UNIVERSIDAD DE LAS CIENCIAS INFORMATICAS

FACULTAD 9

Diseño del Subsistema de Configuración de la Plataforma de Televisión Informativa, PRIMICIA

TRABAJO DE DIPLOMA PARA OPTAR POR EL TÍTULO DE INGENIERO EN INFORMÁTICA

Autor: Carlos de Jesús Andrés González

Tutor: Ing. Ruber Hernández García

Co-Tutor: Ing. Rafael Lorente Miranda

Ciudad de La Habana, Junio del 2010

La inteligencia sin ambición es como un pájaro sin alas.

C. Archie Danielson

DEDICATORIA

Dedico este trabajo a las tres personas más grandes que existen para mí, mis padres y mi hermana.

Hablar de mis padres por separado es imposible e ilógico cuando los dos han dedicado toda su vida para darme una formación de la cual me siento orgulloso. Han sido los que me han enseñado a darle valor a las cosas y por sobre todo el concepto de unidad y familia. Gracias a ustedes puedo diferenciar entre lo bueno y lo malo, no porque me lo dijeron, sino porque me dieron la libertad para poder verlo con mis propios ojos. Nunca me señalaron el camino que debía seguir, por su parte siempre me dieron el apoyo para que tomara el que quisiera, confiando siempre en mi decisión; creo que tome el mejor camino al llegar hasta aquí, dándoles la satisfacción de ver en mi la persona que deseaban que fuera, aunque nada de lo hecho hasta el momento retribuye ni lo más mínimo de lo que han dado por mí.

De mi hermana que decir, eres todo para mí al igual que mis padres y junto a ellos me has dado la fortaleza para salir adelante ante los problemas que tenido que enfrentar en la vida, has sabido hacer tuya mis alegrías y derrotas.

Gracias por cada gota de sudor y cada alegría que hemos vividos juntos para formar esta maravillosa familia.

AGRADECIMIENTOS

Gracias a Dios por acompañarme en cada una de las dificultades y éxitos que he tenido durante mis años de estudiantes.

A mis padres y mi hermana que tanto esfuerzo y empeño han puesto para que yo pueda haber hecho este sueño realidad.

A Fidel y la Revolución Cubana por su visión inmejorable de crear esta universidad, la cual me ha permitido formarme como profesional.

A esa persona tan maravillosa que ha sido mi novia durante los tres últimos años, por todo el amor, comprensión y dedicación que ha tenido hacia mí, llenando de luz cada instante de mi vida.

A Ruber por ser mi tutor y la persona que me dio la oportunidad de formar parte del proyecto PRIMICIA, así como toda la ayuda, comentario y consejo que recibí de su parte, y que a pesar de encontrarse lejos nunca se desligó del trabajo que yo realizaba.

A Rafael por estar conmigo en cada uno de los cortes y seguir mi trabajo desde el seno del proyecto.

A todos los que han formado parte del tribunal que me han evaluado y a Enrique que más que mi oponente ha sido otra ayuda para obtener un mejor trabajo.

A todo los que han formado parte de PRIMICIA y de los cuales siempre algo aprendí.

A mis compañeros de la universidad a lo largo de estos cinco años, por los momentos que pasamos juntos, así como los profesores que con su buen ejemplo contribuyeron a mi formación.

A todas las buenas personas que siempre han confiado en mí, brindándome su apoyo incondicional en los momentos más difíciles.

DECLARACIÓN DE AUTORÍA

Yo Carlos de Jesús Andrés González declaro que soy el único autor del trabajo titulado: “Diseño del Subsistema de Configuración de la Plataforma de Televisión Informativa, PRIMICIA” y autorizo al proyecto PRIMICIA, de la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

Para que así conste firmo la presente a los 6 días del mes de julio del año 2010.

_____

Autor: Carlos de Jesús Andrés González

_____ _____

Tutor: Ing. Ruber Hernández García Co-Tutor: Ing. Rafael Lorente Miranda

OPINIÓN DEL TUTOR

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

Título: Diseño del Subsistema de Configuración de la Plataforma de Televisión Informativa, PRIMICIA.

Autor: Carlos de Jesús Andrés González

El tutor considera que durante el desarrollo del presente trabajo el estudiante mostró las cualidades que a continuación se detallan.

Durante las etapas de investigación y desarrollo ha demostrado un alto nivel de independencia y laboriosidad, expresado en los aspectos de avanzada que ha tenido que estudiar para desarrollar los objetivos del trabajo, entre los que se encuentran contenidos de configuración de sistemas, diseño de software utilizando framework, tecnologías de interoperatibilidad e interfaces de sistemas, internacionalización de productos informáticos, entre otros que han contribuido a la formación profesional e investigativa del mismo.

Las soluciones a los diferentes obstáculos que aparecieron demostraron una elevada creatividad y originalidad que se reflejó en el compromiso individual, apoyado en un arduo y continuo trabajo que demostró la gran responsabilidad asumida por el estudiante ante las disímiles dificultades que impuso el desarrollo de la tesis.

En la obtención de la solución el autor mostró una gran receptividad ante las críticas y sugerencias efectuadas, así como capacidad para apropiarse de los cocimientos que necesitaba para lograr la mayor calidad en los resultados alcanzados. Demostró además buenas cualidades para integrarse y dirigir el trabajo en equipo, lo que permitió obtener un subsistema capaz de acoplarse al producto y revolucionar las funcionalidades del mismo.

El documento redactado cumple con los requisitos establecidos, completándose en su totalidad y con una excelente calidad los objetivos trazados para la investigación. Los resultados alcanzados presentan un alto nivel científico, proporcionándole al producto PRIMICIA un elevado grado de valor agregado, lo que posibilitará una satisfactoria expansión en el mercado nacional e internacional.

Por lo anteriormente expresado considero que el estudiante ha cumplido los objetivos propuestos, está listo para ejercer como Ingeniero en Ciencias Informáticas, y propongo que se le otorgue al Trabajo de Diploma, la calificación máxima de **5 puntos.**

_____

Tutor: Ing. Ruber Hernández García

RESUMEN

La televisión ha sido el medio de comunicación que mas rápido ha evolucionado desde que se realizaran las primeras emisiones públicas por la cadena de televisión inglesa BBC en 1927 hasta nuestros días. El número de adeptos que ha logrado incorporar año tras años ha sido increíble, convirtiéndose rápidamente en el medio de preferencia mundial.

Con la evolución de la informática específicamente de Internet, la televisión encontró un fuerte adversario que va creciendo con más fuerza cada día e incorpora nuevas formas de atraer al público. Por su parte la televisión, lejos de ir en detrimento ha logrado aliarse a la informática para mejorar cada uno de sus procesos, posibilitando la emisión de materiales de mejor calidad y encaminada a la interactividad con los televidentes. La Plataforma de Televisión Informativa PRIMICIA es un ejemplo de las ventajas que ofrece la integración de la televisión y la informática, dicho sistema permite gestionar y transmitir noticias en diferentes formatos, a través de una red de televisión y haciendo uso de medios informáticos.

A partir de numerosos inconvenientes detectados en el proceso de configuración de PRIMICIA, producto que se mantiene en constante evolución en cada ciclo de desarrollo, se determinó la ausencia de un conjunto de funcionalidades que permitan la configuración del producto. Guiado por la metodología de desarrollo de software RUP, se construyeron los principales artefactos para obtener el diseño de un Subsistema de Configuración que satisface los problemas actuales de la plataforma, quedando plasmados en el presente trabajo.

PALABRAS CLAVES

PRIMICIA, Configuración, Subsistema, Funcionalidades, Diseño.

DATOS EN INGLÉS

Title: Design Configuration Subsystem for Informative Television Platform, PRIMICIA.

Author: Carlos de Jesús Andrés González.

Tutor: Ing. Ruber Hernández García.

Co-Tutor: Ing. Rafael Lorente Miranda.

Television has been the mean of communication that has developed faster since the first broadcasts were made public by the british television BBC in 1927 until today. The number of followers has been amazing, rapidly becoming the preferred medium of worldwide. With the evolution of information technology, specifically Internet, television found a strong opponent who grows stronger every day, and incorporate new ways for attracting the public. For its part the television, has made an alliance with information technology to improve every one of its processes, enabling the emission of better quality materials and designed for interactivity with viewers.

Informative Television Platform, PRIMICIA, is an example of the advantages of the integration of television and computing. This system lets you manage and deliver news in different formats, across the TV network making use of computers. After several inconvenient found in the process of setting PRIMICIA, it's been determined the absence of a set of features that allow a correct configuration. Guided by the software development methodology RUP, major artifacts were built for the design of a configuration subsystem that meets the current problems of the platform; they are included in this work.

ÍNDICE

INTRODUCCIÓN

El desarrollo alcanzado por la informática en los últimos años, ha hecho posible su integración a disímiles escenarios de la vida humana, uno de esos escenarios han sido los medios de comunicación y en especial la televisión. Debido a la forma de atraer al público, la televisión (TV) ha logrado gran popularidad, llegando a ser durante años el medio de preferencia mundial. Los resultados obtenidos de dicha integración, ha hecho posible la transmisión de materiales audiovisuales de mayor calidad, con nuevos valores añadidos de interactividad y procesos de gestión de medias más eficientes. La Plataforma de Televisión Informativa PRIMICIA, desarrollada en el Centro Productivo de la Facultad 9 en la Universidad de las Ciencias Informáticas (UCI), es un ejemplo de las ventajas que brinda la alianza informática y televisión para el proceso de transmisión y gestión de noticias en diferentes formatos.

PRIMICIA es el resultado de la generalización de un conjunto de soluciones informáticas creadas para la transmisión de noticias. Sus principales antecedentes radican en Señal 3, sistema informativo creado para la red de televisión interna de la UCI; Señal ACN, sistema para la transmisión de noticias a los colaboradores cubanos en el exterior y habitantes de las zonas de silencio de la geografía cubana; y TV Energía, desarrollado para mantener informados a los trabajadores y visitantes de la sede central del Ministerio del Poder Popular para la Energía y Petróleo de Venezuela (MENPET). La evolución y desarrollo de estos sistemas a la medida, introdujeron importantes cambios en los conceptos que los sustentan, logrando finalmente definir un producto informático basado en software libre, capaz de proveer un canal de televisión informativo.

Es difícil hablar de un producto de software que no permita ser configurado a conveniencia del usuario final, debido a las diferentes incompatibilidades que puedan existir entre el flujo de trabajo planteado por la empresa que lo produce y lo que realmente necesita el usuario que adquiere el software. Teniendo en cuenta los objetivos de convertir a PRIMICIA en un producto que pueda ser adaptado a las necesidades de cada uno de los clientes que lo adquiera, se ha determinado el siguiente **problema a resolver,** la Plataforma de Televisión Informativa PRIMICIA no cuenta con las funcionalidades que permitan su configuración. Como **objeto de estudio** para la realización del presente trabajo se tomaron en cuenta los procesos relacionados con la Plataforma de Televisión Informativa PRIMICIA. Su **campo de acción** está centrado en las funcionalidades de configuración de la Plataforma de Televisión Informativa PRIMICIA.

Se planteó como **objetivo general** del trabajo: diseñar un subsistema que permita la configuración de la Plataforma de Televisión Informativa PRIMICIA. Para dar cumplimiento al objetivo general se trazaron los siguientes **objetivos específicos**:

- Definir los conceptos teóricos – técnicos asociados a la configuración de PRIMICIA.
- Definir las funcionalidades que permitan la configuración del producto.
- Modelar la solución propuesta.

Como guía para dar cumplimiento a los objetivos se definieron, las siguientes **tareas** de investigación:

1. Caracterizar los procesos relacionados con la configuración de la Plataforma de Televisión Informativa PRIMICIA a partir del estudio de la conceptualización realizada.
2. Caracterizar el diseño y la arquitectura de software de la Plataforma de Televisión Informativa PRIMICIA.
3. Identificar los requisitos funcionales y no funcionales vinculados a los procesos de configuración de PRIMICIA.
4. Seleccionar herramientas y metodologías para el desarrollo.
5. Diseñar las funcionalidades identificadas.

Teniendo en cuenta lo anterior, se plantea la siguiente **idea a defender**: si se logra diseñar el subsistema de configuración de PRIMICIA, se garantizará que la plataforma cuente con las funcionalidades necesarias que permitan la configuración de la misma.

Para cumplir con las tareas de investigación del trabajo se utilizaron los siguientes métodos científicos.

Métodos teóricos.

- **Analítico – Sintético:** Aportó la posibilidad de buscar la esencia de los fenómenos, los rasgos que caracterizan y distinguen los procesos relacionados con la configuración de PRIMICIA. Dentro de la investigación ha permitido la extracción de los elementos más importantes que se relacionan con la plataforma.

- **Inductivo – deductivo:** Permitió llegar a un grupo de conocimientos generalizadores acerca de los posibles elementos configurables en la plataforma, así como determinar aquellos procesos más específicos implicados en los nuevos cambios que se necesitan para una correcta gestión de la configuración.

Métodos empíricos.

- **Observación:** Este método ha sido utilizado en distintos momentos de la investigación para recoger la información de cada uno de los conceptos o variables definidas en la idea a defender.

- **Entrevista:** Este método fue utilizado para conocer diferentes aspectos que no quedaron bien definidos en trabajos investigativos realizados con anterioridad, para lo cual fueron necesarias entrevistas con los principales desarrolladores de PRIMICIA con el objetivo de profundizar en los elementos y problemática a tratar en la presente investigación.

El trabajo de diploma está estructurado de la siguiente forma:

Capítulo 1. Fundamentación teórica y características de PRIMICIA, durante este capítulo, se lleva a cabo un estudio de los conceptos generales asociados al tema de investigación. Se definen el objeto de estudio de la investigación, las características generales de PRIMICIA, los procesos relacionados con la configuración, la arquitectura y el diseño actual de la plataforma. También se aborda con mayor profundidad la situación problemática relacionada con la investigación y se analizarán las características de configuración de algunos de los sistemas gestores de contenidos.

Capítulo 2. Requisitos y propuesta de solución a la configuración de PRIMICIA, mediante este capítulo se realizará un análisis de la metodología de desarrollo de software, el lenguaje de modelado y la herramienta seleccionada para desarrollar la propuesta de solución. Se realiza el modelo de dominio concerniente al problema planteado, así como un levantamiento de los requisitos de software. Se describe la propuesta de solución a través de los actores, casos de uso y la descripción extendida de cada uno de los caso de uso.

Capítulo 3. Análisis y Diseño del nuevo Subsistema de Configuración de PRIMICIA, en este capítulo se realiza un del *framework*[1] Symfony en cuanto a su arquitectura y algunos de los patrones de diseños que implementa, para de esta forma lograr un mejor entendimiento del diseño desarrollado. Se construirá el Modelo de Diseño a través de los diagramas de clases del diseño y clases persistentes. Y por último se dejan plasmados los cambios a realizar en el modelo de datos para garantizar el funcionamiento correcto de las nuevas funcionalidades.

[1] En español, marco de trabajo es una estructura de soporte definida en el cual un proyecto de software puede ser organizado y desarrollado.

CAPÍTULO 1

Fundamentación teórica y características de PRIMICIA

1.1. Introducción.

En este capítulo se lleva a cabo un estudio de los principales conceptos asociados al tema de investigación. Se define el objeto de estudio de la investigación, las características y funcionalidades generales del producto, se caracterizan los procesos actuales relacionados con la gestión de la configuración, así como el diseño y la arquitectura que sustenta a PRIMICIA. Además se analiza con mayor profundidad la situación problemática que dio surgimiento a la presente investigación y la necesidad de darle solución. Para finalizar el capítulo se hace un estudio de las características presentadas por algunos gestores de contenido y la posibilidad de ser incorporadas a la Plataforma de Televisión Informativa.

1.2. Principales conceptos asociados al tema de investigación.

Siempre que se plantea la tarea de realizar una investigación surgen palabras esenciales y de gran peso a lo largo del tema abordado; estas constituyen conceptos y es de suma importancia su comprensión. Para el presente trabajo se hace necesario abordar tres conceptos fundamentales asociados a la problemática tratada en la investigación. Uno de ellos es módulo, por la importancia que tiene comprender el alcance y envergadura del mismo, debido a la fuerte probabilidad que existe de que las funcionalidades afines sean agrupadas a través de módulos. Mientras que configuración, es el resultado de lo que se pretende lograr con dicho subsistema y con cada uno de los módulos de manera individual. Por lo que la unión de estos conceptos constituye la esencia de todo el trabajo. También se da una noción de lo que son los sistemas de gestión de contenidos, por la estrecha relación que existe entre las características que estos presentan y las nuevas funcionalidades que se pretenden diseñar para la plataforma.

1.2.1. Módulo.

Un módulo es una pieza o un conjunto unitario de piezas que se repiten en una construcción de cualquier tipo, para hacerla más fácil, regular y económica. (1)

Según el Diccionario de la Real Academia de la Lengua Española (DRAE) el término módulo, en cuestiones de programación se define:

> *"Un módulo es un software que agrupa un conjunto de subprogramas y estructuras de datos. Los módulos son unidades que pueden ser compiladas por separado y los hace reusables y permite que múltiples programadores trabajen en diferentes módulos en forma simultánea, produciendo ahorro en los tiempos de desarrollo. Los módulos promueven la modularidad y el encapsulamiento, pudiendo generar programas complejos de fácil comprensión".* (2)

El módulo por lo tanto forma parte de un sistema y mantiene algún tipo de relación o vínculo con el resto de los componentes. Lo modular es fácil de ensamblar y suele ofrecer una amplia flexibilidad no en sus componentes, sino en la manera de armado. Por otra parte, el producto final o sistema puede ser reparado, si se repara el módulo o componente que no funciona. Se conoce como modularidad a la capacidad de un sistema para ser entendido como la unión de varios componentes que interactúan entre sí y que son solidarios, donde cada uno cumple con una tarea en pos de un objetivo común. (3)

La palabra módulo, es muy utilizada en diversas ramas de las ciencias informáticas, su uso se encuentran asociado a la programación, la ingeniería de software y los sistemas operativos entre otros.

En programación un módulo es una parte de un programa, que realizará una o varias tareas del total que debe realizar el programa para cumplir con sus objetivos. Normalmente suelen estar organizados jerárquicamente en niveles, de forma que los módulos de nivel superior realizan las llamadas a los de nivel inferior. Cuando un módulo es llamado, recibe como entrada los datos proporcionados por el de nivel superior que ha hecho la llamada, realiza su tarea, y cuando finaliza devuelve la salida pertinente al módulo superior que lo llamó, un módulo puede llamar a otro u otros módulos de nivel inferior o del mismo nivel, pero nunca uno superior.

En la ingeniería de software, un módulo es una parte de un sistema, que realiza una o varias tareas del conjunto total que se pueden ejecutar, para cumplir con un objetivo específico dentro del mismo y contribuir a cumplir con los objetivos generales de dicho sistema.

En un sistema operativo, un módulo suele ser un conjunto de rutinas que realizan funciones a nivel de sistema, y que pueden cargarse y descargarse dinámicamente desde el núcleo cuando sea requerido. Con frecuencia contienen controladores de dispositivos o interfaces.

De manera general se puede decir que un módulo no es más que una pequeña estructura que agrupa un subconjunto de funcionalidades. Y que su objetivo es hacer más fácil la construcción y mantenimiento de un software. Permite una mayor compresión del trabajo a realizar, por parte de los analistas de sistema y programadores. Además sirve de meta para medir los esfuerzos de un equipo de proyecto.

Cada uno de los módulos de un programa, idealmente deberían cumplir las siguientes características:

Tamaño pequeño: Permite aislar el impacto que pueda tener un cambio en el sistema, ya sea en el momento de corregir un error, o en el rediseño de una parte del código.

Independencia modular: Implica que sea más fácilmente trabajar con ellos, para desarrollar un módulo no es necesario conocer detalles internos de otros.

1.2.2. Configuración.

Según el Diccionario de la Real Academia de la Lengua Española (DRAE) el término configuración se define como:

> *"la disposición de las partes que componen una cosa y le dan su peculiar forma y propiedades anejas"*. (4)

Mientras que su definición para los aspectos relacionados con la informática dado por la DRAE precisa:

> *"conjunto de los aparatos y programas que constituyen un sistema informático".* (4)

Por su parte el Diccionario de Sinónimos y Antónimos relaciona el siguiente conjunto de palabras análogas a configuración:

> *"forma, figura, conformación, estructura, ordenación, distribución, composición"*. (5)

Para la actual investigación se define como:

> *"el conjunto de elementos que determinan la composición de un programa o sistema informático, son opciones que generalmente se establecen cuando se instala un software, pero también pueden implantarse en cualquier otro momento".*

Si un programa o sistema carga la configuración por defecto se puede decir que la misma está predeterminada, si por el contrario la define el usuario podernos decir que se está haciendo uso de una configuración personalizada.

La configuración predeterminada es la que no se ha definido aún, generalmente no es la más recomendada debido que a través de la misma se trata de generalizar todo lo posible, para ser más adaptable a la mayoría de los entornos. Una configuración predeterminada tiene que estar preparada para:

- Seguridad media.
- Generalmente en inglés.
- Nivel de interfaz de usuario medio.
- Usuarios de todas las edades y ambos sexos.

Una configuración personalizada es aquella que será definida por el usuario, para que el sistema o software se adapte mejor al entorno de trabajo donde será usado, esta es guardada generalmente en un archivo o en una base de datos, puede estar cifrada para que solo se pueda modificar por el programa a configurar, o puede ser texto plano para que también se pueda modificar sin depender del programa.

1.2.3. Sistemas de Administración de Contenidos.

Los primeros sistemas de administración de contenidos aparecieron por la necesidad que surgió en las organizaciones que publicaban una gran cantidad de información en Internet y necesitaban de continuas actualizaciones de la información que mostraban; un ejemplo de esto son: revistas en línea, periódicos y publicaciones corporativas. En 1995 el sitio de noticias tecnológicas CNET, sacó su sistema de administración de documentos y publicación, creando una compañía llamada Vignette, pionero de los sistemas de administración de contenidos comerciales. El rápido crecimiento y la extensión de la Internet, la construcción de portales con más contenido y la alta participación de los usuarios directamente a través de blogs y redes sociales, convirtió a los gestores de contenidos en una herramienta esencial en la red de redes, tanto para empresas e instituciones como para las personas sin conocimientos de programación.

El gestor de contenidos es una aplicación informática usada para crear, editar, gestionar y publicar contenido digital en diversos formatos. El gestor de contenidos genera páginas dinámicas

interactuando con el servidor para generar la página web bajo petición del usuario, con el formato predefinido y el contenido extraído de la base de datos del servidor.

Un CMS[2] es un programa que permite crear una estructura de soporte para la creación y administración de contenidos, principalmente en páginas web, por parte de los participantes. Consiste en una interfaz que controla una o varias bases de datos donde se aloja el contenido del sitio. El sistema permite manejar de manera independiente el contenido y el diseño. Así, es posible manejar el contenido y darle en cualquier momento un diseño distinto al sitio sin tener que darle formato al contenido de nuevo, además de permitir la fácil y controlada publicación en el sitio a varios editores. Un ejemplo clásico son los editores que cargan el contenido al sistema y otro de nivel superior que permite que estos contenidos sean visibles a todo el público.

Un CMS siempre funciona mediante el servidor web donde se encuentre alojado el portal. El acceso al gestor se realiza generalmente a través de un navegador web mediante el protocolo HTTP[3], y se puede requerir el uso de FTP[4] para subir contenido al mismo. Cuando un usuario accede a una URL[5], se ejecuta en el servidor esa llamada, se selecciona el esquema gráfico y se introducen los datos que correspondan de la base de datos. La página son generadas dinámicamente para cada usuario, el código HTML[6] final se genera en cada llamada. Normalmente en el gestor se predefine varios formatos de presentación del contenido, para brindar mayor flexibilidad a la hora de crear nuevos apartados e informaciones.

Los CMS brindan una gran cantidad de ventajas, a continuación se mencionan algunas de ellas:

- Facilita el acceso a la publicación de contenidos a un rango mayor de usuarios.
- Permite que sin conocimientos de programación ni maquetación cualquier usuario pueda indexar contenido en sitio web.

[2] Del inglés *Content Managment System*, en español sistema de administración de contenidos.

[3] Del Inglés *Hypertext Transfer Protocol*, en español Protocolo de transferencia de hipertexto, usado en cada transacción de la web.

[4] Del Inglés *File Transfer Protocol*, en español Protocolo de transferencia de archivos, usada en internet para el intercambio de archivos entre dos sistemas conectados a una red TCP.

[5] Del Inglés *Uniform Resource Locator*, localizador uniforme de recursos, es una secuencia de caracteres de acuerdo a un formato estándar que se usa para nombrar recursos en Internet

[6] Del Inglés *HyperText Markup Language*, Lenguaje utilizado para la creación de documentos de hipertexto e hipermedia. Es el estándar usado en el *World Wide Web*.

- Permite la gestión dinámica de usuarios, permisos, y la colaboración de varios usuarios en el mismo trabajo.
- La interacción de los usuarios mediante herramientas de comunicación.
- La maquetación es hecha al inicio del proceso de implantación del gestor de contenidos.
- Los costes de gestión de la información son muchos menores porque se elimina un eslabón de la cadena de publicación, el maquetado.
- Permiten que la actualización y reestructuración de los sitios web sean mucho más sencillas, al tener todos los datos del portal y los contenidos en una base de datos estructurada en el servidor.

1.3. Caracterización de los procesos de configuración PRIMICIA.

PRIMICIA es un sistema basado en dos aplicaciones web, una destinada para la administración de los contenidos audiovisuales y la otra para la transmisión de los mismos. A pesar de estar dividido, actúan como un todo para dar solución a la transmisión de noticias mediante una red de televisión, haciendo uso de medios informáticos. En ambas aplicaciones se realizan un conjunto de procesos que determinan el resultado final del sistema. A continuación se describen las funcionalidades de la plataforma y los procesos críticos relacionados con la configuración.

1.3.1. Descripción general de las funcionalidades de PRIMICIA.

Las dos aplicaciones que conforman PRIMICIA se relacionan entre sí, mediante una misma base de datos. El Subsistema de Administración permite que los usuarios del sistema puedan realizar la gestión de las noticias y recursos audiovisuales del canal. El Subsistema de Transmisión trabaja de manera automática y es el encargado de visualizar las noticias y materiales publicados, así como cualquier elemento adicional que sirva de información a los televidentes. A continuación de se mencionan las funcionalidades que realiza cada uno de los subsistemas.

El Subsistema de Administración tiene las siguientes prestaciones generales:

1. **Gestión de los usuarios del sistema,** permite adicionar y eliminar usuarios, así como establecer y modificar los permisos de acceso en el sistema.

2. **Gestionar las secciones temáticas del canal,** permite establecer el orden de las secciones, horario en que serán mostradas y habilitarlas o deshabilitarlas.

3. **Funcionalidades para la redacción de noticias,** según los formatos definidos para las pantallas; la publicación de las noticias teniendo en cuenta fecha de inicio y fin de la publicación; la gestión de las noticias del canal que permitan modificar, eliminar y archivar las noticias; la administración del archivo de noticias del canal que permitan reutilizar y eliminar las mismas.

4. **Gestionar Recursos Multimedia,** posibilita el almacenamiento, administración y reproducción de recursos multimedia como imágenes, música y video.

5. **Funcionalidades para la creación y administración de cintillos informativos o infocintas,** la administración de los cintillos establece el orden de prioridad de muestra y la habilitación o deshabilitación de los mismos.

6. **Generación de reportes sobre la actividad del sistema,** los reportes se realizarán sobre la actividad de los trabajadores del sistema, permite realizar búsquedas de noticias publicadas atendiendo distintos criterios como fecha de publicación, temática, palabras claves y título. Ofrece facilidades para la impresión de los reportes y la exportación de los reportes a formato digital.

7. **Administración de la señal del canal,** lo cual permite cambiar entre la señal del canal y la televisión en vivo de un canal externo.

El Subsistema de Transmisión presenta las siguientes funcionalidades:

1. **Generar una cartelera,** del ciclo de transmisión mostrando para cada noticia la sección temática y el titular, en el orden que se visualizarán.

2. **Visualizar noticias,** compuestas por pantallas de tipo texto, texto-imagen, imagen y video.

3. **Reproducir fondo musical,** mientras se muestran las noticias, excepto cuando se muestre un video.

4. **Mostrar comentarios,** que oriente al televidente acerca de lo que está observando en las pantallas de tipo imagen.

5. **Mostrar cintillos informativos o infocintas,** para promocionar eventos de última hora o acontecimientos de gran importancia.

6. **Mostrar información adicional a la noticia,** la fecha, hora, tiempo restante de la pantalla y titular de la próxima noticia y sección temática.
7. **Transmitir televisión en vivo,** proveniente de una señal externa.
8. **Mostrar patrón del canal,** cuando este se encuentre fuera de servicio.

1.3.2. Procesos relacionados con la configuración de PRIMICIA.

En el sub-epígrafe anterior se mencionaron un conjunto de funcionalidades presentes en PRIMICIA, las cuales fueron ubicadas en dos subsistemas diferentes de acuerdo a los objetivos que persiguen las mismas. Cada una de las funcionalidades que se mencionaron representa procesos llevados a cabo en el sistema, los cuales interactúan entre sí para que el sistema funcione como un todo.

En la **Figura 1** se muestra a un alto nivel como se relacionan de forma gráfica dichos procesos. Las relaciones entre los mismos se representan por líneas en forma de saetas, que indican como ocurre direccionalmente el flujo de ejecución de los procesos y la dependencia de estos; que a pesar de encontrarse separados existe una estrecha relación y cierto grado de dependencia entre los mismos para lograr un correcto funcionamiento de PRIMICIA. Aunque es válido mencionar que algunos de ellos gozan de gran independencia en su funcionamiento individual, como pueden ser Gestionar Usuarios, Generar Reportes, y Mostrar el Patrón del Canal, hecho que ocurre cuando el canal esta fuera de servicio, se presente un fallo en la transmisión o no existan noticias para transmitir.

Una vez que un usuario haya pasado de forma satisfactoria por el proceso de Autenticar Usuarios, podrá acceder a las funcionalidades habilitadas por el sistema según su rol. Entre estas se encuentran Gestionar Usuarios, Generar Reportes, Gestionar Infocintas, Administrar Recursos Multimedia, Redactar Noticias, Gestionar Secciones Temáticas, y Administrar Señal Externa.

Redactar Noticias está fuertemente ligado a Gestionar Secciones Temáticas y Administrar Recursos Multimedia, debido a que cuando se redacta una noticia, se debe especificar a qué sección temática pertenece y si tiene asociado algún recurso multimedia, ambos previamente gestionados. De igual manera ocurre con Generar Cartelera y Mostrar Información Adicional que dependen de Visualizar Noticias y Gestionar Secciones Temáticas, debido a que para realizar su trabajo requieren conocer todo lo referente a las secciones temáticas y las noticias que están listas para visualizar.

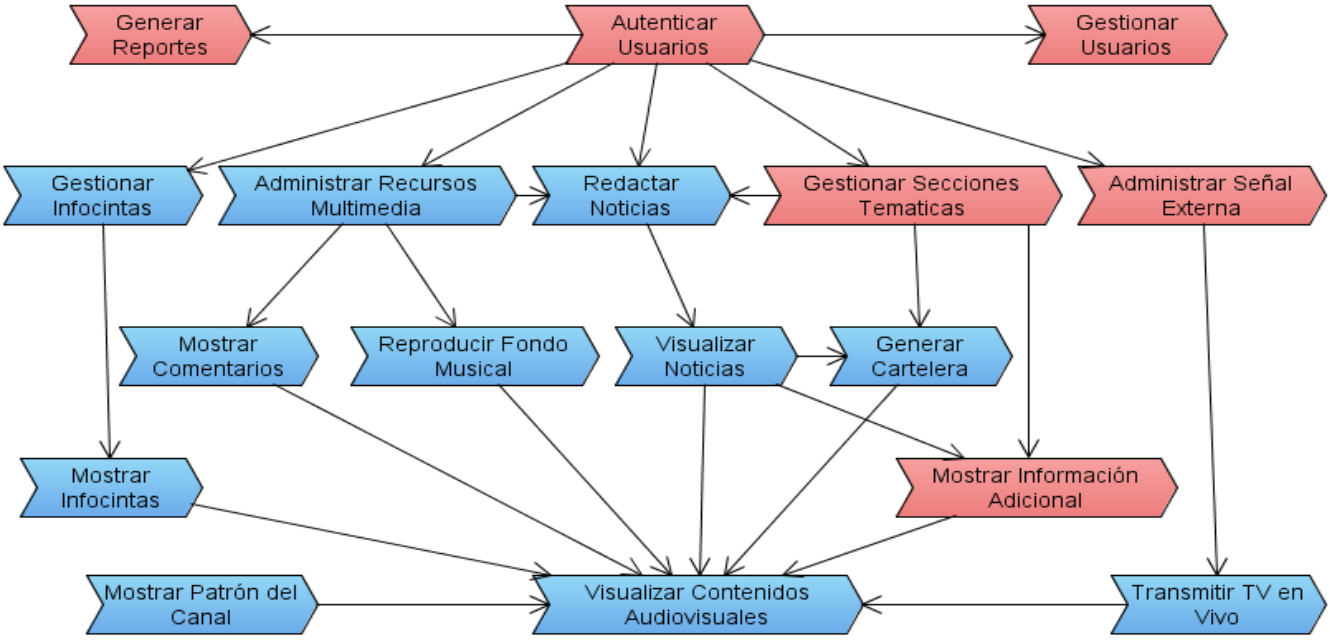

Figura 1: Diagrama de los Procesos Generales de PRIMICIA.

Mostrar Infocintas, Mostrar Comentarios, Reproducir Fondo Musical, Visualizar Noticias, Generar Cartelera de Transmisión, Mostrar Información Adicional, y Transmitir Señal Externa, son procesos llevados a cabo para transmitir las noticias y todo los recursos adicionales que la acompañan, cada uno de ellos depende de la previa ejecución de uno de los procesos pertenecientes a la administración del canal.

El proceso Visualizar Contenidos Audiovisuales, representa de manera general el resultado visual que se puede obtener de la transmisión, no siempre se ejecutan procesos como Mostrar Infocintas, Reproducir Fondo Musical, Transmitir Señal Externa y Mostrar el Patrón del Canal.

Durante el estudio realizado en el trabajo de diploma, “Conceptualización de la automatización de la personalización y configuración de PRIMICIA” del autor Levián Lara Gómez, se determinaron aquellos procesos que son necesarios cambiar para lograr una correcta configuración de la plataforma. Partiendo del hecho que los elementos configurables dentro de la plataforma, son aquellos que pueden ser cambiados de acuerdo a las necesidades de los usuarios que la administren y no repercutan de manera negativa en su funcionamiento, en la **Figura 1** se marcaron en rojo los procesos actuales que debían ser cambiados.

A continuación se analizan los procesos críticos vinculados con configuración teniendo en cuenta lo planteado por el autor antes mencionado.

- Actualmente cuando la plataforma es desplegada se debe definir el tipo de autenticación a utilizar (utilizando dominio o autenticación local). Una de las mejoras que se pudiera incorporar es dar la posibilidad de permitir al usuario determinar el tipo de autenticación que quiere utilizar. (6)

Se hace necesario que el proceso de Autenticar Usuario pueda ser configurado, de manera automática y transparente para utilizar cualquiera de las dos autenticaciones antes mencionadas y que sea el administrador de la plataforma quien determine cuál de las variantes utilizar.

- Los roles de usuarios fueron impuestos por el equipo de desarrollo, y a los usuarios del sistema solo se le podrá asignar dichos roles, esta situación no permite que los administradores puedan asignarles a los usuarios los permisos que ellos realmente estimen convenientes. (6)

Dentro del proceso de Gestión de Usuario, solo se permite insertar, modificar y eliminar usuarios, además de asignarle permisos según los roles predefinidos, un aspecto importante es que el sistema pueda permitir la gestión de los roles con que se trabajará en la aplicación, de acuerdo al flujo de trabajo que pretenda seguir la organización que haga uso de la plataforma.

- Las secciones usadas en la plataforma se definen antes de que el sistema sea instalado, quedando de esta manera siempre asociada una sección a un logotipo antes negociado con el cliente. Esto trae como consecuencia que no se puedan adicionar, modificar y eliminar aquellas que ya no se estén utilizando. Se propone incorporar la funcionalidad de gestionar secciones para posibilitar dicha tarea de forma automática además de permitir modificar el logotipo asociado a la misma de acorde a las necesidades del cliente. (6)

La Gestión de Secciones Temáticas actualmente solo permite establecer el orden en que serán transmitidas, no permite que se creen nuevas secciones, modifiquen y eliminen; lo que trae como consecuencia que en la mayoría de las ocasiones no se correspondan las existentes con el tipo de información que quiere mostrar las nuevas organizaciones que haga uso de PRIMICIA.

- Los enlaces con televisoras externas son especificados directamente en el código fuente. Para ellos se propone la creación de la funcionalidad de gestionar enlaces para una mayor independencia de los enlaces escritos en el código fuente dando la posibilidad poder determinar las distintas televisoras a enlazarse en un determinado momento. Adicionar nuevos enlaces, modificarlos en caso de sufrir algún cambio o eliminarlos de la plataforma una vez no estén más disponibles. (6)

En total acuerdo a la idea anterior, se hace necesario profundizar en esta funcionalidad para que se pueda determinar cuál son los canales con los que enlazará la plataforma para una transmisión en vivo, ya que actualmente solo permite especificar la hora del cambio de transmisión sobre los canales preestablecidos.

- Las informaciones adicionales a la noticia como hora, tiempo restante, número de pantalla, tienen un orden establecido directamente en el código fuente. Esto imposibilita que se puedan ordenar, eliminar o agregar a voluntad, por lo que se propone implementar la funcionalidad gestionar elementos de panel lo que permitirá realizar dichas tareas de forma automática. (6)

Realmente se desea que todo el proceso Mostrar Información Adicional pueda ser configurado de manera que la plataforma pueda utilizar de los servicios web ofrecidos por terceros.

- Generar Reportes, solo genera 3 tipos de reportes, que pueden o no, ser de interés para la entidad que hace uso de PRIMICIA, lo cual requiere que sea redefinido este proceso para obtener mejores resultados, y mayor aporte para la entidad que haga uso del la plataforma.

1.4. Caracterización del Diseño y la Arquitectura de PRIMICIA.

Para entender de manera correcta el funcionamiento de PRIMICIA durante los dos sub-epígrafes a continuación, se hace una caracterización en cuanto al diseño y arquitectura. Entre los que se analiza el patrón arquitectónico utilizado en la plataforma; las herramientas y lenguajes de programación que sustentaron su construcción; las librerías y componentes utilizados; además del rendimiento del gestor de base datos escogido para gestionar toda la información persistente en la plataforma.

1.4.1. Características del Diseño.

PRIMICIA está formado por dos subsistemas y distribuidos en varios paquetes que interactúan para cumplimentar un conjunto de funcionalidades. En la siguiente figura se muestra la dependencia directa entre los subsistemas que conforman la plataforma.

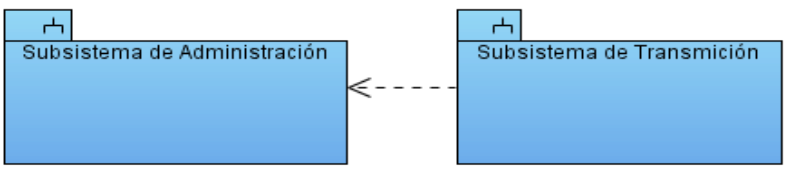

Figura 2: Distribución Lógica de PRIMICIA.

Los paquetes de diseño se utilizan para agrupar los elementos de modelo de diseño relacionados con objetivos organizativos y a menudo, para la gestión de la configuración. En los modelos grandes, el empaquetado es fundamental para facilitar su comprensión. Incluso en modelos más pequeños un empaquetado adecuado puede mejorar sensiblemente la inteligibilidad del modelo. (7)

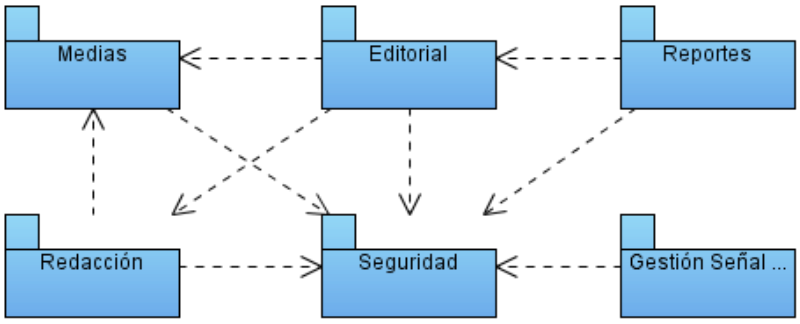

Figura 3: Diagramas de Paquetes del Subsistema de Administración.

PRIMICIA hace uso del empaquetamiento principalmente en el Subsistema de Administración, conformado por los paquetes de redacción, medias, editorial, gestión de señal del canal, seguridad y reportes, ver **Figura 3**. Mientras que el Subsistema de Transmisión está compuesto por un solo paquete que lleva por nombre Transmisión. Cada uno de estos paquetes engloba un conjunto de clases y relaciones que posibilitan el correcto funcionamiento del sistema, y permiten un mayor entendimiento por parte de los desarrolladores. A su vez estos paquetes constituyen módulos de la plataforma.

En la **Figura 3** se puede apreciar diversas dependencias cruzadas que provocan una menor flexibilidad al cambio del modelo. Aunque es válido mencionar que es muy positiva la estructuración actual de la plataforma, permitiendo una buena organización y distribución de las funcionalidades. A pesar de que la incorporación de las nuevas funcionalidades de configuración de PRIMICIA pueda ocasionar algunos cambios en este modelo, se tratará de respectar al máximo la estructura actual del sistema la cual se considera muy favorable.

1.4.2. Características de la Arquitectura.

La arquitectura de software consiste en un conjunto de patrones y abstracciones que proporcionan un marco de referencia para guiar la construcción del software, establece los fundamentos para que los analistas, diseñadores, programadores y todo el personal de desarrollo en general trabajen por una misma línea, para así poder alcanzar los objetivos del sistema. Se representa a través de sus

componentes, las relaciones entre ellos y el ambiente; así como los principios que orientan su diseño y evolución. Está centrada en los requerimientos no funcionales, que son satisfechos mediante los modelos y diseños de la aplicación; es fundamental para el éxito o fracaso de cualquier proyecto. (8)

PRIMICIA fue desarrollada bajo la perspectiva del *framework* Symfony de PHP, en su versión 1.0.11, el cual implementa el patrón arquitectónico Modelo - Vista – Controlador (MVC). MVC se encuentra en muchos de los diseños de aplicaciones reconocidas, debido a sus interfaces sofisticadas, y dando múltiples razones para su adopción. Siendo muy recomendable en situaciones donde la lógica de una interfaz de usuario cambia con más frecuencia, que los almacenes de datos y la lógica de negocio.

Este patrón divide la aplicación interactiva en 3 áreas, para ello utiliza las siguientes abstracciones:

- **Modelo**: Encapsula los datos y las funcionalidades, es independiente de cualquier representación de salida y/o comportamiento de entrada.
- **Vista**: Intercambia la información con el usuario, pueden existir múltiples vistas del modelo y cada vista tiene asociado un componente controlador.
- **Controlador**: Recibe las entradas, traducidas a solicitudes de servicio para el modelo.

PRIMICIA no interactuará con ningún otro sistema externo, su construcción está basada completamente en el uso del software libre; y siendo el resultado del empleo e integración de varias herramientas y lenguajes de programación como son: el *framework* Symfony de PHP, el gestor de base datos PostgreSQL, y los lenguajes PHP, Java Script y HTML.

El sistema está formado por un conjunto de componentes que permiten su correcto funcionamiento. Entre ellos se encuentra el uso del reproductor VLC para la transmisión vía *streaming* de los videos, y la música que acompaña las noticias. Mediante la utilización de la librería de TCPDF, la plataforma pueden exportar a formato PDF los reportes generados del trabajo con el sistema. A través de la librería LDAP[7] permite la autenticación mediante directorios activos. Mientras la utilización de CURL[8] permite la subida de archivos de medias tales como, música, imágenes y videos, mediante el protocolo FTP. En la **Figura 4**, se puede apreciar con mayor claridad la relación de cada uno de los componentes con el sistema.

[7] Del inglés *Lightweight Directory Access Protocol,* en español Protocolo Ligero de Acceso a Directorios.

[8] Herramienta para usar en un intérprete de comandos para transferir archivos con sintaxis URL.

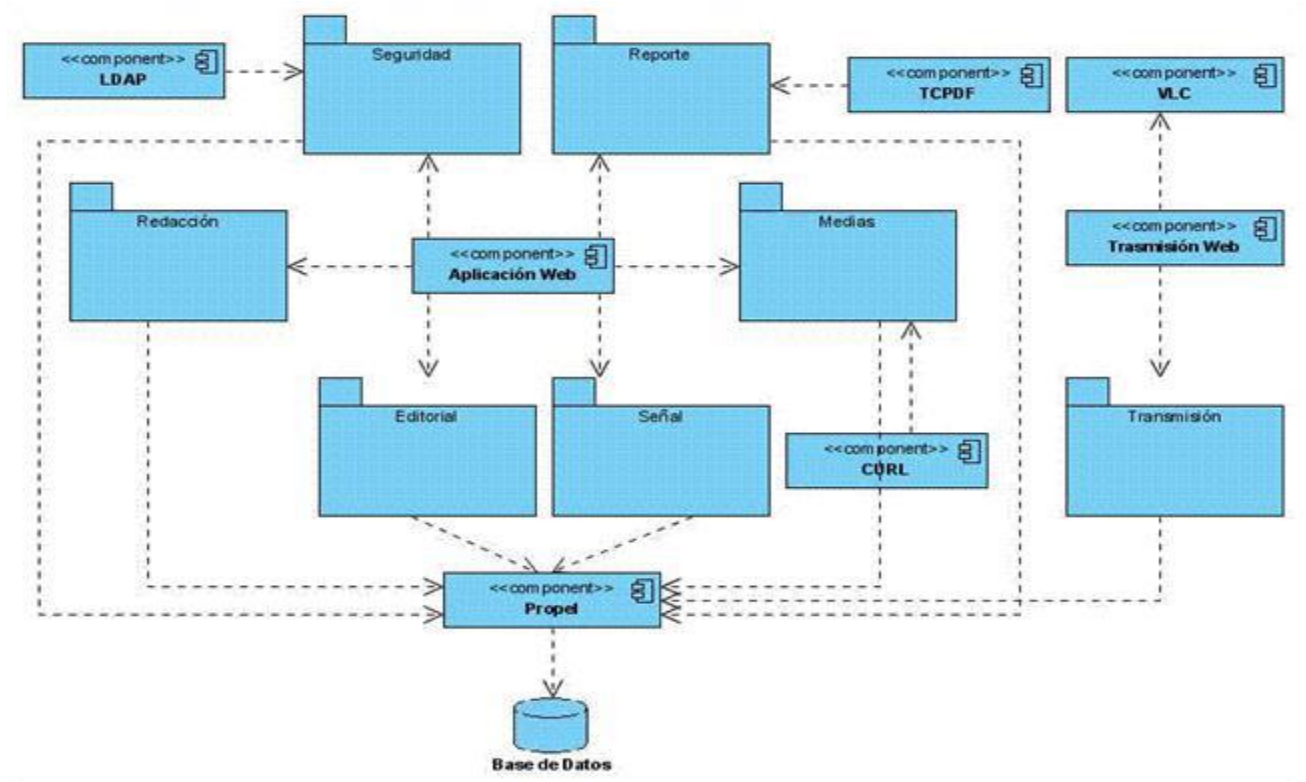

Figura 4: Diagrama de Componentes de PRIMICIA.

Una de las restricciones actuales del sistema es que solo maneja archivos de video en formato MPEG, AVI, MOV y archivos de música MP3. Mientras que las imágenes preferiblemente deben estar en formato JPEG, con proporciones mayores de 480X360.

El sistema debe contar con dos servidores, uno para la gestión de recursos y administración del canal donde estará alojado el Subsistema de Administración. El otro servidor está encargado de la transmisión del canal y es donde estará instalado el Subsistema de Transmisión. La comunicación entre ambos servidores se realizará mediante el protocolo TCP/IP con una conexión de 1 Gigabyte, para así lograr una máxima eficiencia en la transmisión de la información.

El servidor de administración será accedido mediante HTTP debido a que el sistema está implementado con tecnología web. La arquitectura física del sistema puede ser acoplada en dependencia del tipo de red de televisión que exista en la entidad que haga uso de PRIMICIA, para una mejor comprensión se propone visitar el **Anexo 2**.

El sistema consta de 23 tablas que son utilizadas por los subsistemas y módulos para permitir la persistencia de datos. Los datos a guardar crecerán considerablemente con el paso del tiempo debido

a que el sistema deberá guardar todas las noticias generadas por los redactores, las cuales contarán con músicas, imágenes y videos. La base de datos está soportada por el gestor PostgreSQL el cual llega a soportar una ilimitada cantidad de información en general y 32 Terabyte por tabla como máximo. Los recursos multimedia no se insertaran en la base de datos con el objetivo de ahorrar capacidad en la misma, solo se copiará la dirección física donde se encuentran por lo que se requiere altos niveles de almacenamiento en el servidor donde se guarde estos archivos.

La ejecución del sistema no necesita realizar consultas de datos en tiempo real; aunque permitirá algunas operaciones de interfaz de usuario que deberán ser ejecutadas en el cliente para lograr un aumento en el tiempo de respuesta del mismo. El Subsistema de Administración debe posibilitar la visualización de los recursos multimedia almacenados, por esta razón el tiempo de respuesta puede variar en dependencia de la pre-carga de estos archivos.

1.5. Situación Problemática.

La Plataforma de Televisión Informativa PRIMICIA provee un canal de televisión, el cual es soportado y transmitido haciendo uso de medios informáticos. Cuando se pensó en su desarrollo como un producto que pueda establecerse en el mercado del software, se determinaron la ausencia de algunas características vitales con que debía contar el mismo, para obtener un producto integral y que pueda ser ajustable a cualquier entorno de trabajo, como son: televisoras, agencias de noticias, empresas, terminales de transporte, hoteles y cualquier otra entidad que cuente con una red de televisión.

El inconveniente fundamental de la aplicación es su incapacidad para adaptar varias de sus características y funcionalidades de una manera automática y transparente. En muchas ocasiones este inconveniente provoca que PRIMICIA sea considerado como una solución informática y no como un producto, contrario a lo que realmente se pretende.

Actualmente cada vez que se necesita instalar el sistema en un nuevo entorno, requiere que el equipo de desarrollo tenga que modificar el código fuente para ajustar algunas de sus funcionalidades. Las modificaciones para lograr adecuar el producto ocurren tanto a nivel de interfaz de usuario, como lógica del negocio, y base de datos, por lo que el trabajo se vuelve totalmente engorroso. En la mayoría de las ocasiones a pesar de que puedan ser muy pocos los cambios, se requiere de altos costos en tiempo que provoca la pérdida de mercados y clientes para el producto, debido que todo el trabajo de configuración se tiene que hacer de forma manual.

Entre los problemas que más se repiten, se encuentra la gestión de los roles de usuario para la interacción con el canal, los cuales mayormente se ven afectado en cuanto al nombre que reciben y las funcionalidades a las que pueden acceder. Existen casos donde el trabajo con el sistema se hace muy simple en consecuencia del propósito de la entidad, por lo que solicitan no hacer uso de algunas de las funcionalidades que tiene la plataforma. El sistema transmite las noticias por secciones temáticas, las cuales tienen que ser modificadas a menudo porque muchas veces no son compatibles con los contenidos que desean mostrar las organizaciones que hacen uso del mismo. El tipo de transmisión a realizar es un punto clave para cualquier entidad, ya sea de manera continua, por bloques noticiosos, o por secciones temáticas, actualmente la aplicación solo cuenta con esta última y de querer establecer alguna de las otras dos, se necesita un cambio total de los subsistemas que componen PRIMICIA.

El tipo de autenticación es otro de los elementos cambiantes, algunos prefieren que sea a través de usuarios locales para el sistema, mientras otras organizaciones demandan que sea integrado con el dominio que poseen. Una de las ventajas que tiene la plataforma es poder enlazarse con la transmisión de cualquier canal externo, en ocasiones se ve empañada debido a que dichos canales son predefinidos en el momento que se implementa el sistema y en muchas ocasiones no son del interés de la entidad que hará uso de PRIMICIA, y solicitan que sean cambiados.

Entre las características deseadas para el producto, está la obtención de funcionalidades genéricas fácilmente escalables, que no dependan de un entorno dado y no atadas a un diseño gráfico específico; así como la necesidad de que la misma plataforma gestione su configuración, para obtener un producto personalizado a la necesidad de cada cliente.

1.6. Análisis de gestores de contenidos y sus posibilidades de configuración.

A nivel internacional existe un numeroso conjunto de CMS que brindan gran facilidad a la hora de publicar contenidos en la web, entre ellos existen características muy comunes; como pudiera ser que el usuario no tiene que conocer detalles de la programación, y separan por completo el contenido a publicar de la forma en que pudiese mostrarse, dando a los usuarios la posibilidad de mostrar sus contenido de la forma que mejor entiendan. Sin embargo la característica que ha logrado que su utilización se generalice a nivel mundial, es la gestión de la configuración que estos brindan, permitiendo ser adaptados a cualquier entorno de trabajo.

El principal objetivo de analizar algunos de los CMS más utilizados a nivel mundial, es el identificar las características que estos proporcionan para establecer la configuración de un sistema informático y lo

beneficioso que sería incorporar algunas de estas características a la Plataforma de Televisión Informativa PRIMICIA.

1.6.1. Drupal.

Drupal es un sistema de gestión de contenido modular y muy configurable, de código abierto con licencia GNU/GPL escrito en PHP, desarrollado y mantenido por una activa comunidad de usuarios. Destaca por la calidad de su código y de las páginas generadas, el respeto de los estándares de la web, un énfasis especial en la usabilidad y consistencia de todo el sistema. El diseño de Drupal es especialmente idóneo para construir y gestionar comunidades en Internet. No obstante su flexibilidad y adaptabilidad, así como la gran cantidad de módulos adicionales disponibles, hace que sea adecuado para realizar diferentes tipos de sitio web. (9)

La autenticación de usuarios es una de las grandes ventajas de este CMS, el cual permite que los usuarios se puedan registrar e iniciar sesión de forma local o utilizando un sistema de autenticación externo como Jabber, Blogger[9], LiveJournal[10] u otro sitio Drupal. Para su uso en una intranet, Drupal se puede integrar con un servidor LDAP. (10) Esta última variante de autenticación mediante un servidor de dominio LDAP pudiera ser una de las nuevas funcionalidades a incorporar en PRIMICIA, permitiendo que los administradores de la plataforma puedan seleccionar el tipo de autenticación deseada ya sea él antes mencionado o local como se viene realizando hasta este momento.

Los permisos basados en roles es una ventaja para los administradores de Drupal, permitiendo que no se tenga que establecer permisos para cada uno de los usuarios; en lugar de eso pueden asignar permisos a un rol y agrupar los usuarios por roles. (10) Esta característica es de vital importancia poderla incorporar a PRIMICIA, de ser así daría solución a uno de los hechos que dio origen a la situación problemática actual. Aportaría grandes ventajas, automatizando el proceso de crear los roles de los usuarios que trabajarán con la plataforma. Además de poder gestionar los permisos de un rol de forma dinámica y no como han sido preestablecido hasta el momento.

Drupal ha sido diseñado desde el principio para ser multiplataforma. Puede funcionar con Apache o Microsoft IIS como servidor web y en sistemas operativos como Linux, BSD, Solaris, Windows y Mac OS X. Por otro lado, al estar implementado en PHP, es totalmente portable. Se puede decir que

[9] Es una herramienta de publicación de blogs gratuita de Google.

[10] Abreviado como **LJ**, es el nombre de un weblog que permite a los internautas mantener un diario en línea.

actualmente aunque PRIMICIA no ha sido probada, ni concebida para todas estas plataformas, sí funciona perfectamente con el servidor Apache en sistemas operativos GNU/Linux y Windows.

Drupal exporta el contenido en formato RDF[11]/RSS[12] para ser utilizado por otros sitios web. Esto permite que cualquiera con un Agregado de Noticias, tal como NetNewsWire[13] o Radio UserLand[14] visualice el contenido publicado en la web. (10) La sindicación del contenido puede ser otra de las funcionalidades incorporada a la plataforma, permitiendo que las noticias del canal puedan llegar mediante fuentes RSS a un mayor número de personas. Cada vez crece más el número de sitios web que utilizan esta funcionalidad, al punto de convertirse casi en un estándar en Internet.

Drupal incluye una potente herramienta para leer y publicar enlaces a noticias de otros sitios web. Además incorpora un sistema de caché en la base de datos, con temporización configurable. (10) El uso de un lector de fuentes RSS aportaría muchísimo valor a PRIMICIA, a través de la misma plataforma los redactores del canal tendrían la posibilidad de revisar un numeroso grupo de noticias publicadas por otros sitios web sin necesidad de visitar los mismos. Su incorporación debe ser un hecho a materializar en nuevas versiones del producto.

La administración y configuración de Drupal se puede realizar enteramente con un navegador y no precisa de ningún software adicional. (10) Esta característica resume los objetivos que se pretenden lograr con la presente investigación; cuando se realice el diseño de las nuevas funcionalidades se pensará en ella como el factor principal que rija la solución propuesta.

Drupal está pensado para una audiencia internacional y proporciona opciones para crear un portal multilingüe. Todo el texto puede ser fácilmente traducido utilizando una interfaz web, importando traducciones existentes o integrando otras herramientas de traducción como GNU Gettext[15]. (10) Esta característica se requiere en PRIMICIA y su incorporación facilitará que el producto pueda ser usado en cualquier país sin importar su idioma. Además es fundamental para abrirse paso en los mercados de software no hispanohablantes.

[11] Del inglés *Resource Description Framework*, Marco de Descripción de Recursos.

[12] Del inglés *Really Simple Syndication*, formato XML para canalizar las noticias de un sitio web.

[13] Es uno de los mejores lectores RSS y Atom para la plataforma Mac, con soporte para Google Reader.

[14] Herramienta para la creación de blogs, entre sus funcionalidades cuneta con un lector de fuentes RSS.

[15] Biblioteca GNU usada para escribir programas con interfaz en múltiples idiomas.

La mayor parte de las instalaciones de Drupal utilizan MySQL como sistema gestor de base de datos, pero incorpora una capa de abstracción de base de datos que actualmente está implementada y mantenida para MySQL y PostgresSQL, aunque permite incorporar fácilmente soporte para otras bases de datos. La independencia de la base de datos no es un tema a tratar, ni una característica que se necesite dentro de PRIMICIA porque está concebida para PostgresSQL. Sin embargo dar soporte a la misma no es problema debido a que PRIMICIA está soportada por el *framework* Symfony, el cual presenta una capa de abstracción de datos para un gran número de base de datos. Pudiera ponerse en práctica en caso que un cliente prefiera utilizar otro sistema gestor de base datos.

1.6.2. WordPress.

WordPress es un sistema de gestión de contenido enfocado a la creación de blogs[16]. Se ha convertido en uno de los CMS más popular de la blogósfera. Las causas de su enorme crecimiento son, entre otras, su licencia, su facilidad de uso y sus características como gestor de contenidos. Aunque gran parte del proyecto ha sido desarrollado por la comunidad alrededor de WordPress, aún está asociado a Automattic, empresa donde trabajan algunos de los principales contribuyentes de WordPress. (11)

Automattic, es la empresa que funciona detrás de WordPress, tiene asimismo un servicio de alojamiento de blogs gratuito basado en su software llamado WordPress.com. Lo que permite crear y administrar páginas fuera del orden cronológico normal y ha sido el primer paso para transformarse de un software básico de administración de blogs a un completo sistema de administración de contenidos. WordPress nació del deseo de construir un sistema de publicación personal, elegante y con una buena arquitectura. Basado en PHP, MySQL y licenciado bajo GPL, pone especial atención a la estética, estándares web, y usabilidad. (12)

Con el lanzamiento de cada una de las versiones, el sistema ha ido creciendo e incorporando mejoras que lo han hecho mantenerse en la preferencia de los usuarios que hacen uso de la Internet para postear contenidos. A continuación se mencionan algunas de funcionalidades que brinda este gestor de contenidos, las cuales han posibilitado su alta aceptación a nivel mundial y además se analizará la posible incorporación de estas a PRIMICIA.

La posibilidad que tiene este CMS de tener múltiples autores o usuarios junto a sus roles o perfiles, permite que se establezcan distintos niveles de permisos para acceder a las funcionalidades del sitio.

[16] Sitios web periódicamente actualizados que recopila cronológicamente textos o artículos de uno o varios autores, apareciendo primero el más reciente.

Lo cual es una ventaja que se desea añadir en la plataforma, para realizar una mejor gestión de los permisos de los usuarios. La mayoría de los CMS presentan características muy similares en cuanto a la gestión de usuarios y roles, sin entrar en detalles de cómo lo hace cada uno de ellos o comparar cual sería la más factible, la idea es hacer uso de este concepto en PRIMICIA. Su incorporación y funcionamiento estaría determinado por el propio entorno de la plataforma, debido a que ninguna de las opciones serán aplicadas tal y como lo hacen los distintos sistemas analizados.

WordPress hace uso del editor WYSIWYG[17] "TinyMCE", el cual es de gran utilidad para que los usuarios redacten y le den formato a sus textos. (12) Cualquier editor de este tipo pudiera ser incorporado a PRIMICIA posibilitando una mejor redacción de las noticias. Actualmente las noticias son redactadas a través de un formulario que no permite darle formato al texto, con la incorporación de un editor de tipo WYSIWYG sería posible pre-visualizar las noticias de textos e imagen tal y como se mostrarán durante su transmisión.

La publicación mediante correo electrónico, es una de las grandes ventajas de WordPress, siendo de gran utilidad para las personas que no quieran o no puedan entrar a su aplicación. (12) Por el momento esta funcionalidad no es necesaria en la plataforma, pero puede ponerse en práctica en un futuro, dando la posibilidad que las noticias del canal lleguen por varias vías, y que no sea solo la de los redactores que físicamente se encuentren en la sede del canal. De esta manera se lograría crear un entorno colaborativo entre los redactores, periodistas y foto-reporteros que realizan su trabajo en eventos en vivo y la propia plataforma PRIMICIA, extendiendo así el número de colaboradores del canal.

La importación de contenidos desde cualquiera de la los siguientes gestores de contenidos, Blogger, Blogware, Dotclear, Greymatter, LiveJournal, Movable Type y Typepad, Textpattern y desde cualquier fuente RSS, presentada por este CMS es deseada para la interacción de la plataforma con otros agentes de información externa. La cual permitiría que los redactores de las noticias de PRIMICIA puedan leer desde la misma plataforma los Feeds[18] enviados por otros sitios web de interés sin tener que entrar a los mismos, ahorrando tiempo en la búsqueda de información para redactar las noticias del canal.

[17] Del inglés *"What you see is what you get"*, en español "lo que ves es lo que obtienes" característica que tienen generalmente los editores de texto o HTML.

[18] Término inglés cuyo significado original es alimentar, y que en informática suele referirse a un tipo de dato empleado para suministrar información que es actualizada con frecuencia.

Al igual que Drupal, lo que con una gama mayor de protocolos, WordPress permite la distribución de los artículos mediante RDF, RSS 0.92, RSS 2.0 y ATOM 1.0[19]. (12) Como se dijo en el análisis de Drupal, RSS pudiera ser el formato adaptado por la plataforma para la redistribución de sus contenidos, aunque puede tenerse en cuenta el uso de los demás formatos, para que los administradores de PRIMICIA determinen cual prefieren usar.

La gestión de archivos multimedia a través de FTP es de vital importancia para la Plataforma de Televisión Informativa, debido a que una de sus principales funciones es la manipulación de recursos multimedia como música, videos, imágenes y texto, con los cuales se conforma la programación del canal, para ello se necesita una mejor gestión de la que se hace actualmente.

1.6.3. Joomla.

Joomla es un CMS premiado mundialmente, que ayuda a construir sitios web y otras aplicaciones en línea. Es una solución de código abierto y está disponible libremente para cualquiera que desee utilizarlo. Es usado en todo el mundo para generar desde una simple página web personal hasta complejas aplicaciones web corporativas. Entre los diferentes usos que se le dan a Joomla están, webs corporativas o portales, comercio electrónico, pequeños sitios de negocios, webs de organizaciones, aplicaciones gubernamentales, intranets y extranets corporativas, webs de escuelas o agrupaciones, páginas personales o familiares, portales de comunidades, revistas y periódicos.

Joomla se puede usar para gestionar fácilmente cualquiera de los aspectos de un sitio web, desde la introducción de contenidos e imágenes hasta la actualización de un catálogo de productos o la realización de reservas online. Las próximas versiones de Joomla pretenden incorporar un conjunto de mejoras que posibilitarán su mayor expansión, como es aumentar la integración de aplicaciones externas a través de servicios web y la autentificación remota, como LDAP. (13)

La administración de usuarios es jerárquica, y los distintos grupos de usuarios poseen diferentes niveles de facultades/permisos dentro de la gestión y administración del sitio. (14) Ya se hacía énfasis durante el estudio de Drupal y WordPress en la incorporación de la autenticación remota a la plataforma, no solo como única vía de gestión de acceso, sino que junto a la autenticación local, pueda permitir al administrador de la plataforma seleccionar cual utilizar. Joomla es un ejemplo más de la utilización que tiene hoy día esta característica en los sistemas informáticos.

[19] Hace referencia a un formato XML estándar usado para la redifusión web.

La integración de aplicaciones externas a través de servicios web sería fundamental para incrementar las funciones de los elementos que se exhiben durante la transmisión del canal; estos podrían ser varios, como las condiciones climáticas, los resultados deportivos, las tasas de cambio monetario, entre otras que se pueden obtener de forma automática a través de estos servicios. Poder contar con esta característica en PRIMICIA sería un gran paso de avance para competir con los principales productos construidos a nivel internacional.

Joomla ofrece la posibilidad de instalar, desinstalar y administrar componentes y módulos, que agregarán servicios de valor a los visitantes de su sitio web, por ejemplo: galerías de imágenes, foros de discusión, clasificados, etc. (14) Esta característica permite que este CMS sea totalmente configurable y es una de las metas a lograr en PRIMICIA, para lo cual se debería determinar cuáles son las funcionalidades básicas y las adicionales de las que se puede prescindir sin afectar el funcionamiento de la plataforma.

Joomla trae incorporado un sistema de sindicación de noticias por RSS/XMS de generación automática. (14) Con anterioridad se ha mencionado la importancia de la sindicación de las noticias de la plataforma, para la selección del formato a utilizar se requiere un análisis profundo el cual no es objeto del presente trabajo, sin embargo cualquiera de las vías que se adopte finalmente ofrecerá importantes ventajas al sistema.

Las páginas y documentos de Joomla pueden programarse con fecha de publicación y fecha de caducidad. Es decir un documento puede programarse para que se publique automáticamente al llegar una determinada fecha, y luego despublicarse también de forma automática en otra fecha. (14) Esta cualidad es muy positiva en cualquier sistema de gestión de noticias, por su parte PRIMICIA realiza la gestión de noticias de igual manera.

Las publicaciones que hayan perdido vigencia pueden enviarse a un archivo de almacenamiento, sin necesidad de tener que borrarlas. Esto permite también dar la posibilidad a los navegantes de consultar artículos viejos o documentos anteriores en un historial. (14) Muy parecido a Joomla, la plataforma incorpora esta funcionalidad para almacenar las noticias, las cuales son guardadas por tiempo indefinido como noticias archivadas permitiendo su reutilización o posterior consulta. Esta característica puede ser configurable para que los administradores determinen el tiempo y la cantidad de noticias a guardar por la plataforma.

1.7. Conclusiones.

Durante este capítulo se desarrolló un análisis de los elementos teóricos que sirven como base al problema científico y a los objetivos planteados en el presente trabajo de diploma. Para ello se realizó un estudio acerca de los principales conceptos asociados al tema como es, módulo, configuración y los sistemas de administración de contenidos. Se caracterizaron los procesos relacionados con la Plataforma de Televisión Informativa a través de los cuales se describieron las funcionalidades de la misma, así como un estudio crítico y valorativo acerca de los procesos actuales relacionados directamente con la configuración. Se realizó la caracterización del diseño y arquitectura de PRIMICIA, lo cual permitió una mejor compresión del sistema. Todo esto sirvió para dejar plasmada la situación problemática que dio origen a la investigación; así como el análisis de algunas de las características y funcionalidades de configuración presentadas por los gestores de contenidos Drupal, WordPress, y Joomla, y su posible incorporación a la Plataforma de Televisión Informativa PRIMICIA.

CAPÍTULO 2

Propuesta de solución a la configuración de PRIMICIA

2.1. Introducción.

En este capítulo se explican las razones del uso de la metodología de desarrollo RUP, así como la del lenguaje de modelado UML y la herramienta Visual Paradigm, para guiar y dar soporte a la solución que sea propuesta. Con la construcción del modelo de dominio se llega a un acercamiento mayor del problema a resolver. Mientras que el levantamiento de los requisitos de software va a permitir conocer las condiciones y cualidades deseadas por los usuarios. Para finalizar el capítulo se describe la propuesta de solución a los problemas de configuración de PRIMICIA.

2.2. Análisis de la metodología de desarrollo, lenguaje de modelado y herramienta CASE.

Escoger una metodología de desarrollo adecuada para guiar el proceso de construcción de un software, ya sea en su totalidad o una parte de este, es un paso fundamental para obtener resultados favorables. Sin embargo para un analista de sistema lo más importante es contar con un lenguaje de modelado y una herramienta, que le permita obtener diagramas de fácil compresión en un lenguaje común y de manera eficiente. Es por ello que a continuación se análisis las características que facilitaron la elección de la metodología de desarrollo, el lenguaje y la herramienta modelado, para dar apoyo al trabajo a desarrollar.

2.2.1. Metodología de Desarrollo, RUP.

El desarrollo de software es una tarea compleja, y prueba de ello es la existencia de numerosas propuestas metodológicas que inciden de distintas maneras en el proceso de desarrollo. Estas metodologías imponen un proceso disciplinado sobre el desarrollo de software con el fin de hacerlo de forma eficiente. En la actualidad existen varias definiciones de metodología de desarrollo, una de ellas es la dada por Kenneth Kendall.

"… conjunto de procedimientos, técnicas, herramientas, y un soporte documental que ayuda a los desarrolladores a realizar nuevo software". (15)

Las metodologías de desarrollo de software se clasifican en dos grandes grupos, robustas o ágiles. Independientemente de la clasificación que puedan recibir, todas presentan ventajas y desventajas, que determina la utilización de una u otra en dependencia del tipo de proyecto. Por encima de todo, su principal utilidad radica en organizar y guiar un equipo de proyecto.

Entre las metodologías clasificadas como robustas se encuentran, Proceso Unificado de Desarrollo (RUP, por sus siglas en inglés), Métrica y Proceso Unificado Abierto (OpenUP, por sus siglas en inglés) por solo mencionar algunas de las más conocidas. Este tipo metodologías se centran específicamente en el control de los proceso, demostrando ser efectivas y necesarias en proyectos de gran tamaño respecto a tiempo y recursos.

Por su parte las metodologías ágiles, tales como Programación Extrema (XP, por sus siglas en inglés), Scrum, y *Microsoft Solutions Framework* (MSF), ofrecen una buena solución para aquellos proyectos de entorno volátil, donde los requisitos no se conocen con exactitud y pueden sufrir cambios durante el tiempo de vida del proyecto.

De manera general las metodologías de desarrollo de software se encargan de mantener un correcto equilibrio entre clientes y desarrolladores. Sin embargo muchas veces se obvian factores a tener en cuenta en el momento de seleccionar la metodología adecuada. Entre los factores que pueden incidir en la elección de una u otra metodología, se encuentran: el tiempo que se dispone para la realización de un proyecto, su alcance y envergadura, el nivel de documentación requerido, el número de trabajadores en el equipo de desarrollo, la experiencia de los mismos, las restricciones legales y las restricciones impuestas por los clientes, entre otras.

La metodología RUP fue escogida para desarrollar el Subsistema de Configuración de PRIMICIA por ser la metodología que sustenta el desarrollo general del producto. Genera una gran cantidad de artefactos, lo cual representa una garantía para la continuidad del trabajo del proyecto y la futura implementación de las funcionalidades modeladas. Es una de las metodologías más utilizada para el análisis, implementación y documentación de sistemas orientados a objetos.

RUP consiste en un conjunto de actividades para transformar los requisitos de un usuario en un sistema software. Permite especificar, construir, visualizar y documentar los artefactos de un sistema de software orientado a objetos. Está basado en componentes y utiliza el Lenguaje Unificado de Modelado (UML, por sus siglas en inglés) para representar los esquemas de un sistema de software y

describir todo el proceso. Cuenta con una documentación profunda tanto en idioma inglés como en español.

Esta metodología consiste en una recopilación de prácticas de ingeniería de software que se están mejorando continuamente de forma regular, para reflejar los cambios en las prácticas de la industria y aporta ventajas para todos los participantes en un proyecto. Proporciona a los profesionales del desarrollo de software, un entorno de proceso configurable basado en estándares. Este entorno de proceso, permite la publicación de un método personalizado de *Rational Method Composer* y accesible a todo el equipo del proyecto. Permite que ese método se configure para satisfacer las necesidades exclusivas de cada proyecto. (7)

Las principales características en su ciclo de vida son:

- **Guiado por casos de uso:** Teniendo en cuenta que la razón de ser de un sistema es brindar servicios a los usuarios, RUP define caso de uso como la secuencia de acciones realizadas por un sistema que produce un resultado observable de valor para un actor particular. (16)

- **Centrado en la arquitectura:** La arquitectura involucra los aspectos estáticos y dinámicos más significativos del sistema, está relacionada con la toma de decisiones que indican cómo tiene que ser construido el sistema y ayuda a determinar en qué orden. La arquitectura se ve influenciada por la plataforma software, sistema operativo, gestor de bases de datos, protocolos, consideraciones de desarrollo como sistemas heredados, entre otros. (17)

- **Iterativo e incremental:** La alta complejidad de los sistemas actuales hace que sea factible dividir el proceso de desarrollo en varios mini-proyectos. Cada mini-proyecto se puede ver como una iteración (un recorrido más o menos completo a lo largo de todos los flujos de trabajo fundamentales) del cual se obtiene un incremento que produce un crecimiento en el producto. (18)

El proceso de desarrollo según RUP se divide en cuatro fases, cada una concluye con un hito bien definido, y las cuales son:

- **Inicio**: Tiene como objetivo determinar la visión del proyecto.
- **Elaboración**: En esta etapa el objetivo es determinar la arquitectura óptima.
- **Construcción**: El objetivo es obtener la capacidad operacional inicial.
- **Transición**: Debe llegar a obtenerse un entregable del producto.

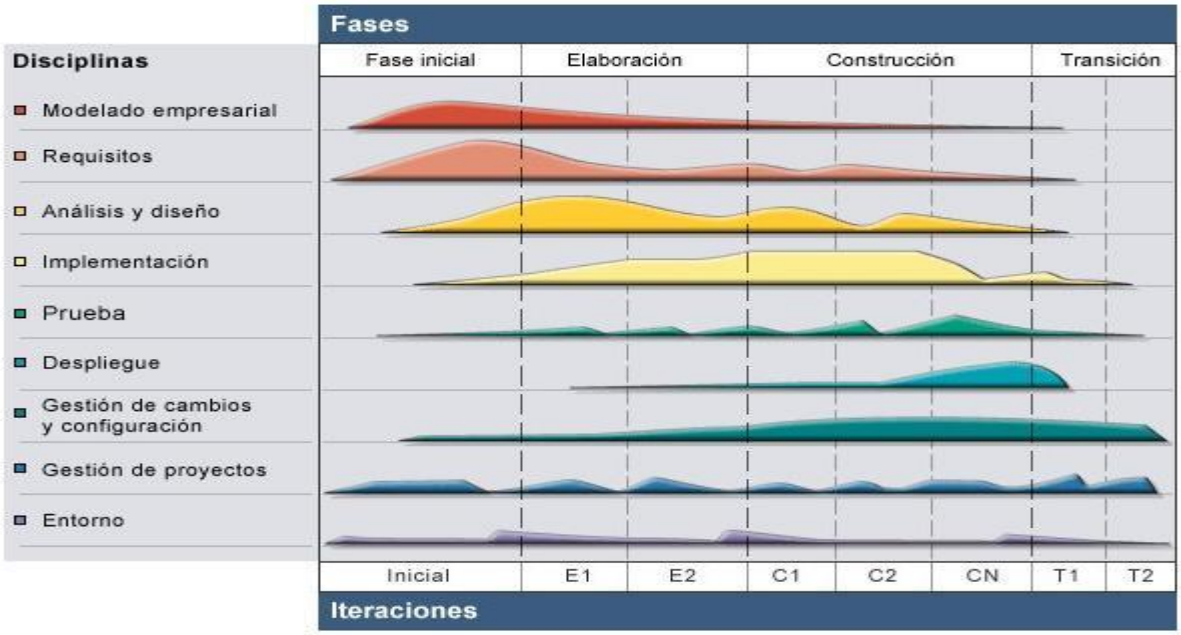

Figura 5: Arquitectura Global de RUP.

Dentro de cada iteración de cada fase se llevan a cabo nueve flujos de trabajo, dentro de los cuales los seis primeros son de ingeniería y los demás son de apoyo.

- **Modelado del negocio:** Se identifican los procesos de negocio, los que estarán sujetos a automatización y quiénes intervienen en los mismos.
- **Requisitos:** Se identifican las restricciones que se imponen y lo que el sistema debe hacer.
- **Análisis y Diseño:** Describe cómo el programa será realizado y define cómo será programado.
- **Implementación:** Define cómo estarán colocados los nodos y la ubicación en paquete de los objetos y clases.
- **Prueba:** Se localizan los defectos del software.
- **Despliegue:** Se entrega una versión operacional.
- **Administración de configuración y cambio:** Describe el uso y la actualización concurrente de los elementos, el control de versiones entre otras actividades.
- **Administración de proyecto:** Encargado de organizar el trabajo y de que se termine el proyecto en el tiempo previsto.
- **Ambiente:** Describe los procesos y herramientas que soportarán el trabajo del proyecto.

Una peculiaridad de esta metodología es que en cada ciclo de iteración se insiste en el uso de artefactos, siendo por este motivo, una de las metodologías más importantes para alcanzar un grado de certificación en el desarrollo del software. Esta metodología es muy utilizada en el proceso de desarrollo de software por ser flexible; además, al ser iterativa, permite que se vaya construyendo el software por ciclos, por lo cual se pueden detectar errores con tiempo de antelación. (19)

2.2.2. Lenguaje de Modelado, UML.

UML es un lenguaje para visualizar, especificar, construir y documentar los artefactos de un sistema. No es una guía para realizar el análisis y diseño orientado a objetos, es decir, no es un proceso; es un lenguaje que permite la modelación de sistemas con tecnología orientada a objetos. (17)

El lenguaje UML tiene una notación gráfica muy expresiva, que permite representar en mayor o menor medida todas las fases de un proyecto informático, desde el análisis con los casos de uso, el diseño con los diagramas de clases y objetos, hasta la implementación y configuración con los diagramas de despliegue. (20)

Está destinado a los sistemas de modelado y puede usarse para distintos tipos de sistemas de software, de hardware y organizaciones del mundo real. Es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad; está respaldado por el Object Management Group (OMG). Cuenta con una documentación común, sin importar el software con que se desarrolle, tampoco especifica en sí mismo qué metodología o proceso usar.

El éxito de los proyectos de desarrollo de aplicaciones o sistemas, se debe a los que sirven como enlace entre quien tiene la idea y el desarrollador. UML es una herramienta que cumple con esta función, ayuda a capturar la idea de un sistema para comunicarla posteriormente a quien está involucrado en su proceso de desarrollo; esto se lleva a cabo mediante un conjunto de símbolos y diagramas. Cada diagrama tiene fines distintos dentro del proceso de desarrollo. (21)

De manera general UML presenta las siguientes características:

- Tecnología orientada a objetos.
- Viabilidad en la corrección de errores.
- Desarrollo iterativo e incremental.
- Modela estructuras complejas.

- Permite especificar las decisiones de análisis, diseño e implementación, construyéndose así modelos precisos, no ambiguos y completos.
- Puede conectarse con lenguajes de programación (Ingeniería directa e inversa).
- Modela el comportamiento del sistema mediante casos de uso, diagramas de secuencia y de colaboración, que sirven para evaluar el estado de este.
- Permite documentar disímiles artefactos de un proceso de desarrollo.

Tabla 1: Clasificación de los Diagramas UML.

De estructura estática	De comportamiento	De implementación
▪ Diagrama de clases. ▪ Diagrama de objetos. ▪ Diagrama de casos de uso.	▪ Diagramas de interacción. ▪ Diagrama de estados. ▪ Diagrama de actividad.	▪ Diagrama de componentes. ▪ Diagrama de despliegue.

2.2.3. Herramienta CASE, Visual Paradigm.

La Ingeniería de Software Asistida por Computadora (CASE, por sus siglas en inglés) es un tipo de ingeniería de software en la que se intenta aumentar la eficacia de sus procesos, al soportar la realización de las tareas con el uso de tecnologías. (22)

Se pueden definir las herramientas CASE como un conjunto de programas y ayudas que dan asistencia a los analistas, ingenieros de software y desarrolladores, durante todos los pasos del ciclo de vida de desarrollo de un software, también es definido como el conjunto de métodos, utilidades y técnicas que facilitan el mejoramiento del ciclo de vida del desarrollo de sistemas de información, completamente o en alguna de sus fases. (23)

Visual Paradigm es una herramienta profesional para el modelado con UML, ha sido diseñada para un gran número de usuarios, incluyendo ingenieros de sistemas, analistas de sistemas, analistas de negocio y arquitectos de sistemas. Permite representar todos los tipos de diagramas de clases, código inverso, generar documentación y código desde diagramas. (24)

Mediante esta herramienta se puede desarrollar un producto de calidad que facilita la comunicación entre desarrolladores, utilizando un lenguaje de estándar común a todo el equipo de desarrollo. Tiene disponible distintas versiones: Enterprise, Professional, Standard, Modeler, Personal y Community la

cual es gratuita. También facilita licencias especiales para fines académicos, tiene varios idiomas, es fácil de instalar y actualizar. (25)

Además se integra con Entornos Integrados de Desarrollo (IDE, por sus siglas en inglés) como Eclipse, JBuilder, NetBeans, IntelliJ IDEA, Jdeveloper, entre otros. Esto ayuda a soportar la fase de implementación del desarrollo de software. Incorpora también el soporte para trabajo en equipo, que permite que varios desarrolladores trabajen a la vez en el mismo diagrama y vean en tiempo real los cambios realizados por sus compañeros. Permite además la generación de código e ingeniería inversa a un numeroso conjunto de lenguajes de programación.

Todas las características antes mencionadas, sustentan la elección de esta herramienta CASE, pero la principal razón de su utilización se debe a que es una de las restricciones del diseño e implementación impuestas a este trabajo, debido a que PRIMICIA está sustentada bajo esta herramienta y la solución que se obtenga debe ser compatible con esa estrategia. Otra de las razones de su elección es la necesidad de realizar la ingeniería inversa, desde el sistema gestor de bases de datos a los diagramas de entidad-relación para obtener el modelo actual e incorporarle los nuevos cambios y a su vez generar de forma automática la nueva base de datos del sistema.

2.3. Modelo de Dominio.

Utilizando la notación UML, un modelo del dominio se representa como un conjunto de diagramas de clases en los que no se define ninguna operación. Y pueden mostrar objetos del dominio o clases conceptuales, así como asociaciones y atributos. Su objetivo es lograr la representación de las clases conceptuales del mundo real significativas en un dominio del problema, no de componentes de software, aunque algunos objetos del modelo pueden terminar siéndolo. (26)

Este tipo de modelo puede utilizarse para capturar y expresar el entendimiento obtenido en un área cualquiera. Son similares a los mapas conceptuales utilizados en el aprendizaje y son utilizados por el analista como un medio para comprender el negocio a informatizar por el sistema.

2.3.1. Conceptos y principales eventos del entorno.

PRIMICIA es una plataforma que permite la administración y **transmisión** de **noticias** en diferentes formatos mediante un canal de televisión, con el objetivo de mantener actualizada cualquier comunidad, empresa, o entidad que requiera de información clara, precisa y de forma inmediata.

La **transmisión** del canal puede realizarse de tres formas diferentes: **continua**, **bloque de noticias** o **sección temática**. La entidad que haga uso de la plataforma debe seleccionar cual es el tipo de transmisión que desea usar de las antes mencionadas.

La transmisión por **sección temática** permite establecer el orden en que serán mostradas las secciones, se comienza con la primera sección temática y las noticias asociadas a la misma, una vez que termine con estas se pasa a la otra y así sucesivamente, creando un ciclo de transmisión mientras exista vigencia en las noticias publicadas. En caso que exista una sección temática que no tenga noticias asociadas esta no es mostrada.

La transmisión por **bloque de noticias** funciona estableciendo el horario en que saldrá al aire un bloque de noticias y durante ese tiempo se muestran todas las noticias asociadas al mismo, creando de esta manera un ciclo durante el horario establecido. Cuando concluye el período establecido se dejan de transmitir las noticias.

La transmisión **continua** es aquella donde las noticias se muestran en el mismo orden en que sean publicadas en el sistema, creando un ciclo de transmisión mientras existan.

Las **noticias** son gestionadas desde el Subsistema de Administración de la plataforma, cuando se redactan se asignan a un bloque de noticias o a una sección temática en caso que la transmisión se esté realizando por una de estas dos variantes. Si por lo contrario se realiza una transmisión continua, al ser publicadas estas noticias pasan a formar automáticamente parte de la misma.

Las **fuentes web** son gestionadas por el equipo de realización del canal y es un modo de proveer información al sistema. La información obtenida de las diferentes subscripciones puede utilizarse en la elaboración de las noticias.

El sistema también permite gestionar **canales externos** los cuales sirven para enlazar la transmisión de la plataforma con estos canales en el momento deseado, ya sea para la transmisión directa de una información inmediata de gran repercusión o simplemente para completar la programación del canal cuando no existan noticias.

El modo de **autenticación** de los usuarios de PRIMICIA puede realizarse por dos vías: **local** al sistema o por **dominio** utilizando los servicios LDAP de un directorio activo de la red.

La plataforma está compuesta por numerosas interfaces de usuarios que permiten la interacción de los usuarios con el sistema. La **interfaz de usuario** normalmente es personalizada en dependencia de los usuarios finales a los que va dirigido el sistema.

El **idioma** es un elemento a tener en cuenta en las interfaces de usuario de un producto, este puede determinar el número de usuarios que puedan hacer uso del sistema, normalmente se piensa en el inglés como idioma universal y de este modo se garantiza que muchas personas puedan utilizarlo, pero si el sistema es construido para un conjunto de idiomas diferentes puede crecer considerablemente el uso del producto en diferentes países y clientes.

A continuación se muestra el modelo de dominio, obtenido gracias a los conocimientos adquiridos de los principales conceptos que sustentan el problema actual de configuración.

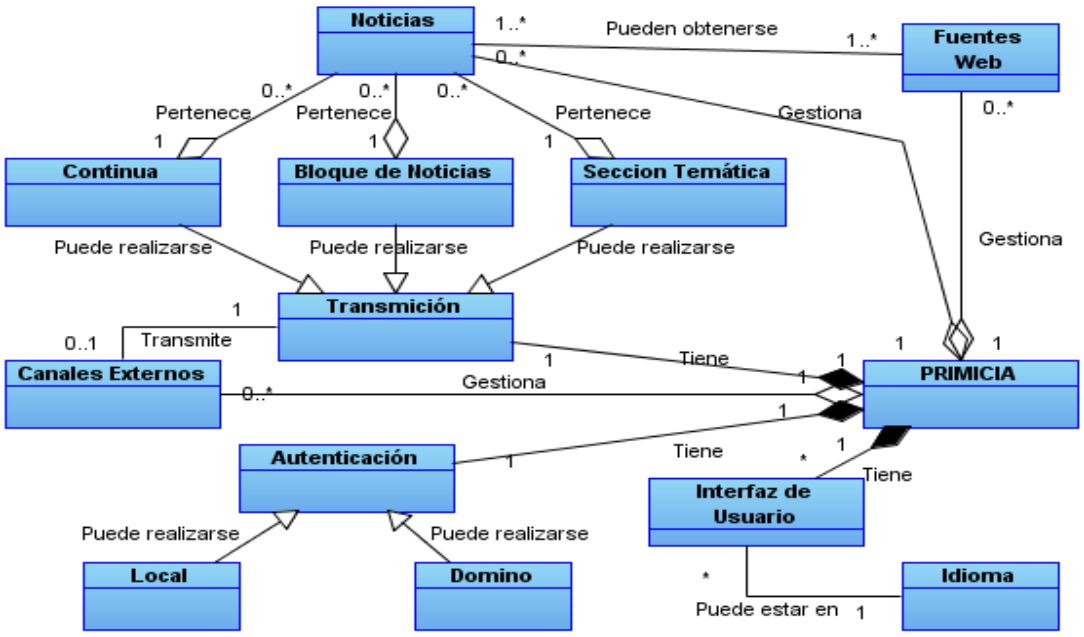

Figura 6: Diagrama Modelo de Dominio.

2.3.2. Glosario de Términos del Dominio.

El siguiente glosario de términos ayudará a una mejor compresión de los conceptos presentes en el entorno del problema.

- **Noticias:** Conjunto de información que llega a las personas de forma inmediata y actualizada ya sea a través de texto, imagen, sonido, video o la integración de estos.

- **Fuentes Web:** Sitios web que a través del formato RSS u otros formatos que pueden distribuir sus contenidos y/o ser leídos mediante un lector de fuentes, sin necesidad de acceder al sitio web.
- **Transmisión:** Envío y recepción de la información en forma de noticias a través de un equipo informático por medio de señales encaminadas en un canal.
- **Transmisión Continua:** Flujo de información que se mantiene enviándose constantemente sin importar el contenido de las noticias.
- **Transmisión por Bloque de Noticias:** La información enviada y recibida se realiza a través de los bloques de noticias que hayan sido establecidos en el canal, teniendo en cuenta el horario de transmisión y el público al que va dirigido.
- **Transmisión por Secciones Temáticas:** La información enviada y recibida se realiza agrupando las noticias por un tema de interés común.
- **Canales Externos:** Cualquier canal de televisión existentes que pertenece a una entidad en particular y no sea el generado por las noticias de PRIMICIA.
- **Autenticación:** Procedimiento de comprobación de identidad de un usuario para acceder a un recurso o realizar determinada actividad en el sistema.
- **Autenticación Local:** Autenticación que se realiza mediante el propio sistema a través de una base de datos local.
- **Autenticación por Dominio:** Autenticación realizada mediante un servidor de dominio a través de los servicios LDAP de un directorio activo.

2.4. Especificación de los requisitos de software.

Según la IEEE *Standard Glossary of Software Engineering Terminology* 610, define un requerimiento como:

> *"Condición o capacidad necesaria para que un usuario resuelva su problema o alcanzar un objetivo que debe cumplirse o poseído por un sistema o componente del sistema para satisfacer un contrato, norma, especificación o formalmente impuesto en un documento"*. (27)

Los requerimientos son los que permiten guiar el desarrollo de cualquier software hacia un sistema correcto. A través de ellos se definen los objetivos generales concretos de forma tal que tanto el negocio como sus actores se beneficien; se obtiene una descripción correcta de lo que debe hacer el

sistema y delimita su alcance; se aumenta la comunicación entre un cliente y el grupo de desarrollo; y sirven de base para la validación, verificación y procedimientos de aceptación de cualquier producto de software. A continuación se mencionan a través de requerimientos funcionales y no funcionales, las acciones que el sistema debe ser capaz de realizar.

2.4.1. Requerimientos Funcionales.

Los requerimientos funcionales son capacidades o condiciones que el sistema debe cumplir. Se mantienen invariables sin importar con que propiedades o cualidades se relacionen. (28)

Son los que definen las acciones que debe realizar el sistema desde el punto de vista de las necesidades del usuario. Durante el estudio de los procesos actuales, así como el análisis de algunos de los gestores de contenidos más importantes en la actualidad, se determinaron los requisitos con que debía contar el Subsistema de Configuración para mejorar el funcionamiento actual de PRIMICIA. A continuación se relacionan dichos requerimientos:

R1 Permitir que se pueda establecer el idioma para la interfaz de usuario de la plataforma.
R2 Dar la posibilidad que se pueda establecer el tipo de autenticación a usar en la plataforma.
R3 Permitir que a través de la plataforma se gestione el nombre y los permisos para los roles de usuarios.
- **R3.1** Crear roles de usuarios.
- **R3.2** Modificar roles de usuarios.
- **R3.3** Eliminar roles de usuarios.
- **R3.4** Ver los permisos establecidos para cada uno de los roles de usuarios.

R4 Establecer el tipo de transmisión usada por la plataforma, ya sea por secciones temáticas, bloque de noticias o de manera continua.
R5 Gestionar las secciones temáticas usadas por la plataforma para la transmisión.
- **R5.1** Crear sección temática.
- **R5.2** Modificar sección temática.
- **R5.3** Eliminar sección temática.
- **R5.4** Ver el listado de las secciones temáticas habilitadas para la transmisión.

R6 Gestionar los bloques noticiosos usados por la plataforma para la transmisión.
- **R6.1** Crear bloques de noticias.
- **R6.2** Modificar bloques de noticias.

R6.3 Eliminar bloques de noticias.

R6.4 Ver el listado de los bloques de noticias habilitados para la transmisión.

R7 Gestionar los canales externos, con los que la plataforma puede enlazarse para cambiar su transmisión.

R7.1 Adicionar canal externo.

R7.2 Modificar canal externo.

R7.3 Eliminar canal externo.

R7.4 Ver el listado de los canales externos habilitados para enlazarse.

R8 Establecer los servicios web que serán usados para obtener las informaciones adicionales que se muestran durante la transmisión.

R9 Gestionar los sitios web que serán usados por el lector de noticias de la plataforma.

R9.1 Adicionar fuentes de información.

R9.2 Eliminar fuentes de información.

R9.3 Ver el listado de fuentes de información.

R10 Permitir que se puedan habilitar y deshabilitar los módulos y funcionalidades de la plataforma.

R11 Permitir que se puedan habilitar y deshabilitar los reportes que se instalen con la plataforma y seleccionar el formato de salida de los mismos.

2.4.2. Requerimientos No Funcionales.

Los requisitos no funcionales son propiedades o cualidades que el producto debe de tener. Son características que hacen al producto, atractivo, usable, rápido y confiable. En muchos casos los requerimientos no funcionales son fundamentales en el éxito del producto. Normalmente están vinculados a requerimientos funcionales, es decir una vez que se conozca lo que el sistema debe hacer se puede determinar cómo ha de comportarse, qué cualidades debe tener o cuán rápido o grande debe ser. Son importantes para que clientes y usuarios puedan valorar las características no funcionales del producto. Las propiedades no funcionales, como cuán usable, seguro y agradable, pueden marcar la diferencia entre un producto bien aceptado y uno con poca aceptación. (28)

Los requerimientos no funcionales se clasifican en múltiples categorías, a continuación se mencionan aquellos necesarios para el Subsistema de Configuración sea integrado de forma exitosa a PRIMICIA.

Requerimientos de apariencia o interfaz externa: La apariencia del módulo debe estar restringida por las mismas condiciones que PRIMICIA, haciendo uso de colores apropiados que denoten un

producto profesional. El módulo debe contar con una interfaz amigable, intuitiva, interactiva y simple de usar.

Requerimientos de usabilidad: El sistema de forma general debe brindar gran facilidad de uso para personas con poca experiencia con las computadoras, pero con nivel calificado. Para trabajar con el Subsistema de Configuración se requiere de conocimientos mínimos en informática, televisión y uso de la web, ya que a través de este se gestiona el funcionamiento de toda la plataforma. Las funcionalidades deben ser claras y se debe mostrar la información de forma lógica y correctamente estructurada. El nuevo subsistema debe permitir una mejor configuración del sistema adaptándolo a las necesidades de la entidad donde se implante, y el cual reduzca los costos en tiempo y complejidad para estas operaciones con relación a la actualidad. Se requiere que pueda ser usado por personas que hablen diferentes idiomas.

Requerimientos de confiabilidad: Se debe garantizar un tratamiento adecuado de las excepciones y validación de las entradas del usuario.

Requerimientos de portabilidad: El sistema estará realizado para funcionar en los sistemas operativos GNU/Linux o Windows, garantizando de esta manera un producto multiplataforma.

Requerimientos de seguridad: La información manejada por el sistema y el uso del mismo estará protegida del acceso de personas no autorizadas. Solo se accederá a las opciones autorizadas según el rol que desempeñe. El Subsistema de Configuración solo debe ser accedido por un súper administrador. Solo se accederá al sistema de administración desde las computadoras autorizadas. Los usuarios autorizados tendrán garantizado el acceso pleno a la información, de acuerdo a los permisos que tengan establecidos.

La información manejada por el sistema será objeto de cuidadosa protección contra la corrupción o estado inconsistente. Deberán existir mecanismos de chequeo de integridad. Se deberán hacer copias de respaldo que puedan restaurar el sistema en caso de fallo crítico o pérdida total de la información. El sistema debe estar disponible los 7 días de la semana y las 24 horas del día, garantizando el buen funcionamiento de la Plataforma. De esta manera se debe garantizar que las modificaciones realizadas en la configuración de la plataforma se hagan en tiempo real y que no afecten en ningún sentido el correcto funcionamiento de la misma.

Requerimientos Legales: El Subsistema de Configuración estará sujeto a las mismas restricciones legales que el resto de la Plataforma de Televisión Informativa, PRIMICIA. Sus derechos de autor y otros están determinados por la empresa comercializadora del producto, ALBET S.A y la entidad desarrolladora, la UCI.

Requerimientos de Ayudas y Documentación en línea: El sistema debe presentar un manual de usuario donde indique al usuario los pasos que tiene que seguir para obtener una correcta configuración.

Requerimientos de Software: Se debe utilizar Apache como servidor web y PostgresSQL 8.4 o superior como sistema gestor de base de datos. El Subsistema de Administración y de Configuración deben ser accedidos a través del cualquier navegador, fundamentalmente Internet Explorer 7 o superior, o Mozilla Firefox.

Requerimientos de Hardware: Serán los mismos que para el servidor donde corra el Subsistema de Administración de la plataforma, el subsistema responde a un elemento que se debe integrar a dicho sistema, los requerimientos son los siguientes.

- Procesador: Dual-Core Xeon 2.33 GHz.
- Memoria RAM: 4Gb.
- Disco Duro: 500Gb.
- Tarjeta de Red: Ethernet Gigabit con 2 puertos.
- Tarjeta de Video: Capturadora Hauppauge WinTV PVR 350.

Restricciones en el diseño y la implementación: Para la modelación del sistema se utilizará el lenguaje UML y a través de la herramienta Visual Paradigm. Se requiere el uso de la arquitectura Modelo-Vista-Controlador implementada por el *framework* Symfony de PHP. La arquitectura debe soportar migrar la interfaz de usuario de forma rápida, para lograr visualizar cualquiera de los cambios que se produzcan.

2.5. Descripción de la Solución Propuesta.

Después de haber realizado el levantamiento de requisitos del sistema y tener en cuenta varias restricciones, se propone el desarrollo de un subsistema que gestione toda la configuración de la plataforma, posibilitando realizar cambios en la misma en cualquier momento que se desee. Dicho

subsistema dará solución al problema de configuración actual en PRIMICIA, automatizando así un conjunto de funcionalidades que hasta el momento se vienen realizando manualmente.

El Subsistema de Configuración para PRIMICIA solo debe ser accedido por una única persona que sea la encargada de configurar todo el sistema, lo que requiere de un alto nivel de seguridad para rechazar los usuarios no autorizados. Para su incorporación a la plataforma se respetará el diseño actual de la base datos, pero sin negar la posibilidad de realizar pequeños cambios o añadir nuevas tablas que ayuden a mejorar el desempeño del sistema y posibiliten las labores de configuración requeridas. Este subsistema tendrá una alta repercusión en el funcionamiento del Subsistema de Administración, conllevando a modificar varias de sus funcionalidades en futuras versiones del producto. La administración y configuración del sistema se podrá realizar enteramente con un navegador web y no precisara de ningún software adicional.

2.5.1. Actores Propuestos para el Sistema.

Encontrar los actores es uno de los primeros pasos en la definición del uso del sistema. Cada tipo de fenómeno externo con el que debe interactuar el sistema está representado por un actor. Se debe definir cada actor escribiendo una breve descripción que incluya el área de responsabilidad del actor y para qué necesita el actor el sistema. Como los actores representan elementos externos al sistema, no es necesario describirlos en detalle. (7)

Los actores propuestos para el nuevo funcionamiento de la plataforma son los que aparecen en la **Tabla 2**. Su elección está determinada por la aparición del Subsistema de Configuración el cual permitirá la gestión de los roles de usuarios. Por lo que se representa de forma genérica el actor Usuario debido a que el rol que desempeñe en el sistema estará en dependencia de las necesidades de cada cliente. A continuación se proponen los actores y la función que estos desempeñarán.

Tabla 2: Propuesta de Actores del Sistema.

Actor	Descripción
Administrador de Configuración	Persona responsable de la gestión de la configuración de la plataforma.
Usuario	Persona con acceso al sistema encargada de cualquiera de los procesos que se realizan en el mismo.

2.5.2. Casos de Uso del Subsistema de Configuración.

La mejor forma de encontrar casos de uso es considerar qué necesita cada actor del sistema, es importante tener presente que el sistema sólo existe para los usuarios y que por lo tanto debe basarse en las necesidades de éstos. Se debe reconocer muchas de las necesidades de los actores a través de los requisitos funcionales realizados sobre el sistema. (7)

Los casos de uso son el componente clave del modelado. Permiten ilustrar las funcionalidades de un sistema y la relación con el actor para cumplir un objetivo. A continuación se expone una descripción general de cada uno de los casos de uso que conforman el Subsistema de Configuración.

Tabla 3: Descripción del CUS Determinar Idioma de la Interfaz de Usuario.

CU-1	Determinar Idioma de la Interfaz de Usuario.
Actor	Administrador de Configuración.
Descripción	Permitirá cambiar el idioma de la interfaz de usuario para que la plataforma pueda ser usada por personas que hablen diferentes idiomas.
Referencia	R1

Tabla 4: Descripción del CUS Determinar Tipo de Autenticación.

CU-2	Determinar Tipo de Autenticación.
Actor	Administrador de Configuración.
Descripción	Dará la posibilidad de establecer el tipo de autenticación a usar en la plataforma ya sea local o por dominio.
Referencia	R2

Tabla 5: Descripción del CUS Gestionar Roles de Usuarios.

CU-3	Gestionar Roles de Usuarios.
Actor	Administrador de Configuración.
Descripción	Permitirá crear, modificar, eliminar y listar los roles de los usuarios de la plataforma, indicando las funcionalidades a las que puede tener acceso.
Referencia	R3, R3.1, R3.2, R3.3, R3.4

Tabla 6: Descripción del CUS Determinar Tipo de Transmisión.

CU-4	Determinar Tipo de Transmisión.
Actor	Administrador de Configuración.
Descripción	Permitirá establecer el tipo de transmisión que realice la plataforma, ya sea por, secciones temáticas, bloques noticiosos o transmisión continua.
Referencia	R4

Tabla 7: Descripción del CUS Gestionar Secciones Temáticas.

CU-5	Gestionar Secciones Temáticas.
Actor	Administrador de Configuración.
Descripción	Permitirá que se pueda crear, modificar, eliminar y listar las secciones temáticas que se utilizarán para la trasmisión de las noticias.
Referencia	R5, R5.1, R5.2, R5.3, R5.4

Tabla 8: Descripción del CUS Gestionar Bloques Noticiosos.

CU-6	Gestionar Bloques Noticiosos.
Actor	Administrador de Configuración.
Descripción	Permitirá que se pueda crear, modificar, eliminar y listar los bloques de noticias que se utilizarán para la trasmisión de las noticias.
Referencia	R6, R6.1, R6.2, R6.3, R6.4

Tabla 9: Descripción del CUS Gestionar Canales Externos.

CU-7	Gestionar Canales Externos.
Actor	Administrador de Configuración.
Descripción	Permitirá adicionar, modificar, eliminar y listar los canales de televisión externos con los que se puede enlazar la plataforma.
Referencia	R7, R7.1, R7.2, R7.3, R7.4

Tabla 10: Descripción del CUS Configurar Elementos de Información.

CU-8	Configurar Elementos de Información.
Actor	Administrador de Configuración.
Descripción	Permitirá establecer los servicios web que serán usados para obtener la información mostrada por los elementos de información adicional del canal.
Referencia	R8

Tabla 11: Descripción del CUS Gestionar Fuentes Web.

CU-9	Gestionar Fuentes Web.
Actor	Administrador de Configuración.
Descripción	Permitirá adicionar, eliminar y listar los sitios web que servirán de agentes de información para el lector de noticias.
Referencia	R9, R9.1, R9.2, R9.3

Tabla 12: Descripción del CUS Habilitar/Deshabilitar Módulos y Funcionalidades.

CU-10	Habilitar/Deshabilitar Módulos y Funcionalidades.
Actor	Administrador de Configuración.
Descripción	Permitirá que se puedan ocultar o visualizar los módulos y funcionalidades de la plataforma.
Referencia	R10

Tabla 13: Descripción del CUS Configurar Reportes.

CU-11	Configurar Reportes.
Actor	Administrador de Configuración.
Descripción	Permitirá que se puedan ocultar o visualizar los reportes de la plataforma así como configurar el formato de salida de los mismos.
Referencia	R11

El detallar un caso de uso es una de las tareas del flujo de trabajo de Requisitos en RUP y tiene por objetivos, describir uno o varios de los flujos de sucesos del caso de uso con el detalle suficiente para que el desarrollo de software pueda empezar en él. (7)

La plantilla de especificación de caso de uso permite que a través de la descripción textual se pueda detallar cada uno de los aspectos a tener en cuenta en la realización de un caso de uso. Entre esos aspectos se pueden encontrar la descripción del flujo de eventos normal y alterno, la interacción entre actores y el sistema, pre-condiciones y pos-condiciones que deben cumplirse para la realización del mismo, así como el prototipo de interfaz de usuario para cada uno de los escenarios del caso de uso en cuestión.

Para lograr una mayor comprensión de los casos de uso que se describieron anteriormente resulta necesario el estudio de la especificación de cada uno de ellos. Debido a lo extenso que resulta esta tarea, no se realizará en este apartado del documento y para ello se propone visitar el **Anexo 1**.

2.5.3. Diagrama de Caso de Usos del Subsistema Configuración.

El diagrama de Casos de Uso del Sistema (CUS) es un modelo de las funciones deseadas para el sistema y su entorno, sirve como contrato entre el cliente y los desarrolladores. Se utiliza como entrada esencial para las actividades de análisis, diseño y prueba. (7)

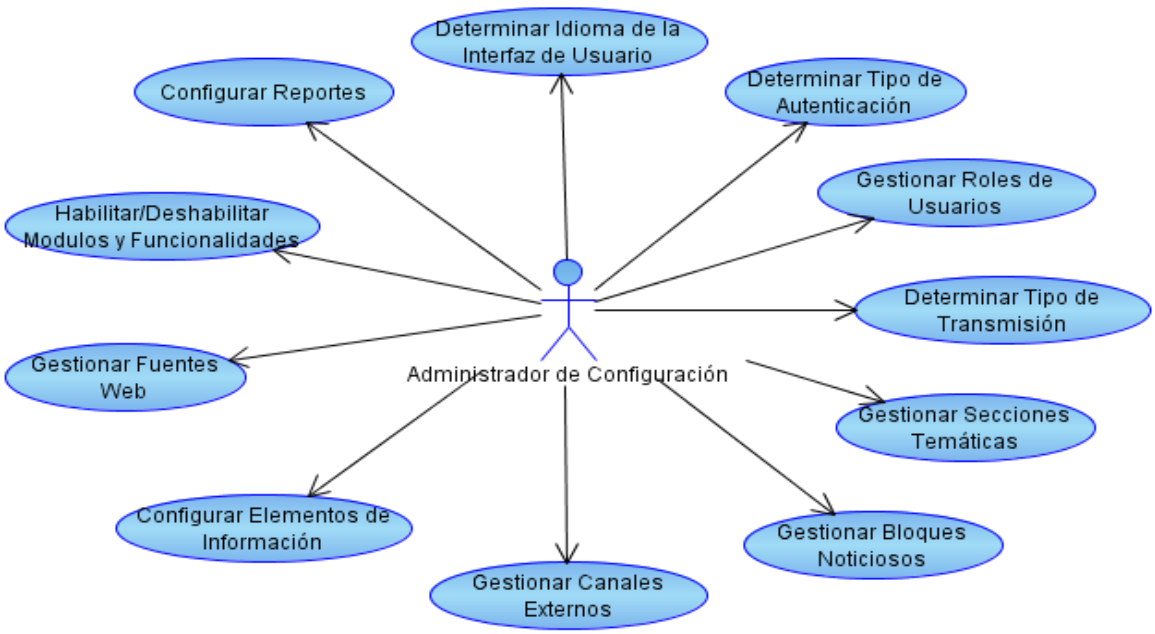

Figura 7: Diagrama de CUS Subsistema de Configuración.

2.6. Conclusiones.

Este capítulo se realizó el modelo de dominio el cual se utiliza para lograr la compresión de los conceptos utilizados por los usuarios y con los que deberá trabajar la aplicación. Permitió que se dejaran plasmados los argumentos de la elección de UML y Visual Paradigm para la construcción de algunos de los artefactos propuestos por la metodología de desarrollo del software RUP. Se analizaron y valoraron todos los requisitos de software planteados por los usuarios finales del sistema. Y como resultado principal de este capítulo quedó plasmada la propuesta de solución para el problema actual del sistema, detallada a través de los actores, casos de usos, diagrama de casos de uso, así como las especificaciones de los casos de uso.

CAPÍTULO 3

Diseño del Subsistema de Configuración de PRIMICIA

3.1. Introducción.

Después de haber especificado los requisitos y alcanzar un entendimiento de lo que se quiere lograr a través de los casos de uso descritos con anterioridad, en este capítulo se realiza un análisis de la arquitectura y los patrones de diseño usados por el *framework* Symfony para de esta forma alcanzar un mejor entendimiento del diseño que se realice. Se construyen los diagramas de clases del diseño y clases persistentes como parte del modelado de la solución propuesta. Para finalizar se especifican los datos contenidos por los ficheros de configuración del proyecto así como las modificaciones que se necesitan realizar en el modelo de datos para una correcta incorporación de las funcionalidades de configuración a la plataforma.

3.2. Análisis del Framework Symfony.

Para lograr un diseño correcto de las funcionalidades de configuración de PRIMICIA se hace necesario conocer algunos detalles del *framework* Symfony para comprender con exactitud cuáles son los elementos esenciales que se deben representar en el Modelo de Diseño a desarrollar. Para ello se analizará la arquitectura del framework y las particularidades del mismo, así como los patrones de diseños aplicados en su núcleo.

3.2.1. Arquitectura de Symfony.

Symfony está basado en el patrón arquitectónico MVC, el cuál que está formado por tres niveles:

- 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 para que el usuario pueda interactuar con la aplicación.
- El controlador, se encarga de procesar las interacciones del usuario y realiza los cambios apropiados en el modelo o en la vista.

Esta arquitectura separa la lógica de negocio (el modelo) y la presentación (la vista) por lo que se consigue un mantenimiento más sencillo de las aplicaciones. El controlador se encarga de aislar al modelo y a la vista de los detalles del protocolo utilizado para las peticiones (HTTP, consola de comandos, email, etc.). El modelo se encarga de la abstracción de la lógica relacionada con los datos, haciendo que la vista y las acciones sean independientes del tipo de gestor de bases de datos utilizado por la aplicación. Symfony realiza su propia implementación del MVC, para ello toma lo mejor de esta arquitectura logrando que el desarrollo de las aplicaciones sea más rápido y sencillo.

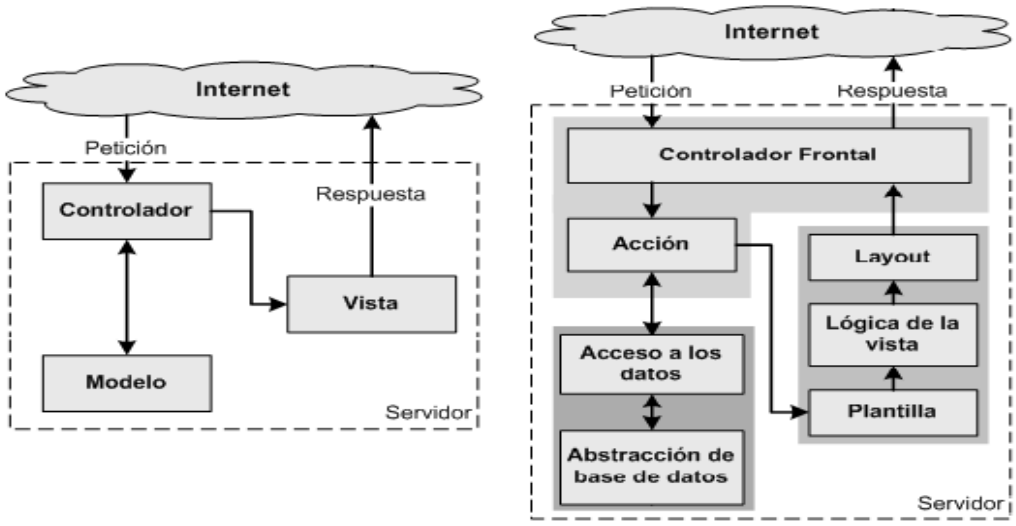

Figura 8: Patrón MVC implementado por Symfony.

Normalmente el controlador realiza muchas tareas en las aplicaciones web por lo que suele tener mucho trabajo. Por este motivo el controlador de Symfony se divide en un controlador frontal que es único para cada aplicación y las acciones que incluyen el código específico del controlador de cada página. El controlador frontal ofrece un punto de entrada único para toda la aplicación y se encarga de las tareas comunes como son, el manejo de las peticiones del usuario, de la seguridad y cargar la configuración de la aplicación entre otras tareas similares.

Las páginas web suelen contener elementos que se muestran de forma idéntica a lo largo de toda la aplicación como son, las cabeceras de las páginas, el *layout* genérico, el pie de página y la navegación global, cambiando solamente el interior de la página. Por este motivo, la vista se separa en un *layout* generalmente global en toda la aplicación o al menos en un grupo de páginas y en una plantilla que sólo se encarga de visualizar las variables definidas en el controlador. Para que estos componentes interaccionen entre sí correctamente, es necesario añadir cierto código llamado lógica de la vista que determina el *layout* a usar para cada plantilla.

La capa del modelo se divide en dos, la de acceso a los datos y la de abstracción de la base de datos. De esta forma, las funciones que acceden a los datos no utilizan sentencias ni consultas que dependen de una base de datos, sino que utilizan otras funciones para realizar las consultas. De tal manera que si se cambia de sistema gestor de bases de datos, solamente es necesario actualizar la capa de abstracción de la base de datos.

Afortunadamente la implementación que realiza Symfony simplifica el proceso de desarrollo de una aplicación; debido a que el controlador frontal y el *layout* son comunes para todas las acciones de la aplicación, aunque se pueden tener varios controladores y varios *layout*. El controlador frontal es un componente que sólo tiene código relativo al MVC, por lo que no es necesario crear uno, Symfony lo genera de forma automática. Por último la lógica de la vista se puede transformar en un archivo de configuración sencillo sin necesidad de programarla.

Las clases de la capa del modelo son generan automáticamente con la utilización del mapeo de objetos a base datos (ORM, por sus siglas en inglés *"Object-Relational Mapping"*). Mientras la abstracción de la base de datos es completamente transparente para el programador, de esta manera si se cambia el sistema gestor de bases de datos en cualquier momento no se debe reescribir ni una línea de código, sólo es necesario modificar un parámetro en un archivo de configuración. En un principio se usaba el ORM Propel, pero la evolución del *framework* ha permitido alternar entre este y Doctrine, convirtiéndose este ultimo en el ORM por defecto para las versiones más actuales.

3.2.2. Patrones de Diseño en Symfony.

Un patrón es una descripción de un problema y la solución, a la cual se le da un nombre y se puede aplicar a nuevos contextos. (26) Los patrones de diseño se dividen en dos grandes grupos los *"General Responsibility Assignment Software Patterns"* (GRASP) y los *"Gang of Four"* (GOF).

Patrones GRASP.

Los patrones GRASP describen los principios fundamentales de la asignación de responsabilidades a objetos, además constituyen un apoyo para entender el diseño y aplica el razonamiento para el diseño de una forma sistemática, racional y aplicable. En Symfony se aplican los cinco patrones que conforman este grupo y son con los que interactúan mayormente tanto diseñadores como programadores. A continuación se describe la solución que brinda estos patrones, el problema que resuelve y en qué partes del *framework* se puede evidenciar su uso.

- **Alta Cohesión:** Asignar una responsabilidad de manera que la cohesión permanezca alta. ¿Cómo mantener la complejidad manejable? (26) Permite una alta colaboración entre las clases de un sistema. Symfony permite asignar responsabilidades a una clase con una alta cohesión siempre que se realice un trabajo de acuerdo a la estructura planteada por el *framework*, en el cual debe existir una acción para cada una de las plantillas. Las acciones contenidas en cada una de las clases poseen una fuerte relación, teniendo un sentido común y donde su mayor responsabilidad radica en definir las acciones para cada una de las plantillas de un módulo. Permitiendo así que dichas clases sean flexibles y aporten un alto nivel de modularidad a los sistemas que se implementen con el *framework* en cuestión.

- **Bajo Acoplamiento:** Asignar una responsabilidad de manera que el acoplamiento permanezca bajo. ¿Cómo soportar bajas dependencias, bajo impacto del cambio e incremento de la reutilización? (26) Permite mantener un bajo nivel de dependencias entre las clases que conforman un sistema y una alta probabilidad de que puedan ser reutilizadas. Symfony cumple con este patrón de diseño casi en su totalidad, debido que el nivel de dependencia entre las clases que contienen las acciones es muy bajo por no decir que ninguno. Esto es posible gracias a la sintaxis y la propia estructura del *framework*, dando al traste con el patrón anterior. Las acciones son métodos con el nombre “executeNombreAccion” de una clase llamada “nombreMóduloActions” que hereda de la clase “sfActions” y se encuentran agrupadas por módulos. Lo cual brinda gran nivel de reutilización a cada uno de los módulos que se implemente en una aplicación. Además si se quiere obtener la reutilización a nivel de acciones, se puede utilizar una sintaxis alternativa para distribuir las acciones en archivos separados. En este caso, cada clase acción extiende de “sfAction” y su nombre es “nombreAccionAction”. El nombre del método es simplemente “execute”. Mientras el nombre del archivo es el mismo que el de la clase a diferencia del anterior que es “action.class.php”.

- **Creador:** Se encarga de asignar a la clase B la responsabilidad de crear una instancia de clase A. ¿Quién debería ser el responsable de la creación de una nueva instancia de alguna clase? (26) Este patrón al igual que los anteriores se puede evidenciar en las clases que contienen las acciones, las cuales son encargadas de instanciar a las clases del modelo que representan las entidades. Su uso también se puede percibir en las clases del modelo, las cuales son generadas automáticamente por cualquiera de los ORM utilizado por el *framework*, y aunque el mismo es transparente al programador, se puede apreciar cuando se debe crear una clase B que agrega, contiene, registra instancias, o tiene los datos de inicialización que se pasarán a otro objeto A. El caso típico es cuando se desea guardar una tupla en una tabla de la base de datos que tiene

relación de uno a muchos con otra tabla, la cual es mapeada como una relación de agregación, donde solo es necesario crear un solo objeto específicamente de la clase de uno y esta será la responsable de crear la de muchos.

- **Controlador:** Asignar la responsabilidad de recibir o manejar un mensaje de evento del sistema a una clase. ¿Quién debe ser el responsable de gestionar un evento de entrada al sistema? (26) Este patrón facilita la centralización de actividades, no porque las realice sino porque las delega en otras clases con las que mantiene un modelo de alta cohesión. Todas las peticiones web en el *framework* son manejadas por un solo controlador frontal, que es el punto de entrada único de toda la aplicación. El controlador frontal es imprescindible en Symfony, se encarga de numerosas tareas entre las que se encuentran: crear la configuración de la aplicación y el contexto; cargar e inicializar las clases del núcleo del *framework*; decodificar la URL de la petición para determinar la acción a ejecutar y los parámetros de la petición; ejecutar los filtros; ejecutar la acción y producir la vista. Las acciones también hacen uso de este patrón conteniendo la lógica de la aplicación; verifican además la integridad de las peticiones y preparan los datos requeridos por la vista utilizando el modelo y definiendo las variables para la vista, por lo que se puede decir que son el corazón de cada aplicación. Este patrón se encuentra ejemplificado en las clases “sfFrontController”, “sfActions”, así como todos los archivos “actions.class.php” que son creados en cada uno de los módulos de la aplicación.

- **Experto:** Asignar una responsabilidad al experto en información -la clase que tiene la información necesaria para realizar la responsabilidad-. ¿Cuál es el principio general para asignar responsabilidades a los objetos? (26) Su uso se encuentra enmarcado fuertemente en el mapeo y abstracción de la base de datos, se puede reflejar siempre que se generan de forma automática ambas capas del modelo, es uno de los patrones más utilizados, debido a que tanto Propel como Doctrine, generan las clases de acceso a datos asignándoles las responsabilidades de ejecutar todas las funcionalidades comunes de las entidades que representan y de la cual poseen información. Es válido mencionar que este patrón suele ser usado por el programador, pero es el propio *framework* quien se encarga de implementarlo a través de los dos ORM de los que se puede hacer uso.

Patrones GOF.

A principios de los años 90 con la publicación del libro *Design Patterns*, se establecen veintitrés patrones de diseño GOF. El número exacto de estos patrones ha variado como consecuencia de los

años de experiencia alcanzados en la materia, las soluciones encontradas para resolver problemas específicos de un lenguaje de programación y los cuales se han considerado patrones, así como las distintas publicaciones referentes al tema y autores que se han pronunciado. Sin embargo son los patrones del libro *Design Patterns* los considerados comunes y aceptados por la mayoría de los especialistas del tema.

Los patrones GOF según su propósito se clasifican en tres grupos. Los Creacionales tratan la creación de instancias. Los Estructurales tratan la relación entre clases, la combinación clases y la formación de estructuras de mayor complejidad. Mientras que los de Comportamiento tratan la interacción y cooperación entre clases. A su vez según su ámbito se pueden clasificar de Clase, basados en la herencia de clases y de Objetos basados en la utilización dinámica de objetos.

Seguidamente se relacionan algunos de los patrones GOF que se pueden encontrar implementados en el interior de Symfony.

- **Solitario (*Singleton*):** Patrón creacional a nivel de objetos. Su propósito es garantizar que una clase sólo tenga una única instancia, proporcionando un punto de acceso global a la misma. (29) Este patrón es utilizado en la clase “sfContext” la cual almacena una referencia a todos los objetos que forman el núcleo de Symfony. Debido que se encarga de enrutar todas las peticiones que se realizan a la aplicación.

- **Fábrica Abstracta (*Abstract Factory*):** Patrón creacional a nivel de objetos. Su propósito es proporcionar una interfaz para la creación de familias de objetos interdependientes o interrelacionados, sin especificar sus clases concretas. (29) Su aplicación se puede evidenciar en la clase “sfContext” utilizada cuando el *framework* necesita crear un nuevo objeto para una petición, busca en la definición de la factoría el nombre de la clase que se debe utilizar para esta tarea, de esta forma se simplifica la decisión abstrayéndola dentro de una clase especializada.

- **Decorador (*Decorator*):** Patrón estructural a nivel de objetos. Su propósito es añadir responsabilidades adicionales a un objeto dinámicamente, proporcionando una alternativa flexible a la especialización mediante herencia, cuando se trata de añadir funcionalidades. (29) Es aplicado a las vistas donde el contenido de la plantilla se integra en el *layout*, o si se mira desde el otro punto de vista, el *layout* decora la plantilla. De manera que el *layout* contiene el código HTML y los elementos comunes a todas las páginas, mientras las plantillas se encargan de mostrar el resultado

de las acciones. Las clases “sfView”, “sfFilter”, y “sfWidgetFormSchemaDecorator” implementa dicho patrón.

- **Fachada (*Facade*):** Patrón estructural a nivel de objetos. Su propósito es proporcionar una interfaz unificada de alto nivel que, representando a todo un subsistema, facilite su uso. La “fachada” satisface a la mayoría de los clientes, sin ocultar las funciones de menor nivel a aquellos que necesiten acceder a ellas. (29) Se evidencia en la clase “sfConfig”, la cual crea un objeto que proporciona métodos estáticos para poder acceder a los parámetros de configuración desde cualquier punto de la aplicación.

- **Acción (*Command*):** Patrón de comportamiento a nivel de objetos. Su propósito es encapsular en un objeto la acción que satisface una petición, permitiendo ejecutar dicha operación sin necesidad de conocer el contenido de la misma. (29) Su aplicación se evidencia en la clase “sfFrontWebController” esta clase es la encargada de determinar cual módulo y acción debe responder a las solicitudes de los usuarios. Encapsula las peticiones en forma de objetos permitiendo así parametrizar los clientes utilizando distintas peticiones, encolar las peticiones y ofrecer la posibilidad de deshacer las operaciones.

- **Cadena de Responsabilidades (*Chain of Responsibility*):** Patrón de comportamiento a nivel de objetos. Su propósito es proporcionar a más de un objeto la capacidad de atender una petición, para así evitar el acoplamiento con el objeto que hace la petición. Se forma con estos objetos una cadena, en la cual cada objeto satisface la petición o la pasa al siguiente. Su aplicación se encuentra reflejada en la clase “sfEventDispatcher”, la cual rige el funcionamiento casi completo del *framework*.

3.3. Modelo de Diseño.

El modelo de diseño es una abstracción de la implementación del sistema. Se utiliza para concebir y documentar el diseño del sistema de software. Es un producto de trabajo integral y compuesto que contiene los artefactos, clase de diseño, interfaz, paquete de diseño, subsistema de diseño, sucesos, señal, clase de comprobabilidad, ejecución de guiones de uso, operación, realización de una operación, y componente de servicio. (7)

RUP permite a los arquitectos de software y diseñadores decidir sobre los siguientes aspectos de un modelo de diseño: las propiedades que desea incluir; si son necesarias o no ampliaciones de UML; el

nivel de formalidad aplicado al modelo; la personalización aplicable a subproductos de trabajo individuales; y cómo se correlaciona el modelo con el modelo de análisis; si se utilizará un modelo único o múltiples modelos; si el modelo será una especificación abstracta, una especificación detallada, un diseño detallado, o alguna combinación; y cómo se correlaciona con el modelo de implementación. (7)

3.3.1. Diagrama de Clases del Diseño.

Los diagramas de clases del diseño (DCD) son subproductos del modelo de diseño, para el presente trabajo fue necesario personalizarlos haciendo uso de las posibilidades permitidas por la metodología RUP, para así lograr una mejor representación del diseño del nuevo Subsistema de Configuración. Después de realizar un estudio del *framework* Symfony, donde se analizaron los elementos principales de su arquitectura, así como los patrones de diseño utilizados por el mismo; se ha decidido solo representar los elementos considerados esenciales, con los que se van a relacionar el programador para la implementación de cada una de las funcionalidades, los cuales se describen a continuación.

- **Modelo, Vista, y Controlador:** representan paquetes lógicos en correspondencia con la arquitectura del *framework*.
- **Doctrine:** representa de manera lógica el ORM seleccionado para implementación del nuevo sistema y responde al deseo del proyecto PRIMICIA de utilizar las versiones más actuales del *framework*.
- **Symfony:** paquete estereotipado <<framework>> representa todos los patrones de diseño inmersos en el interior del mismo, así como las clases y elementos que permiten su funcionamiento.
- **i18n:** representación física del lugar donde tienen que estar ubicados los diccionarios de traducción.
- **index:** este elemento es único y representa el controlador frontal, es creado de manera automática para toda la aplicación, todo el flujo de trabajo pasa a través del mismo, considerándolo de suma importancia para el funcionamiento del *framework* y de ahí el motivo de su representación.
- **[Nombre]Actions**: representa las nuevas clases que se deben crear para contener las acciones que controlan cada una de las funcionalidades del nuevo módulo.
- **[Nombre], [Nombre]Table, Base[Nombre]:** constituyen las clases de acceso a datos y son generadas de forma automática por el ORM Doctrine. Los programadores deben escribir la mayor

parte de sus códigos de lógica de negocio en la clase [Nombre], quedando en las acciones de la capa controlador la menor parte.

- **[nombre]Success:** son los archivos que contienen todo el código referente a la presentación del contenido, deben ser creadas por los programadores.
- **layout:** elemento utilizado para decorar las páginas de presentación del contenido, su representación confirma que dichas páginas deben ser decorados por un único elemento de este tipo.
- **app.yml, module.yml:** estos dos archivos guardaran los datos de configuración del proyecto a nivel de la aplicación y de módulo.

A continuación en la **Tabla 14**, se describen los estereotipos utilizados para relacionar las clases y paquetes antes mencionados. Los mismos fueron definidos desde las primeras versiones de UML en el libro "El Lenguaje Unificado de Modelado" escrito por Booch, Rumbaugh y Jacobson, aún continúan formando parte de la versión 2.1 de UML el cual se encuentra integrado a la herramienta de modelado Visual Paradigm utilizada para la creación de los diagramas.

Tabla 14: Estereotipos UML usados en los Diagramas de Clases del Diseño.

Estereotipo	Descripción
<<instantiate>>	Especifica que hay operaciones en la clase origen que crean instancias de la clase destino.
<<use>>	Especifica que la semántica del elemento origen depende de la semántica de la parte pública del destino.

A finales de los años 90 cuando el desarrollo de aplicaciones web se hizo más importante Jim Conallen describe la extensión de UML para el modelado de aplicaciones web en el artículo "*Modelling Web Applications Architectures with UML*",. Sus extensiones presentan como elementos más significativos a 3 clases de UML estereotipadas como *"Server Page"*, *"Client Page"*, y *"Form"* empleados para el código servidor, código cliente y formularios respectivamente, las cuales son usadas en la construcción de los diagramas en cuestión. Para relacionar estos tipos de clase fue necesario hacer uso de los estereotipos relacionados en la **Tabla 15**, definidos por el propio Conallen.

Tabla 15: Estereotipos Web usados en los Diagramas de Clases del Diseño.

Estereotipo	Descripción
<<build>>	Relaciona una página cliente con la del servidor, se expresa como que las páginas que se encuentran en el servidor construyen las páginas en el cliente.

<<submit>>	Relaciona un formulario y una página del servidor, donde el primero manda los valores de sus campos al servidor, para ser procesados por la página servidor.

A continuación se muestran los diagramas de clase del diseño separados por caso de uso.

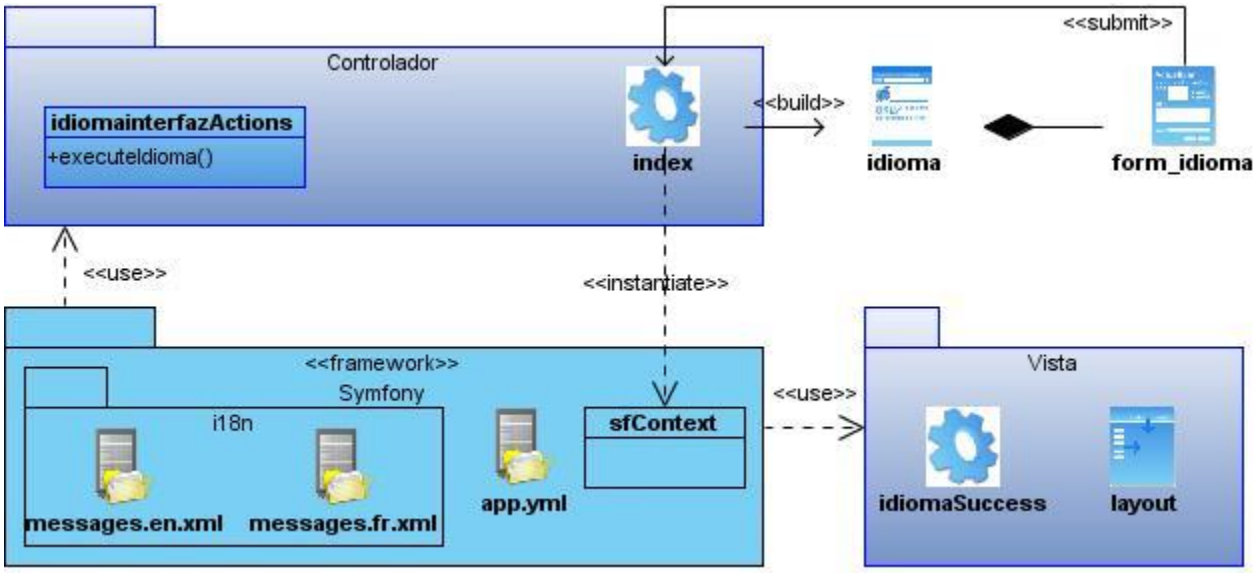

Figura 9: DCD Determinar Idioma de la Interfaz de Usuario.

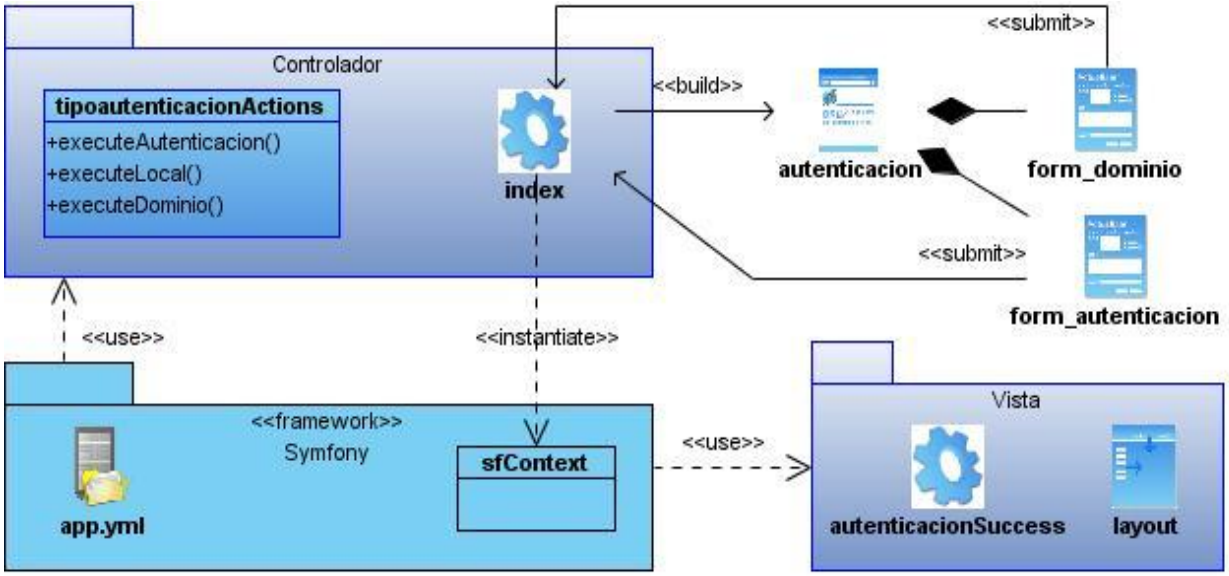

Figura 10: DCD Determinar Tipo de Autenticación.

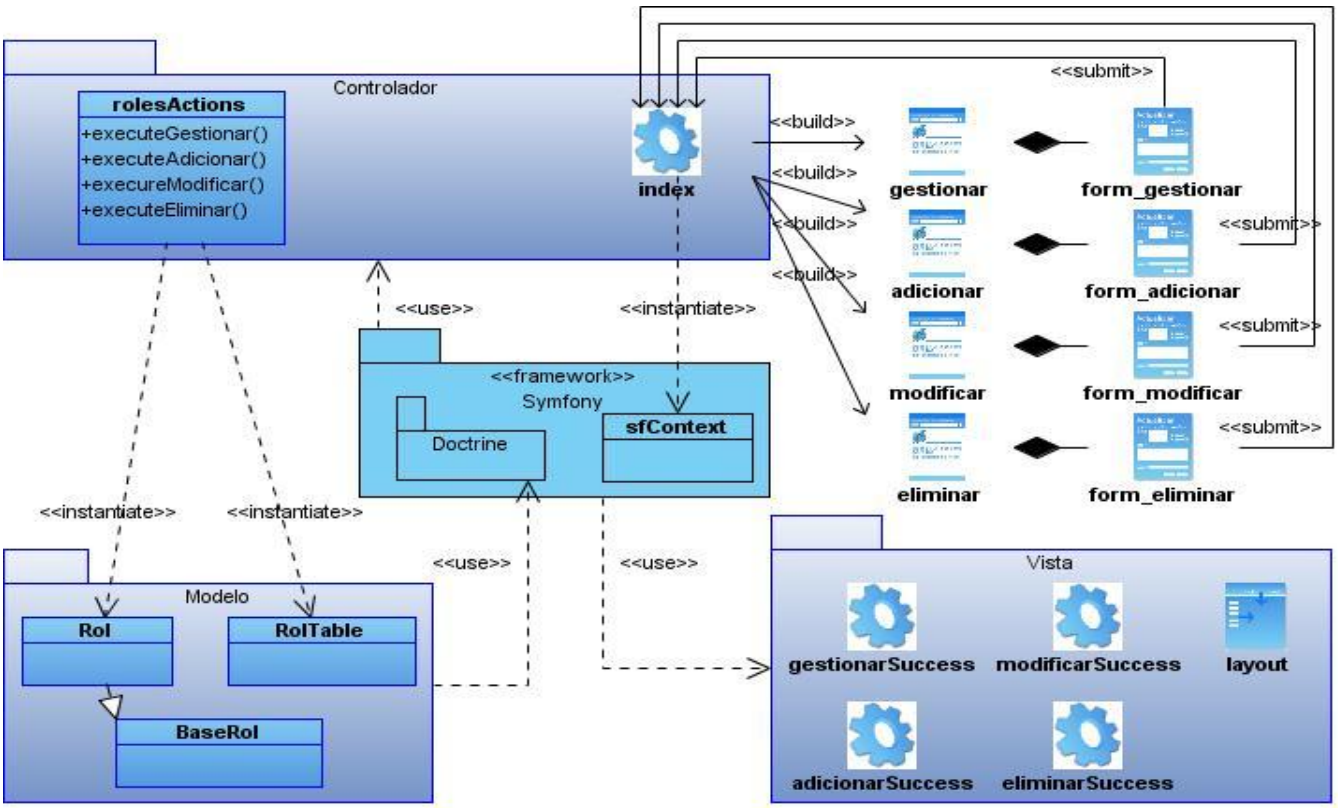

Figura 11: DCD Gestionar Roles de Usuarios.

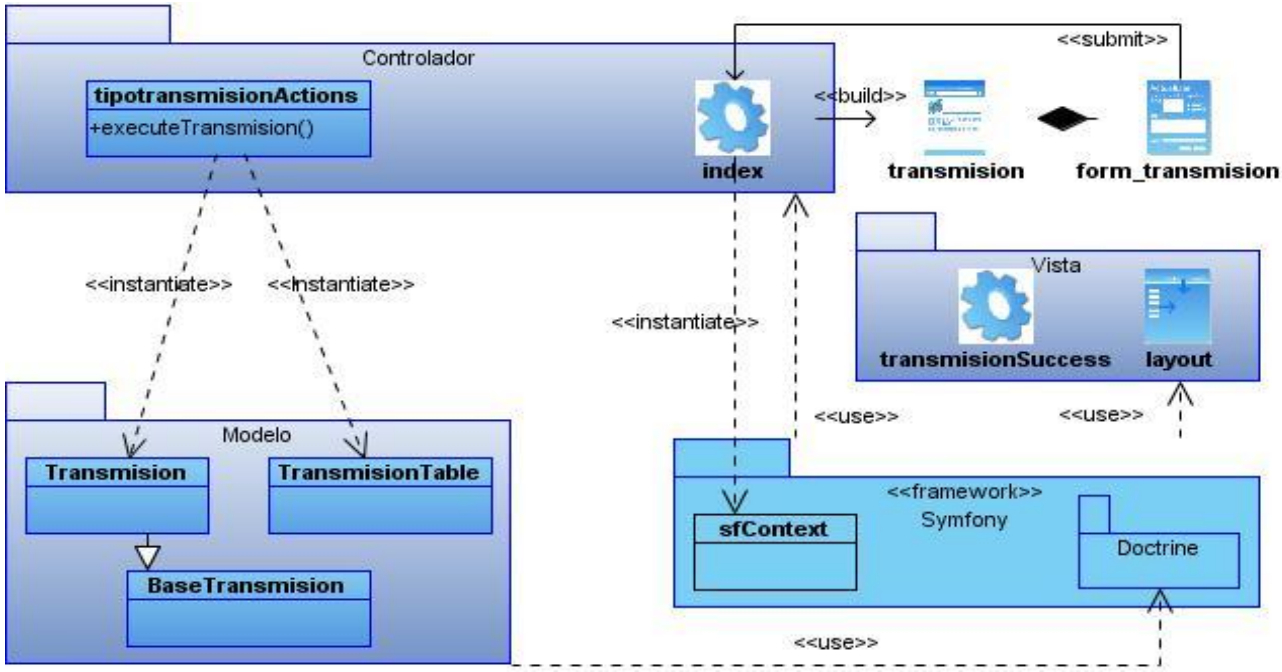

Figura 12: DCD Determinar Tipo de Transmisión.

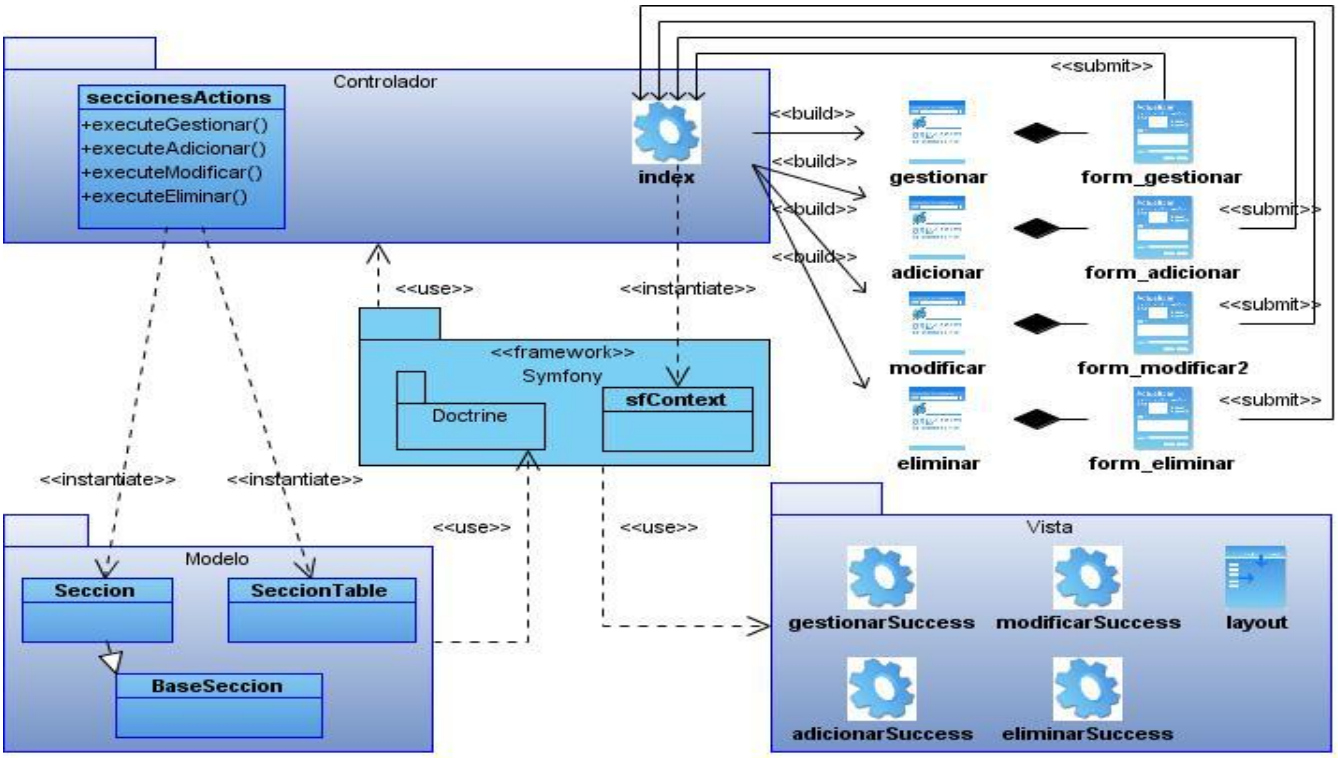

Figura 13: DCD Gestionar Secciones Temáticas.

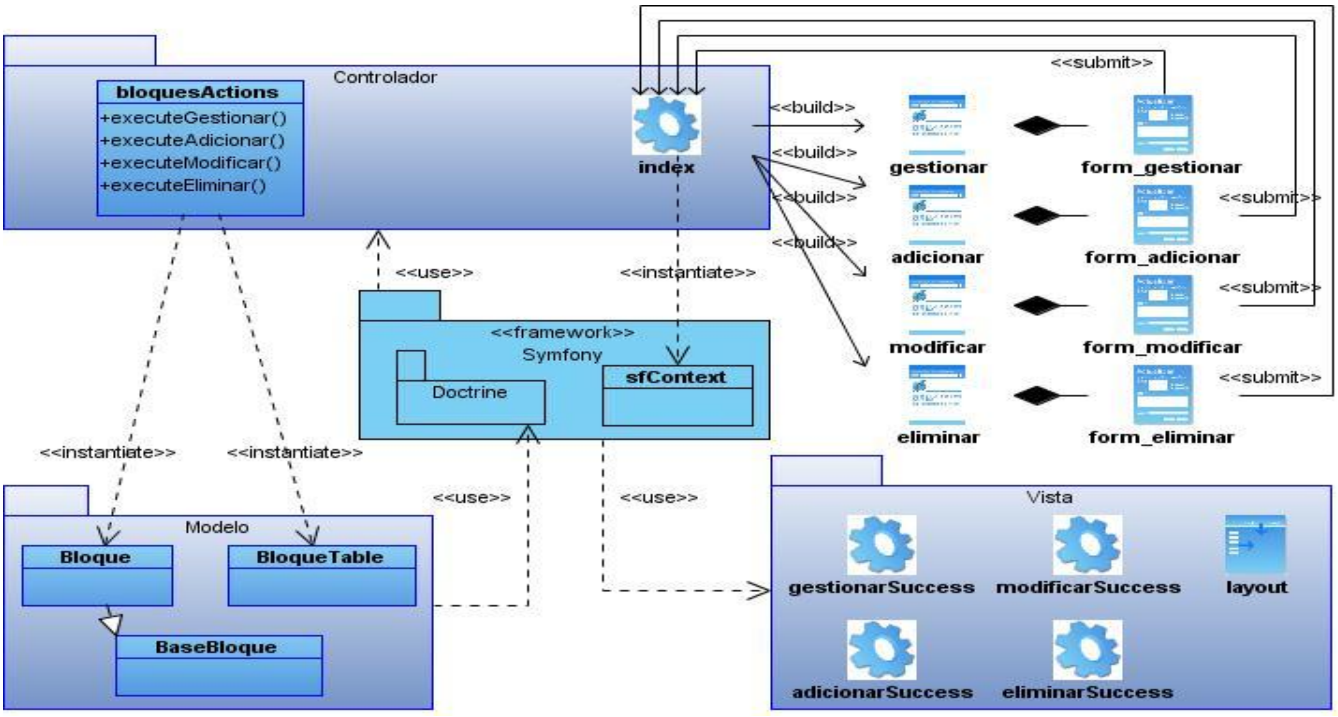

Figura 14: DCD Gestionar Bloques Noticiosos.

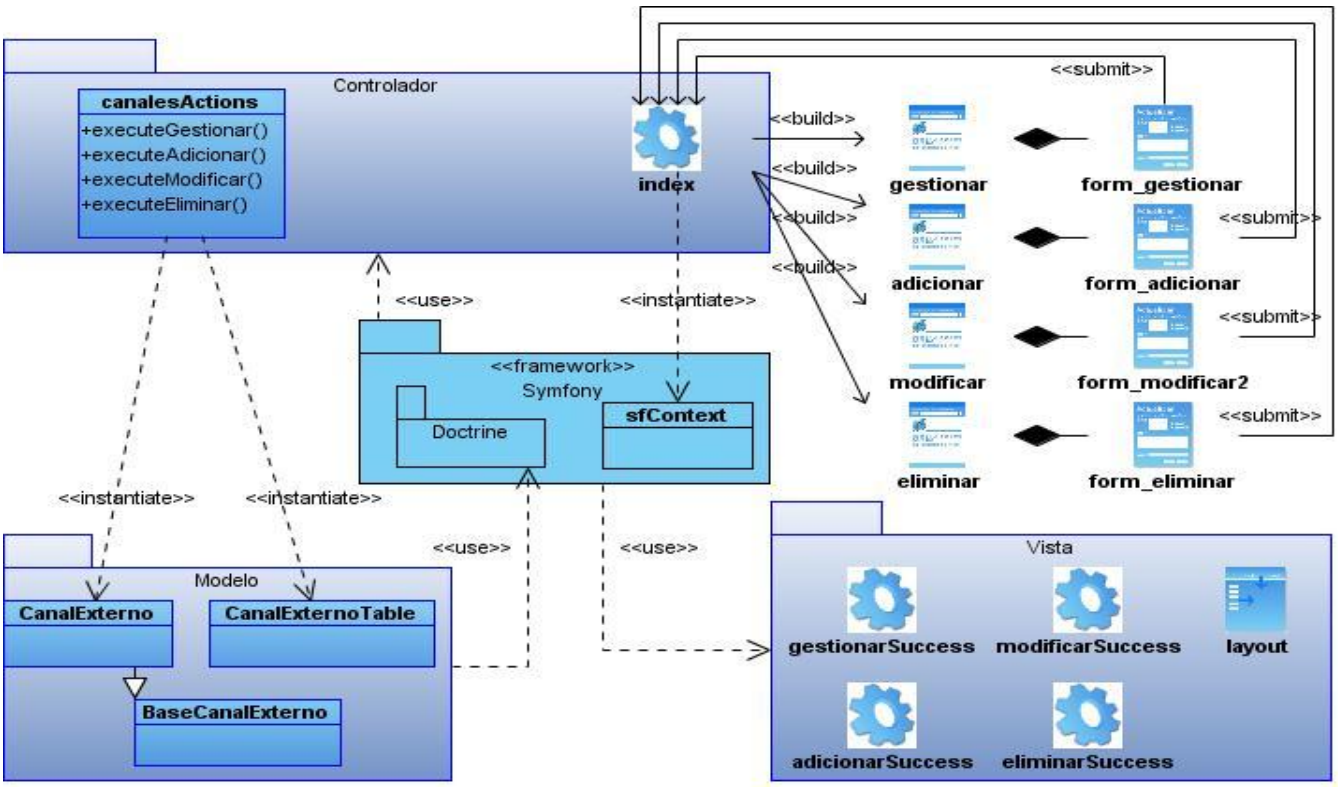

Figura 15: DCD Gestionar Canales Externos.

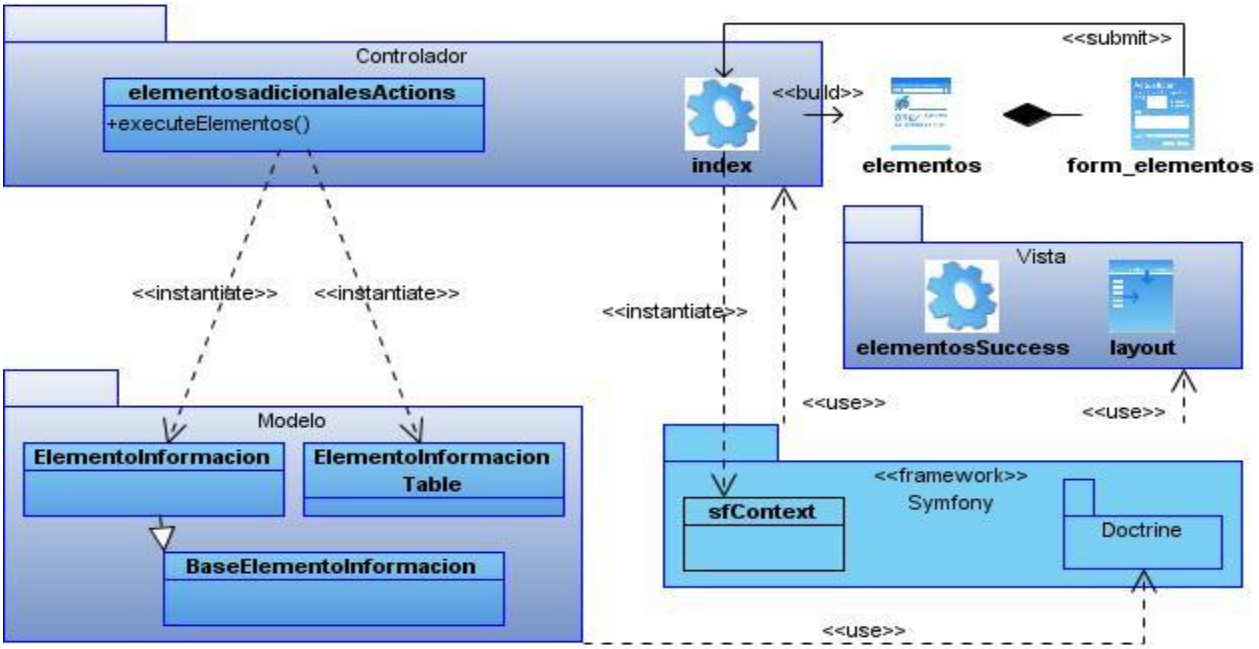

Figura 16: DCD Configurar Elementos de Información.

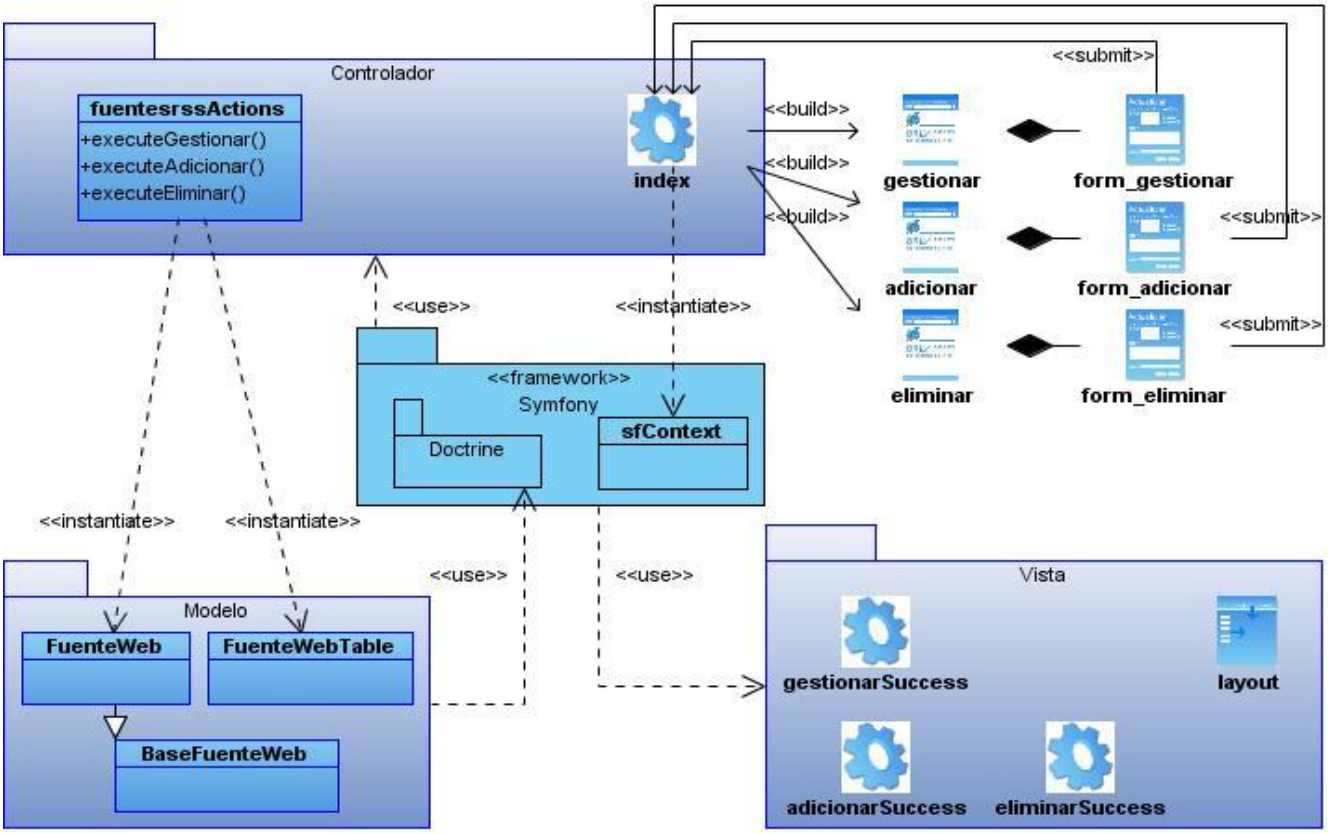

Figura 17: DCD Gestionar Fuentes Web.

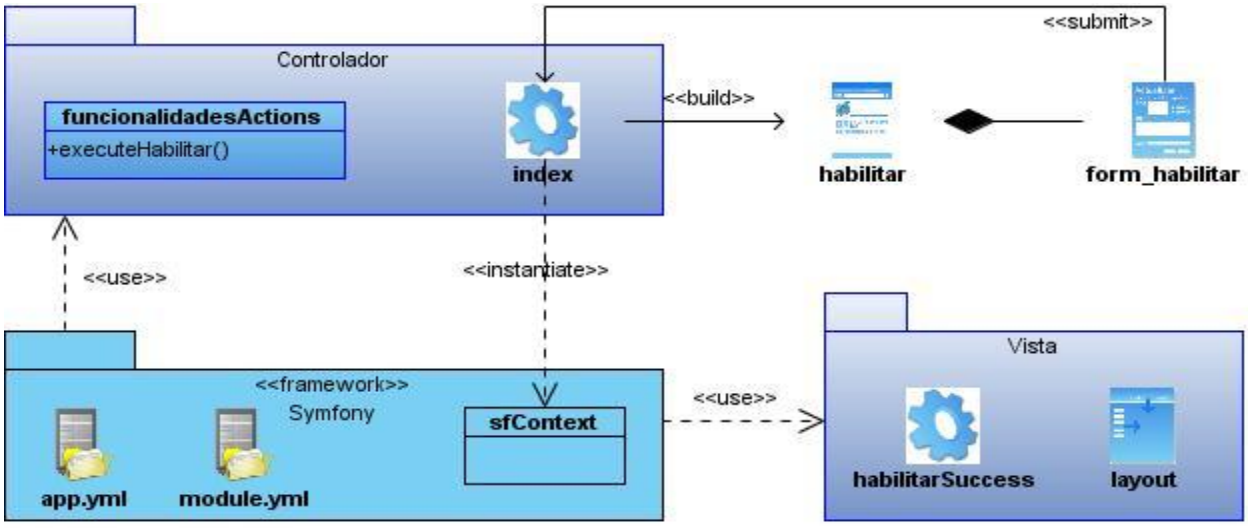

Figura 18: DCD Habilitar/Deshabilitar Módulos y Funcionalidades.

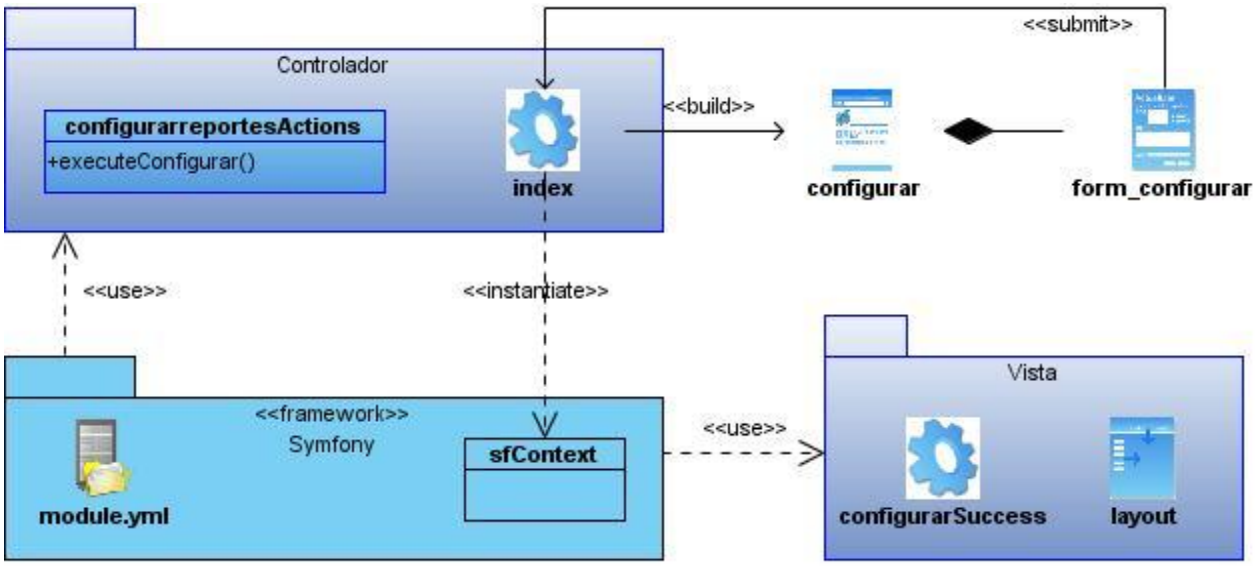

Figura 19: DCD Configurar Reportes.

3.3.2. Diagrama de Clases Persistentes.

Las clases que deben almacenar su estado en un medio permanente se conocen como persistentes. La necesidad de almacenar su estado puede ser para el registro permanente de información de las clases, como copia de seguridad en el caso de una anomalía del sistema, o para el intercambio de información. Los objetos persistentes puede que no se deriven solo de las clases de entidad; los objetos persistentes también pueden ser necesarios para manejar requisitos no funcionales en general. Algunos ejemplos son los objetos persistentes, necesarios para mantener la información relevante para el control de procesos o para mantener información de estado entre transacciones.

La identificación de clases persistentes, permite notificar al diseñador de base de datos que la clase necesita especial atención, según las características de almacenamiento físico. También notifica al arquitecto de software que la clase debe ser persistente y al diseñador responsable del mecanismo de persistencia, que las instancias de la clase deben convertirse en persistentes.

En el diagrama que se muestra a continuación se representas las nuevas clases persistentes que se obtuvieron del diseño, así como aquellas clases que ya existían con anterioridad y tenían relación directa con las nuevas clases o requerían de alguna modificación.

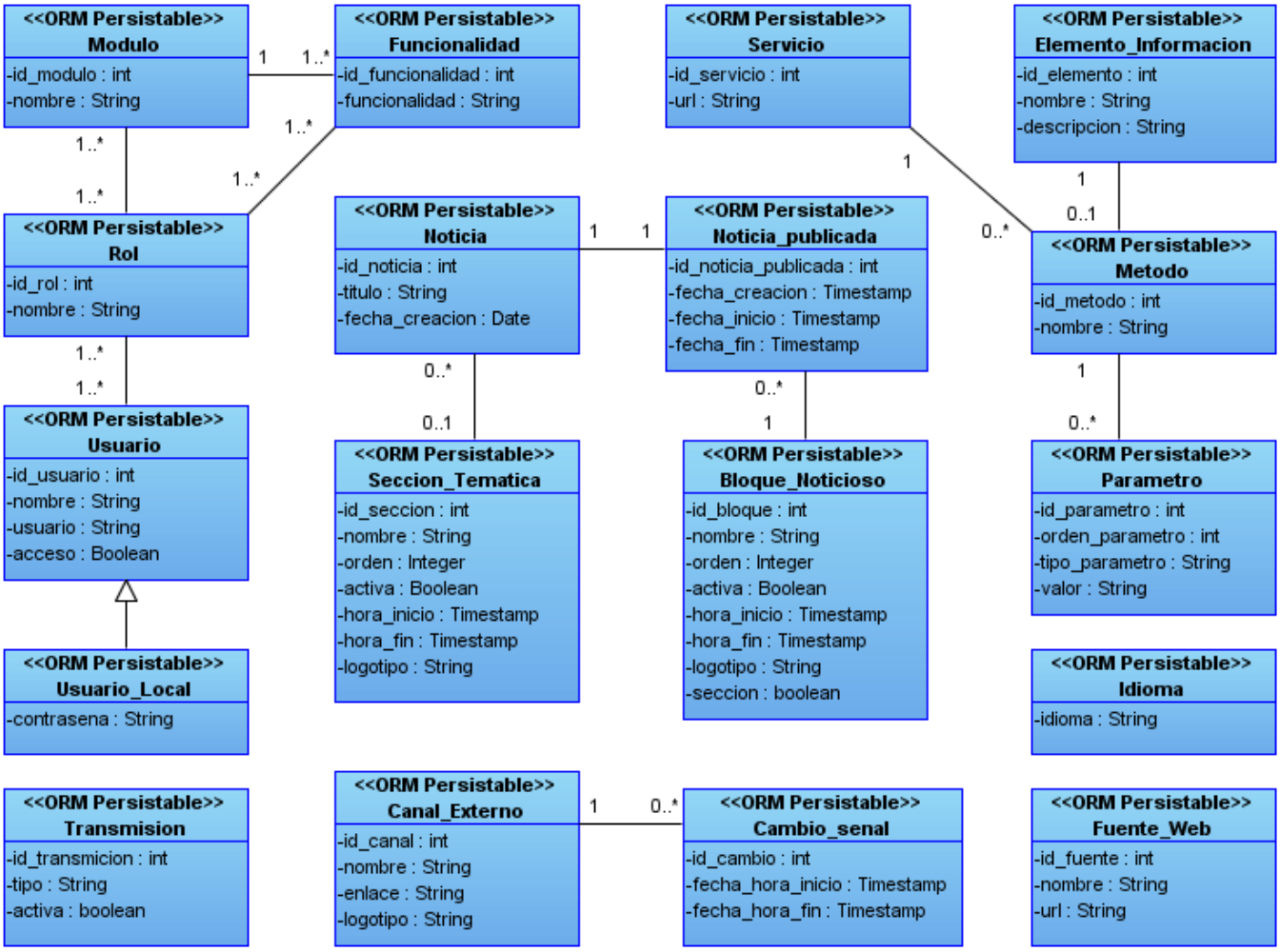

Figura 20: Diagrama de Clases Persistentes.

3.4. Modelo de Datos.

El modelo de datos se puede crear revirtiendo la ingeniería de los almacenes de datos persistentes "bases de datos" o se puede crear inicialmente desde un conjunto de Clases de diseño persistentes en el Modelo de diseño. (7) En este trabajo se ha empleado las dos variantes anteriormente mencionadas, la primera variante debido a que la plataforma ya contaba con una base de datos bien definida, y la segunda para incorporadas las nuevas tablas obtenidas de las clases persistentes del diseño.

Este artefacto describe las representaciones lógicas y físicas de datos persistentes gestionados por el sistema. Se utiliza para definir la correlación entre las clases de diseño persistentes y las estructuras de datos persistentes, y para definir las estructuras de datos persistentes. Es necesario siempre que el mecanismo de almacenamiento persistente se base en alguna tecnología no orientada al objeto. (7)

3.4.1. Diagrama Entidad-Relación

Los diagramas entidad relación representan física y lógicamente los datos que son guardados en cada una de las tablas de la base de datos que usa el sistema. Teniendo en cuenta que PRIMICIA es un sistema que posee una base de datos ya definida y con este trabajo se va incorporar nuevas funcionalidades al sistema, se representan las entidades del siguiente modo: color naranja las nuevas tablas que se incorporan, color verde las tablas que ya existían pero necesitan ser modificadas y color azul las tablas existentes que no requieren ningún cambio pero tiene una estrecha relación con alguna de las tablas incorporadas o modificadas.

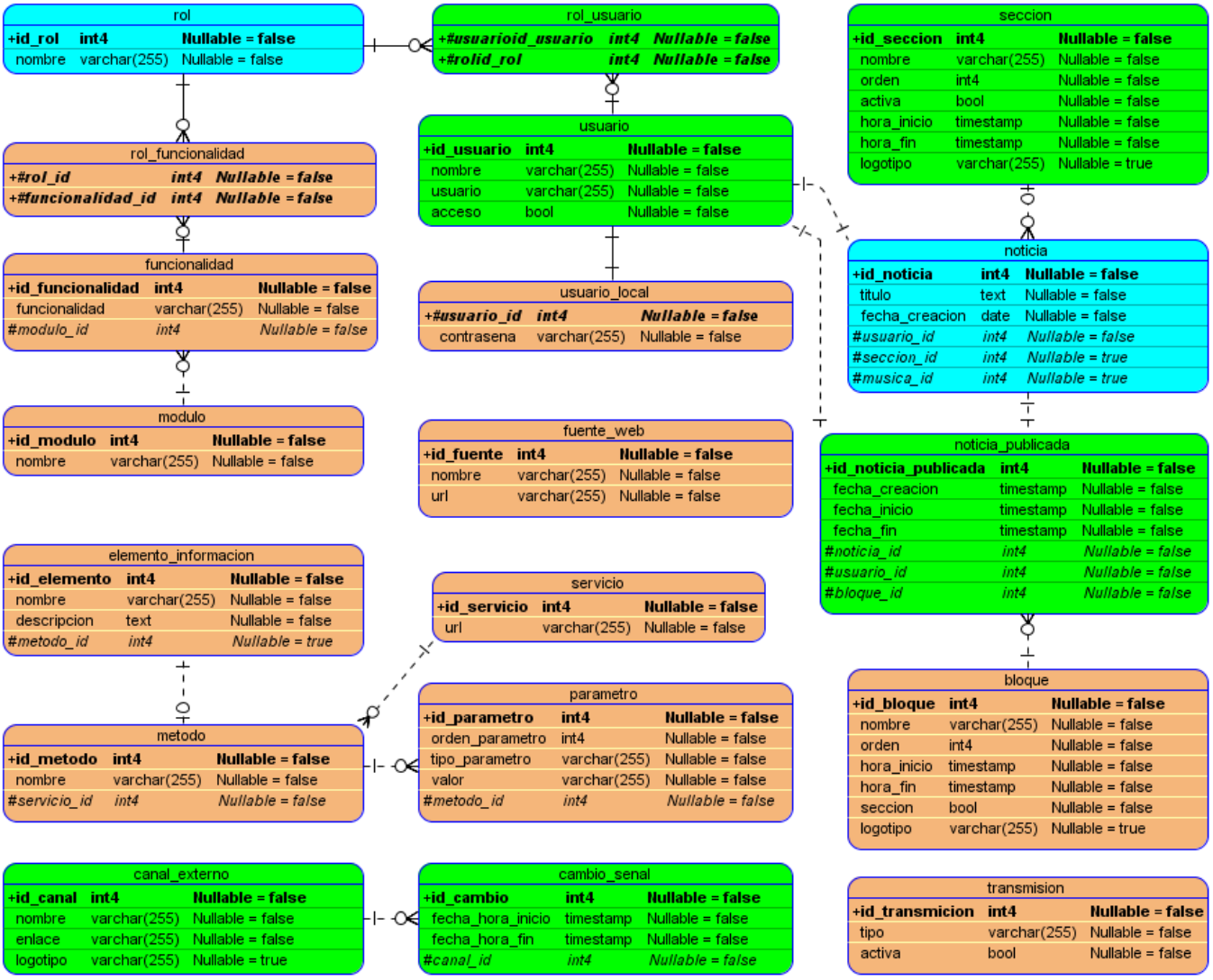

Figura 21: Diagrama Entidad - Relación.

3.4.2. Ficheros de Symfony.

Una de las grandes ventajas de Symfony es que desde cualquier punto de la aplicación se puede acceder a los valores establecidos en los archivos de configuración. Teniendo en cuenta esta característica, se decidió potenciar el uso de los ficheros de configuración del *framework* a nivel aplicación y módulo. En varias de las funcionalidades diseñadas quedo plasmados el uso de dichos archivos. Por lo general se encuentran escrito en YAML, aunque puede usarse otro formato como el caso de XML y PHP.

En los diagramas de diseño se hizo uso de los archivos “app.yml” y “module.yml”, los cuales tendrán un significado y uso muy especial para servir de apoyo a cuatro de las funcionalidades de configuración y a la aplicación en general. A continuación se muestra el código que deberán contener cada uno de estos archivos.

```yaml
all:
  idioma_iu: es

  .modulos:
    modulo_1: true
    modulo_2: true
    modulo_3: false

  .tipo_autenticacion:
    local:    true
    dominio:  false

  .autenticacion_dominio:
    servidor_ldap:    10.0.0.3
    servidor_dns:     uci.cu
    tipo_servidor:    ActiveDirectory
    usuario_ldap:     samaccountname
    clave_ldap:       uF2SODWAHiW0eJboFFQEAvVzJ
    version_ldap:     3
```

Figura 22: Contenido del archivo “app.yml”.

```yaml
all:
  .funcionalidades:
    funcionalidad_1: true
    funcionalidad_2: true
    funcionalidad_3: false
```

Figura 23: Contenido del archivo “module.yml”.

3.5. Conclusiones.

Durante este capítulo se obtuvo los diagramas de clases de análisis, como primer paso para llegar a la construcción de los diagramas de clases del diseño y para reflejar una visión general de las clases a tener en cuenta para implementar las nuevas funcionalidades. Se desarrolló un análisis de la arquitectura y diseño de Symfony, el cual fue necesario para comprender la misma, el uso de los patrones de diseño y los elementos principales del *framework*. Todo lo anterior sirvió para determinar los elementos esenciales con los que interactúan el programador y el diseñador, de esta manera obtener una mejor representación de los diagramas de clases del diseño. También se identificaron las clases persistentes quedando plasmadas en su diagrama correspondiente y las cuales dieron paso a la construcción del modelo de datos. Además se especificaron los datos que deben contener los ficheros “app.yml” y “module.yml”. Con el diagrama entidad relación se concluyó el capítulo, determinando todos los cambios necesarios a realizar en la base de datos de PRIMICIA, para obtener de forma satisfactoria las funcionalidades de configuración requeridas.

CONCLUSIONES GENERALES

Después de haber desarrollado y analizado los resultados obtenidos con la elaboración del presente trabajo y la culminación del diseño de las funcionalidades de configuración para la Plataforma de Televisión Informativa PRIMICIA, se llegaron a las siguientes conclusiones:

- Con el empleo de los métodos de investigación teóricos y empíricos se logró conocer el estado del objeto de estudio de la investigación. Analizando los principales conceptos asociados al tema de investigación y las características de configuración presentadas por tres de los gestores de contenidos más importantes; así como la identificación de los procesos y funcionalidades configurables presentes en la plataforma, se pudo comprender de manera satisfactoria toda la situación problemática y definieron los conceptos teóricos – técnicos asociados a la configuración de PRIMICIA.
- Como parte de las tareas a cumplir en el presente trabajo quedaron expuestos los elementos que sustentan la selección de la metodología RUP para guiar todo el proceso de desarrollo, así como el lenguaje de modelado UML, permitiendo elaborar cada uno de los artefactos realizados a lo largo de todo el trabajo. Siguiendo con lo planteado por la metodología RUP, se realizó el modelo de dominio como parte del flujo de trabajo Modelado del Negocio, el cual permitió identificar los principales conceptos y sus relaciones, involucrados en el contexto del problema. De igual manera se desarrolló el levantamiento de requisitos, dando la posibilidad de identificar los casos de uso del sistema como elemento clave en el desarrollo de software apoyado en la metodología RUP.
- A través del análisis de la solución propuesta se pudo estructurar los requisitos definidos previamente, con la construcción del modelo de análisis y a su vez este constituye un primer acercamiento al diseño. Luego se realizo un análisis del *framework* Symfony dando paso a realizar un diseño personalizado y orientado a objetos con la utilización del mismo. Con la identificación de las clases persistentes fue posible construir el modelo de datos, para de esta forma dejar plasmado de manera fiable los cambios necesarios en la base de datos, para la implementación de las nuevas funcionalidades de configuración.

Por todo lo anterior se concluye que los objetivos propuestos para el presente trabajo han sido cumplidos satisfactoriamente.

RECOMENDACIONES

Una vez cumplido con los objetivos del presente trabajo y en correspondencia con los resultados obtenidos, los cuales se pusieron en práctica en el propio documento, se recomienda:

- Realizar la implementación del nuevo subsistema a partir del resultado de esta investigación, bajo las restricciones y especificaciones plasmadas en este documento.
- Continuar documentando la realización de los flujos de trabajo Implementación, Prueba y Despliegue propuestos por la metodología RUP, partiendo de los artefactos generados en el presente trabajo.
- Aplicar al proyecto PRIMICIA los nuevos conocimientos adquiridos en el análisis y diseño de las funcionalidades de configuración de la propia plataforma.

REFERENCIAS BIBLIOGRÁFICAS

1. **Definicion.de.** *Definicion.de.* [En línea] 2008. [Citado el: 11 de diciembre de 2009.] http://definicion.de/modulo.

2. **Real Academia Española.** Real Academia Española. *Real Academia Española.* [En línea] [Citado el: 9 de diciembre de 2009.] http://buscon.rae.es/draeI/SrvltConsulta?TIPO_BUS=3&LEMA=modulo.

3. **ALEGSA.** Diccionario Informatico. *Diccionario Informatico.* [En línea] 2009. [Citado el: 11 de diciembre de 2009.] http://www.alegsa.com.ar/Dic/modulo.php.

4. **Real Academia Española.** Real Academia Española. *Real Academia Española.* [En línea] [Citado el: 10 de diciembre de 2009.] http://buscon.rae.es/draeI/SrvltConsulta?TIPO_BUS=3&LEMA=configuraci%C3%B3n.

5. **Diccionario de sinónimos y antónimos Espasa-Calpe.** WordReference.com. *WordReference.com.* [En línea] [Citado el: 10 de Diciembre de 2009.] http://www.wordreference.com/sinonimos/configuracion.

6. **Gómez, Levián Lara.** *CONCEPTUALIZACIÓN DE LA AUTOMATIZACIÓN DE LA PERSONALIZACIÓN Y CONFIGURACIÓN DE PRIMICIA.* Ciudad de La Habana : s.n., 2009.

7. **IBM Corp.** *Ayuda del Rational(Español).* [Sitio Web] s.l. : Rational Unified Process Versión 7.0.1, 2006.

8. **Almaguer, Bernardo Rey.** *Documento de Arquitectura de Software. Plataforma de Televisión Informativa, PRIMICIA.Versión 1.0.* 2008.

9. **Reyero, Jose A.** Drupal Hispano. *Comunidad de usuarios de Drupal.* [En línea] 16 de Marzo de 2006. [Citado el: 12 de Diciembre de 2009.] http://drupal.org.es/drupal.

10. —. Drupal Hispano. *Comunidad de usuarios de Drupal.* [En línea] 17 de Julio de 2005. [Citado el: 12 de Diciembre de 2009.] http://drupal.org.es/caracteristicas.

11. **WordPress.org.** WordPress.org. *WordPress.org.* [En línea] [Citado el: 12 de Diciembre de 2009.] http://codex.wordpress.org/Shortcode_API.

12. **WordPress.** *WordPress Español.* [En línea] [Citado el: 12 de Diciembre de 2009.] http://es.wordpress.org/.

13. **Joomla Spanish.** Joomla Spanish. *Joomla Spanish.* [En línea] [Citado el: 12 de Diciembre de 2009.] http://ayuda.joomlaspanish.org.

14. **Ayllapan, Walter Ulises.** Joomlaos.net. *Joomlaos.net.* [En línea] 2009. [Citado el: 13 de Diciembre de 2009.] http://www.joomlaos.net/caracteristicas-de-joomla.php.

15. **Kendall, Kenneth.** *Análisis y Diseño de Sistemas.* 1997.

16. **ETEROVIC, Y.** *El Proceso Unificado (RUP): Técnicas Modernas para Desarrollar Aplicaciones.* [En línea] 2001. [Citado el: 2 de Marzo de 2010.] http://www2.ing.puc.cl/~iic3194/rup.ppt.

17. **Jacobson, Ivar, Booch, Grady y Rumbaugh, James.** *El Proceso Unificado de Desarrollo de Software.* 2000.

18. **VALENCIA, U.P.D.** *Introduccion a RUP.* [En línea] 2007. https://pid.dsic.upv.es/C1/Material/Documentos%20Disponibles/Introducci%C3%B3n%20a%20RUP.doc.

19. **Sanchez, María A Mendoza.** *Metodologías De Desarrollo De Software.* Perú S.A.C. : s.n., 2004.

20. **Orallo, E. H.** Entorno Virtual de Aprendizaje. *El Lenguaje Unificado de Modelado (UML).* [En línea] 2008. [Citado el: 4 de Marzo de 2010.] http://eva.uci.cu/file.php/102/Curso_2008-2009/Materiales_Complementarios/Materiales_Complementarios_Conf_1/UML.pdf.

21. **Schmuller, Joseph.** *Aprendiendo UML en 24 Horas.* México : Pearson Educación , 2000.

22. **Pérez, M.** *Arquitectura para Ambientes CASE Integrados.* Universidad Central de Las Villas : s.n., 1999.

23. **Martínez Borges, Lisyen y Martínez Méndez, Anisley.** *Sistema para informatizar el proceso judicial de los Tribunales Militares Regionales en Cuba. Rol Analista de Sistemas.* Ciudad de La Habana : s.n., 2007.

24. **Visual Paradigm.** [En línea] http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_%5Bcuenta_de_Plataforma_de_Java_14715_p/.

25. **Garcia Duardo, Ivanni.** *PRIMICIA, Plataforma de Televisión Informativa. Rol Analista.* Ciudad de La Habana : s.n., 2009.

26. **Larman, Craig.** *UML Y PATRONES.* Mexico : s.n., 1999.

27. **IEEE 610.** Thiyagarajan Veluchamy. *Thiyagarajan Veluchamy.* [En línea] [Citado el: 6 de Febrero de 2010.] http://thiyagarajan.wordpress.com/glossary/.

28. *Flujo de trabajo de requerimientos.* **Universidad de las Ciencias Informaticas.** Ciudad de La Habana : s.n., 2007.

29. **Facultad de informática - Universidad Politecnica de Madrid.** *Patrones "Gang of Four" .* s.l. : Unidad Docente de Ingeniería del Software.