Universidad de las Ciencias Informáticas

“Vocabulario controlado para la representación del contenido en el sistema de gestión documental de audio y vídeos digitales TeVeo Plus V1.0.”

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

Autores:
Maivys Ramirez Castañet

Joel Ayata Escalona

Tutora:

Ing. Yuneisy Barrios Pérez

La Habana, 2013

"Sueña y serás libre en espíritu, lucha y serás libre en vida"

DECLARACIÓN DE AUTORÍA

Declaramos ser los autores legítimos del trabajo titulado "*Vocabulario Controlado para la representación del contenido en el sistema de gestión documental de audio y vídeos digitales TeVeo Plus v1.0"* y delegamos al centro de Ideo-Informática y a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste se firma el presente a los ____ días del mes de _____ del año 2013.

_____ _____

Firma del autor
Maivys Ramirez Castañet

Firma del autor
Joel Ayata Escalona

_____

Firma de la tutora
Ing. Yuneisy Barrios Pérez

DATOS DE CONTACTO

Maivys Ramirez Castañet
Correo: mcastanet@estudiantes.uci.cu
La Habana, Cuba

Joel Ayata Escalona
Correo: jayata@estudiantes.uci.cu
La Habana, Cuba

Ing. Yuneisy Barrios Pérez
Ingeniera en Ciencias Informáticas
Correo: ybperez@uci.cu, teléfono 8372132
Universidad de las Ciencias Informáticas, La Habana, Cuba

AGRADECIMIENTOS

Maivys:

A mi mamá, por ser la luz de mis ojos, por estar siempre a mi lado brindándome su amor infinito. Por luchar siempre por mí y junto a mí, por sacrificarse y dármelo todo. Por ser la inspiración de todos mis logros, por hacer de mí una mejor persona con sus sabios consejos. Gracias por ser especial mami.

A mi papá por ser tan especial conmigo, por estar siempre presente y brindarme todo su amor y su cariño a pesar de la distancia.

A mi hermano Robertico porque aunque siempre estemos fajados lo quiero con la vida.

A mis hermanas Meivys y Melani por el simple hecho de tenerme siempre presente a pesar de la distancia.

A mi abuela Vive por ser esa viejita tan peleona a la que quiero con la vida porque lo ha dado todo por mí.

A Robe por ser mi segundo papá, por haberme acogido en sus brazos como a su hija, por darme su cariño a pesar de mi carácter y mis berrinches.

A mis tías Duny, Bulé y Baby por ser tan cariñosas y especiales conmigo.

A mi novio Yohansy por su amor y paciencia, por saber esperar por mí durante todos estos años.

A mis primos por siempre preocuparse por mí.

A mi abuela María por brindarme todo su apoyo y hacerme tener fe en que todo me salga bien.

A la pinareña de Yenny , por halarme los pelos cada vez que hacía falta, por compartir conmigo tantos viajes ,por estar siempre cuando la necesité y hacerme creer que yo si podía, en fin por convertirse en mi amiga en tan poquito tiempo.

A la insoportable de Lidis por ser esa personita tan especial para mí, por estar siempre a mi lado en las buenas y hacerme reír en las malas. Por hacer que yo cambiara un poco mi carácter, por tantas peleas, por soportarme durante todos estos años. Por ser más que mi amiga, mi hermana.

A mis amigos (Alejandro, Omarito, Miguel,Andy,Pedro J., Tomás, Guillermo, Elizabet e Ivis) por estar siempre a mi lado cuando los necesite durante estos 5 años.

A Joel por ser el mejor compañero de tesis, sin ti nada de esto hubiese sido posible.

A nuestra tutora Yuneisy por su gran ayuda y paciencia.

A todos los que de una forma u otra hicieron posible la realización de este sueño.

Joel:

A mi madre por ser lo más grande y hermoso que me ha pasado en la vida, por su apoyo y confianza incondicional.

A mi padre por haberme forjado con su carácter y criarme como un hombre de bien.

A mi hermano por inculcarme su pasión por el conocimiento.

A mi sobrina por ser un regalo e inspiración en mi vida.

A mis primos Araim y Pedro por estar ahí en el momento que lo necesitara y ayudarme en estos 5 años.

A Maivys, mi compañera de tesis que ha demostrado ser muy capaz.

A nuestra tutora Yuneisy por su gran ayuda, paciencia y dedicación.

A los amigos que han estado ahí en las buenas y las malas.

A todos los que de una manera u otra han tenido que ver en mi formación como persona y como ingeniero.

DEDICATORIA

Maiwys:

A mis padres por ser mis tesoros.

A mis hermanos y mi familia por su cariño.

Joel:

A mis padres

RESUMEN

En la actualidad realizar búsquedas en un lugar donde existe gran cantidad de información es un reto para los desarrolladores de software. A partir de la introducción y utilización de los vocabularios controlados en los procesos que conforman la fase de organización y representación de la información, los resultados de las búsquedas serán mejores y el usuario podrá obtener de una manera más fácil y automática lo que necesita. Las búsquedas basadas en esquemas conceptuales generan resultados coherentes con el criterio del usuario sobre uno o varios dominios de información. Todo ello depende de que la información se describa mediante un mecanismo de representación del conocimiento donde los datos se relacionen entre sí. El objetivo de este trabajo es desarrollar un Vocabulario Controlado para el sistema de gestión documental de audio y vídeos digitales TeVeo Plus v1.0 que sea capaz de asegurar que cada concepto distinto sea descrito por una única forma lingüística. Durante la investigación se especifican los lenguajes y herramientas seleccionadas para su implementación, la metodología por la cual se rige el desarrollo de la misma y se describen los procesos. Se realizan pruebas a la aplicación desarrollada, obteniéndose resultados satisfactorios.

Palabras clave: Búsquedas, representación del conocimiento, vocabulario controlado.

ÍNDICE

INTRODUCCIÓN

Desde el inicio de la era de la información se hizo necesario el almacenamiento de grandes volúmenes de información para su posterior manejo y gestión. Esto trajo consigo la creación de gran cantidad de aplicaciones informáticas con gran número de funcionalidades para interactuar en tiempo real con la información previamente almacenada. El trabajo diario permitió verificar que, en ocasiones, debido a aspectos de la lengua y el idioma, se cometían errores en los procesamientos de datos, no se obtenían resultados esperados, los criterios de búsqueda eran incorrectos, pues simplemente no había una centralización de los términos y palabras que se debían o permitían usar en determinados entornos digitales.

El área de la gestión y representación del conocimiento comprende aristas, que cercanas a la inteligencia artificial, permiten la inferencia del conocimiento con la finalidad de facilitar la comunicación y el intercambio de información entre diferentes sistemas y entidades dentro de uno o varios dominios de datos. Los vocabularios controlados surgen con la finalidad de centralizar y normalizar la terminología a usar en ciertos ámbitos pues contienen los términos empleados para representar los conceptos, temas o contenidos de los documentos, con miras a mejorar el canal de acceso y comunicación entre los usuarios y las Unidades de Información (entiéndase unidad de información como: biblioteca, archivo o centro de documentación). Aunque en la práctica tradicional se habla de unitérminos, en la actualidad se han efectuado grandes variaciones dando la posibilidad de incorporar términos o descriptores compuestos, es decir, descriptores que se componen de dos o más palabras.

Un intermediario entre el lenguaje que se encuentra en los documentos (lenguaje natural) y el que emplean los especialistas de un determinado campo del saber (lenguaje controlado), es la gestión documental y archivística, que en el caso de la presente investigación, dirigida a los medios periodísticos, donde las tendencias actuales de este sector están marcadas por un medio donde prima la inmediatez en la publicación y la calidad en la noticia, el Archivo Documental juega un papel primordial como herramienta de búsqueda y gestión de la información y el conocimiento. Para efectuar la búsqueda de la información almacenada en los sistemas de gestión documental, se necesita contar con un vocabulario controlado para así realizarla de manera organizada.

Un vocabulario controlado tiene tres funciones fundamentales:

- ✓ Reducir las ambigüedades semánticas.

- ✓ Mejorar la consistencia en la representación de la materia.
- ✓ Facilitar la realización de búsquedas amplias.

Actualmente los sistemas de gestión documental presentan ambigüedades semánticas y problemas en el momento de la realización de búsquedas amplias. Dichos sistemas carecen de herramientas que ayuden a mejorar o resolver estos problemas. También está el desconocimiento por parte de los usuarios del significado de múltiples términos que son utilizados. Existen deficiencias tales como la presencia de diferentes respuestas para una misma búsqueda ya que no están estandarizados los términos, rudimentarios sistemas organizativos que impiden un uso óptimo de estos recursos, falta de criterios y políticas que rijan los procesos en sí, poca práctica en el uso de estándares para la representación de la información.

De la situación antes planteada se identifica como **problema de investigación:** ¿Cómo contribuir a la representación de los contenidos en el sistema de gestión documental de audio y vídeos digitales TeVeo Plus V1.0?

El **objeto de estudio** se centra en los sistemas de gestión del conocimiento y el **campo de acción** está enmarcado en los vocabularios controlados para entornos web.

Se propone como **objetivo general** desarrollar una aplicación web para la representación del contenido en el sistema de gestión documental de audio y vídeos digitales TeVeo Plus V1.0. Del objetivo de la investigación se derivan los siguientes **objetivos específicos** que posibilitarán el desarrollo del mismo:

- ✓ Caracterizar las tendencias actuales del manejo de ontologías y vocabularios controlados en la Gestión Documental y Archivística.
- ✓ Diseñar las funcionalidades de la solución.
- ✓ Implementar las funcionalidades de la solución.
- ✓ Validar mediante pruebas de software la solución.

Para lograr el cumplimiento de los objetivos se trazaron las siguientes **tareas de la investigación:**

1. Caracterización de las tendencias actuales de las ontologías y vocabularios controlados en el ámbito documental y archivístico.

2. Identificación de los requerimientos funcionales y no funcionales de la solución.
3. Selección de las tecnologías, herramientas, estándares, patrones y metodologías necesarias para el desarrollo de la solución.
4. Elaboración de la arquitectura de información de la solución.
5. Realización del análisis y diseño de la solución.
6. Implementación de las funcionalidades de la solución.
7. Aplicación de las pruebas de funcionalidad a la solución.

Los **métodos científicos** utilizados fueron los teóricos y los empíricos, facilitando la investigación y guía para un mejor trabajo, logrando de esta forma un mejor entendimiento del problema.

Métodos teóricos:

Histórico-lógico: Mediante esta práctica se analizaron las tendencias de los sistemas de gestión de vocabularios controlados existentes, permitiendo documentar los elementos necesarios para garantizar la comprensión del funcionamiento de estos sistemas.

Analítico-Sintético: Se utilizó en la revisión de documentos y artículos, donde se identificaron las ideas centrales relacionadas con el funcionamiento de los vocabularios controlados; permitiendo obtener el conocimiento esencial para generar una propuesta de solución adecuada a las exigencias de los medios de prensa.

Métodos Empíricos:

Entrevista: Se identificaron los principales problemas a los que se enfrentan los medios de prensa, así como obtener toda la información referente a la descripción de flujos de trabajos y definición de responsabilidades de los trabajadores del negocio, para diseñar e implementar la solución propuesta.

Como **posible resultado** se tiene una aplicación web, para la gestión del vocabulario controlado que permita identificar, describir, catalogar, recuperar, comentar, publicar y compartir el mismo, utilizado en el sistema de gestión documental de audio y vídeos digitales TeVeo Plus V1.0.

El trabajo de diploma está estructurado de la siguiente manera: introducción, tres capítulos, conclusiones, recomendaciones, bibliografía, anexos y glosario de términos.

Capítulo 1: **"Fundamentación Teórica":** Este capítulo recoge toda la fundamentación teórica que sustenta el desarrollo de la investigación para el posterior desarrollo del sistema propuesto. En el mismo se hace un estudio del estado del arte de sistemas existentes con características similares en cuanto a objetivos de uso y funcionamiento. Se hace referencia a las tendencias actuales de las tecnologías y las metodologías que se aplican durante el del proceso de investigación y de desarrollo del sistema.

Capítulo 2: "Características y diseño del sistema": Se describe la solución propuesta para la situación problemática. Se presentan las características y funcionalidades del sistema a partir de los requisitos funcionales y no funcionales capturados. Además se realizan todos los diagramas correspondientes al diseño del sistema.

Capítulo 3: "Implementación y validación del Sistema": En este capítulo se incluye la programación realizada a partir de los requerimientos, así como las pruebas realizadas para su validación.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

En el presente capítulo se precisan elementos teóricos que sustentan la investigación y el desarrollo del tema propuesto, a través del estudio y análisis de soluciones existentes. Se tratan las principales definiciones relacionadas con los vocabularios controlados. Se describen las tecnologías, herramientas, lenguajes y metodologías de desarrollo utilizadas para el análisis y personalización del sistema.

1.1 Vocabulario Controlado

1.1.1 Definiciones de Vocabulario Controlado

Un vocabulario controlado es una lista o índice de términos que establece relaciones unívocas y precisas entre ellos, así como con los conceptos representados [1].

Un vocabulario controlado es una lista de valores, en la que cada uno de ellos es expresado por medio de una cadena de caracteres [2].

Un vocabulario controlado es una lista de términos que ha sido elaborada explícitamente por una autoridad [3].

1.1.2 Objetivo y propósitos del Vocabulario Controlado

Objetivo

El objetivo principal de un vocabulario controlado es asegurar que cada concepto distinto sea descrito por una única forma lingüística. Si existen múltiples formas, estas deben ser controladas o reguladas de tal forma que la información o contenido que es entregado al usuario no sea distribuido por el sistema bajo múltiples puntos de acceso, pero sean almacenados juntos en un mismo lugar [4].

Propósitos

El propósito de los vocabularios controlados es proveer una forma de organización de la información. A través del proceso de asignación de términos seleccionados para describir documentos y otros tipos de contenidos, los materiales son organizados de acuerdo con varios elementos que han sido elegidos para describirlos [4].

Los vocabularios controlados sirven a cinco propósitos:

1. **Traducción:** Provee la forma de convertir el lenguaje natural de autores, personal encargado de la indización y usuarios en un vocabulario que puede ser usado para indizar y obtener a través de búsquedas.

2. **Consistencia:** Provee uniformidad en el formato de los términos y en la asignación de estos.

3. **Indicación de relaciones:** Indica relaciones semánticas entre términos.

4. **Etiquetado y navegación:** Provee jerarquías consistentes y limpias en un sistema de navegación para ayudar a los usuarios a encontrar los contenidos deseados.

5. **Obtención:** Sirve como ayuda en la búsqueda para la localización de objetos de contenido.

1.1.3 Principios que guían el desarrollo de un Vocabulario Controlado

Existen cuatro principios fundamentales que guían el diseño y desarrollo de un vocabulario controlado [4]. Estos son:

1. Eliminación de ambigüedades.

2. Controlar las sinonimias.

3. Establecimiento de relaciones entre términos donde sea necesario.

4. Pruebas y validación de términos.

1.1.4 Norma para la gestión y construcción de Vocabularios Controlados

ANSI/NISO Z39.19 – 2005 (R2010) Directrices para la construcción, formato y administración de vocabularios controlados monolingües.

Con el objetivo de realizar buenas prácticas en el desarrollo del sistema se decide utilizar esta norma internacional que presenta las directrices y convenciones de los contenidos, visualización, construcción, pruebas, mantenimiento y gestión de los vocabularios controlados monolingües, la cual se centra en vocabularios controlados que son utilizados para la representación de objetos de contenido en los sistemas de organización del conocimiento incluyendo listas de sinónimos, anillos, taxonomías y tesauros [4].

Este estándar manifiesta que el control de un vocabulario se basa en tres métodos principales:

1. Definiendo el ámbito de los términos.

2. Usando la relación de equivalencia para enlazar sinónimos y términos cercanamente sinónimos.

3. Distinguiendo entre homógrafos.

1.2 Componentes del Vocabulario Controlado y sus relaciones

Descriptor

Término o palabra que luego de un proceso de revisión y posterior aprobación por un conjunto de especialistas se adiciona al vocabulario controlado, además es considerado una palabra o conjunto de palabras del lenguaje corriente que responden a un concepto. Los descriptores son términos o unidades lingüísticas que expresan conceptos. Un concepto se expresa con un único término y ese término responde a un único concepto. Dicho término será utilizado para el análisis y la recuperación de información [4].

No descriptor

Término o palabra que está en uno de los siguientes estados:

- ✓ Esperando revisión (por el grupo de especialistas para ser aprobado o no como un descriptor).
- ✓ Eliminado o prohibido totalmente del Vocabulario Controlado [4].

1.2.1 Relaciones entre los términos del Vocabulario Controlado

Los términos que los conforman se interrelacionan entre ellos bajo tres modalidades de relación [4]:

Relaciones jerárquicas: Establecen subdivisiones que generalmente reflejan estructuras de TODO/Parte.

Estas relaciones cubren tres tipos situaciones lógicamente diferentes y mutuamente exclusivas:

- ✓ La relación genérica.
- ✓ La relación de instancia.
- ✓ La relación todo-parte.

Relaciones de equivalencia: Controlan la sinonimia, homonimia, antonimia y polisemia entre los términos.

Estas relaciones cubren cinco tipos básicos:

- ✓ Sinónimos.
- ✓ Variantes léxicas.
- ✓ Sinónimos cercanos.
- ✓ Publicaciones genéricas.
- ✓ Referencias cruzadas a elementos de términos compuestos.

Relaciones asociativas: Mejoran las estrategias de recuperación y ayudan a reducir la poli jerarquía entre los términos.

Estas pueden ser cualquier relación comprendida en las siguientes:

- ✓ Aquellas que pertenecen a la misma jerarquía.
- ✓ Aquellas que pertenecen a diferentes jerarquías.

1.3 Indización

La indización es el proceso de describir o representar el contenido temático de un recurso de información. Este proceso da como resultado un índice de términos de indización que será utilizado como herramienta de búsqueda y acceso al contenido de recursos en sistemas de recuperación de información.

La indización es una actividad técnica-intelectual integrada por procedimientos dirigidos a desglosar, descifrar, analizar y resumir el contenido de los documentos, todo esto con el objetivo de hacer posible el almacenamiento, recuperación, acceso y difusión de la información, facilitando su recuperación, ya sea en forma directa por los usuarios o a través de índices, catálogos, base de datos, etc., y se cumple en forma controlada y normalizada a través de un proceso conformado por cuatro etapas [5]:

1. Revisión del contenido del documento.
2. Selección de los conceptos.

3. Traducción de los conceptos en descriptores (vocabulario controlado).

4. Establecimiento de enlaces sintácticos entre los descriptores (temas relacionados).

En el sistema de gestión documental de audios y vídeos digitales TeVeo Plus V1.0 se manejan gran cantidad de archivos digitales, los cuales poseen metadatos, por lo que su gestión, organización, búsqueda y recuperación sería más fácil si se emplearan estos metadatos como material de organización y reconocimiento, pues estos describen de manera sintética y específica los principales aspectos del contenido que representan. Un vocabulario controlado en este ámbito permitirá mantener aún más cerrado el espectro de conocimiento necesario para la gestión realizada en el sistema TeVeo Plus V1.0 mediante la indización en diferentes categorías de los términos que representaran a estos contenidos, de tal forma que si se deseara realizar una búsqueda por temáticas, personas, fechas, lugares, cámara, u otro elemento representado por un metadato y estos términos estuvieran indizados de antemano en un vocabulario controlado, la rapidez y eficacia sería mucho mayor.

1.4 Problemas a resolver con el Vocabulario Controlado

Ambigüedad

La ambigüedad ocurre en el lenguaje natural cuando una palabra o frase tiene más de un significado y puede ser usada para representar más de un concepto diferente [4].

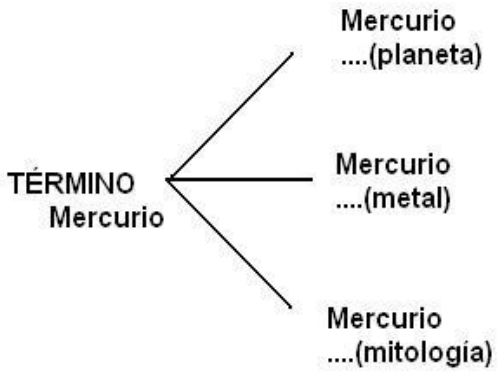


Figura 1. Ejemplo de una Ambigüedad.

Un vocabulario controlado debe compensar los problemas causados por la ambigüedad asegurando que cada término tengo un único e incomparable significado.

Sinonimia

La sinonimia ocurre cuando un concepto puede ser representado por múltiples términos teniendo el mismo significado o uno similar, esto puede dificultar la obtención del contenido deseado pues este puede ser descrito por términos diferentes pero equivalentes [4].

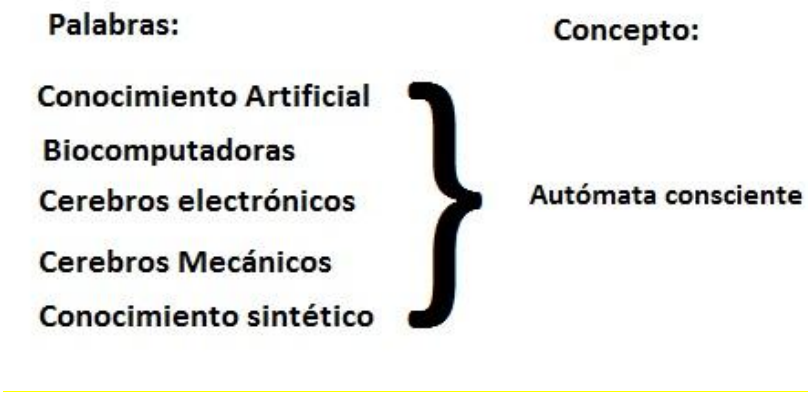


Figura 2. Ejemplo de Sinonimia.

Un vocabulario controlado debe compensar los problemas causados por la sinonimia asegurándose que cada concepto sea representado por un único término principal y debe listar cada uno de los otros sinónimos y variantes como términos no principales con referencias de tipo USE al término principal.

1.5 Tipos de Vocabularios Controlados

Los vocabularios controlados están estructurados para habilitar y manejar los diferentes tipos de relaciones entre los términos que ellos contienen. Hay cuatro tipos de vocabularios controlados, determinados por su estructura crecientemente compleja [4], estos son:

- ✓ Lista
- ✓ Anillos de sinónimos
- ✓ Taxonomía
- ✓ Tesauros

Figura 3. Ejemplo del aumento de la complejidad estructural de los Vocabularios Controlados.

1.5.1 Lista

Una lista (a veces llamadas listas de selección) es un conjunto limitado de términos, organizada como una simple lista alfabética o en algún otro orden lógico evidente. Las listas son usadas para describir aspectos de objetos de contenido o entidades que poseen un número limitado de posibilidades [4]. Ejemplos, incluyen geografía (país, estado, cuidad), idioma (Inglés, Francés, Suizo) o formato (texto, imagen, sonido).

Ejemplo 1: Lista alfabética simple (Organización alfabética de nombres de países)

Alabama

Alaska

Arkansas

California

Connecticut

Ejemplo 2: Lista lógica simple (Organización de los planetas de acuerdo a su cercanía con el Sol)

Mercurio

Venus

La Tierra

Marte

Júpiter

Saturno

Urano

Neptuno

Plutón

1.5.2 Anillo de sinónimos

Un anillo de sinónimos es considerado un vocabulario controlado, este juega un rol diferente a los de otros tipos de vocabulario controlados. Los anillos de sinónimos no pueden ser usados durante el proceso de indización, por lo que son usados solo durante la recuperación. El uso de estos asegura que un concepto que pueda ser descrito por múltiples sinónimos o términos equivalentes va a ser devuelto si cualquiera de ellos es usado en una búsqueda.

Un anillo de sinónimos, por lo tanto, es un conjunto de términos que son considerados equivalentes para propósitos de recuperación, que permite a los usuarios acceder a todos los objetos de contenido o entradas de la base de datos de cualquiera de ellos [4].

1.5.3 Taxonomía

Una taxonomía es un vocabulario controlado consistente en términos principales, todos ellos conectados en jerarquía o poli-jerarquía. No exige que sus componentes estén conectados mediante un tipo específico de relaciones; simplemente requiere que sus componentes estén organizados.

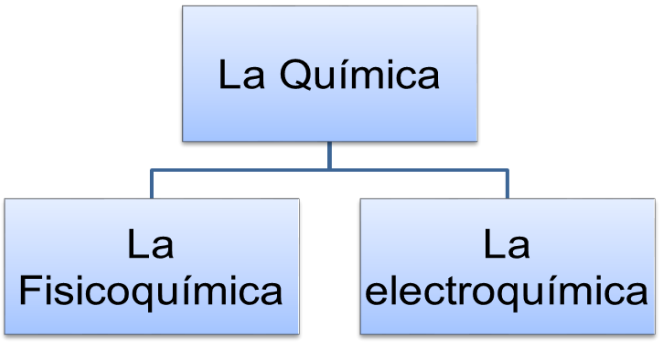


Figura 4. La jerarquía de una taxonomía.

La utilización de taxonomías en la categorización de recursos de información ofrece los puntos fuertes generales de los lenguajes controlados, como son: el tratamiento de los aspectos semánticos y sintácticos del lenguaje; la representación de conceptos implícitos; la creación de una visión global de los dominios que son objeto de representación; la exhaustividad en la indización; y la solución de los problemas que conllevan los contextos multilingües. Desde el punto de vista de la gestión de sitios web, la utilización de taxonomías en la categorización de los recursos ofrece dos importantes beneficios adicionales:

- ✓ Rentabiliza los esfuerzos de construcción y mantenimiento de la taxonomía y de categorización de recursos; ya que una misma herramienta puede ser reutilizada en el desarrollo de diferentes aplicaciones de búsqueda, navegación, personalización, etc.

- ✓ Permite mantener la consistencia conceptual y designativa en la representación de los elementos de un mismo dominio, lo cual crea en los usuarios una imagen de consistencia en el conjunto del sitio web, y en la entidad que lo crea y lo mantiene.

1.5.4 Tesauros

Un tesauro es un vocabulario controlado organizado en un orden conocido y estructurado de manera que varias relaciones entre términos son mostradas claramente e identificadas por indicadores de relaciones estandarizados. Estos indicadores deben ser empleados recíprocamente [4].

Características principales de los tesauros [6]:

- ✓ Proporcionan una representación del conocimiento compartido de un dominio con el fin de facilitar la comunicación eficiente.

- ✓ Son sistemas basados en conceptos que representan conocimiento altamente complejo.

- ✓ Pueden englobarse dentro de los lenguajes controlados. Puesto que ambos definen el vocabulario de un dominio específico mediante un conjunto de términos básicos y las relaciones entre dichos términos.

- ✓ Están relacionados con la terminología empleada para representar los conceptos de un dominio particular.

- ✓ Utilizan jerarquías para agrupar términos en categorías y subcategorías.

- ✓ Pueden utilizarse para catalogar y organizar recursos de información.

Aunque vocabularios controlados y ontologías no responden a un mismo concepto no están totalmente reñidos. Una ontología surge con el fin de moldear el conocimiento del mundo, compartirlo y obtener inferencias a partir de él, lo cual es básicamente lo que se quiere lograr con un vocabulario controlado pero centrándose en un área más pequeña definida por el sistema de gestión documental de audio y vídeos digitales TeVeo Plus V1.0, un vocabulario controlado permitirá indizar y recuperar información a partir de las relaciones establecidas entre sus términos y sus conceptos por lo que el uso de cualquiera de las herramientas para construir ontologías estaría subutilizado, es decir estas conforman un concepto más amplio y abarcador del cual un vocabulario controlado solo sería una pieza fundamental, que no es lo que se espera conseguir con el desarrollo de esta investigación. Un vocabulario controlado en forma de Tesauro se ajusta más a las necesidades que el sistema TeVeo Plus V1.0 tiene hoy en día, pues resuelve de manera directa los problemas de sinonimia y ambigüedades, además de permitir establecer relaciones entre los términos tanto de jerarquía como de asociación.

Un tesauro es el punto culminante en cuanto a especialización de vocabularios controlados, pues integra las funcionalidades y características de las listas, anillos de sinónimos y taxonomías en una sola y más completa estructura organizativa de términos indizados, en pocas palabras su finalidad es indizar documentos y recuperar información y va dirigido a los profesionales de la Información y Documentación, y buscadores de información.

1.6 Herramientas para la construcción y gestión de Tesauros

1.6.1 Cognatrix

Cognatrix es una aplicación nativa de Mac OS X (“Cocoa/Aqua”) para la construcción de tesauros. Es una herramienta auto contenida, es decir no depende de paquetes de bases de datos o aplicaciones de terceros, incluye capacidades multi-lenguaje que permiten definir relaciones de equivalencia entre conceptos, expresadas en varios lenguajes, no posee prevención de nombres duplicados pero los marca para poder revisarlos. Está compilado como un "binario universal". Corre sobre Mac OS X 10.5 (Leopard) y superiores. El hardware mínimo requerido es el mismo que para el propio Mac OS X [7].

Este sistema tiene características muy favorables pero presenta la limitación de que solamente está hecho para MacOS lo cual implica una traba para su uso.

1.6.2 Midos Thesaurus

Midos Thesaurus se basa en un nuevo enfoque para la gestión y tratamiento de los tesauros. Está compuesto por una base de datos de texto completo en el que todas las relaciones son una expresión almacenada en un formato de datos variables.

No contiene una base de datos relacional, está completamente en la memoria y se reproduce como un modelo de datos relacional. Dado que no se requiere ningún otro acceso a los archivos a través del programa de ejecución, todas las funciones del tesauro son muy rápidas [8].

Al ser desarrollado por una empresa alemana toda la implementación y documentación están en alemán, lo cual supone un impedimento importante debido al tiempo que se emplearía solamente en su traducción.

1.6.3 MultiTes

Es un compendio de tecnologías que incluyen:

- ✓ MultiTes Pro (Creación de tesauros)
- ✓ MultiTes Web Deployment Kit (WDK, Publicación de tesauros en Internet/Intranet, sin necesidad de servidor SQL)
- ✓ MultiTes EDK (Publicación de tesauros en servidores basados en SQL o en aplicaciones en Internet)

Estas tecnologías permiten a los creadores de tesauros, gracias a la gran cantidad de características que posee, incrementar su productividad, publicar tesauros en Internet/Intranet permitiendo a los usuarios navegar por el sistema ya sea sobre un servidor SQL o sin él [9].

La adquisición y uso de esta herramienta se hace difícil ya que la misma no es libre, hay que pagar por usar cualquiera de sus servicios; ya sea construcción y administración de tesauros o la publicación de los mismos. Además hay que pagar por el mantenimiento y soporte.

1.6.4 STRIDE

Es uno de los más potentes y versátiles sistemas de administración de tesauros existentes. Permite construir y mantener vocabularios para su uso en indización y obtención de información. Es simple de usar y provee respuestas rápidas cuando se accede a tesauros muy grandes. Soporta todas relaciones de los tesauros y permiten la creación de relaciones especiales propias [10].

Esta herramienta solo permite usar una versión de prueba o hay que suscribirse y usarlo bajos sus términos.

De manera general estos sistemas para la gestión de tesauros online presentan más limitaciones que ventajas para su uso, al menos en la Universidad de las Ciencias Informáticas (UCI) no sería conveniente su uso, pues los recursos a emplear en su utilización serían demasiados, díganse tiempo total perdido en su dominio debido a una curva de aprendizaje demasiado alta y pago de licencias de uso, además la falta de comunidades de desarrollo, el uso de software privativo y la falta de una capa de servicios que se integre con el sistema que los va a consumir son severas limitantes que llevan a la decisión de rechazarlas completamente**.**

Un enfoque más directo a las necesidades de este sistema, en cuanto a normalización de terminología se refiere, permite concluir que la mejor solución sería la implementación de un subsistema independiente que se ajuste a las necesidades del entorno donde será desplegado, permitiendo así una mayor personalización de los servicios y de las funciones que brindaría, y no el uso de una de las herramientas antes mencionadas, las cuales sí cumplen con los requisitos para ser usadas pero no satisfacen de manera explícita los requisitos del cliente, dígase integración con el sistema TeVeo Plus mediante una capa de servicios que permita a este último acceder a las funcionalidades de básicas del vocabulario como son obtener descriptores, sugerir un nuevo término y obtener los términos relacionados a un descriptor específico, ajustarse al hardware existente en el CIP (Centro de Informatización para la Prensa) y la facilidad de uso en cuanto a navegación al orientarlo a usuarios con poca experiencia en sistemas informáticos.

1.7 Tecnologías, herramientas y metodologías

1.7.1 Lenguajes utilizados

PHP 5

Es un lenguaje de programación utilizado para la creación de sitios web. PHP es un acrónimo recursivo que significa (*PHP Hypertext Pre-processor*). Surgió en 1995, desarrollado por PHP Group. Es un lenguaje interpretado en el lado servidor utilizado para la generación de páginas web dinámicas, embebidas en páginas HTML y ejecutadas en el servidor. No necesita ser compilado para ejecutarse. Para su funcionamiento necesita tener instalado Apache o IIS con las librerías de PHP. La mayor parte de su sintaxis ha sido tomada de C, Java y Perl con algunas características específicas.

PHP está diseñado específicamente para ser un lenguaje más seguro, y con la selección correcta de opciones de configuración en tiempos de compilación y ejecución y siguiendo algunas prácticas correctas de programación [11].

A continuación se relacionan algunas de sus características más relevantes:

- ✓ Es multiplataforma, puede ser utilizado sobre los sistemas operativos: GNU/Linux, Windows, entre otros.
- ✓ Funcionalidades de conexión con la mayoría de los sistemas de gestión de bases de datos: MySQL, PostgreSQL, Oracle, MS SQL Server, entre otras.
- ✓ Soporta la orientación a objeto y sus paradigmas.
- ✓ Su potencial se puede expandir utilizando extensiones.
- ✓ No requiere definición de tipos de variables ni manejo detallado del bajo nivel.
- ✓ Fácil de aprender.
- ✓ Posee una extensa documentación con ejemplos de uso de sus funciones.
- ✓ Es distribuido bajo licencia libre.
- ✓ La versión más reciente soportada es la 5.3 sobre la que se basan varios *frameworks* de desarrollo y sistemas de gestión de contenidos desarrolladas sobre este lenguaje.

XML

XML es un lenguaje orientado a identificar estructuras de datos en un documento. El término extensible en su propia definición, significa que a diferencia de HTML, en este tipo de documentos el usuario tiene la posibilidad de crear sus propias etiquetas, ordenar los datos, actualizarlos en tiempo real o realizar un pedido. Además, permite agrupar una variedad amplia de aplicaciones, desde páginas web hasta bases de datos; así como la gestión y manipulación de los datos desde el propio cliente web. Provee múltiples vistas para los datos. En efecto, los documentos XML definen el modelo de datos y luego se puede crear más de una vista basado en las necesidades de la aplicación que se está desarrollando [12].

JSON

JSON (*JavaScript Object Notation*) Notación de Objetos de JavaScript es un formato ligero de intercambio de datos. Leerlo y escribirlo es simple para humanos, mientras que para las máquinas es simple interpretarlo y generarlo. Está basado en un subconjunto del Lenguaje de Programación JavaScript. JSON es un formato de texto que es completamente independiente del lenguaje pero utiliza convenciones que son ampliamente conocidos por los programadores de la familia de lenguajes C, incluyendo C, C++, C#,

Java, JavaScript, Perl, Python, y muchos otros. Estas propiedades hacen que JSON sea un lenguaje ideal para el intercambio de datos [13].

JavaScript

JavaScript es un lenguaje de programación que se utiliza principalmente para crear páginas web dinámicas. Técnicamente, es un lenguaje de programación interpretado, por lo que no es necesario compilar los programas para ejecutarlos. En otras palabras, los programas escritos con *JavaScript* se pueden probar directamente en cualquier navegador sin necesidad de procesos intermedios. La ventaja básica de JavaScript consiste en su relativa sencillez, en tanto que el punto débil es que el código no es "seguro": estando intercalado en las mismas páginas, es perfectamente posible salvarlo y re-utilizarlo para otros fines [14]

CSS 3

Hojas de Estilo en Cascada (*Cascading Style Sheets*), es un mecanismo simple que describe cómo se va a mostrar un documento en la pantalla, o cómo se va a imprimir, o incluso cómo va a ser pronunciada la información presente en ese documento a través de un dispositivo de lectura. Esta forma de descripción de estilos ofrece a los desarrolladores el control total sobre estilo y formato de sus documentos. En el desarrollo de la aplicación fue utilizado CCS 3 [15].

HTML 5

HTML es el lenguaje de marcado predominante en el desarrollo de páginas web; se utiliza en la traducción y descripción de la estructura y la información en forma de texto, así como para complementar este con otros elementos [16].

HTML 5 es una actualización de HTML, el lenguaje en el que es creada la Web. También es un término de marketing para agrupar las nuevas tecnologías de desarrollo de aplicaciones web: HTML5, CSS3 y Javascript. Aunque aún es un lenguaje en desarrollo, los principales navegadores, en sus últimas versiones, reconocen muchos de los elementos que aporta. Entre sus características resalta la reducción de la necesidad de *plugins* externos y un mejor manejo de errores.

1.7.2 Framework de Desarrollo

Los *frameworks* son desarrollados con el objetivo de brindarles a los programadores y diseñadores una mejor organización y estructura a sus proyectos.

Symfony 2.1

Es un *framework* PHP basado en la arquitectura MVC (*Model-View-Controller*). Fue escrito desde un origen para ser utilizado sobre la versión 5 de PHP ya que hace ampliamente uso de la orientación a objetos que caracteriza a esta versión y desde la versión 2 de Symfony se necesita mínimamente PHP 5.3.3 [17].

Symfony puede ser utilizado para otros tipos de desarrollos no orientados a la Web, aunque fue diseñado para optimizar el desarrollo de aplicaciones web, proporcionando herramientas para agilizar aplicaciones complejas y guiando al desarrollador a acostumbrarse al orden y buenas prácticas dentro del proyecto.

Symfony se diseñó para que se ajustara a los siguientes requisitos:

- ✓ Fácil de instalar y configurar en la mayoría de plataformas.
- ✓ Independiente del sistema gestor de bases de datos.
- ✓ Sencillo de usar en la mayoría de casos, pero lo suficientemente flexible como para adaptarse a los casos más complejos.
- ✓ Basado en la premisa de “convenir en vez de configurar”, en la que el desarrollador solo debe configurar aquello que no es convencional.
- ✓ Sigue la mayoría de mejores prácticas y patrones de diseño para la Web.
- ✓ Preparado para aplicaciones empresariales y adaptables a las políticas y arquitecturas propias de cada empresa, además de ser lo suficientemente estable como para desarrollar aplicaciones a largo plazo.
- ✓ Código fácil de leer que incluye comentarios de phpDocumentor y permite un mantenimiento muy sencillo.
- ✓ Fácil de extender, lo que permite su integración con librerías desarrolladas por terceros.

Bootstrap 2.0

Bootstrap es un *framework* que simplifica el proceso de creación de diseños web combinando CSS y JavaScript. Ha sido desarrollado por Twitter que recientemente liberó su versión 2.0. La mayor ventaja es

que permite crear interfaces que se adapten a los distintos navegadores (*responsive design*) gracias a que es un *framework* potente con numerosos componentes web, se ahorrarán mucho esfuerzo y tiempo [18].

Características principales de Bootstrap:

- ✓ Ofrece una serie de plantillas CSS y ficheros Javascript que permiten integrar el *framework* de forma sencilla y potente en los proyectos web.
- ✓ Permite crear interfaces que se adapten a los diferentes navegadores, tanto de escritorio como *tablets* y móviles a distintas escalas y resoluciones.
- ✓ Se integra perfectamente con las principales librerías Javascript, por ejemplo JQuery.
- ✓ Ofrece un diseño sólido usando LESS y estándares como CSS3/HTML5.
- ✓ Es un *framework* ligero que se integra de forma limpia a cualquier proyecto.
- ✓ Funciona con todos los navegadores, incluido Internet Explorer usando HTML Shim para que reconozca los *tags* HTML5.
- ✓ Dispone de distintos *layout* predefinidos con estructuras fijas a 940 píxeles de distintas columnas o diseños fluidos.

1.7.3 Sistema Gestor de Bases de Datos

El software que permite la utilización y/o la actualización de los datos almacenados en una o varias bases de datos por uno o varios usuarios desde diferentes puntos de vista y a la vez, se denomina Sistema Gestor de Bases de Datos (SGBD). Su principal objetivo es suministrar al usuario herramientas que le permitan manipular, en términos abstractos, los datos, o sea de forma que no le sea necesario conocer el modo de almacenamiento de los datos en la computadora, ni el método de acceso empleado.

Las principales funciones que debe cumplir un SGBD se relacionan con la creación y mantenimiento de la base de datos, el control de accesos, la manipulación de datos de acuerdo con las necesidades del usuario, el cumplimiento de las normas de tratamiento de datos, evitar redundancias e inconsistencias y mantener la integridad [19].

MongoDB 2.0.4

Es un sistema de base de datos NoSQL orientado a documentos, desarrollado bajo el concepto de código abierto. MongoDB forma parte de la nueva familia de sistemas de base de datos NoSQL. En vez de

guardar los datos en tablas como se hace en las base de datos relacionales, MongoDB guarda estructuras de datos en documentos tipo JSON con un esquema dinámico (MongoDB llama ese formato BSON), haciendo que la integración de los datos en ciertas aplicaciones sea más fácil y rápida [20].

Algunas de las características de este sistema:

- ✓ Almacenamiento orientado a documentos.

  - Documentos estilo JSON con esquemas dinámicos ofrecen simplicidad y poder.

- ✓ Soporte Full índex.

  - Índices sobre cualquier atributo.

- ✓ Replicación y alta disponibilidad.

  - Espejos entre LANs y WANs.

- ✓ Auto-Sharding.

  - Escalabilidad horizontal sin comprometer la funcionalidad.

- ✓ Consultas.

  - Ricas y basadas en documentos, rápidas para consultas de lectura y escritura básica.

- ✓ Indexación.

  - Cualquier campo en un documento de MongoDB puede ser indexado, al igual que es posible hacer índices secundarios. El concepto de índices en MongoDB es similar a los encontrados en base de datos relacionales.

- ✓ Mapeo y reducción.

  - Agregación flexible y procesamiento de datos.

- ✓ Soporta consultas dinámicas.

  - Se pueden formular sobre cualquier valor de los documentos y no solamente en los indexados.

Otras de las características importantes de MongoDB

- ✓ Presenta interfaz para varios lenguajes programación como C++, Java, PHP, C# entre otros
- ✓ Utiliza un fichero asignado en memoria (ocupa una porción de la memoria virtual) como mecanismo de almacenamiento y consulta de la información. Lo anterior hace que delegue gran parte del manejo de la memoria al sistema operativo donde se ejecuta, por lo que el código de MongoDB para gestionar la memoria es pequeño y claro. Las operaciones de entrada y salida son mucho más rápidas con esta variante utilizada por MongoDB.
- ✓ Cuenta con una comunidad estable y amplia, de experiencia en el desarrollo con la herramienta que brinda un espacio para consultar y solucionar los problemas que se presentan en el desarrollo con MongoDB.
- ✓ Mantiene algunas características de SQL que permiten que los usuarios que desarrollan con MongoDB luego de haberlo hecho con SQL perciban de un modo mucho más sencillo el cambio de paradigma.

1.7.4 Metodologías de desarrollo de software

RUP

RUP, (del inglés *Rational Unified Process)*, es un producto comercial desarrollado y comercializado por *Rational Software*, una compañía de IBM. El proceso de software propuesto por RUP tiene tres características esenciales, es un proceso dirigido por casos de uso, centrado en la arquitectura y sigue una secuencia iterativa e incremental. RUP divide el proceso en cuatro fases, dentro de las cuales se realizan varias iteraciones en número variable según el proyecto, y en las que se hace un mayor o menor hincapié en los distintas actividades.

Los aspectos que caracterizan a RUP se puede decir que cuenta con una forma disciplinada de asignar tareas y responsabilidades, que pretende implementar mejores prácticas, propone un desarrollo iterativo y el uso de la arquitectura basada en componentes, además de un riguroso control de cambios, se basa en el modelado visual y le da gran importancia a la verificación de la calidad del software. RUP se enfatiza en el uso exhaustivo de documentación durante todo el ciclo de vida del proyecto y es recomendada para los proyectos con grandes equipos de desarrollo [21].

OpenUP

OpenUP es una metodología que contiene el conjunto mínimo de prácticas que ayudan a los equipos a ser más eficaces en el desarrollo de software. OpenUP abraza una filosofía pragmática y ágil que se centra en la naturaleza colaborativa de desarrollo de software. Es un proceso iterativo que es Mínimo, Completo y Extensible que puede utilizarse tal cual o ampliarse para tratar una amplia variedad de tipos de proyecto. Se caracteriza por ser iterativo e incremental, estar centrado en la arquitectura y guiado por los casos de uso. Está organizada dentro de cuatro áreas principales de contenido: Comunicación y Colaboración, Intención, Solución y Administración.

OpenUP se caracteriza por cuatro principios básicos que se soportan mutuamente:

- ✓ Colaboración para alinear los intereses y un entendimiento compartido.
- ✓ Balance para confrontar las prioridades (necesidades y costos técnicos) para maximizar el valor para los *stakeholders* [22].

Ventajas de usar OpenUP

- ✓ Metodología de desarrollo de software de código abierto diseñado para pequeños equipos organizados quienes quieren tomar una aproximación ágil del desarrollo.
- ✓ Proceso iterativo e incremental que es Mínimo, Completo y Extensible donde practicantes de desarrollo de software (desarrolladores, administradores de proyectos, analistas y probadores) trabajan juntos como un equipo de proyecto.
- ✓ Permite detectar errores tempranos a través de un ciclo iterativo.
- ✓ Evita la elaboración de documentación, diagramas e iteraciones innecesarios requeridos en la metodología RUP.
- ✓ Por ser una metodología ágil tiene un enfoque centrado al cliente y con iteraciones cortas [22].

1.7.5 Arquitectura de Información

ForeUI

ForeUI permite diseñar prototipos de interfaces web o de aplicaciones mediante la creación de maquetas de imágenes para la visualización de las ideas. La aplicación ofrece un conjunto de componentes *drag* y *drop* que incluyen botones, menús, pestañas, barras de desplazamiento, combo box y otros elementos comunes, además de herramientas gráficas para insertar formas, líneas, textos, imágenes y mucho más.

Se pueden utilizar estos elementos para diseñar rápidamente una GUI estándar y añadir notas o comentarios al diseño. ForeUI puede renderizar maquetas utilizando Windows XP, Mac OS X o temas

Wireframe para luego exportarlo como diseño final en formato de imagen PNG. Otras características incluyen soporte para capas, niveles de opacidad y una librería de iconos integrada [23].
Es desarrollado sobre Java, por lo que es multiplataforma. Cuenta con una licencia privativa. La versión más reciente es la 2.8.

1.7.6 Sistemas para la indización y búsqueda

ElasticSearch

ElasticSearch es un servidor de búsqueda distribuido RESTful libre y de código abierto basado en Apache Lucene. Está hecho en Java bajo los términos de Apache License [24].

Lucene

Es una API de código abierto para recuperación de información, originalmente implementada en Java por Doug Cutting. Está apoyado por el Apache Software Foundation y se distribuye bajo la Apache Software License. Lucene tiene versiones para otros lenguajes incluyendo Delphi, Perl, C#, C++, Python, Ruby y PHP.
Es útil para cualquier aplicación que requiera indexado y búsqueda a texto completo. Lucene ha sido ampliamente usado por su utilidad en la implementación de motores de búsquedas. Por ello, a veces se confunde Lucene con un motor de búsquedas con funciones de "*crawling*" y análisis de documentos en HTML incorporadas.
El centro de la arquitectura lógica de Lucene se encuentra el concepto de Documento (*Document*) que contiene Campos (*Fields*) de texto. Esta flexibilidad permite a Lucene ser independiente del formato del fichero. Textos que se encuentran en PDFs, páginas HTML, documentos de Microsoft Word, así como muchos otros pueden ser indexados mientras que se pueda extraer información de ellos [24].

1.7.7 Herramienta CASE

Visual Paradigm para UML 8.0

Visual Paradigm para UML es una herramienta profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño, construcción, pruebas y despliegue. Garantiza una rápida construcción de aplicaciones con una mayor calidad, y a un menor coste. Permite dibujar todos los tipos de diagramas de clases, generar código desde diagramas y generar documentación. La herramienta UML CASE también proporciona abundantes tutoriales, demostraciones interactivas y proyectos UML.

Es una herramienta multiplataforma distribuida bajo licencia privativa en la Visual Paradigm Suite. Es orientada a objetos y se puede integrar con Eclipse, Netbeans, IntelliJ IDEA entre otras herramientas de desarrollo.

Proporciona un entorno ágil y eficiente para el diseño. Permite el diseño de base de datos y el diseño del sistema con el diagrama de clases UML. También apoya la generación de bases de datos, Java e Hibernate, así como técnicas de ingeniería inversa de bases de datos. Modelado colaborativo con CVS y Subversion permite el trabajo en equipo, puede soportar la edición simultánea de un mismo proyecto o diagrama. En Visual Paradigm Suite 5.0 se distribuye la versión 8.0 la más reciente de Visual Paradigm para UML. El uso de esta herramienta trae consigo varios beneficios [25].

Beneficios:

1. Navegación intuitiva entre el modelo visual y el código.
2. Poderosa herramienta de generación de PDF/HTML a partir de diagramas UML.
3. Sincronización entre el código fuente y el modelo en tiempo real o bajo demanda.
4. Entorno visual de modelado superior.
5. Soporte para toda la notación UML.
6. Sofisticados y automáticos diagramas de capas.
7. Diseño centrado en casos de uso y enfocado al negocio que genera un software de mayor calidad.
8. Uso de un lenguaje estándar común a todo el equipo de desarrollo que facilita la comunicación.
9. Capacidades de ingeniería directa (versión profesional) e inversa.
10. Modelo y código que permanece sincronizado en todo el ciclo de desarrollo.
11. Disponibilidad de integrarse en los principales IDEs (*Integrated Development Environment*).
12. Disponibilidad en múltiples plataformas.

1.7.8 Entorno Integrado de desarrollo

Un entorno de desarrollo integrado (IDE) es un programa compuesto por una serie de herramientas que utilizan los programadores para desarrollar código. Esta herramienta puede estar pensada para su utilización con un único lenguaje de programación o bien puede dar cabida a varios de estos.

NetBeans 7.3

Es un Entorno de Desarrollo Integrado gratuito y de código abierto para el desarrollo de aplicaciones web, de escritorio y para móviles, disponible en los sistemas operativos Windows, Linux, Mac OS X y Solaris. La base en la que se sustenta su elección es que permite desarrollar aplicaciones utilizando el *framework* Symfony y ejecutar los comandos del mismo directamente desde la interfaz del IDE. NetBeans IDE es de

fácil instalación y multiplataforma ya que se ejecuta en Windows y GNU/Linux. Las funcionalidades de NetBeans son extensibles mediante *plugins*, por ejemplo existen extensiones para la creación de aplicaciones web, para la creación de módulos para Drupal y para la creación, modificación y compilación de módulos (AMP) para el ECM Alfresco WingIDE [26].

1.7.9 Estilo de Arquitectura

REST

REST se refiere estrictamente a una colección de principios para el diseño de arquitecturas en red. Estos principios resumen cómo los recursos son definidos y diseccionados. El término frecuentemente es utilizado en el sentido de describir a cualquier interfaz que transmite datos específicos de un domino sobre HTTP sin una capa adicional [27].

Éste estilo arquitectural no está limitado al desarrollo de aplicaciones para el uso directo por personas. Actualmente se está utilizando más y más en el desarrollo de arquitecturas orientadas a servicios utilizando servicios web destinados a la comunicación entre máquinas. REST supone una alternativa al estilo arquitectural *Remote Procedure Call* (RPC).

Las respuestas de los *requests* a diferencia de SOAP y XML-RPC no han de ser obligatoriamente XML. Las respuestas en formato JSON u objetos de Java Serializados son perfectamente aceptables [28].

Aunque REST no es un estándar está basado en los estándares siguientes:

- ✓ HTTP
- ✓ URL
- ✓ Representación de los recursos: XML/HTML/GIF/JPEG/…
- ✓ Tipos MIME: text/xml, text/html.

1.8 Conclusiones Parciales

- ✓ Se realizó un análisis de la situación actual de las herramientas que hacen posible la realización de los vocabularios controlados llegando a la conclusión de que no cumplen con las necesidades requeridas, por lo tanto no pueden ser utilizadas.

- ✓ Los fundamentos teóricos de la investigación realizada evidenciaron la necesidad de desarrollar una aplicación que permita la gestión de la representación del contenido en el sistema de gestión documental de audio y vídeos digitales TeVeo Plus V1.0.

- ✓ En este capítulo se profundizó en algunos conceptos relacionados con la problemática en cuestión los cuales sentaron las bases necesarias para el inicio de la implementación del vocabulario controlado.

- ✓ Después de estudiar una serie de tecnologías, herramientas y metodologías se concluye que las más idóneas para el desarrollo del sistema son: como *framework* de desarrollo Symfony 2.1 y Bootstrap 2.0, como IDE de desarrollo Netbeans 7.3, MongoDB como SGBD, como servidor web Apache 2 y ElasticSearch como servidor de búsqueda. Como herramienta CASE Visual Paradigm 8 y como lenguaje de modelado UML 2.

CAPÍTULO 2: CARACTERÍSTICAS Y DISEÑO DEL SISTEMA

En el presente capítulo se expresan las características principales del sistema a desarrollar. Se definen los requerimientos del sistema (requisitos funcionales y no funcionales) con que debe cumplir el mismo. Se describe el flujo de trabajo a partir del modelado por procesos. Se realizan una serie de diagramas que permiten entender la lógica del sistema, entre ellos podemos mencionar el de Caso de Uso del Sistema (CU), el de Secuencia y el de Despliegue. Además, se hace una descripción de los estilos arquitectónicos y los patrones de diseño.

2.1 Descripción del sistema

El Vocabulario Controlado constituye una herramienta básica para la representación del contenido y forma, y el control terminológico de la información publicada en los medios masivos de comunicación en la Universidad de las Ciencias Informáticas.

El Vocabulario Controlado está compuesto por palabras o grupos de palabras que identifican el contenido y la forma del documento, los cuales son denominados términos. Cada concepto sólo puede ser representado por un término. Los conceptos se relacionan sobre bases lógicas, ontológicas, funcionales y secuenciales, uniendo los términos entre sí. El término puede ser un descriptor o un no descriptor.

El sistema brindará la posibilidad directa de gestionar tantos vocabularios controlados, es decir las categorías donde estarán agrupados los términos según algún tipo de relación establecida entre ellos, descriptores, no descriptores y sugerencias. También, a través de servicios web usando la arquitectura para la comunicación entre sistemas **RE**presentational **S**tate **T**ransfer (REST), se podrán realizar acciones básicas para la gestión de términos; en pleno uso del sistema de gestión documental TeVeo Plus se podrán realizar acciones tales como obtener todos los descriptores en los formatos XML y JSON, sugerir un nuevo término a un vocabulario controlado específico y dado un término específico obtener todos los términos relacionados a él.

Se usará la arquitectura REST para la publicación de los servicios web pues esta posee ventajas que la hicieron la candidata ideal para la construcción de los mismos [28]:

- ✓ La aplicación es más simple de mantener, los vínculos son más estructurados y su forma es más común. Es también más común que los vínculos sean el modo de llevar el estado de una aplicación web.

- ✓ A ausencia de gestión de estado de cliente en el servidor conduce un consumo más eficiente de la memoria, una mayor simplificación de la funcionalidad y una capacidad más grande de gestionar un gran número de *requests* al mismo tiempo.

- ✓ La ausencia de la gestión de estado del cliente en el servidor permite implementar un balance de carga más simple: una sesión del cliente no tiene que ser mantenida por un servidor en particular, ni propagada a otros servidores. Ésta permite también una mejor evolución y tolerancia a fallos.

- ✓ La utilización de protocolo HTTP y todas las partes de sus encabezamientos (la mayor parte de la gramática HTTP).

2.2 Análisis de la Estructura Organizativa

A continuación se describen los procesos que se realizan en el Vocabulario Controlado y los roles que intervienen en cada uno de ellos; los cuales se modelarán seguidamente a través de los diagramas de procesos, para una mejor personalización del sistema propuesto.

2.2.1 Roles

Administrador: Es el encargado de la gestión de los usuarios y la administración del sistema.

Anónimo: Usuario que no está autenticado en el sistema

Colaborador: Usuario que tiene como objetivo fundamental sugerir los descriptores.

Gestor: Es el responsable de la gestión de los contenidos en el vocabulario controlado.

2.2.2 Procesos

- ✓ **Adición:** Tiene como objetivo sugerir términos los cuales serán almacenados en una lista de sugerencias.

- ✓ **Revisión/Aprobación:** Permite la revisión de las sugerencias y después de un análisis se decide su aprobación o no.

- ✓ **Almacenamiento:** Su objetivo es almacenar los términos según su tipo (Descriptor o No Descriptor).
- ✓ **Visualización:** Permite la visualización y edición de los términos y sus propiedades.

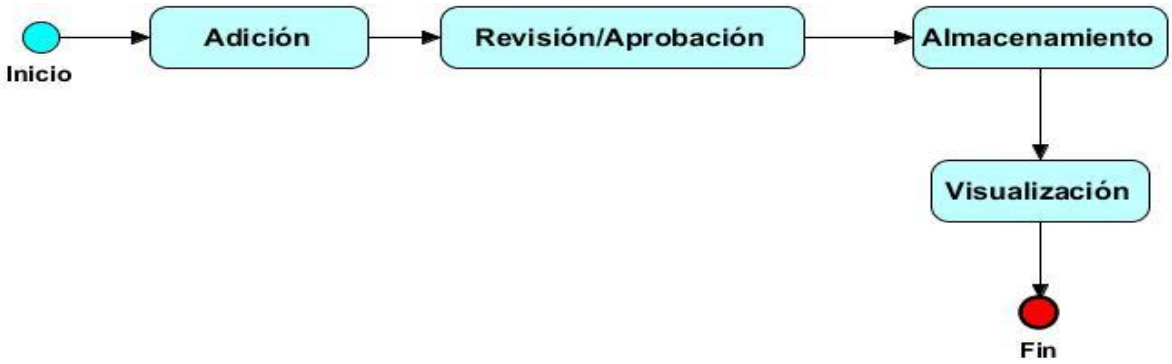


Figura 5. Diagrama de procesos (General).

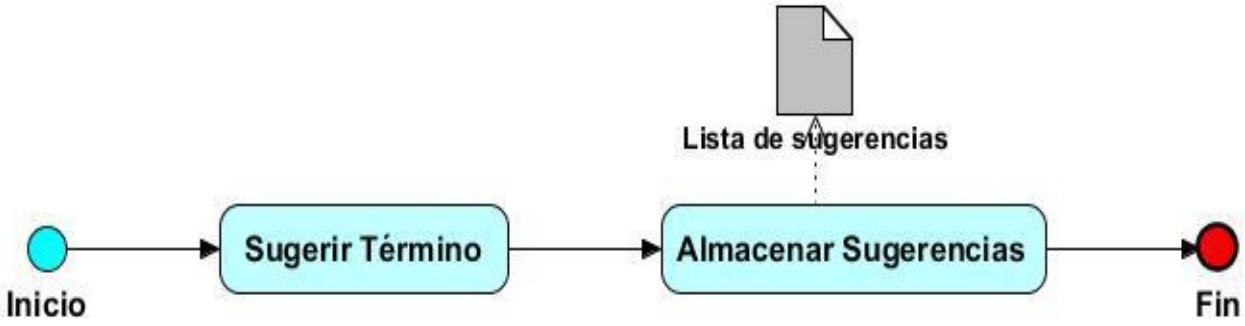


Figura 6. Diagrama de procesos (Adición).

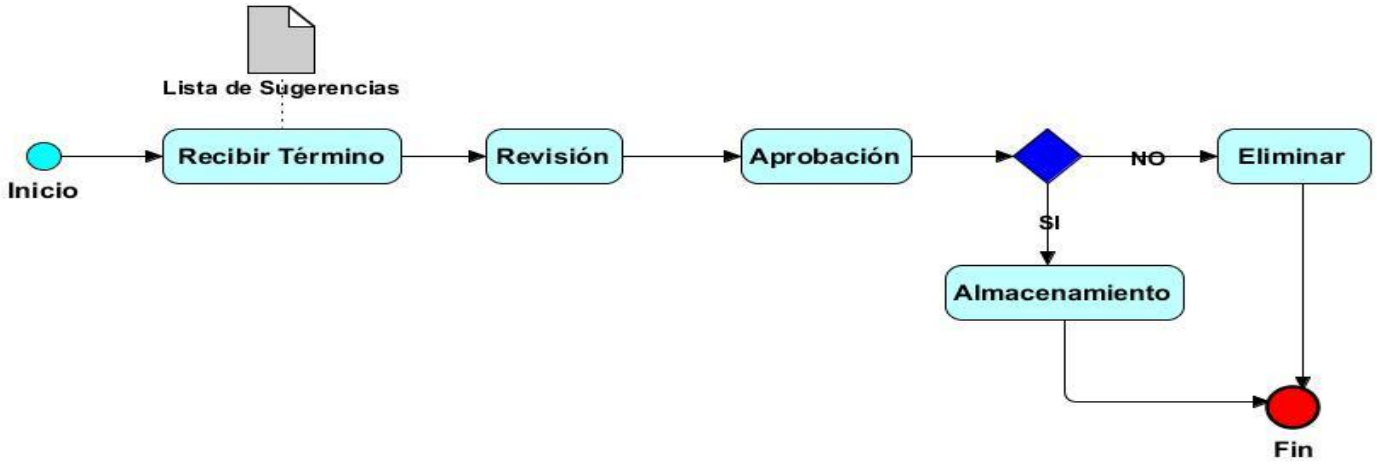


Figura 7. Diagrama de procesos (Revisión/Aprobación).

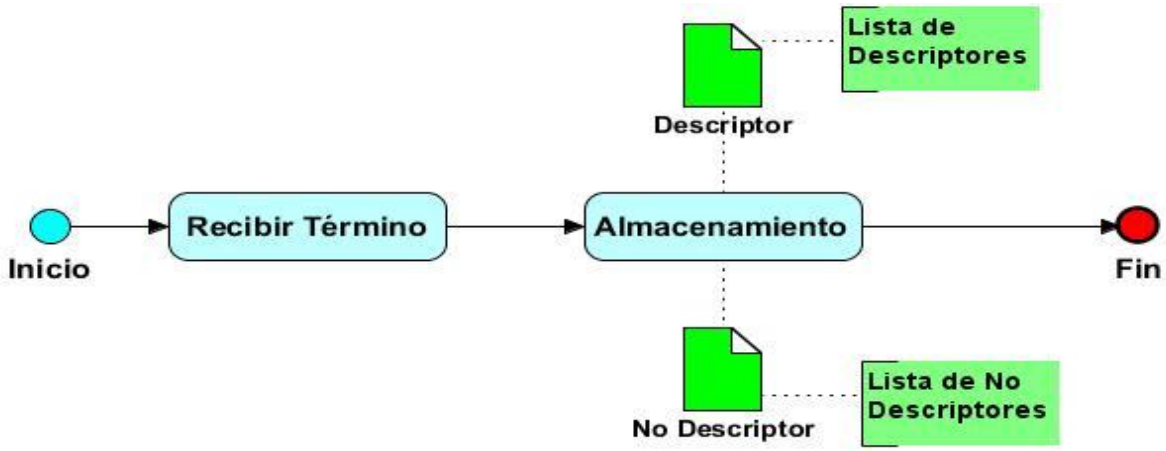


Figura 8. Diagrama de procesos (Almacenamiento).

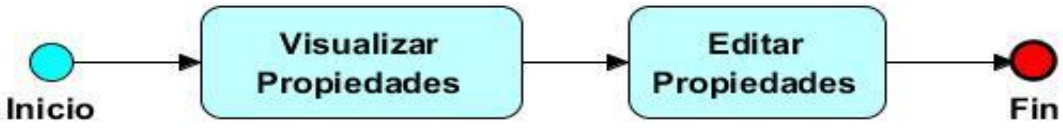


Figura 9. Diagrama de procesos (Visualización).

2.3 Requerimientos del sistema

Las condiciones que el sistema debe cumplir o capacidad que debe tener con el objetivo de establecer un entendimiento común entre el usuario y el proyecto de software son los requerimientos. Los requerimientos se clasifican en requerimientos funcionales y no funcionales [29].

Requerimientos funcionales

Los requerimientos funcionales son la determinación exacta de qué debe ser capaz de hacer el sistema, éstas se corresponden con opciones que ejecutará el software, operaciones realizadas de forma oculta o condiciones extremas a determinar por el sistema [29].

Requisitos Funcionales	Prioridad
RF1_Autenticar usuario	Alta
RF2_Adicionar usuario	Alta
RF3_Editar usuario	Alta
RF4_Eliminar usuario	Alta
RF5_Deshabilitar cuenta de usuario	Alta
RF6_Auditar cuenta de usuario	Media
RF7_Filtrar cuentas de usuarios	Alta
RF8_Listar cuentas de usuarios	Alta
RF9_Solicitar cuentas de usuarios	Media
RF10_Realizar búsqueda simple	Media
RF11_Realizar búsqueda avanzada	Media
RF12_Salvar criterio de la búsqueda	Media
RF13_Visualizar servicios web	Media
RF14_Configurar vocabulario	Media
RF15_Adicionar contenidos	Alta
RF16_Editar contenidos	Alta
RF17_Eliminar contenidos	Alta
RF18_Aprobar contenidos	Alta
RF19_Sugerir descriptor	Alta
RF20_Visualizar lista de descriptores	Alta
RF21_Importar descriptores	Media

RF22_Exportar descriptores	Media
RF23_Visualizar lista de no descriptores	Alta
RF24_Visualizar lista de vocabularios controlados	Alta
RF25_Eliminar notificaciones	Baja
RF26_Visualizar notificaciones	Baja
RF27_Ayuda Online	Baja
RF28_Contáctanos	Media
RF29_Visualizar ruta de navegación	Baja

Tabla 1. Requisitos funcionales.

Requisitos No Funcionales

Los requerimientos no funcionales son propiedades o cualidades que el producto debe tener y que de una u otra forma puedan limitar el sistema. Debe pensarse en estas propiedades como las características que hacen al producto atractivo, usable, rápido o confiable. Normalmente están vinculados a requerimientos funcionales [29].

Apariencia e Interfaz externa:

- ✓ Se requiere una interfaz sencilla, amigable, de apariencia intuitiva. Desde una perspectiva más amplia del diseño visual de la aplicación, debe mantener una coherencia y estilo común entre todas las ventanas, proporcionando una consistencia visual a la aplicación.

Requerimiento de Seguridad:

- ✓ Autentificación SSL.
- ✓ Sesión única.

Requerimiento de Rendimiento:

- ✓ El sistema debe ser rápido y el tiempo de respuesta debe ser el mínimo posible, de 1 segundo.

Requerimiento de Software:

- ✓ Como sistema operativo Ubuntu 12.04.
- ✓ Como gestor de base de datos: MongoDB.
- ✓ El servidor de aplicación: Apache.
- ✓ Para el desarrollo de la aplicación se utilizará el *framework* Symfony 2.1.
- ✓ Navegador web: Mozilla Firefox, Chrome.

Requerimiento de Hardware:

- ✓ Se necesitan 2 servidores Core I3 con 2GB de RAM, de 3.0 GHz o superior. Se requiere de 160 GB de espacio de disco duro.

Requerimiento de Diseño e implementación:

- ✓ Como gestor de base de datos: MongoDB
- ✓ El servidor de aplicaciones: Apache
- ✓ El lenguaje de programación utilizado: PHP, utilizando Symfony como *framework* de desarrollo.

Requerimiento de Usabilidad.

- ✓ Garantizar la confidencialidad de los datos mediante el cierre automático de la sesión transcurrido un tiempo de 15 minutos.

2.4 Diagrama de Casos de Uso del Sistema

Los casos de uso del sistema son una técnica para especificar el comportamiento del software y se parte de la identificación de los requerimientos del sistema. El modelo de casos de uso describe lo que hace el sistema para el usuario; la forma en que éste usa el sistema se representa con casos de uso, derivados de los requisitos funcionales que los relacionan. La representación de cada caso de uso facilita especificar la secuencia de acciones que el sistema puede llevar a cabo [30]

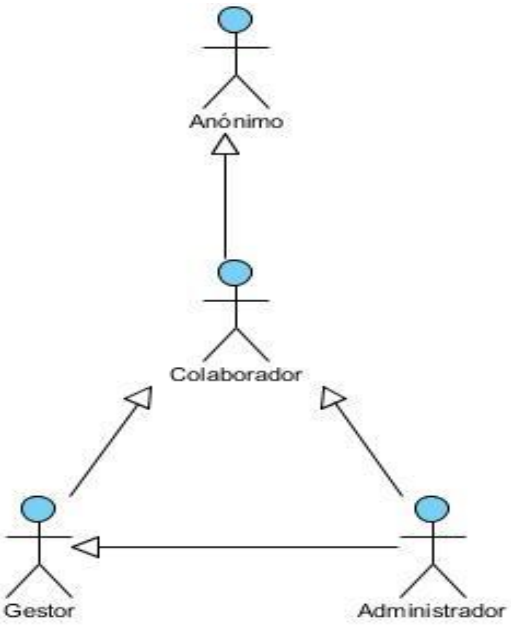


Figura 10. Caso de Uso por actores.

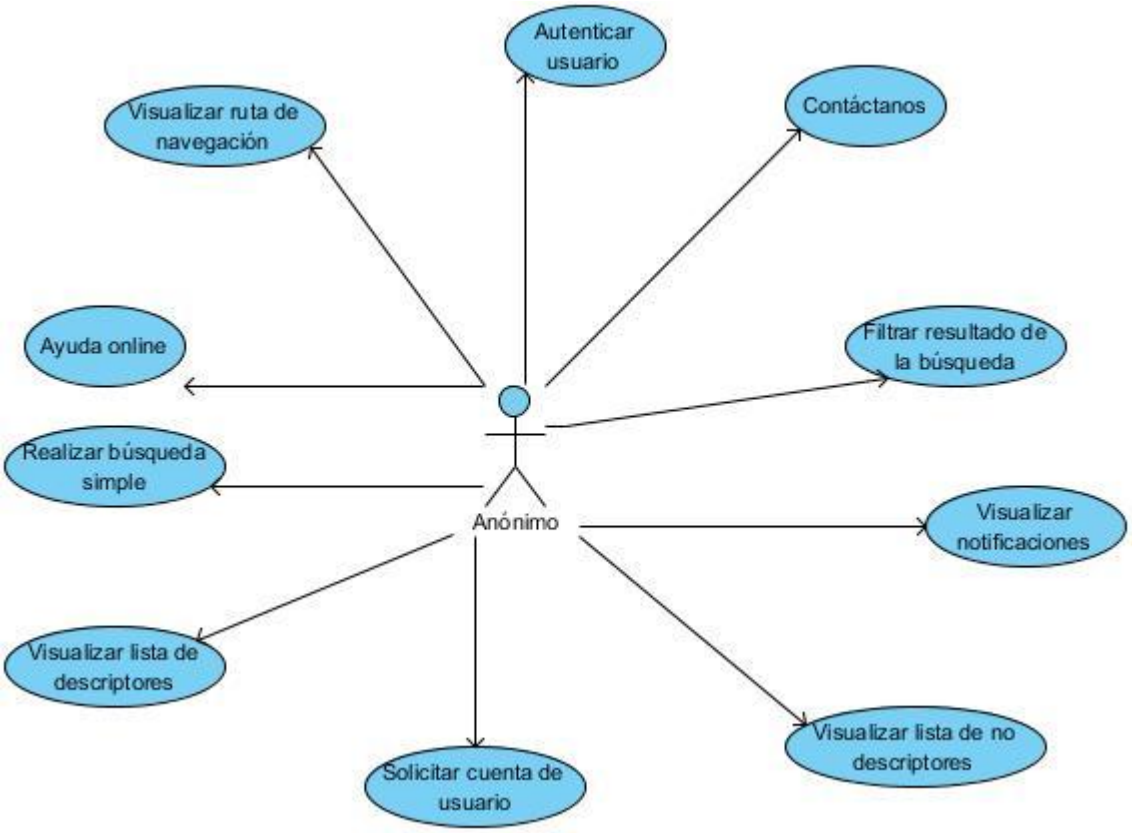


Figura 11. Diagrama de Caso de Uso específico por cada rol (Anónimo).

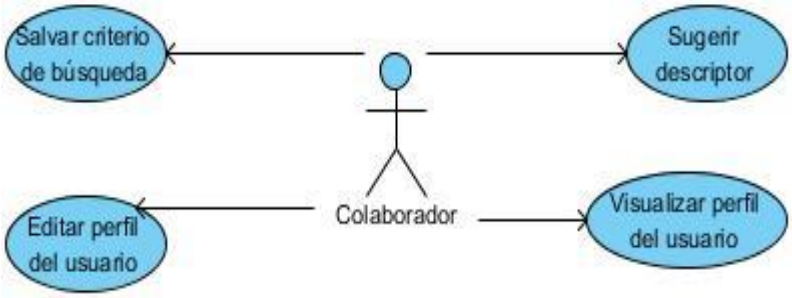


Figura 12. Diagrama de Caso de Uso específico por cada rol (Colaborador).

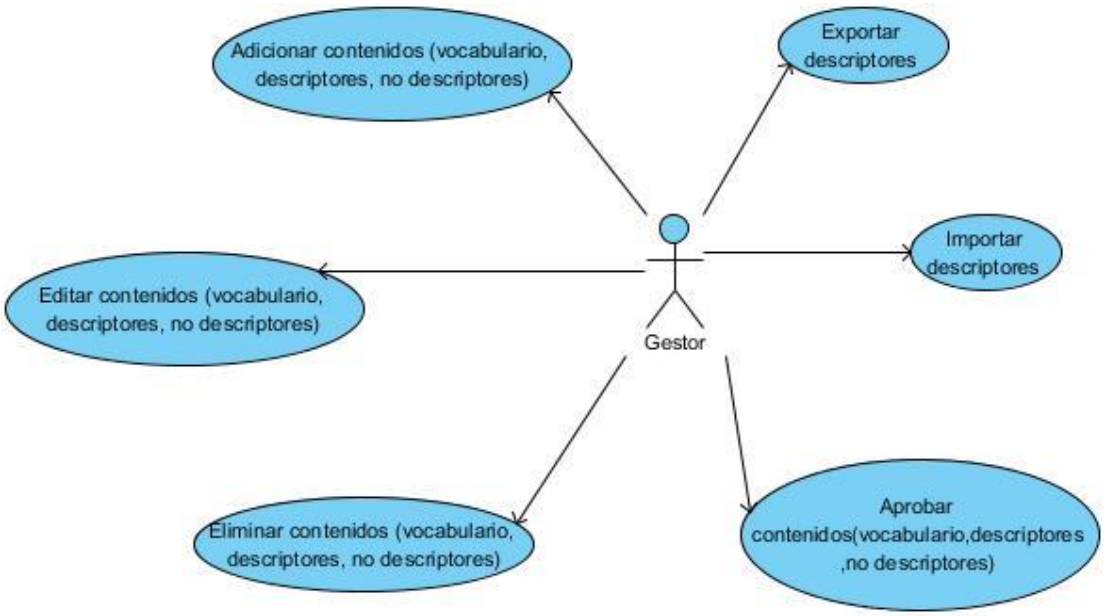


Figura 13. Diagrama de Caso de Uso específico por cada rol (Gestor).

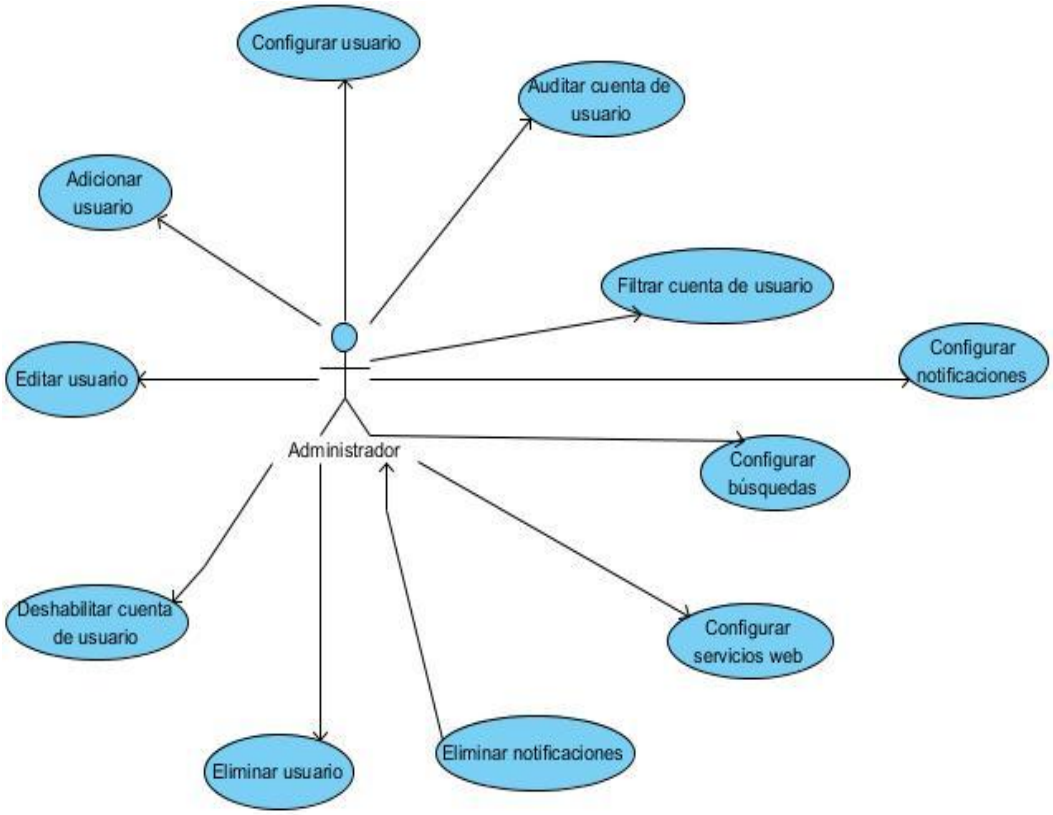


Figura 14. Diagrama de Caso de Uso específico por cada rol (Administrador).

Con el objetivo de detallar las relaciones entre los casos de usos y los actores, y cómo juntos constituyen el modelo de casos de uso, se desarrollan las descripciones de los principales casos de uso.

CU 1. Autenticar usuario

Objetivo	Autenticación en el sistema.
Actores	Administrador: (Inicia) se autentica con su usuario y contraseña en el sistema. Colaborador: (Inicia) se autentica con su usuario y contraseña en el sistema. Gestor: (Inicia) se autentica con su usuario y contraseña en el sistema.
Resumen	El CU permite el acceso al sistema mediante la inserción de usuario y contraseña según el rol.
Complejidad	Alta
Prioridad	
Precondiciones	Los datos introducidos han sido validados.

Post-condiciones		
Flujo de eventos		
Flujo básico “Autenticar Usuario”		
	Actor	**Sistema**
1.	Introduce datos de autenticación.	Valida que los datos están completos. Valida que los datos sean correctos.
2.		Permite acceso al sistema.
1- Los datos son incorrectos		
	Actor	**Sistema**
1.	Introduce usuario o contraseña incorrecta.	Muestra mensaje de error.
2.		Termina el caso de uso.
3.		
2- Los campos están vacíos.		
	Actor	**Sistema**
1.	Deja campos vacíos.	Muestra mensaje de error.
2.		Termina el caso de uso.

Tabla 2. Descripción del caso de uso Autenticar usuario.

Objetivo	Realizar búsqueda simple.
Actores	Administrador: (Inicia) Realiza una búsqueda simple. Anónimo: (Inicia) Realiza una búsqueda simple. Colaborador: (Inicia) Realiza una búsqueda simple. Gestor: (Inicia) Realiza una búsqueda simple.
Resumen	El CU permite realizar una búsqueda simple sobre los contenidos del sistema.
Complejidad	Alta
Prioridad	Crítico
Precondiciones	-
Post-condiciones	Resultados de la búsqueda disponibles.
Flujo de eventos	
Flujo básico “Realizar búsqueda simple”	
Actor	**Sistema**

1.	Selecciona el/los tipos de contenido sobre los que desea realizar la búsqueda e introduce el criterio para buscar.	Valida que el campo del criterio de búsqueda no esté vacío.
2.		Ejecuta la búsqueda y muestra los resultados.
3.		Termina el caso de uso.
Flujos alternos		
1. Se dejaron campos vacíos.		
	Actor	**Sistema**
1.	Deja el campo de criterio de búsqueda sin rellenar.	Muestra mensaje de error.
2.		Termina el caso de uso.

Tabla 3. Descripción del caso de uso Realizar búsqueda simple.

2.5 Estilos arquitectónicos y patrones de diseño

2.5.1 Patrón MVC

Symfony está basado en un patrón clásico del diseño web conocido como MVC (Modelo Vista Controlador). Este patrón nos permite y obliga a separar la lógica de control (sabe qué cosas hay que hacer pero no cómo), la lógica de negocio (sabe cómo se hacen las cosas) y la lógica de presentación (sabe cómo interactuar con el usuario). Está organizado en tres modelos separados.

- ✓ El Modelo representa la información con la que trabaja la aplicación, es decir, su lógica de negocio.
- ✓ La Vista transforma el modelo en una página web que permite al usuario interactuar con ella.
- ✓ El Controlador se encarga de procesar las interacciones del usuario y realiza los cambios apropiados en el modelo o en la vista [31].

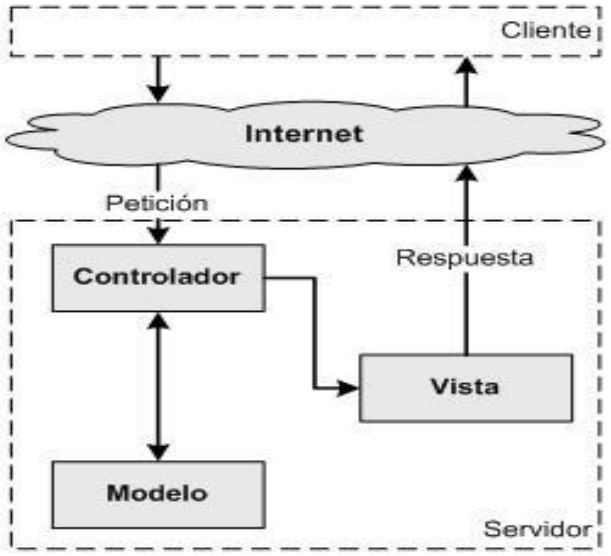


Figura 15. Patrón MVC.

2.5.2 Patrones de diseño

Los patrones de diseño proponen una forma de reutilizar la experiencia de los desarrolladores, para ello clasifican y describen formas de solucionar problemas que ocurren de forma frecuente en el desarrollo. Por tanto, están basados en la recopilación del conocimiento de los expertos en desarrollo de software [32].

Entre los patrones utilizados por Symfony se encuentran los siguientes:

Controlador (GRASP)

El patrón controlador es un patrón que sirve como intermediario entre una determinada interfaz y el algoritmo que la implementa, de tal forma que es la que recibe los datos del usuario y la que los envía a las distintas clases según el método llamado.

Este patrón sugiere que la lógica de negocios debe estar separada de la capa de presentación, esto para aumentar la reutilización de código y a la vez tener un mayor control. Se recomienda dividir los eventos del sistema en el mayor número de controladores para poder aumentar la cohesión y disminuir el acoplamiento [33].

Experto (GRASP)

Permite asignar una responsabilidad al experto en información: la clase que cuenta con la información necesaria para cumplir la responsabilidad [33].

Alta Cohesión (GRASP)

Symfony permite la organización del trabajo en cuanto a la estructura del proyecto y la asignación de responsabilidades con una alta cohesión. Un ejemplo de ello es la clase *Actions,* la cual está formada por varias funcionalidades que están estrechamente relacionadas, siendo esta la responsable de definir las acciones para las plantillas y colaborar con otras para realizar diferentes operaciones, instanciar objetos y acceder a las propiedades [33].

Decorador (GoF)

Permite añadir responsabilidades adicionales a un objeto dinámicamente, proporcionando una alternativa flexible a la especialización mediante herencia, cuando se trata de añadir funcionalidades [33].

En Symfony 2 la vista se separa en una plantilla base y varias plantillas que heredan de esta. Normalmente, la plantilla base es global en toda la aplicación y contiene el código HTML que es común en la mayoría de las páginas, lo cual es una implementación del patrón decorador. Su uso aporta una mayor flexibilidad que la herencia estática, permitiendo, entre otras cosas, añadir una funcionalidad dos o más veces.

Además evita concentrar en lo alto de la jerarquía clases “guiadas por las responsabilidades”, es decir, que pretenden satisfacer todas las posibilidades. De esta forma las nuevas funcionalidades se componen de piezas simples que se crean y se combinan con facilidad [31].

2.6 Diagramas de Clases del Diseño

Un diagrama de clases muestra un conjunto de clases e interfaces, así como sus relaciones. Los diagramas de clases se utilizan para modelar la vista de diseño estática de un sistema, esto incluye modelar el vocabulario del sistema, modelar las colaboraciones o modelar esquemas.

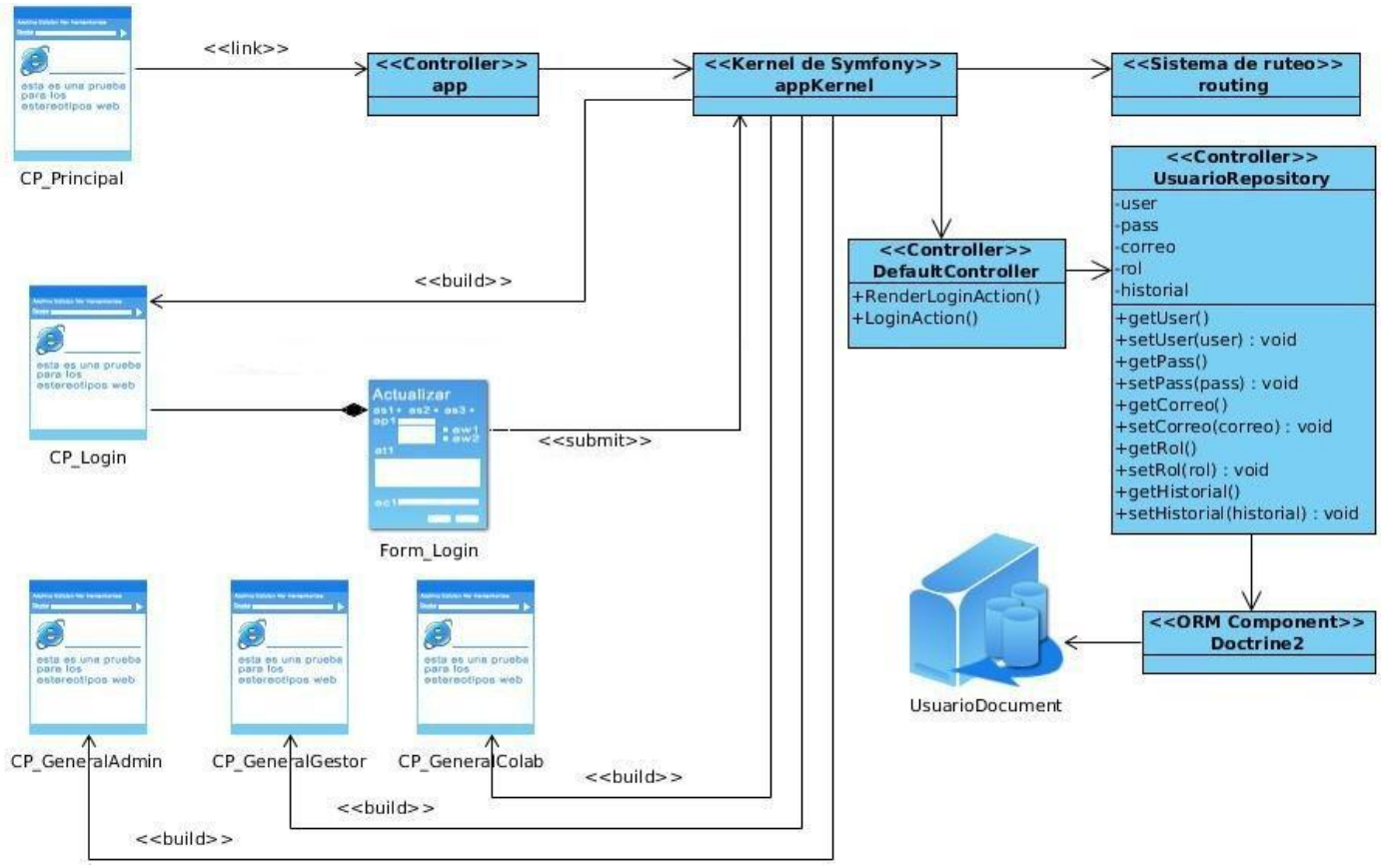


Figura 16. Diagrama de Clase del Diseño del caso de uso Autenticar Usuario.

2.7 Diagrama de Interacción

Un diagrama de interacción explica gráficamente las interacciones existentes entre las instancias (las clases) del modelo de estas. El punto de partida de las interacciones es el cumplimiento de las pos condiciones de los contratos de operación. Los diagramas de interacción pueden ser de secuencia o de colaboración.

Diagrama de Secuencia

Un diagrama de secuencia es una representación que muestra, en determinado escenario de un caso de uso, los eventos generados por actores externos, su orden y los eventos internos del sistema [34].

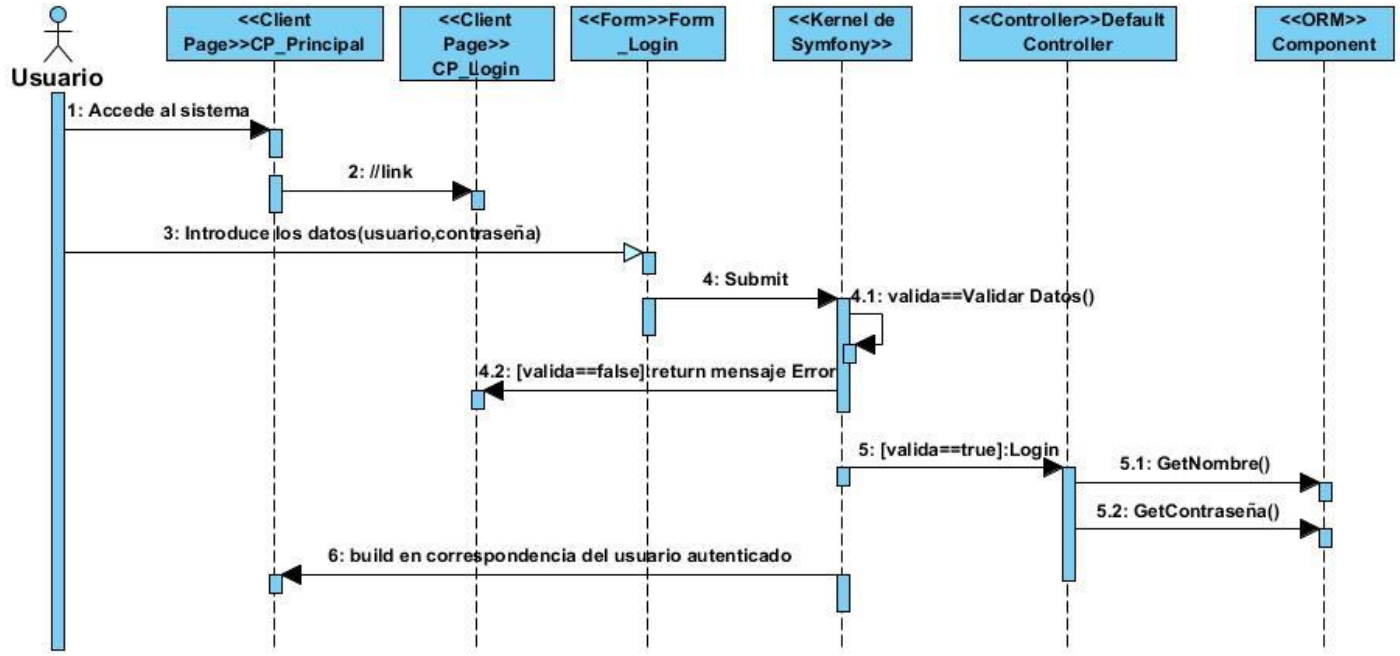


Figura 17. Diagrama de secuencia del caso de uso Autenticar Usuario.

2.8 Modelo de Despliegue

El modelo de despliegue es un modelo de objetos que describe la distribución física del sistema en términos de cómo se distribuye la funcionalidad entre los nodos de cómputo. Se utiliza como entrada fundamental en las actividades de diseño e implementación, debido a que la distribución del sistema tiene una influencia principal en su diseño [35]. En él se representan una serie de nodos y arcos, donde cada nodo representa un equipo de cómputo (procesador o equipo de hardware similar).

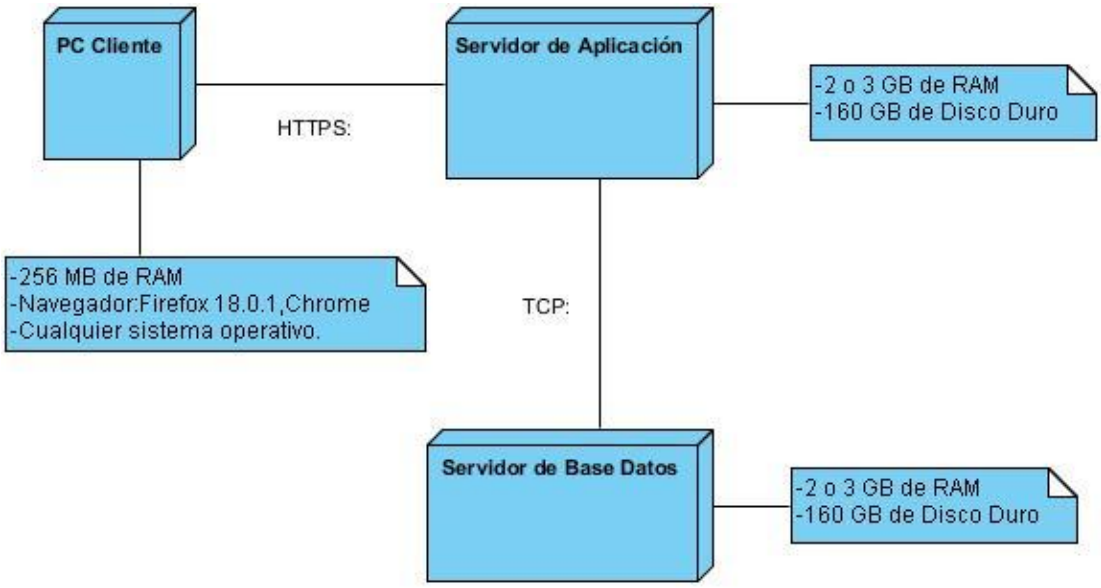


Figura 18. Diagrama de despliegue.

El despliegue de la aplicación está compuesto por tres nodos:

1. PC cliente, que accederá al servidor web del Vocabulario Controlado mediante un navegador web, y a través del protocolo seguro HTTPS.
2. El servidor web donde se instala la aplicación, estará conectado al servidor de Base de Datos mediante el protocolo TCP.
3. El servidor de Base de Datos es donde se debe albergar toda la información que perdurará en el tiempo.

2.9 Conclusiones Parciales

- ✓ En el presente capítulo se plantearon las características principales del sistema a desarrollar para de esta forma lograr un mayor acercamiento al mismo.
- ✓ Se llevó a cabo el levantamiento de los requerimientos de la aplicación, llevándolos a CU para la posterior implementación de las funcionalidades del Vocabulario Controlado.
- ✓ Se realizó una la descripción del estilo arquitectónico a utilizar MVC y los patrones de diseño, ambos vinculados con el *framework* de desarrollo Symfony 2.1, permitiendo así una mejor organización y estructura del sistema.

- ✓ Además se representan los diagramas correspondientes al diseño de la aplicación, Diagrama de Proceso, Diagrama de Clases del Diseño, Diagrama de Secuencia y Diagrama de Despliegue, los cuales permitieron dar un mejor enfoque de cómo estará diseñado el Vocabulario Controlado.

CAPÍTULO 3: IMPLEMENTACIÓN Y VALIDACIÓN DEL SISTEMA

Un producto listo para ser entregado, requiere la completa implementación y validación de las funcionalidades previamente definidas. En el presente capítulo se presentan los Casos de Prueba a los cuales fue sometido el sistema en cada una de las iteraciones, el diagrama de componente que muestra la relación entre los distintos componentes del sistema. Finalmente se realizan los distintos tipos de pruebas (Pruebas de seguridad, Pruebas de rendimiento y Pruebas funcionales) y se exponen los resultados obtenidos.

3.1 Diagrama de componentes

Un Diagrama de Componente es un esquema o diagrama que muestra las interacciones y relaciones de los componentes de un modelo. Describe los elementos físicos del sistema y sus relaciones, muestra opciones de realización, incluyendo código fuente, binario y ejecutable. Los componentes representan todos los tipos de elementos de software que entran en la fabricación de aplicaciones informáticas, pueden ser simples archivos, paquetes o bibliotecas [36].

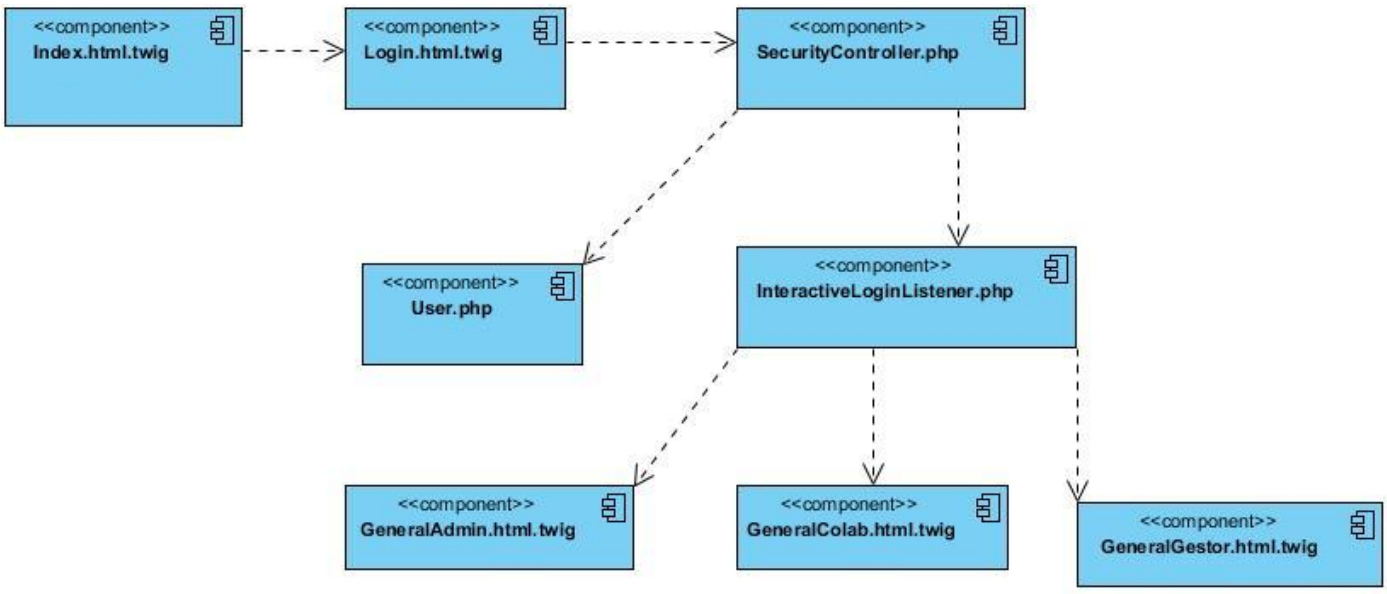


Figura 19. Diagrama de componente.

3.2 Código fuente

El código fuente es un conjunto de líneas que conforman un bloque de texto, escrito según las reglas sintácticas de algún lenguaje de programación destinado a ser legible por humanos. Es un programa en su forma original, tal y como fue escrito por el programador, no es ejecutable directamente por el computador, debe convertirse en lenguaje de maquina mediante compiladores, ensambladores o intérpretes [37].

El vocabulario controlado posee un código fuente claro y legible por cualquier desarrollador.

3.2.1 Clases o implementaciones relevantes

Como un aporte, que se considera relevante en la investigación, está el uso de la tecnología ElasticSearch.

ElasticSearch es un servidor de búsqueda distribuido RESTful libre y de código abierto basado en Apache Lucene. Está hecho en Java bajo los términos de Apache License.

Para su integración con Symfony se usó el bundle ElasticaBundle el cual usa el servicio de Elastica previamente instalado para realizar las búsquedas en el sistema.

```
public function searchAction(Request $request) {

    $inputSearch = $request->request->get('inputSearch');

    $finder_descriptor = $this->container->get('foq_elastica.finder.vocab.descriptor');
    $finder_sugerencia = $this->container->get('foq_elastica.finder.vocab.sugerencia');
    $finder_no_descriptor = $this->container->get('foq_elastica.finder.vocab.nodescriptor');

    $descriptores = $finder_descriptor->find('', 10000);
    $sugerencias = $finder_sugerencia->find('', 10000);
    $no_descriptores = $finder_no_descriptor->find('', 10000);

    $all = array_merge((array) $descriptores, (array) $no_descriptores, (array) $sugerencias);
    $results = array();

    foreach ($all as $value) {
        if (preg_match(strtolower("/.($inputSearch)|($inputSearch)+/"), strtolower($value->getT
            $results[] = $value;
    }

    $paginator = $this->get('knp_paginator');
    $pagination = $paginator->paginate($results, $this->get('request')->query->get('page', 1),
    $rol = null;
    if ($this->container->get('security.context')->isGranted('ROLE_ADMIN'))
        $rol = "admin";
    if ($this->container->get('security.context')->isGranted('ROLE_GESTOR'))
        $rol = "gestor";
    if ($this->container->get('security.context')->isGranted('ROLE_COLAB'))
```

Figura 20. Ejemplo de código de Elastica.

3.2.2 Estándar de codificación

Con el fin de garantizar una uniformidad en el código de la aplicación se utilizó el estándar de codificación basado en Pautas de Codificación PHP de Zend Framework.

Dicho estándar presenta tres temas relacionados con la estructura y confección del código usando el lenguaje PHP.

Formato a los archivos php

Los archivos PHP mantienen el formato según lo pacta dicho estándar, cada línea de código posee una identación compuesta por 4 espacios y su longitud está entre los 80 y 120 caracteres como máximo.

```
public function addVocabularioAction($rol, $option, Request $request) {
    $inputName = $request->request->get('inputName');
    $inputDesc = $request->request->get('inputDesc');
    $vocabulario = new Vocabulario();

    $vocabulario->setName($inputName);
    $vocabulario->setDescripcion($inputDesc);
    $dm = $this->get('doctrine_mongodb')->getManager();
    $dm->persist($vocabulario);
    $dm->flush();
    return new RedirectResponse($this->generateUrl('_gestionar_vocabularios'
}

public function add_no_descriptorAction($rol, $id) {
    $dm = $this->get('doctrine_mongodb')->getManager();

    $sug = $dm->getRepository('VocabularioBundle:Sugerencia')->find($id);

    $no_descriptor = new Nodescriptor();

    $no_descriptor->setTE($sug->getTE());
    $no_descriptor->setDescrip($sug->getDescrip());
    $no_descriptor->setTR($sug->getTR());
    $no_descriptor->setVocabulario($sug->getVocabulario());
    $no_descriptor->setType('no descriptor');
```

Figura 21. Ejemplo de código de Identación.

Convenciones de nombre

Los nombres de las clases contienen solo caracteres alfanuméricos, en su mayoría se encuentran compuestos por más de una palabra por lo que la primera letra de cada una aparece en mayúscula.

Las funcionalidades implementadas tienen nombres que sugieren el comportamiento de la acción a realizar. Específicamente en los métodos de acceso para las instancias o variables estáticas, los caracteres que comienzan sus nombres son "*get*" o "*set*", indistintamente por la acción que se vaya a realizar.

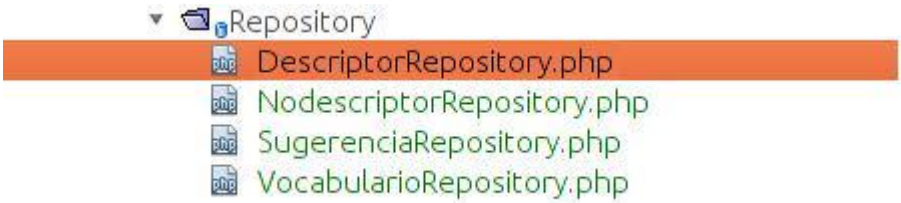


Figura 22. Ejemplo de los nombres de las clases.

Estilos de código

El código PHP se encuentra delimitado por la forma completa de las etiquetas PHP estándar.

```
<?php
namespace Vocabulario\VocabularioBundle\Controller;

use Symfony\Bundle\FrameworkBundle\Controller\Controller;
use Symfony\Component\HttpFoundation\Request;
```

Figura 23. Ejemplo del uso de la etiquetas php.

3.3 Interfaces principales de la aplicación

El vocabulario controlado cuenta con una interfaz web sencilla que permite acceder a los contenidos de forma rápida.

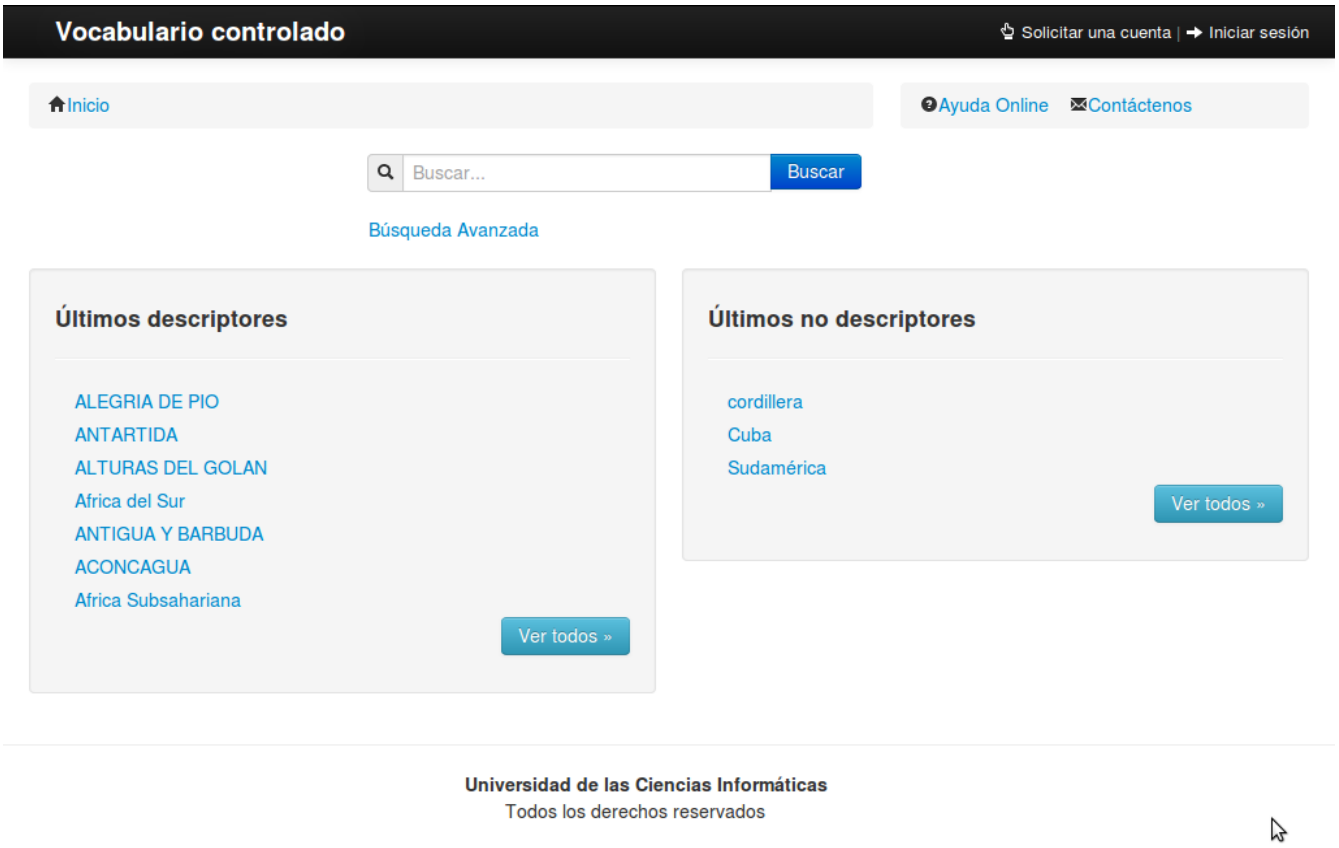


Figura 24. Interfaz Principal de usuarios anónimos.

Figura 25. Interfaz del Colaborador.

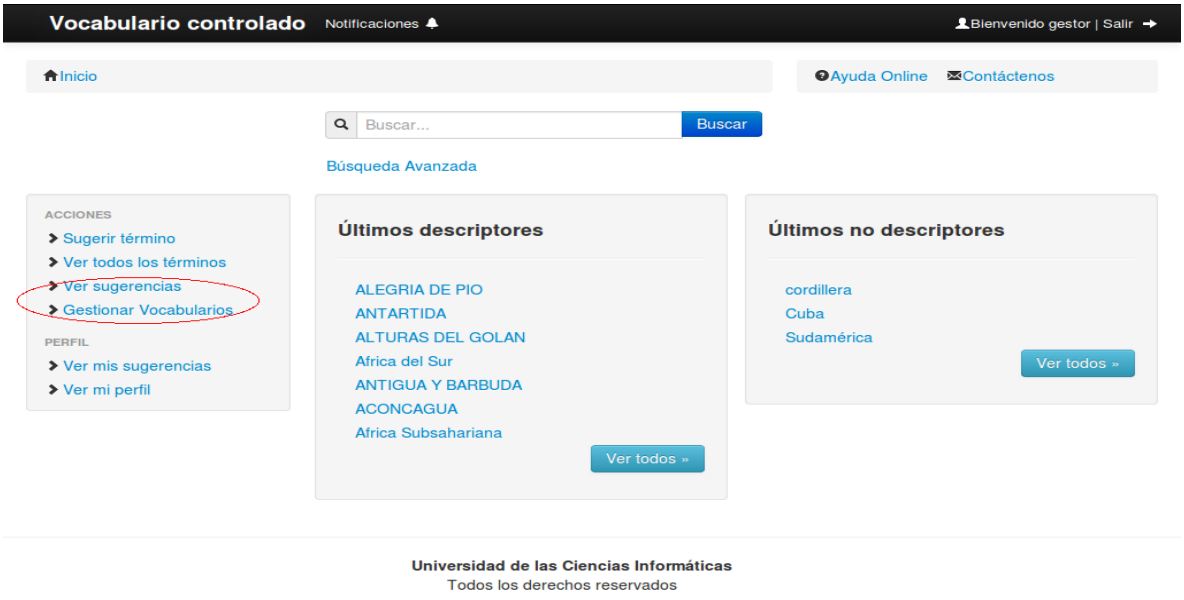


Figura 26. Interfaz del gestor.

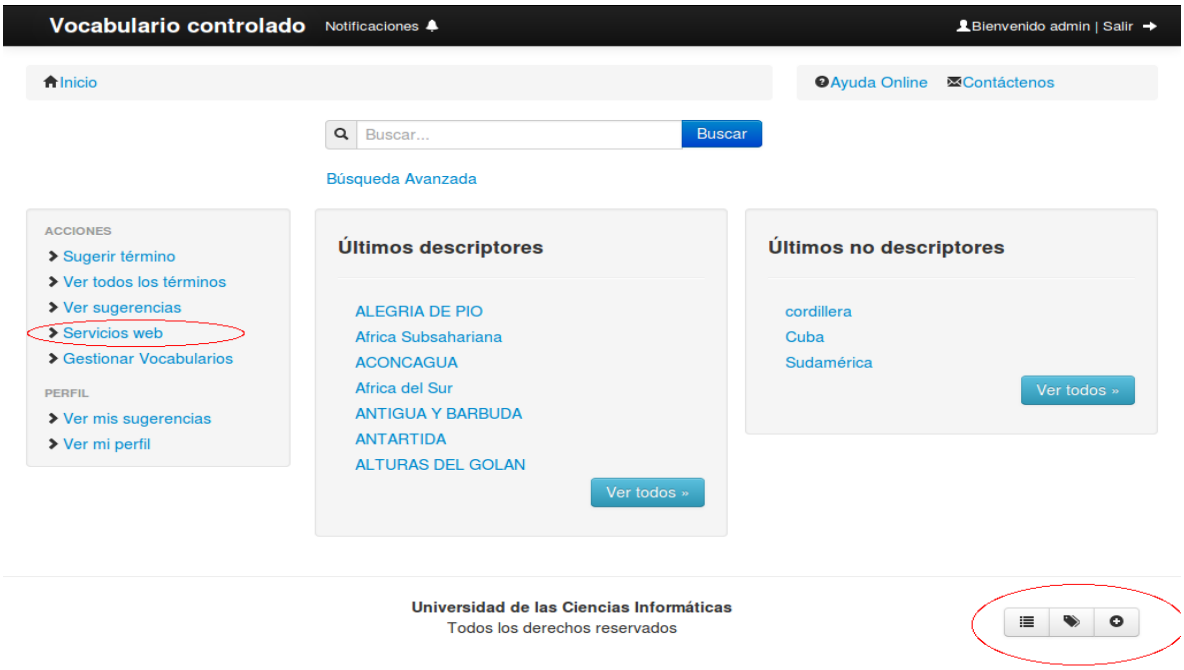


Figura 27. Interfaz del administrador.

3.4 Validación del sistema

La validación del software implica procesos de comprobación, como las inspecciones y revisiones, en cada etapa del proceso de software desde la definición de requerimientos hasta el desarrollo del programa. Sin embargo, la mayoría de los costos de validación aparecen después de la implementación, cuando se prueba el funcionamiento del sistema.

A continuación se presentan las pruebas realizadas al vocabulario controlado y los resultados obtenidos luego de su ejecución.

3.4.1 Pruebas de software

Las pruebas de software constituyen una fase del proceso de desarrollo de un software centrada en el favorecimiento a la calidad, fiabilidad y robustez del mismo, dentro del contexto o escenario previsto para ser utilizado [38].

3.4.1.1 Pruebas funcionales

Mediante las pruebas funcionales se valida el cumplimiento de las aplicaciones desarrolladas contra las funciones detalladas en el documento de requisitos del cliente, buscando reducir los defectos en la etapa

de operación y permitiendo la corrección de los mismos a costos reducidos al ser encontrados en etapas tempranas. A este tipo de pruebas se les denomina también pruebas de caja negra, debido a que los analistas de pruebas enfocan su atención a las respuestas del sistema de acuerdo a los datos de entrada y su resultado en los datos de salida [39].

Las metas de estas pruebas son:

- ✓ Verificar el procesamiento, recuperación e implementación adecuada de las reglas del negocio.
- ✓ Verificar la apropiada aceptación de datos.

Resultados de las pruebas funcionales

A continuación se muestran los casos de prueba funcionales a realizar sobre las interfaces Autenticar Usuario y Realizar búsqueda simple.

Caso de Prueba 1				
Nombre de la sección: Autenticar usuario				
Condiciones de ejecución: • El usuario debe acceder a través del navegador al sistema, conectándose a la página: http://localhost/VC/web/app.php/				
Pasos de ejecución: Se deben introducir los caracteres correspondientes a los siguientes campos: • Nombre de Usuario • Contraseña Luego presionar el botón Entrar.				
Escenarios	**Descripción**	**Usuario**	**Contraseña**	**Respuesta del sistema**
Entrada de los datos de autenticación correctos.		admin	admin	Muestra la interfaz correspondiente al usuario autenticado.
Los datos introducidos en el campo Nombre de usuario son incorrectos.		administrador	admin	Muestra el mensaje de error. “Nombre de usuario o Contraseña inválido”
Los datos introducidos en el				Muestra el mensaje de error.

campo Contraseña son incorrectos.		admin	administrador	“Nombre de usuario o Contraseña inválido”
El campo Nombre de usuario se encuentra vacío.			admin	Muestra el mensaje de error: “Por favor rellene este campo”.
El campo Contraseña está vacío.		admin		Muestra el mensaje de error: “Por favor rellene este campo”.

Tabla 4. Descripción del caso de prueba Autenticar usuario.

Caso de Prueba 2			
Nombre de la sección: Realizar búsqueda simple.			
Condiciones de ejecución: • El usuario debe acceder a través del navegador al sistema, conectándose a la página: http://localhost/VC/web/app.php/			
Pasos de ejecución: Se deben introducir un criterio de búsqueda en el campo que hace referencia a dicha función. Luego presionar el botón buscar.			
Escenarios	**Descripción**	**Criterio de búsqueda**	**Respuesta del sistema**
Respuesta al criterio de búsqueda introducido.		ter	Muestra una interfaz con los resultados de la búsqueda.
Criterio de búsqueda no encontrado.		maivys	Muestra el mensaje de error: ”No hay resultados para el criterio de búsqueda maivys ”
Criterio de búsqueda no introducido.			Muestra el mensaje de error: “No debe dejar el campo de búsqueda vacío.”

Tabla 5. Descripción del caso de prueba Realizar búsqueda simple.

A partir del diseño de los casos de prueba se realizaron 2 iteraciones de prueba. A continuación se muestran los resultados obtenidos.

No conformidades	**Primera iteración**	**Segunda iteración**
Detectadas	24	10

Resueltas	20	10
Pendientes	4	0

Tabla 6. Resultados de las No Conformidades encontradas.

A continuación se muestran algunas de las no conformidades encontradas durante las iteraciones.

- ✓ Mensaje en inglés cuando un rol esta deshabilitado.
- ✓ Cuando se edita un vocabulario existe falta de ortografía en la palabra vocabulario.
- ✓ Al regresar de la búsqueda avanzada, cuando no encuentra resultados se muestra un error.

3.4.1.2 Pruebas de Seguridad

Las pruebas de seguridad buscan medir la Confidencialidad, Integridad y Disponibilidad de los datos, desde la perspectiva del aplicativo. Una vez ejecutadas las pruebas de seguridad es posible medir y cuantificar los riesgos a los cuales se ven expuestos los aplicativos tanto en la infraestructura interna como externa, valiéndose de la filosofía del Hacking ético.

En dichas pruebas se mide el nivel de seguridad en cuanto a:

1. La aplicación: al verificar que un actor solo pueda acceder a las funciones y datos que su usuario tiene permitido.

2. El sistema: al verificar que solo los actores con acceso al sistema y a la aplicación están habilitados para accederla.

Las pruebas de seguridad garantizan:

- ✓ Que los usuarios están restringidos a funciones específicas o su acceso está limitado únicamente a los datos que está autorizado a acceder.
- ✓ Que solo aquellos usuarios autorizados a acceder al sistema son capaces de ejecutar las funciones del sistema [40].

Resultados de las pruebas de seguridad

La seguridad de la aplicación está basada en roles definidos para accederla y asegurarse que solo los usuarios que posean dichos roles accedan a la aplicación y realicen las funciones que para cada uno de ellos estén definidas. Ningún usuario es capaz de acceder a las interfaces ni funcionalidades de otro usuario con roles diferentes. Están establecidos tres roles principales, ROLE_ADMIN para aquellos usuarios destinados a administrar las preferencias del sistema así como la gestión de los demás usuarios, también serán capaces de acceder a todas las funcionalidades de la aplicación y gestionar todo el contenido de esta, ROLE_GESTOR y ROLE_COLAB agrupan a los usuarios que se desempeñarán como gestores y colaboradores en el proceso de gestión del vocabulario controlado, cada uno es capaz de acceder solo a sus funciones siendo, el primer rol mayor en la jerarquía de estos y el ROLE_ADMIN el mayor de todos. Cada uno de ellos es capaz de realizar las funciones del inmediato inferior, no así de manera contraria, y de manera adicional se encuentran los usuarios anónimos, que no representan un rol en específico pero si agrupa a los usuarios que no se han autenticado.

Para probar la seguridad del sistema se le aplicaron una serie de pruebas usando el software que para ello está destinado Acunetix WVS Reporter v8.0 el cual generó un reporte tras haberlo escaneado en busca de las fallas de seguridad más importantes y comunes como son Cross Site Scripting, SQL injection, ataques por fuerza bruta y otros. Durante una primera revisión se detectaron una serie de fallos los cuales estuvieron repartidos en enlaces rotos, formularios sin seguridad, envío de información sensible en texto claro y la no utilización del protocolo seguro https, que en dependencia del riesgo estaban clasificados en alto, medio y bajo. De ellos no se detectó ningún riesgo de clasificación alto, cinco de nivel medio y cuatro de nivel bajo, las cuales fueron solucionadas para luego ejecutar otra prueba de seguridad y ver corregidos todos estos fallos, para así concluir que el mecanismo de seguridad del sistema funciona correctamente y garantiza la integridad y disponibilidad de los recursos a gestionar.

3.4.1.3 Pruebas de rendimiento

Pruebas de carga: Mediante la ejecución de las pruebas de Carga es posible identificar la capacidad de recuperación de un sistema cuando es sometido a cargas variables, tanto de usuarios como de procesos. Al realizar las pruebas de carga se puede determinar el tiempo de respuesta de todas las transacciones críticas del sistema y encontrar cuellos de botella de la aplicación [41].

Prueba de estrés: Mediante las pruebas de estrés es posible identificar la capacidad de respuesta de un sistema bajo condiciones de carga extrema, representadas por una alta concurrencia de Usuarios y/o Procesos. Una vez realizadas las pruebas de estrés se podrá conocer el punto de quiebre del aplicativo en términos de capacidad de respuesta, con lo cual será posible establecer acciones de optimización en

diferentes niveles para asegurar una mejor capacidad de concurrencia de usuarios y/o procesos, que se verá reflejada en una óptima operación de negocio [42].

Resultados de las Pruebas de carga y estrés

Para las pruebas de rendimiento, específicamente pruebas de Carga y Estrés, que se le realizaron a la aplicación se usó la herramienta JMeter versión 2.3.4 la cual es una herramienta Java que permite realizar pruebas de rendimiento y pruebas funcionales sobre aplicaciones web.

JMeter es un software de código abierto diseñado para pruebas de carga de comportamiento funcionales y la medición del rendimiento. Originalmente fue diseñado para probar las aplicaciones web, pero desde entonces se ha expandido a otras funciones de prueba. Es utilizado para probar el rendimiento tanto en los recursos estáticos como dinámicos. Puede ser utilizado para simular una carga pesada en un servidor, de red o un objeto para poner a prueba su resistencia o para analizar el rendimiento general en diferentes tipos de carga.

Puede usarse además para hacer un análisis gráfico de rendimiento o para probar el comportamiento de su servidor /script/objeto con carga pesada concurrente.

Para la realización de las pruebas se hizo necesario tener en cuenta las condiciones del escenario, tanto del hardware como software, donde se encuentra la aplicación; para obtener una correcta información de comportamiento y resultados en general. Las características del escenario son las siguientes:

Hardware:

Tipo de procesador: Pentium (3) Dual-Core CPU E5400, 2.70Hz (2CPUs)

Memoria: 2 GB RAM

Tipo de Red: Ethernet 100Mbps

Software:

Tipo de servidor web: Apache Apache/2.2.22 (Ubuntu)

Memoria máxima: 256 MB

Máximo de hilos concurrentes: 50

Plataforma: SO Ubuntu 12.04.2 LTS (precise)

Servidor de BD: MongoDB

En un ambiente con las características anteriormente especificadas se obtuvo que para un total de 50 usuarios (hilos) concurrentes a un intervalo de un segundo, la aplicación tuvo un mínimo de 0,293 segundo de tiempo de respuesta y como máximo demoró 7.888 en responder a una petición y en total es capaz de responder a 21,7 peticiones por segundo con una media de 2074 respuestas satisfactorias. Se demuestra que la aplicación es estable para estos 50 usuarios pues se mantuvo prestando servicios todo

el tiempo para un 0.0% de errores, lo cual muestra que se mantiene funcional aun en situaciones donde concurren gran cantidad de usuarios trabajando sobre ella, accediendo y modificando información de manera simultánea. Para la aplicación del plan de pruebas se probó una secuencia de acciones las cuales son ejecutadas de manera aleatoria por el Jmeter, simulando el trabajo de 50 usuarios conectados, el flujo que se probó incluye la mayoría de los requisitos funcionales de la aplicación, lo que demostró la capacidad de respuesta y procesamiento del sistema. (Ver **Figura 28, Figura 29**)

	Label	Media	Mediana	Linea d...	Mín	Máx	% Error	Rendimiento	Kb/sec
100	/VC/web/app.php/	3248	2016	7159	372	7888	0.00%	2.3/sec	25.4
50	/VC/web/app.php/login	1567	2040	2158	335	2218	0.00%	4.8/sec	13.6
50	/VC/web/app.php/login_check	3080	3798	4002	606	4061	0.00%	3.5/sec	13.2
150	/VC/web/app.php/admin/genera...	1874	2045	2256	293	2516	0.00%	4.7/sec	4.3
50	/VC/web/app.php/admin/sugerir...	1623	1916	2116	394	2160	0.00%	2.9/sec	2.7
50	/VC/web/app.php/admin/sugerir...	1891	2144	2343	441	2404	0.00%	2.6/sec	2.4
50	/VC/web/app.php/admin/all/2	2153	2256	2453	789	2514	0.00%	2.2/sec	2.0
50	/VC/web/app.php/admin/detalle...	1894	1946	2078	789	2196	0.00%	2.1/sec	1.9
50	/VC/web/app.php/admin/sugere...	2162	2173	2322	1851	2412	0.00%	2.0/sec	1.8
50	/VC/web/app.php/admin/no_des...	1751	1753	1893	1261	1985	0.00%	2.0/sec	1.9
50	/VC/web/app.php/admin/web_se...	1643	1733	1957	532	2045	0.00%	2.1/sec	1.9
50	/VC/web/app.php/admin/vocab/8	1604	1599	1995	446	2102	0.00%	2.1/sec	1.9
50	/VC/web/app.php/admin/vocab_...	1708	1633	1998	1388	2164	0.00%	2.1/sec	1.9
50	/VC/web/app.php/admin/mis_su...	1491	1428	1862	1229	1993	0.00%	2.2/sec	2.0
50	/VC/web/app.php/admin/help	1472	1451	1957	615	1995	0.00%	2.2/sec	2.0
50	/VC/web/app.php/logout	3252	3124	4163	2306	4502	0.00%	2.1/sec	25.6
950	TOTAL	2074	1952	3176	293	7888	0.00%	21.7/sec	61.0

Figura 28. Resultados de las pruebas de rendimiento.

Peticiones hechas a la aplicación y sus respuestas

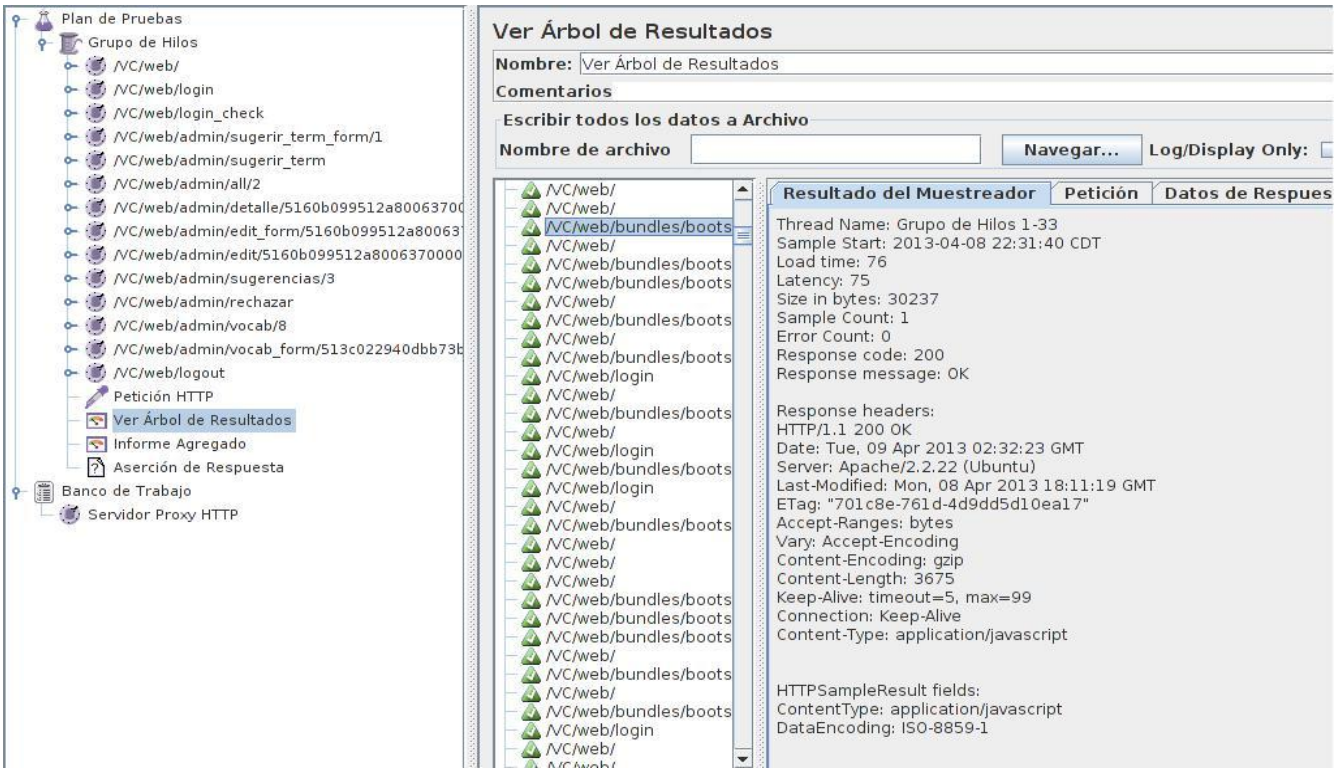


Figura 29. Resultados de las peticiones hechas a la aplicación y sus respuestas.

3.5 Conclusiones Parciales

- ✓ Tras realizar la implementación y validación del sistema, se ha obtenido un producto con una estructura organizacional amigable y segura para cada uno de los usuarios, partiendo a partir de cada uno de los roles y los permisos definidos.
- ✓ Las pruebas efectuadas al producto permitieron corregir algunos errores, mejorando así la calidad de la solución.

CONCLUSIONES GENERALES

- ✓ Con el propósito de darle cumplimiento al objetivo general y los objetivos específicos de este trabajo se llevaron a cabo una serie de tareas que permitieron el buen desarrollo del sistema.
- ✓ Se realizó una previa investigación sobre los vocabularios controlados lo cual permitió reflejar la situación actual de las herramientas que posibilitan la implementación de los mismos, arribando a la conclusión de que era necesaria la implementación de un vocabulario controlado.
- ✓ Se realizaron varios estudios para elegir las tecnologías, lenguaje de programación y metodología de desarrollo a utilizar donde se escogieron las que resultaron más adecuadas para el desarrollo del sistema.
- ✓ Se llevaron a cabo una serie de pruebas con las cuales se pudo comprobar el adecuado funcionamiento de la aplicación.
- ✓ Como resultado de este trabajo se desarrolló un Vocabulario Controlado cumpliendo con los requerimientos establecidos inicialmente.

RECOMENDACIONES

- ✓ Realizar otros estudios acerca de los vocabularios controlados.
- ✓ Implementar otras funcionalidades que permitan la gestión de la información tales como:
  - Convertir un No Descriptor en Descriptor

REFERENCIAS BIBLIOGRÁFICAS

[1] Lancaster, F.W. (1986). Vocabulary Control for Information Retrieval (2nd Edition).

[2] Gil Urdiciain, 2004

[3] Benítez, Pamela Faber, Terminología multilingüe y ontologías, Pamela Faber Benítez

[4] ANSI/NISO Z39.19 – 2005 (R2010) Guidelines for the Construction, Format, and Management of Monolingual Controlled Vocabularies.

[5] BuenasTareas.com. 2012. Indización. [Disponible en: http://www.buenastareas.com/ensayos/Indizacion/5944819.html]

[6] D.L McGuinness, J. W. (1998) "Conceptual Modeling for Configuration: A Description Logic-based Approach. Artificial Intelligence for Engineering Design, Analysis".

[7] Cognatrix [Disponible en: http://www.lgosys.com/products/Cognatrix]

[8] Midos Thesaurus [Disponible en: http://www.progris.de/index.html?/midost.htm]

[9] MultiTes [Disponible en: http://www.multites.com/]

[10]STRIDE [Disponible en: http://www.questans.co.uk/p100l2.html]

[11] Introducción, definición y evolución de PHP. [Disponible en: http://php.ciberaula.com/articulo/introduccion.php]

[12] Álvarez, 2001

[13] Introducción a JSON. [Disponible en: http://www.json.org/json-es.html]

[14]Pérez, Eguíluz, Javier. Introducción a JavaScript

[Disponible en: http://www.librosweb.es/javascript/]

[15] Guía breve de CSS. [Disponible en: http://www.w3c.es/Divulgacion/GuiasBreves/HojasEstilo]

[16] Páginas web [Disponible en: http://www.paginaswebs.com.mx/formatos/html.html]

[17] Symfony Framework PHP orientado a objetos. [Disponible en: http://www.maestrosdelweb.com/editorial/curso-symfony2-introduccion-instalacion/]

[18] Genbeta: dev Desarrollo y software [Disponible en: http://www.genbetadev.com/frameworks/bootstrap]

[19] MATO, 2006

[20] Bases de datos no relacionales [Disponible en: http://www.slideshare.net/dipina/nosql-cassandra-couchdb-mongodb-y-neo4j]

[21] Computación., Departamento de Sistemas Informáticos y Rational Unified Process (RUP). Universidad Politécnica de Valencia.: © P.Letelier.

[22] Hernández, Yulainne Alonso. Configuración de la Metodología OpenUP. Ciudad Habana: Centro Ideoinformática, 2012.

[23][Disponible en: http://www.dosbit.com/plataformas/windows/foreui-disena-maquetas-de-interfaces-y-renderizalas-en-png]

[24] Elastic Search [Disponible en: http://www.elasticsearch.org/]

[25] León, Eduardo. Tutorial Visual Paradigm for UML. [Disponible en: http://es.scribd.com/doc/36636137/Tutorial-Visual-Paradigm]

[26] [Disponible en: https://netbeans.org]

[27] Navarro Marset, Rafael. Rest vs Servicios Web. ELP-DSIC-UPV Modelado, Diseño e Implementación de Servicios Web. 2006-07.

[28] Roy Thomas Fielding (2000)Architectural Styles and the Design of Network-based Software Architectures [Disponible en: http://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm]

[29] Ivar Jacobson, 2004

[30] Mireles, Gabriel Alberto García. Material didáctico para la asignatura Introducción a la Computación II. [Disponible en: http://www.mat.uson.mx/mireles/Casos%20de%20usonota.htm]

[31] Libros Web Disponible en:[http://librosweb.es/symfony_1_2/capitulo_2/el_patron_mvc.html][23]

[32]Asignatura Ingeniería de Software II. Tema 1. Conferencia No. 1: Patrones del Diseño. Ciudad de La Habana. Cuba. Curso 2006-2007.

[33] Larman, Craig. UML y Patrones. Introducción al análisis y diseño orientado a objetos.

[34] LARMAN, 2004

[35] [Disponible en: http://ocw.usal.es/ensenanzas-tecnicas/ingenieria-del-software/Tema6-DOO]

[36] Ramírez., Lic. Elisa Arizaca. Disponible en: http://virtual.usalesiana.edu.bo/web/practica/archiv/compon.doc]

[37] Pergamino Virtual Disponible en: [http://www.pergaminovirtual.com.ar/definicion/Codigo_Fuente.html]

[38] Jacobson, Booch, Rumbaugh. El Proceso Unificado de Desarrollo de Software, Madrid, 2003, Pearson Education S.A.

[39] PRUEBAS FUNCIONALES [Disponible en: http://www.vyvquality.com/w1/index.php/servicios/pruebas-funcionales.html]

[40] PRUEBAS DE SEGURIDAD.

[Disponible en:http://www.vyvquality.com/w1/index.php/servicios/pruebas-de-seguridad.html]

[31] PRUEBAS DE CARGA

[Disponible en: http://www.vyvquality.com/w1/index.php/servicios/pruebas-de-rendimiento/pruebas-de-carga.html]

[42] PRUEBAS DE ESTRÉS. [Disponible en:

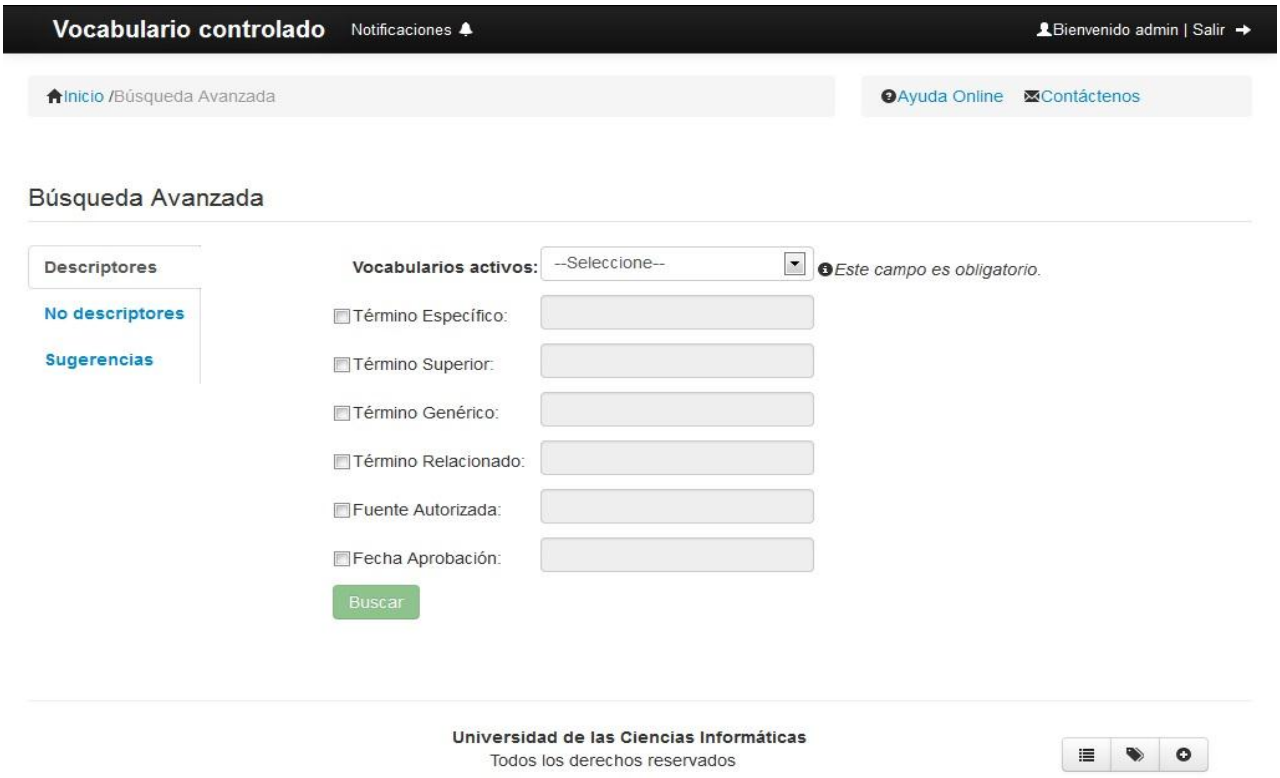
http://www.vyvquality.com/w1/index.php/servicios/pruebas-de-rendimiento/pruebas-de-estres.html]

BIBLIOGRAFÍA

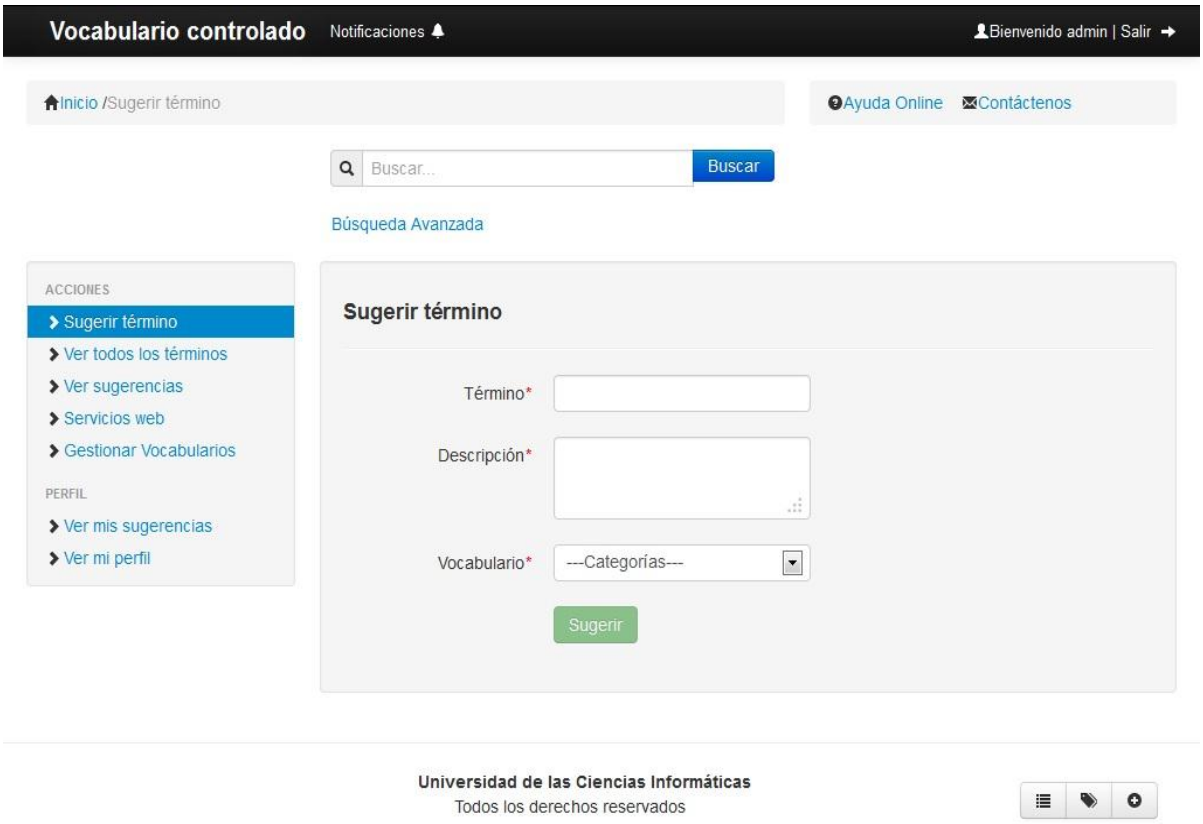
- ✓ A García Jiménez - Anales de documentación, 2004 - revistas.um.es
- ✓ ARANO, Silvia. "Los tesauros y las ontologías en la Biblioteconomía y la Documentación". Hipertex.net, núm, 3, 2005.
- ✓ BG Urdiaciaín - Revista general de información y documentación, 1998 - revistas.ucm.es
- ✓ Cimino, J.J., Desiderata for controlled medical vocabularies in the twenty-first century. Methods Inf Med, 1998. 37(4-5): p. 394-403
- ✓ E Currás - 1998 - dialnet.unirioja.es
- ✓ Herbst, T. (1986). «Defining with a controlled defining vocabulary in a foreign learners' dictionary». Lexicographica, 2, pp. 101-119.
- ✓ LOPEZ ALONSO, Miguel. "Integración de herramientas conceptuales de recuperación en la web semántica: tesauros conceptuales, ontologías, metadatos y mapas conceptuales". VII Encuentros Internacionales sobre Sistemas de Información y Documentación. Zaragoza, 2003.
- ✓ Michiels, A. y Nöel, J. (1984). «The pro's and con's of a controlled defining vocabulary in a learner's dictionary». En Hartman, R. K. K. (ed.) LEXeter'83 Proceedings: papers from the International Conference in Lexicography at Exeter. Tubinguen: Max Niemeyer, pp. 385-394
- ✓ MOREIRO GONZÁLEZ, J. A. [et al.]. Nuevos patrones en la representación y la visualización de la información para entornos distribuidos: del tesauro al Topic Map. Códice: Revista de la Facultad de Sistemas de Información y Documentación, 2005, n. 1
- ✓ Richards, J. (1974). «Word lists: problems and prospects». En RELC Journal, 5, 2, pp. 69-84.
- ✓ Schmitt, N. (2000). Vocabulary in Language Teaching. Cambridge: Cambridge University Press.

ANEXOS

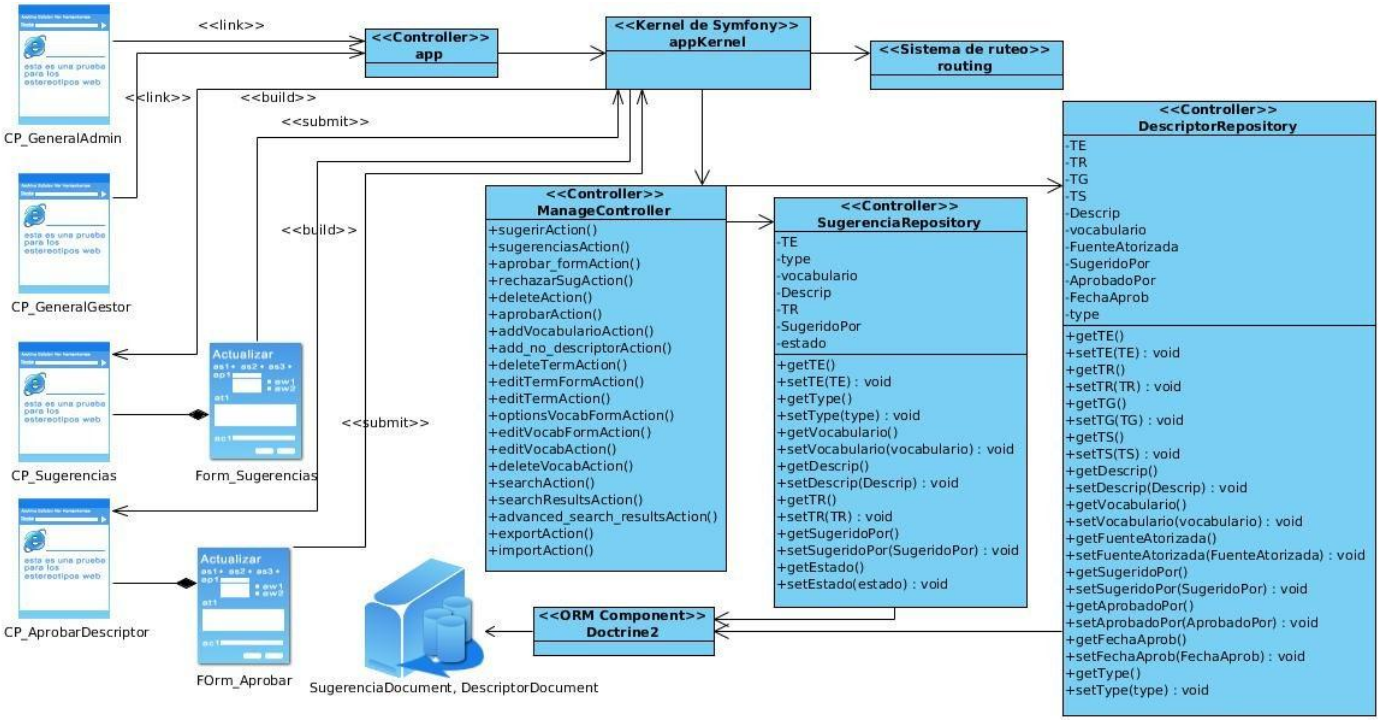
Anexo 1. Interfaz de la búsqueda avanzada.



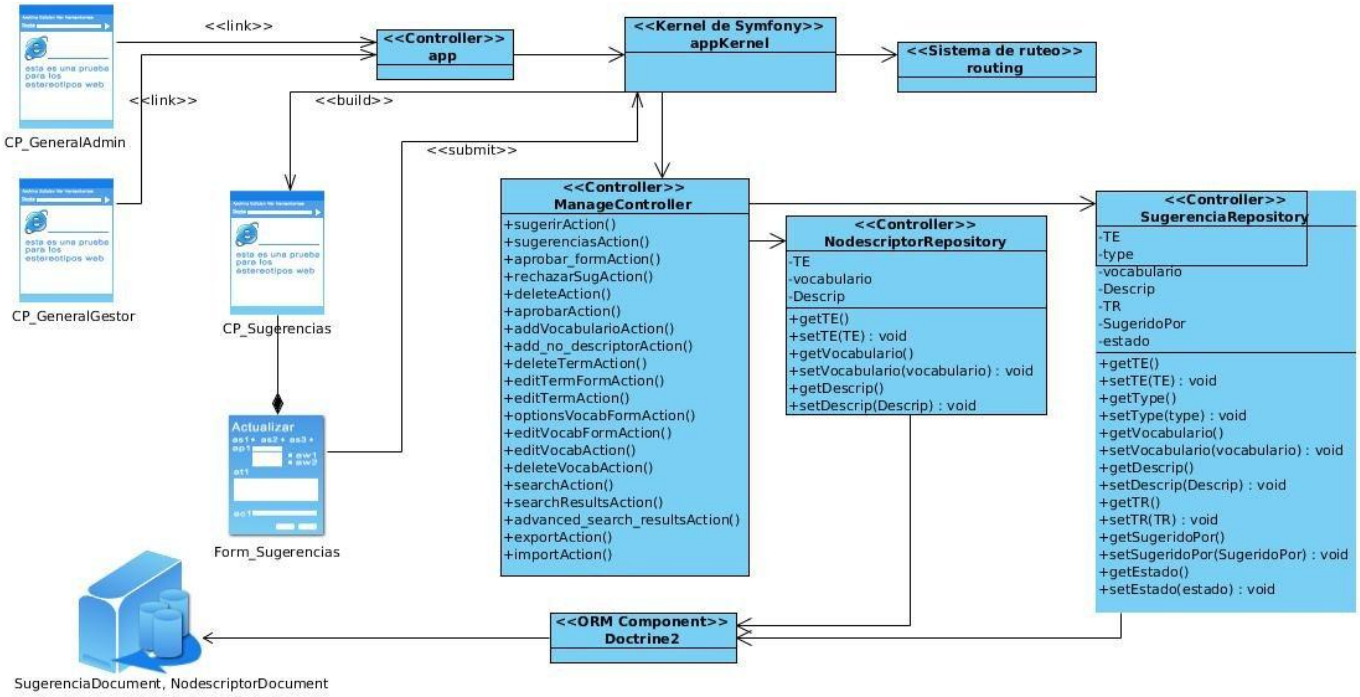
Anexo 2. Interfaz de Sugerir Término.



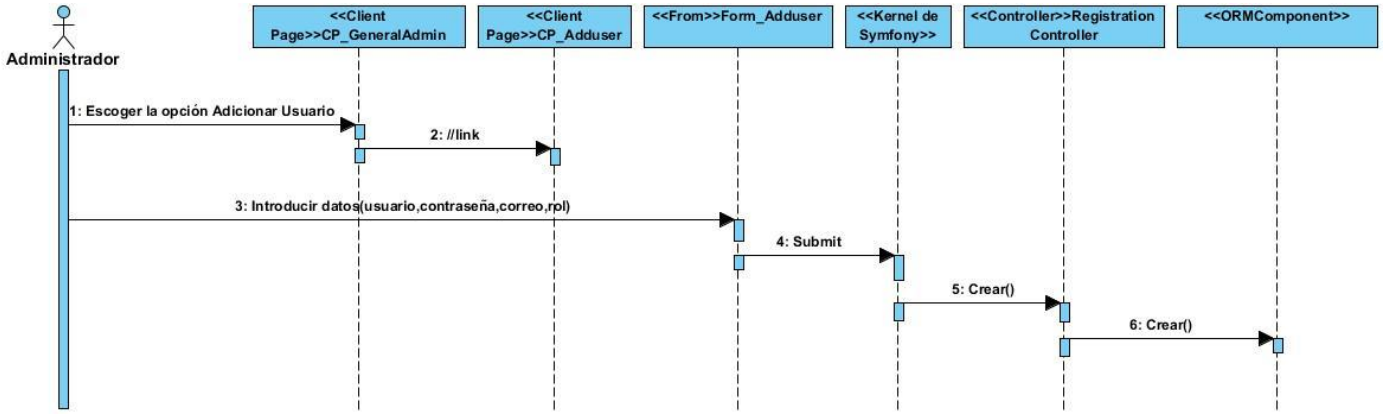
Anexo 3. Diagrama de clases del diseño Adicionar Descriptor.



Anexo 4. Diagrama de clases del diseño Adicionar No Descriptor.



Anexo 5. Diagrama de secuencia



GLOSARIO DE TÉRMINOS

A

Antonimia

Relación que se establece entre dos palabras cuyos significados son opuestos.

C

CASE

Acrónimo en inglés de Computer Aided Software Engineering (Ingeniería de Software Asistida por Computadora).

CSS

Acrónimo en inglés de Cascading Style Sheets (Hojas de Estilo en Cascada). Es un conjunto de instrucciones escritas en HTML, que definen las apariencias de una página web con el objetivo de que sus estilos aparezcan.

D

Descriptores.

Es el término o símbolo autorizado y formalizado que figura en un tesauro, que se utiliza para representar sin ambigüedad los conceptos contenidos en los documentos y en las peticiones de recuperación de la información.

F

Framework

Es una estructura de soporte definida, en la cual un proyecto de software puede ser organizado y desarrollado.

H

Homonimia

Es la cualidad de dos palabras, de distinto origen y significado por evolución histórica, que tienen la misma forma, es decir, la misma pronunciación o la misma escritura.

I

IDE

Acrónimo en inglés de Integrated Development Environment (Entorno de Desarrollo Integrado), es una aplicación compuesta por un conjunto de herramientas útiles para un programador.

Indización

La indización es el proceso de describir o representar el contenido temático de un recurso de información.

J

JSON

Acrónimo en inglés de JavaScript Object Notation *(*Notación de Objetos de JavaScript*),* es un formato ligero de intercambio de datos.

N

No descriptores.

Son sinónimos o cuasi-sinónimos de los descriptores o términos que designan en el lenguaje de uso conceptos afines a los que cubren los descriptores.

P

PHP

Acrónimo en inglés de Hypertext Preprocessor, lenguaje interpretado para el desarrollo de páginas web dinámicas del lado del servidor.

Polisemia

Se presenta cuando una misma palabra o signo lingüístico tiene varias acepciones o significados

Plugins

Un plugin es un programa o aplicación que añade funcionalidad al programa principal donde está hospedado.

R

RUP

Acrónimo en inglés de Rational Unified Process (Proceso Unificado de Racional o Metodología RUP), es una metodología de desarrollo de software.

Requisitos funcionales

Son las capacidades o condiciones que el sistema debe cumplir.

Requisitos no funcionales

Son las propiedades o cualidades que el producto debe tener.

S

SGBD

Sistema Gestor de Bases de Datos.

Sinonimia

La sinonimia es la relación semántica que se da entre palabras o expresiones que presentan significados equivalentes.

Stakeholders

Es cualquier persona o entidad que es afectada o concernida por las actividades o la marcha de una organización.

T

Tags

Los tags, a veces llamados "etiquetas" en español, son los "comandos" que los programas navegadores leen e interpretan para armar y dar forma a las páginas web.

Tesauro

Es un vocabulario controlado organizado en un orden conocido y estructurado de manera que varias relaciones entre términos son mostradas claramente e identificadas por indicadores de relaciones estandarizados.

U

UML

Acrónimo en inglés de Unified Modeling Language (Lenguaje Unificado de Modelado).

X

XML

Acrónimo en inglés de Extensible Markup Language (Lenguaje de Etiquetado Extensible).