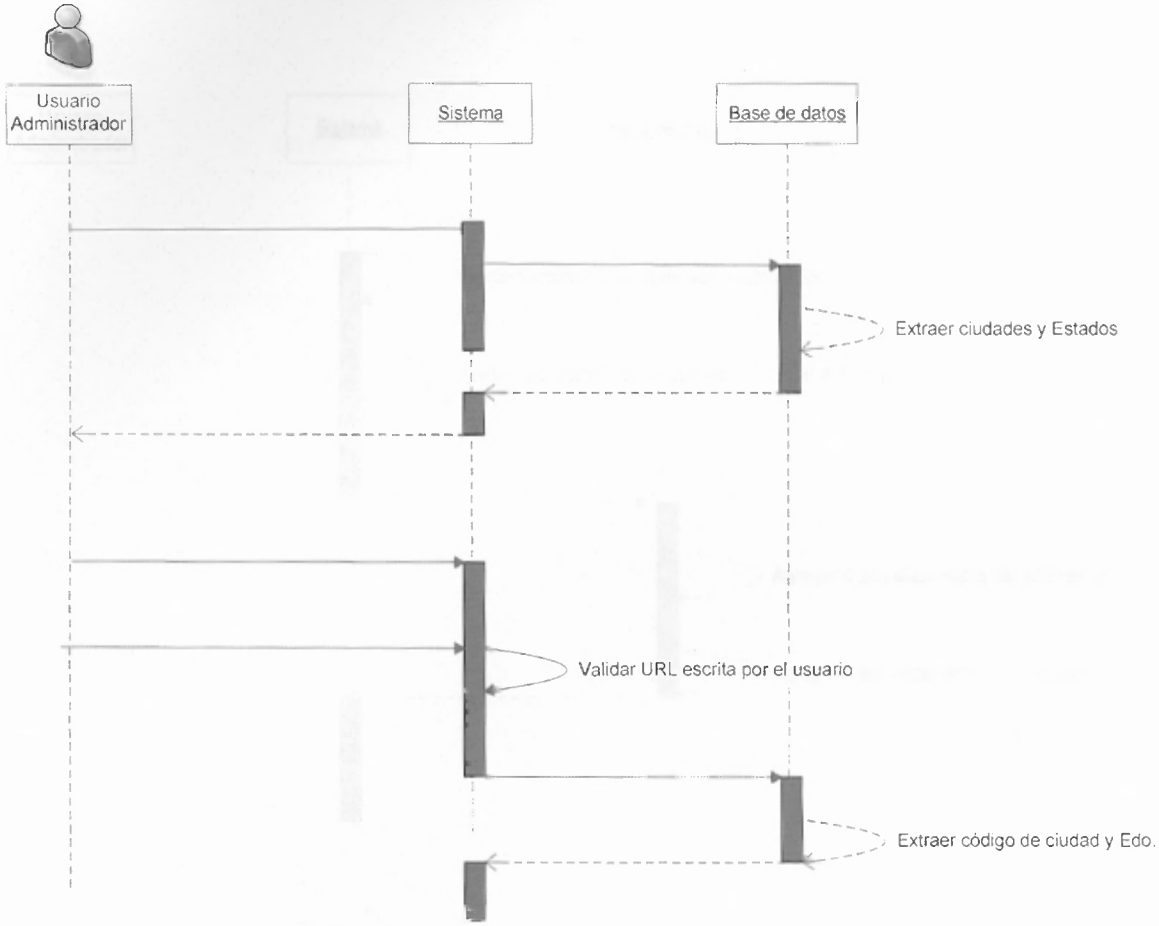

Figura H.1.1. Diagrama de secuencia "iniciar indexación"
Fuente: elaborado por el autor

H.1.2 Caso de Uso "Procesar indexación"

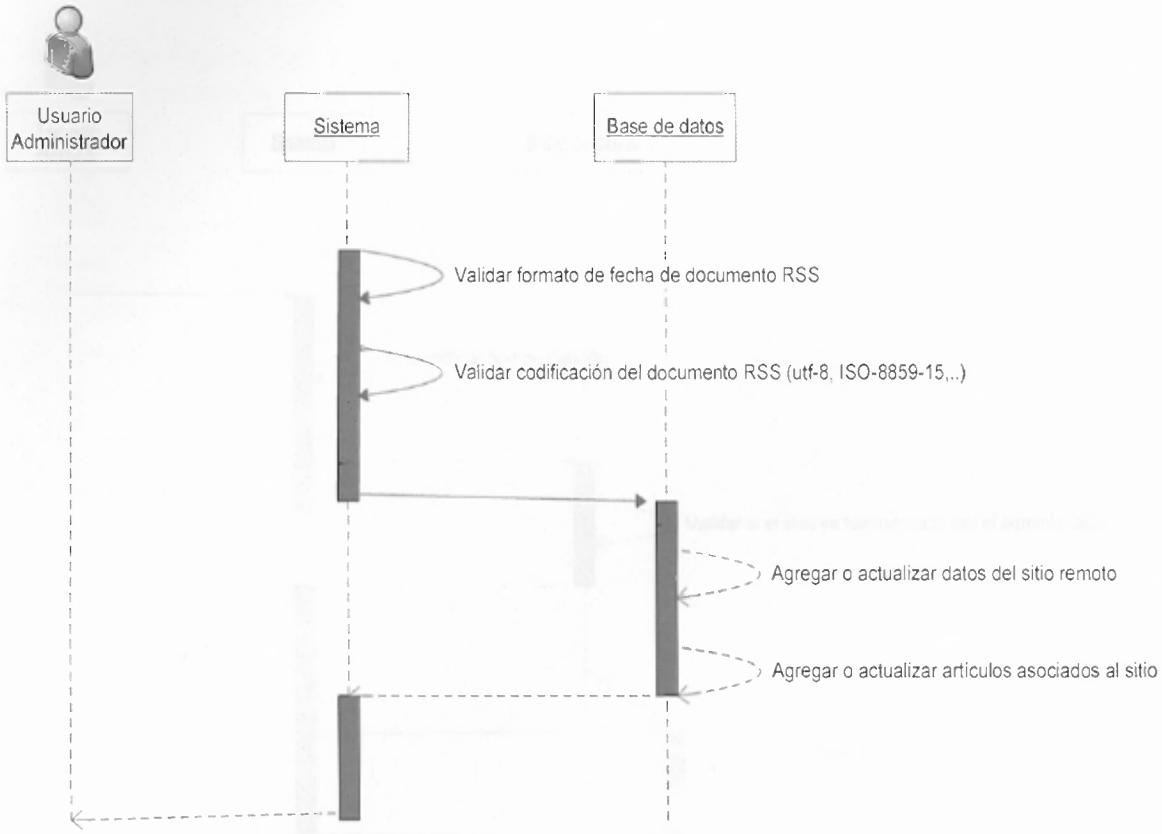

Figura H.1.2. Diagrama de secuencia "procesar indexación"
Fuente: elaborado por el autor

H.1.3 Caso de Uso "Sincronizar indexación"

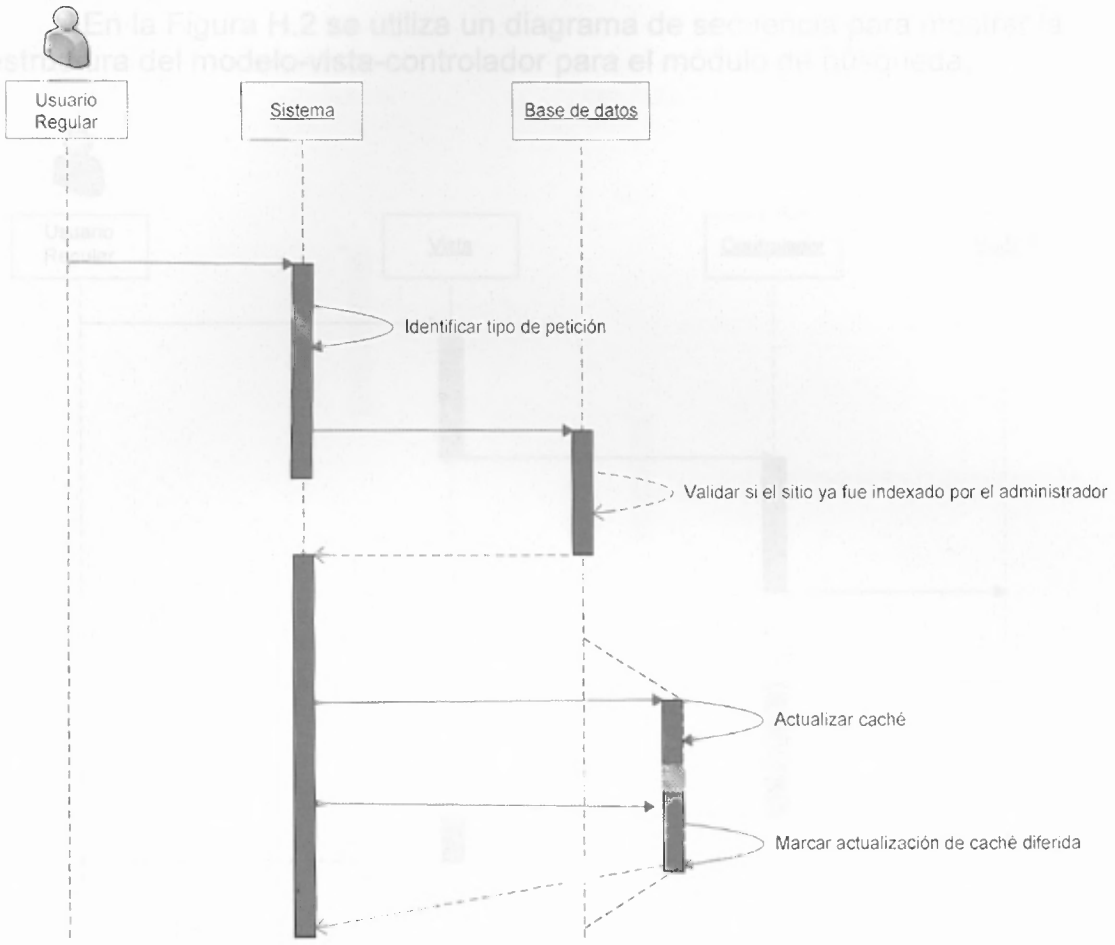

Figura H.1.3. Diagrama de secuencia "sincronizar indexación"
Fuente: elaborado por el autor

H.2 Módulo de búsqueda

En la Figura H.2 se utiliza un diagrama de secuencia para mostrar la estructura del modelo-vista-controlador para el módulo de búsqueda.

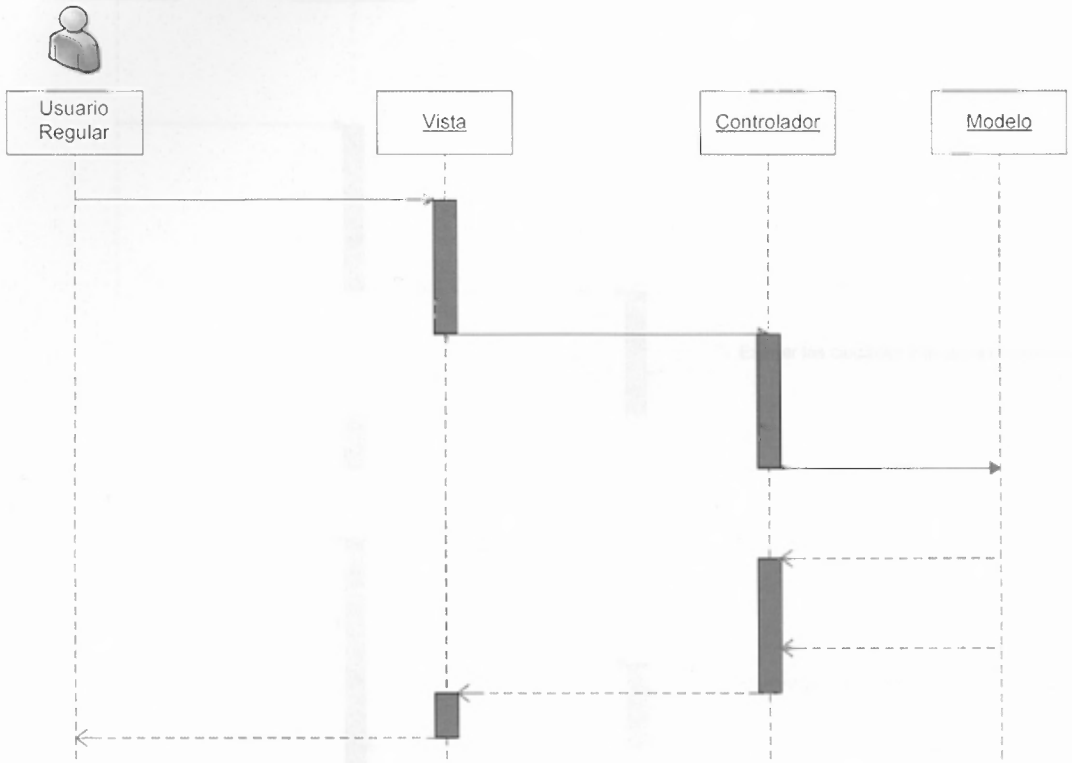

Figura H.2. Diagrama de secuencia "módulo de búsqueda"
Fuente: elaborado por el autor

H.2.1 Caso de uso "Iniciar búsqueda"

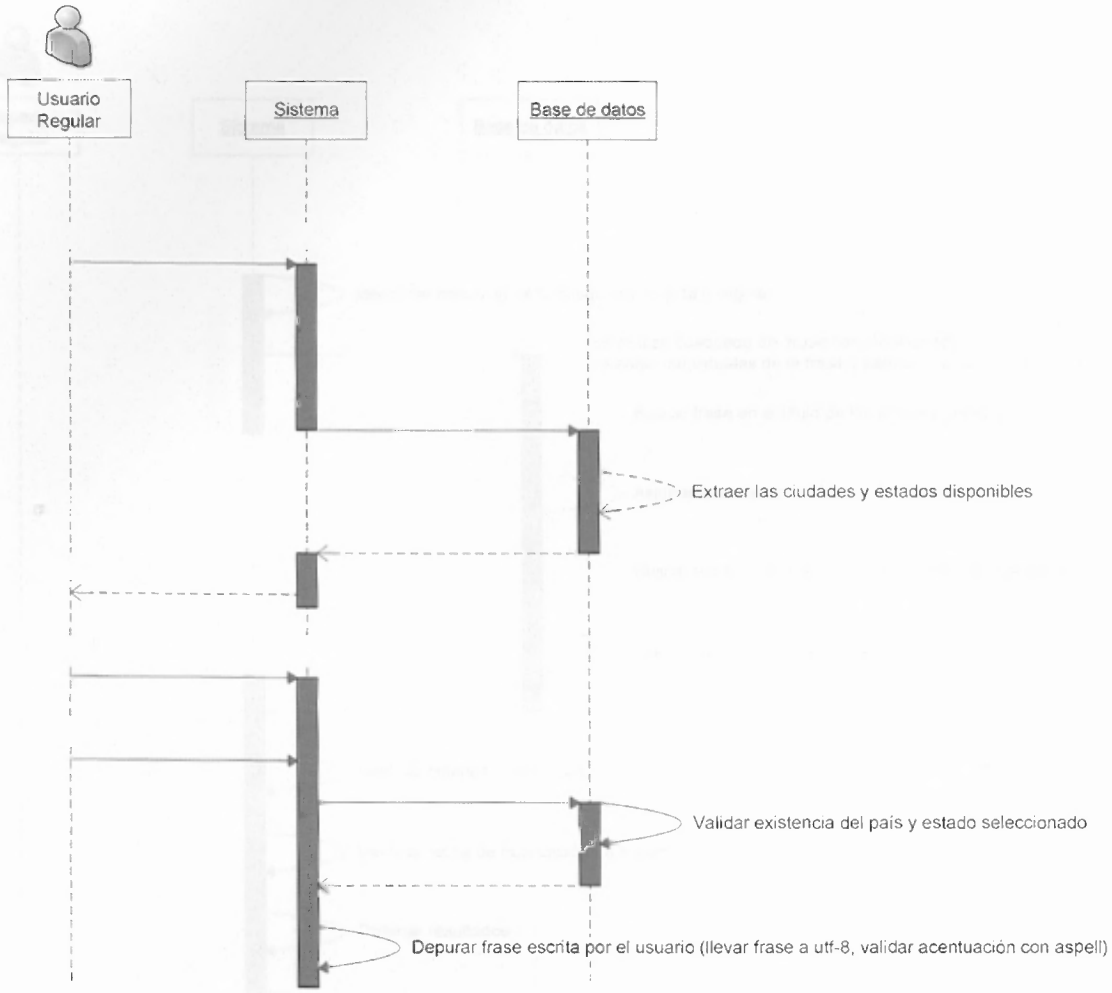

Figura H.2.1. Diagrama de secuencia "iniciar búsqueda"
Fuente: elaborado por el autor

H.2.2 Caso de uso "Realizar búsqueda"

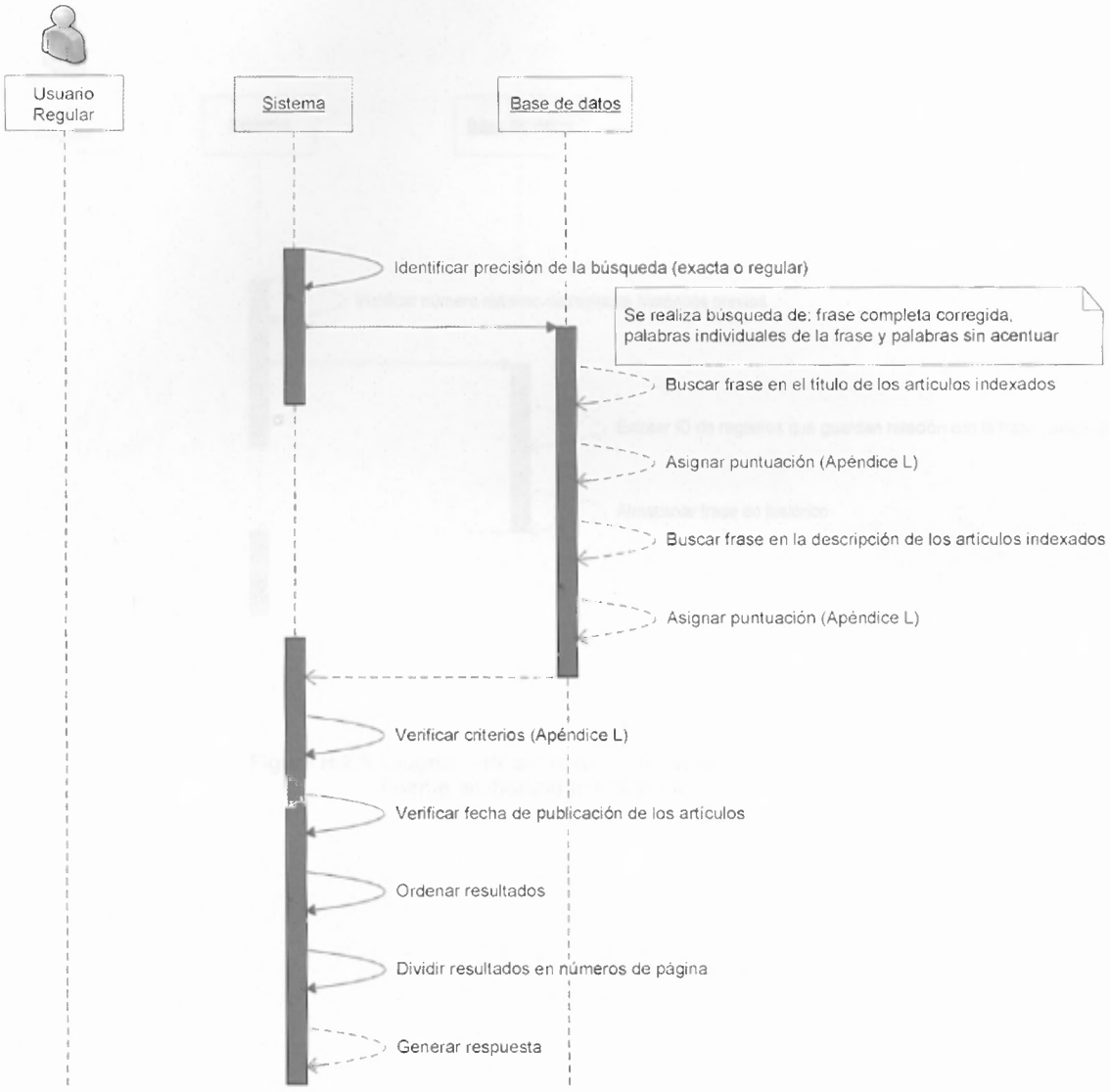

Figura H.2.2. Diagrama de secuencia "realizar búsqueda"
Fuente: elaborado por el autor

H.2.3 Caso de uso "Guardar histórico"

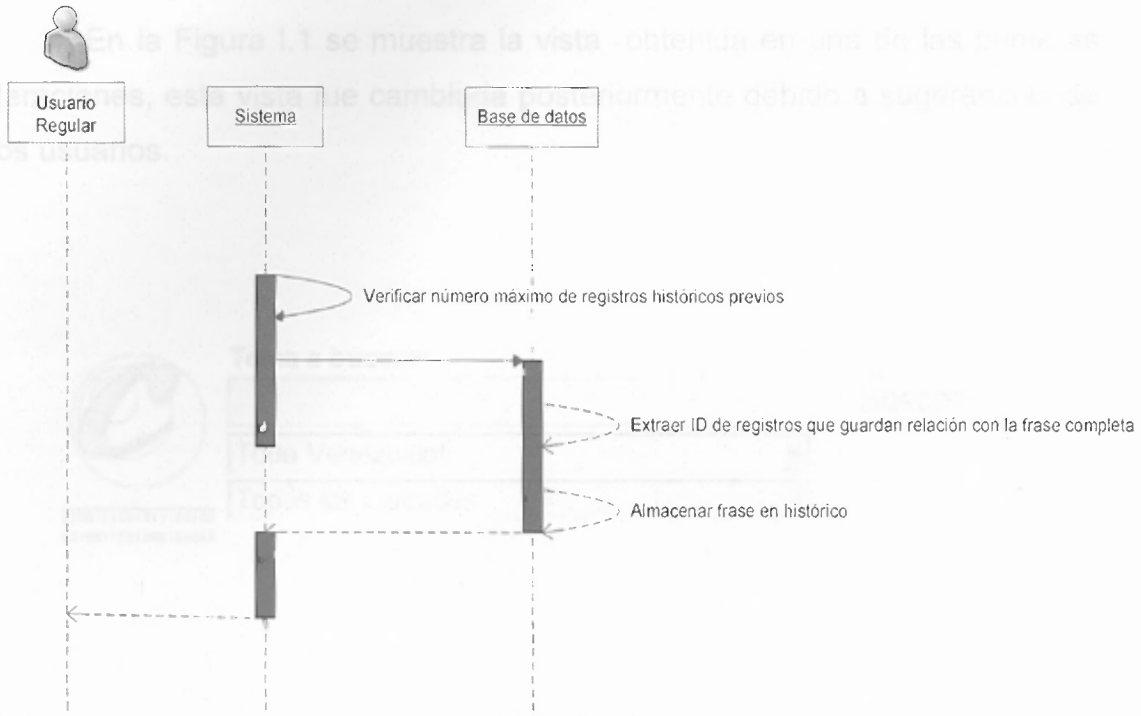

Figura H.2.3. Diagrama de secuencia "guardar histórico"
Fuente: elaborado por el autor

Apéndice I – Versiones preliminares

En la Figura I.1 se muestra la vista obtenida en una de las primeras iteraciones, esta vista fue cambiada posteriormente debido a sugerencias de los usuarios.

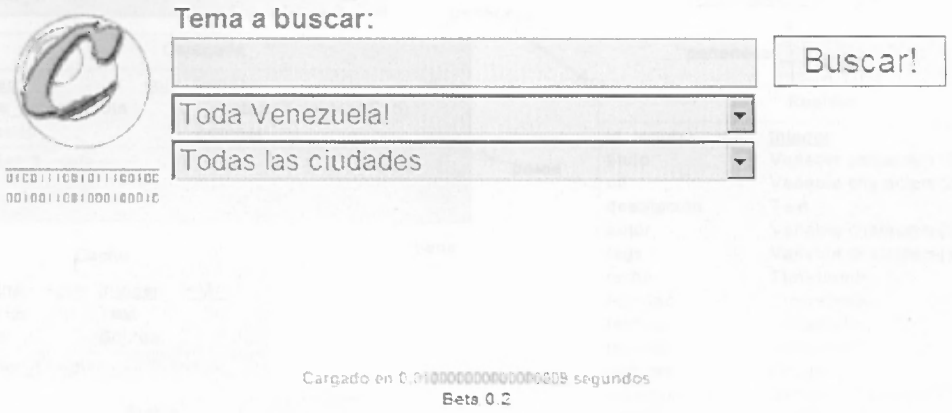

Figura I.1. Vista preliminar del sistema (1era iteración de "Construcción")
Fuente: elaborado por el autor

Apéndice J – Diagrama entidad-relación

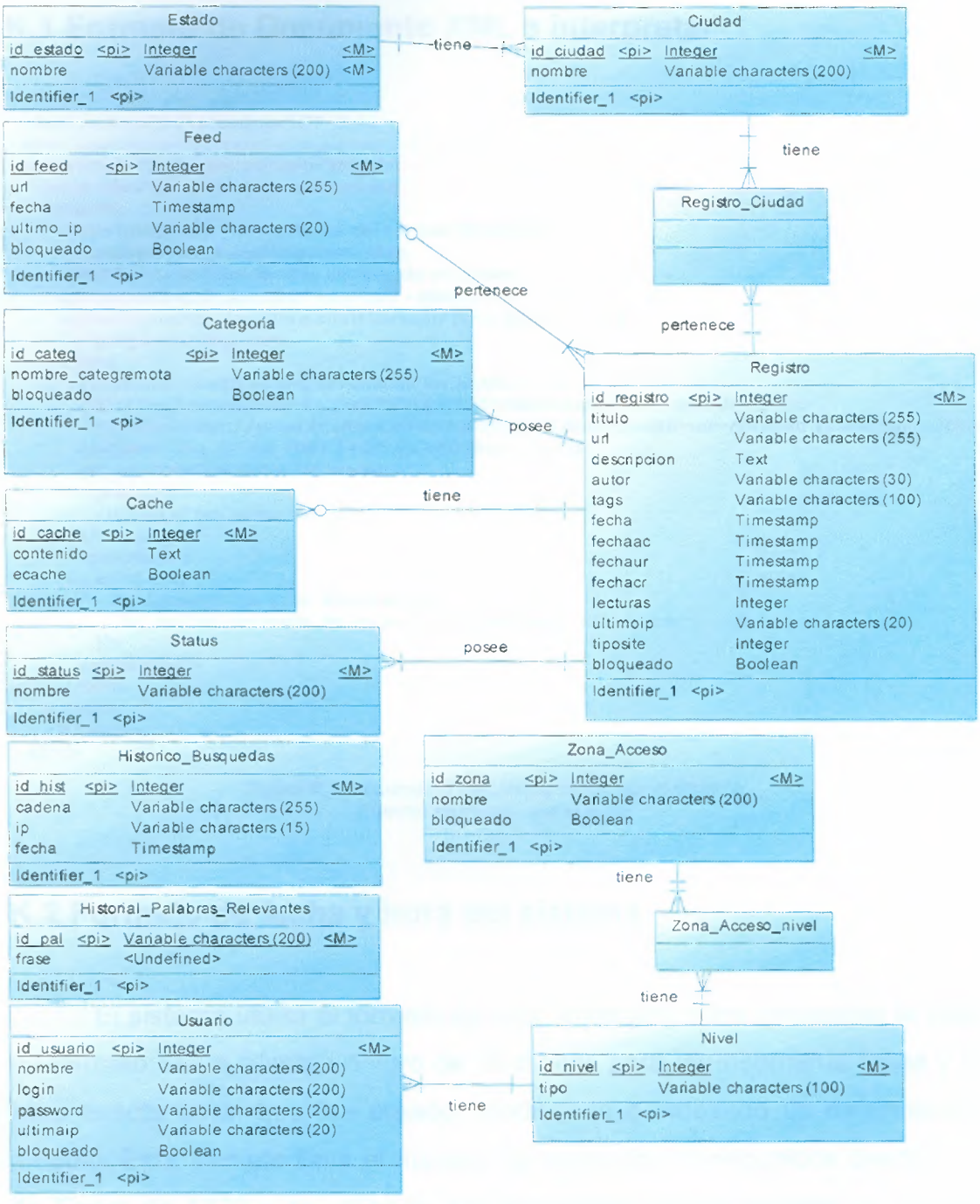

Figura J.1. Diagrama Entidad-Relación del sistema
Fuente: elaborado por el autor

Apéndice K – Formatos utilizados en el sistema

K.1 Formato de Documento XML a interpretar

```
<?xml version="1.0" encoding="UTF-8" ?>
<!--  generator="SistemaDeBusquedas/1.9"  -->
- <rss version="2.0" xmlns:content="http://purl.org/rss/1.0/modules/content/"
  xmlns:wfw="http://wellformedweb.org/CommentAPI/" xmlns:dc="http://purl.org/dc/elements/1.1/">
  - <channel>
      <title>COOLFATA - Las noticias del Estado Miranda</title>
      <link>http://www.coolfata.com</link>
      <description>Noticias locales del estado miranda</description>
      <pubDate>Thu, 07 Jun 2007 19:04:07 +0000</pubDate>
      <generator>http://coolfata.com/version/?v=1.9</generator>
      <language>es</language>
    - <item>
        <title>León suelto en San Antonio de los Altos</title>
        <link>http://www.coolfata.com/2007/06/07/leon-suelto-san-antonio/</link>
        <comments>http://www.coolfata.com/2007/06/07/leon-suelto-san-antonio/#comments</comments>
        <pubDate>Thu, 07 Jun 2007 14:25:01 +0000</pubDate>
        <dc:creator>Fernando</dc:creator>
      - <category>
          <![CDATA[ San Antonio ]]>
        </category>
      - <description>
        - <![CDATA[
            Finalmente ha sido fotografiado
            el felino escapado de un zoológico cercano, al parecer [...]
          ]]>
        </description>
      </item>
    + <item>
    </channel>
</rss>
```

Figura K.1. Documento XML interpretado por el sistema
Fuente: elaborado por el autor

K.2 Formato de fecha y hora del sistema

El sistema utiliza el formato de unix *timestamp* para almacenar la hora en formato de un número entero de 10 dígitos para representar la fecha y la hora exacta en la que fue creado, modificado o indexado un determinado artículo. Este formato lleva el número de segundos transcurridos desde el 1 de Enero de 1970 y es utilizado con frecuencia porque permite un manejo más eficiente de las operaciones de cálculo de fecha debido a que no hay que realizar conversión entre los distintos tipos de formatos.

Apéndice L – Criterios de valoración de resultados

L.1 Manejo de puntuación

A fin de asignarle una mayor prioridad a los artículos, que presenten coincidencias más cercanas a la solicitud del usuario, se establecen los siguientes criterios de asignación de puntos:

Criterio	Puntuación
- Frase a buscar coincide exactamente con el título de un artículo.	+10
- Frase contiene más de 2 palabras y se encuentra contenida en la descripción de un artículo.	+9
- Frase coincide parcialmente con el título de un artículo, es decir, está contenida en el título (misma posición de las palabras) pero no es exactamente el título del artículo.	+7
- Palabras de la frase a buscar están contenidas en el título del artículo pero no coincide el orden.	+3
- Frase a buscar está contenida en la descripción del artículo.	+2
- Frase a buscar contiene palabras relevantes (frecuentes en los últimos artículos publicados).	+1

Tabla L.1. Criterios de asignación de puntos

Fuente: elaborado por el autor

Apéndice M – Implementación del sistema

M.1 Herramientas de edición y lenguajes utilizados

Módulo	Modelo		Vista		Controlador	
	Lenguaje	Editor	Lenguaje	Editor	Lenguaje	Editor
- Indexación - Búsqueda - Mantenimiento	Python	Jext	Javascript (AJAX) HTML dHTML CSS	Macromedia Dreamweaver	Python	Jext, emacs

Tabla M.1. Herramientas utilizadas en implementación

Fuente: elaborado por el autor

Apéndice N – Funcionamiento XML—RPC

Comunicación entre el sistema local y el sistema remoto

En la Figura N.1 se muestran los 2 tipos de comunicaciones para la actualización de los artículos mediante el módulo de indexación.

- Los pasos 1 y 2 describen como se realiza una petición por demanda del sistema local cuando el usuario visualiza el sistema local desde el navegador.

- Los pasos A y B muestran como se actualiza, en tiempo real, un documento en la base de datos del sistema local cuando el autor de un sitio remoto publica un nuevo artículo y se comunica con el servidor XML—RPC.

Figura N.1. Comunicación entre el sistema local y el remoto

Fuente: elaborado por el autor

Apéndice O – Estructuras de almacenamiento

O.1 Estructura diccionario

El diccionario del sistema ofrece la posibilidad de conseguir sinónimos de palabras y realizar correcciones de manera rápida mediante un vector capaz de almacenar palabras y asociar un conjunto de palabras equivalentes o sugeridas. Esta estructura se apoya en la herramienta *GNU Aspell*[7] que consiste en una librería capaz de verificar la ortografía de un conjunto de palabras suministradas.

O.2 Estructura para almacenar registros encontrados

Una vez que los registros son encontrados se procede a almacenar la información en una estructura como la mostrada en la figura O.1:

Puntuación	Título del artículo	URL	Descripción
Número decimal	Cadena de caracteres	Dirección del artículo	Pequeño sumario que describe el artículo

Figura O.1. Vector de registros encontrados en la búsqueda

Fuente: elaborado por el autor

Cada una de las estructuras de este tipo es almacenada, a su vez, en un vector. Finalmente se reordena este vector, de acuerdo a la puntuación obtenida, a fin de mostrar los resultados al usuario final.

[7] GNU Aspell soporta diversos idiomas y adicionalmente ofrece la posibilidad de reconocer palabras acentuadas, letra ñ y caracteres especiales debido a que es compatible con el formato utf-8. Se puede obtener mayor información en: http://aspell.net

Apéndice P – Casos de prueba

P.1 Módulo de indexación

- Caso de prueba "procesar indexación"

Prueba de curso típico:

Entrada: dentro de la sesión del administrador, se escribe la url del sitio remoto, se selecciona ubicación.

Salida: se muestra un mensaje de "url agregada correctamente".

Figura P.1. Solicitud de dirección a indexar
Fuente: elaborado por el autor

Prueba de curso alterno:

Entrada: se escribe una dirección no válida.

Salida: el sistema detecta que la dirección no es correcta.

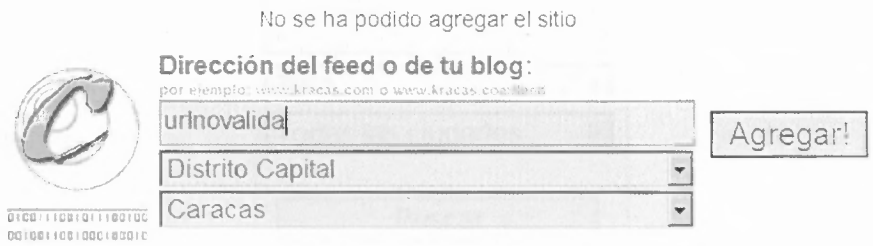

Figura P.2. Respuesta al intentar indexar
Fuente: elaborado por el autor

P.2 Módulo de búsqueda

- Caso de prueba "iniciar búsqueda"

Prueba de curso típico:

Entrada: El usuario escribe una palabra clave o frase y presiona "Buscar".

Salida: El sistema se prepara para realizar la búsqueda.

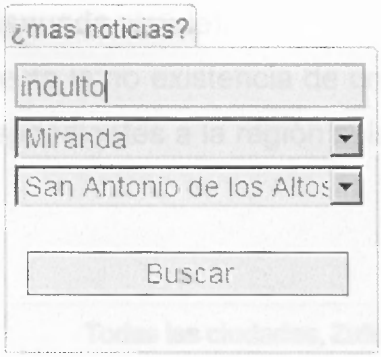

Figura P.3. inicio de búsqueda con frase
Fuente: elaborado por el autor

Prueba de curso alterno:

Entrada: el usuario no escribe ninguna frase o palabra clave y selecciona la opción "Buscar" (búsqueda simple).

Salida: el sistema se prepara para mostrar las últimas noticias registradas en la región seleccionada.

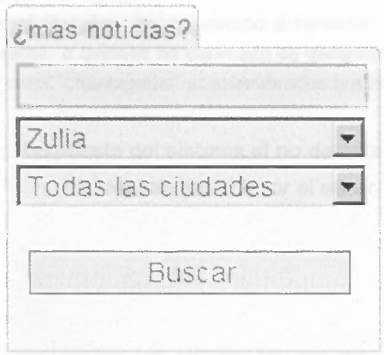

Figura P.4. Solicitud de palabra clave al usuario
Fuente: elaborado por el autor

- Caso de prueba "realizar búsqueda"

Prueba de curso típico:

Entrada: Se recibe una palabra clave o frase.

Salida: El sistema devuelve los resultados de la búsqueda.

Prueba de curso alterno:

Entrada: el usuario no escribe ninguna frase o palabra clave y selecciona la opción "Buscar" (búsqueda simple).

Salida: el sistema detecta la no existencia de una palabra clave y lista las últimas noticias correspondientes a la región seleccionada.

Figura P.5. Respuesta del sistema al no detectar una cadena escrita

Fuente: elaborado por el autor

P.3 Módulo de mantenimiento

- Caso de prueba "iniciar mantenimiento"

Prueba de curso típico:

Entrada: El usuario escribe su usuario y contraseña correctamente.

Salida: El sistema se prepara para cargar la ventana principal.

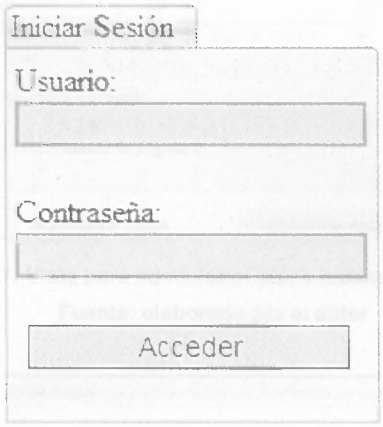

Figura P.6. Solicitud de nombre de usuario y contraseña
Fuente: elaborado por el autor

Prueba de curso alterno:

Entrada: El usuario escribe un usuario o contraseña incorrecta.

Salida: El sistema se muestra un error y no permite continuar.

En la Figura P.7 y P.8, que se muestra a continuación, se puede visualizar la vista principal del sistema que permite la administración de sitios y artículos previamente indexados.

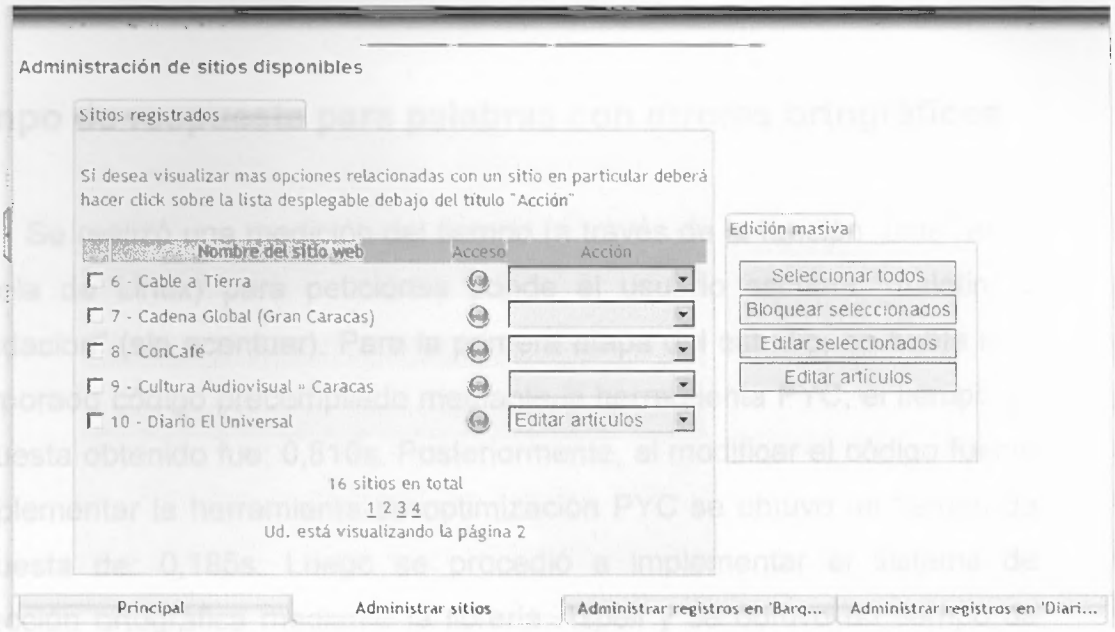

Figura P.7. Vista para administrar sitios indexados en el sistema

Fuente: elaborado por el autor

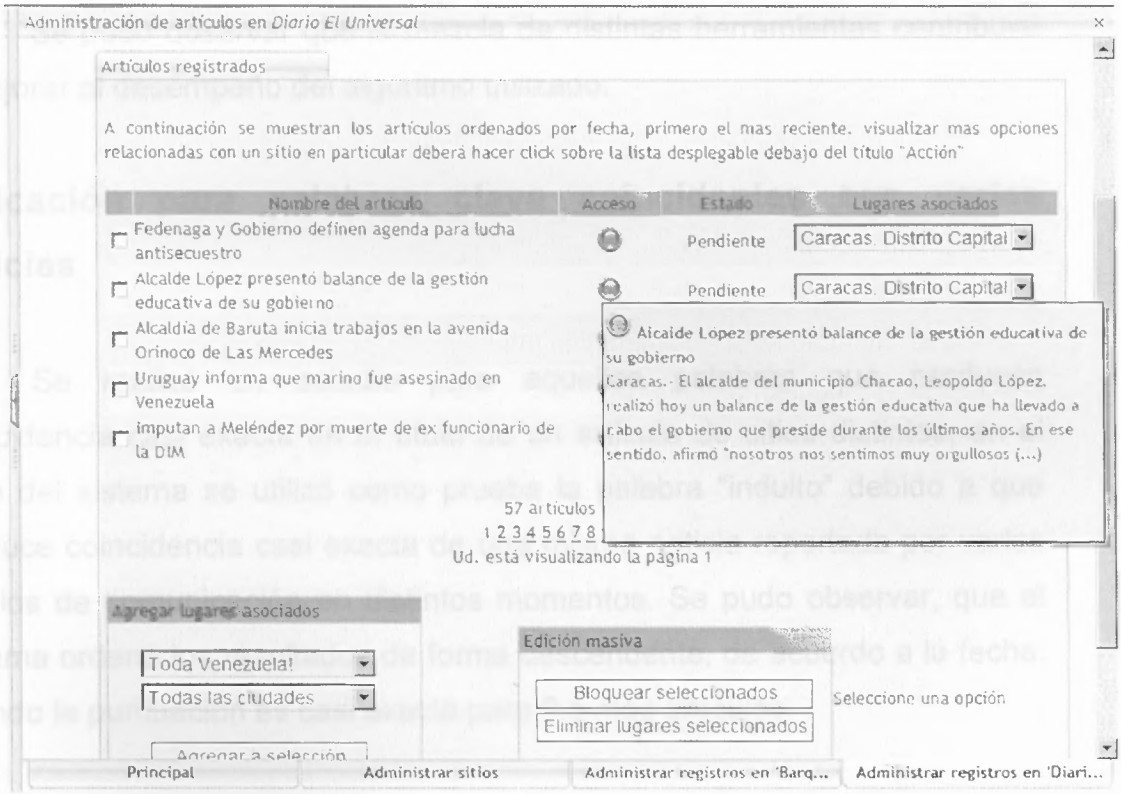

Figura P.8. Administrar artículos a "El Universal"

Fuente: elaborado por el autor

Apéndice Q – Estudio de casos

Tiempo de respuesta para palabras con errores ortográficos

Se realizó una medición del tiempo (a través de la función "time" en la consola de Linux) para peticiones donde el usuario escribía "maletin" o "inundacion" (sin acentuar). Para la primera etapa del estudio, no había sido incorporado código precompilado mediante la herramienta PYC, el tiempo de respuesta obtenido fue: 0,810s. Posteriormente, al modificar el código fuente e implementar la herramienta de optimización PYC se obtuvo un tiempo de respuesta de: 0,185s. Luego se procedió a implementar el sistema de corrección ortográfica mediante la librería *Aspell* y se obtuvo un tiempo de respuesta de: 0,125s.

Se pudo observar que la mezcla de distintas herramientas contribuye a mejorar el desempeño del algoritmo utilizado.

Aplicación para palabras clave coincidentes con varias noticias

Se realizó un estudio para aquellas palabras que producen coincidencia casi exacta en el título de un artículo de sitios distintos, en el caso del sistema se utilizó como prueba la palabra "indulto" debido a que produce coincidencia casi exacta de una misma noticia reportada por varios medios de comunicación en distintos momentos. Se pudo observar, que el sistema ordena los resultados de forma descendente, de acuerdo a la fecha, cuando la puntuación es casi exacta para 2 o más artículos.

Apéndice R – Documento visión

El documento visión está diseñado para mostrar al cliente de manera precisa los siguientes aspectos relacionados al sistema desarrollado: propósito, alcance, definiciones, oportunidades de negocio, entre otros aspectos.

Durante el desarrollo del presente trabajo se obtuvieron herramientas fundamentales para la creación del documento visión, alguno de ellas, son: los distintos diagramas, procesos de análisis, diseño e implementación así como la visión general y alcance del proyecto expresado en el Capítulo I y en la sinopsis.