UNIVERSIDAD DE ORIENTE

NÚCLEO DE ANZOÁTEGUI

ESCUELA DE INGENIERÍA Y CIENCIAS APLICADAS

DEPARTAMENTO DE COMPUTACIÓN Y SISTEMAS

DESARROLLO DE UNA HERRAMIENTA DE SOFTWARE QUE DISMINUYA LOS TIEMPOS DE CREACIÓN Y ADMINISTRACIÓN DE CONTENIDO DINÁMICO EN WEBSITES Y APLICACIONES WEB PARA UNA EMPRESA DE DESARROLLO DE SOFTWARE

Realizado por:

_____

Sánchez Vallenilla, Wilfredo Eduardo

C.I. V-17.262.710

Trabajo de grado presentado como requisito parcial para optar al título de

INGENIERO EN COMPUTACIÓN

BARCELONA, mayo de 2009

UNIVERSIDAD DE ORIENTE

NÚCLEO DE ANZOÁTEGUI

ESCUELA DE INGENIERÍA Y CIENCIAS APLICADAS

DEPARTAMENTO DE COMPUTACIÓN Y SISTEMAS

DESARROLLO DE UNA HERRAMIENTA DE SOFTWARE QUE DISMINUYA LOS TIEMPOS DE CREACIÓN Y ADMINISTRACIÓN DE CONTENIDO DINÁMICO EN WEBSITES Y APLICACIONES WEB PARA UNA EMPRESA DE DESARROLLO DE SOFTWARE

Asesor:

_____

Ing. Zulirais García

Asesor Académico

BARCELONA, mayo de 2009

UNIVERSIDAD DE ORIENTE

NÚCLEO DE ANZOÁTEGUI

ESCUELA DE INGENIERÍA Y CIENCIAS APLICADAS

DEPARTAMENTO DE COMPUTACIÓN Y SISTEMAS

DESARROLLO DE UNA HERRAMIENTA DE SOFTWARE QUE DISMINUYA LOS TIEMPOS DE CREACIÓN Y ADMINISTRACIÓN DE CONTENIDO DINÁMICO EN WEBSITES Y APLICACIONES WEB PARA UNA EMPRESA DE DESARROLLO DE SOFTWARE

JURADO CALIFICADOR

Ing. Zulirais García
Asesor Académico

Ing. Claudio Cortínez
Jurado Principal

Ing. Rhonald Rodríguez
Jurado Principal

BARCELONA, mayo de 2009

RESOLUCIÓN

De acuerdo con el artículo 44 del reglamento de trabajo de grado:

“Los trabajos de grado son de exclusiva propiedad de la Universidad de Oriente y sólo podrán ser utilizados a otros fines con el consentimiento del consejo de núcleo respectivo, quien lo participará al Consejo Universitario”.

DEDICATORIA

Dedico la satisfactoria culminación de este trabajo de grado a Dios, quien me ha dado las fuerzas y sabiduría necesarias para cumplir con todas las metas que actualmente me he trazado, y por la certeza de que seguirá bendiciéndome para lograr los éxitos que me esperan.

A mis padres, Miriam y Wilfredo, quienes siempre demostraron con su esfuerzo y dedicación ese inigualable amor que como padres nos tienen a mí y a mis hermanos.

Muy especialmente a mi prometida, Johana; te amo y dedico este logro a ti y a la vida de bendición que forjaremos juntos.

A todas las personas que han creído en mí, así como a todos aquellos que estén leyendo este trabajo.

Wilfredo Sánchez

AGRADECIMIENTOS

En primer lugar, mi más profundo agradecimiento a Dios porque sin su sabiduría, sus fuerzas ni la convicción de su compañía no habría llegado a ser quien soy. Él es mi más profunda base y, a su vez, mi más alta meta y corona. Dios, siempre te amaré y te adoraré.

Agradezco a mis padres, Miriam y Wilfredo, quienes desde pequeños nos enseñaron dos de los consejos más grandes que atesoro: confiarle a Dios nuestra vida y nuestros planes, y pedirle a Él sabiduría todos los días, pues Él nos la da si con fe se la pedimos. Papá, mamá, su amor y consejo son una sólida base en la que he podido edificar una vida de logros y estoy súper agradecido de ustedes por suplirme con amor y cariño de esa firme plataforma.

Agradezco a mis hermanos, Angel, Carlos y Javier, por su apoyo y su cariño. Sé que esa sana competitividad que desde pequeños tuvimos nos ha llevado a todos, incluyéndome, a dar lo máximo de nosotros mismos, y eso se ve materializado en los logros que juntos hemos tenido en Saetha. Somos un buen equipo. Los amo mucho.

Johana, estoy súper agradecido contigo por lo tanto que me has complementado como persona. Gracias por ser tan especial como eres; no habrían sido iguales estos pasos de mi vida y de mi carrera profesional sin tus palabras de aliento, tu cariño y comprensión; ¡te amo demasiado!

Agradezco a mis tíos y primos, en especial a Vanessa, por ese cariño tan único que nos ha unido desde pequeños. Los amo a todos.

A mis amigos Marcela, Alisabel, José Andrés, Leonel, Josué, Adelys, Abelardo, Joselín, Yosibel, Pr. Omar, Yrismar, Jhoana, Cristina, Yarnila, Angelly, Carlos, Glenda, Alexis, Marcos; a todos los amo muchísimo y estoy agradecido de las tantas cosas que he aprendido y compartido con ustedes; muchísimas gracias.

A mis compañeros y amigos, María José, Edgar, Evelyn, Aivett, Jesús, Mariela, Luis Eduardo, Franco, David, Carlos; tuvimos la dicha de estudiar juntos esta carrera, superando todas las dificultades y logrando con excelencia nuestros objetivos.

A Máxima, quien, además de ser mi primera maestra, siempre ha creído en mí y en mi familia y ha demostrado siempre el amor que nos tiene; eres muy especial.

Agradezco a mi asesora, Zulirais, por su apoyo mano a mano en la realización de este trabajo. Siempre le he admirado como profesora; prosiga dando ese ejemplo de educación responsable y de excelencia.

A todos ustedes, muchísimas gracias.

Wilfredo Sánchez

ÍNDICE

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

RESUMEN

La presente investigación consiste en el desarrollo de un sistema web para una empresa desarrolladora de software que permitirá a los analistas del departamento de diseño web implantar funcionalidad de contenido dinámico a los sitios web diseñados, sin necesidad de escribir código imperativo. Para lograr esto sólo se requiere establecer en un archivo XML de configuraciones las estructuras de las secciones y de los datos a almacenar en cada una de ellas, y hacer referencia a las instancias de contenido desde los formularios web con el uso de etiquetas XML.

El núcleo del sistema analiza el archivo de configuración y genera automáticamente el *Sistema administrador de contenidos* (*CMS*, por sus siglas en inglés), así como la estructura de la base de datos. Se pueden registrar usuarios cliente que accederán al CMS y establecer las secciones y aplicaciones específicas que estos podrán actualizar.

El diseño y desarrollo del proyecto se realizó según las fases del Proceso Unificado de Desarrollo de Software y utilizando el Lenguaje Unificado de Modelado UML 2.0. La aplicación se construyó e implementó utilizando la tecnología ASP.NET 3.5, el lenguaje de programación C# 3.0 y el manejador de base de datos SQL Server 2005 Express Edition.

CAPÍTULO 1
INTRODUCCIÓN

1.1. Planteamiento del problema

La empresa **Saetha Business Group** está radicada en Lechería, estado Anzoátegui. En su estructura organizacional se encuentra la **Dirección de Diseño y Sistemas**, que define sus dos áreas principales de producción de la siguiente manera:

- Software – desarrollo de aplicaciones a medida
- Web – creación, publicación y mantenimiento de páginas web

Cada línea de servicio cuenta con personal especializado, siendo conformado el sector *software* por ingenieros en computación y técnicos programadores, y el sector *web* por licenciados y técnicos en diseño gráfico y comunicación visual.

El área *web,* en término casi constante, refleja una gran demanda, por lo que se debe garantizar una máxima eficiencia en el proceso de desarrollo para poder mantener y lograr estabilidad pese a la cantidad de proyectos en ejecución.

Adicional a todos los conocimientos gráficos requeridos dentro del sector *web*, se ha visto frecuentemente añadido el conocimiento de *programación*, puesto que en muchos proyectos web se incluyen secciones de contenido dinámico; esto, debido a la cantidad masiva de información que manejarán y por la necesidad de que dicho contenido sea administrado por el

cliente. Esto obliga a incluir dentro de dichos proyectos a miembros del personal de *software*, que puedan implementar o adaptar código para dar vida a las secciones dinámicas que ya hayan sido conceptualizadas por los diseñadores.

Se crea entonces un verdadero cuello de botella dentro de todo el proceso de producción pues los programadores, que deben ocuparse principalmente de los proyectos de software a medida para empresas, necesitan dedicar tiempo para la creación de las secciones dinámicas de los websites, ocasionando retrasos tanto en sus cronogramas de desarrollo como en los tiempos de entrega de los websites.

Sin embargo, las secciones dinámicas que con frecuencia son requeridas son similares y categorizables, generándose una posible solución para la problemática: el desarrollo de una herramienta —o conjunto de ellas– que lleve a una mínima expresión la participación de los programadores en la creación de secciones web dinámicas al ofrecerle al diseñador la posibilidad de implementar la funcionalidad requerida con el uso de *controles web* parametrizados. Estos controles web se refieren a bloques y archivos de código abstraídos, que harán uso de marcas XHTML para su implementación en el etiquetado del web. De esta forma, se elimina la necesidad de que el diseñador adquiera conocimientos avanzados de programación y le posibilita la configuración de estas funciones con el uso de formatos ya conocidos por ellos, XML/HTML.

Puesto que dos de los valores de la empresa son la visión artística y la competitividad, se debe buscar un esquema para la inclusión de los controles que no limite la flexibilidad creativa de los diseños; se debe garantizar un

grado importante de originalidad en el producto finalizado para que le ofrezca al cliente una ventaja competitiva ante sus similares en el mercado.

La mayor ventaja que ofrecerá el desarrollo de esta herramienta consiste en la mejor segmentación de los sectores de trabajo, al evitar que los programadores de *software* tengan que intervenir en los proyectos del área *web*, mejorando el rendimiento del personal y los tiempos de respuesta.

Otra ventaja implícita radica en la disminución de los tiempos durante la fase de construcción de portales de colaboración, intranets y proyectos de desarrollo de software que vayan a ser codificados como aplicaciones web, pues la herramienta facilitará la implementación de módulos muy comunes en este tipo de programas, como son los de seguridad y acceso a los usuarios finales del sistema.

Los bloques de función principalmente requeridos son:

- **Acceso de usuarios y seguridad**: controles para inicio de sesión y administración de usuarios, con el fin de limitar el acceso a las secciones que así lo requieran y garantizar así la seguridad del sitio web.

- **Navegación**: generación dinámica de los menús de acceso, mapa del sitio.

- **Herramientas para búsqueda**: acceso rápido a contenidos específicos a través de la búsqueda de texto.

- **Contenido general**: requiere la visualización de título, fecha, autor, contenido con formato enriquecido, fotografías y despieces, útil para secciones de noticias, avisos y secciones de texto.

- **Galería multimedia**: visualización de fotografías, videos y audio, organizados en galerías. Opción para ampliar las fotografías y videos, permitir preferiblemente varios tipos de *layout* para ello.

- **Documentos y archivos**: listado y carga de archivos y documentos en formatos populares (DOC, PDF, XLS, entre otros).

- **Mailer**: control que servirá de base para el envío de correo electrónico (útil en secciones como contáctenos, sugerencias, soporte técnico, etc.)

Como punto importante a considerar, se encuentra la amplia utilidad que tendrá el uso de esta herramienta tanto para el proceso de diseño de websites para internet como para intranets corporativas.

Toda la herramienta debe ser desarrollada bajo los esquemas de Web 2.0, permitiéndose la generación de salidas XHTML, con el uso de XML como formato para compartir información entre aplicaciones y servicios, y aprovechamiento de las tecnologías AJAX para el mejoramiento de la experiencia de usuario.

1.2. Objetivos

1.2.1. Objetivo general

Desarrollar una herramienta de software que disminuya los tiempos de creación y administración de contenido dinámico en websites y aplicaciones web para una empresa de desarrollo de software.

1.2.2. Objetivos específicos

1. Definir los aspectos configurables para cada control que flexibilicen al máximo su adaptabilidad gráfica.

2. Realizar la arquitectura base para los componentes y el acceso a datos desde diferentes orígenes.

3. Diseñar la arquitectura final del software, estructura de cada control y de la base de datos relacional.

4. Codificar los componentes diseñados.

5. Realizar las pruebas tanto de unidad como de integración.

6. Realizar la integración y documentación de funcionamiento o instructivo para usuarios de todos los módulos que conforman el sistema.

CAPÍTULO 2
MARCO TEÓRICO

2.1. Antecedentes

Los antecedentes de este trabajo son proyectos que presentan visualización de gráficas de variables y utilización de tecnología Web.

- **Desarrollo de un software basado en aplicaciones web para el monitoreo de dispositivos de la plataforma de telecomunicaciones de PDVSA-GAS** (1)**.** Trabajo de Grado presentado como requisito parcial para optar al título de Ingeniero en Computación en febrero de 2007 por Juan C. Tenías en la Universidad de Oriente, Núcleo Anzoátegui. Se implementó un sistema que se propone monitorear los servidores y dispositivos de redes LAN y WAN de la AIT de PDVSA-GAS, permitiéndose generar reportes y vistas acerca de la salud y uso de los equipos, así como la planeación de mejoras a la red. El autor utilizó la metodología de Proceso Unificado y el lenguaje UML para el modelado de la estructura del software durante el desarrollo, y PHP como tecnología web.

- **Desarrollo de un software para la automatización de reportes y consultas de archivos históricos del tráfico de conexiones de red realizada por la superintendencia de seguridad lógica de una empresa petrolera utilizando tecnología web** (2). Trabajo de Grado presentado como requisito parcial para optar al título de Ingeniero en Computación en julio de 2007 por Pedro E. Salazar en la Universidad de Oriente, Núcleo Anzoátegui. Este sistema busca

disminuir el tiempo de respuesta y la obtención rápida de información útil para la toma de decisiones en el departamento de Protección Lógica de la Superintendencia de Seguridad Lógica PCP Oriente – PDVSA. Se utilizó la metodología de Proceso Unificado y el lenguaje UML para el modelado de la estructura del software, así como la plataforma Microsoft .NET para la implementación.

» **Desarrollo de un sistema de soporte y atención al cliente para una empresa desarrolladora de software** (3). Trabajo de Grado presentado como requisito parcial para optar al título de Ingeniero en Computación en diciembre de 2007 por Aivett Bilbao y Edgar Arriojas en la Universidad de Oriente, Núcleo Anzoátegui. El desarrollo busca mejorar el servicio a los clientes de la empresa Saetha Design & Systems y gestionar eficientemente sus solicitudes y requerimientos, así como llevar el seguimiento de atención de los detalles o fallas surgidas, con respecto a los sistemas implementados. Se utilizó la metodología de Proceso Unificado y UML para el modelado de la estructura del software, implementado bajo Microsoft .NET 2.0.

2.2. Programación Orientada a Objetos

La programación orientada a objetos es un conjunto completo de conceptos e ideas. Es una manera de pensar en el problema al que va dirigido un programa de ordenador y de enfrentarlo de modo más intuitivo e incluso más productivo. En un lenguaje orientado a objetos verdadero, toda entidad del domino del problema se expresa a través del concepto de objetos. Para el dominio del problema, los programas escritos en forma de simular los objetos del mundo real son mucho más fáciles de diseñar y escribir porque permiten pensar de un modo más natural.

Por definición, los objetos comprenden datos y métodos que trabajan con esos datos. Una clase es un diseño para un determinado conjunto de funcionalidad, y un objeto creado tomando como base una determinada clase tiene toda la funcionalidad de esa clase a partir de la que se ha construido. Un objeto es una instancia o ejemplar de una clase.(4)

2.2.1. Principios de la programación orientada a objetos

Según Bjarne Stroustrup, autor del lenguaje de programación C++, para que un lenguaje se llame a sí mismo orientado a objetos debe soportar tres conceptos: objetos, clases y herencia. Sin embargo, ha llegado a pensarse más comúnmente que los lenguajes orientados a objetos son lenguajes construidos sobre el trípode encapsulación, herencia y polimorfismo. La razón de este cambio de filosofía es que con el paso de los años, los desarrolladores de software han llegado a darse cuenta que la encapsulación

y el polimorfismo son partes tan integrantes de la construcción de sistemas orientados a objetos como la clase y la herencia.(4)

2.2.2. Encapsulación

Habilidad de un objeto para esconder sus datos y métodos internos y de presentar una interfaz que hace accesibles las partes importantes del objeto, hablando desde el punto de vista del programa. Los procedimientos internos sobres cómo lleva a cabo un objeto su trabajo no son importantes mientras que ese objeto pueda desempeñar su trabajo.(4)

2.2.3. Herencia

Está relacionada con la habilidad del programador para especificar que una clase tiene una relación “especie de” con otra clase. A través de la herencia, se puede crear (o derivar) una nueva clase que esté basada en una ya existente. Se puede modificar la clase de la manera que se quiera y crear objetos nuevos del tipo derivado. Esta habilidad es la esencia de la creación de la jerarquía de clases. Una clase derivada es la nueva que se está creando y la clase base es desde la que se deriva la nueva clase. La clase derivada hereda todos los miembros de la clase base, para así posibilitar la reutilización del trabajo anterior.(4)

2.2.4. Polimorfismo

Funcionalidad que permite al código antiguo invocar código nuevo. Este es probablemente el mayor beneficio de la programación orientada a objetos, porque permite extender o mejorar un sistema sin romper o modificar el código existente.(4)

El polimorfismo proporciona al menos dos beneficios. Primero, permite la capacidad de agrupar objetos que tienen una clase base en común y tratarlos consistentemente. La segunda ventaja es la que ya se ha mencionado: el código antiguo puede utilizar código nuevo.

2.3. Ingeniería de software

La ingeniería del software es el establecimiento y uso de principios de la ingeniería para obtener económicamente un software confiable y que funcione de modo eficiente en máquinas reales. Está estratificada en cuatro etapas claves:

- **Un enfoque de calidad:** cualquier enfoque de la ingeniería (incluido el de la ingeniería del software) debe estar sustentado en un compromiso de calidad. Como por ejemplo: *La Gestión de Calidad Total*, *Sigma Seis* y otros enfoques similares que fomentan una cultura de mejora continua del proceso. Esta cultura es la que al final conduce al desarrollo de enfoques muy efectivos para la ingeniería del software.

- **El proceso:** el proceso del software forma la base para el control de la gestión de los proyectos de software y establece el contexto en el cual se aplican los métodos técnicos, se generan los productos del trabajo (modelos, documentos, datos, reportes, formatos, etcétera), se establecen los fundamentos, se asegura la calidad, y el cambio se maneja de manera apropiada.

- **Los métodos:** los métodos abarcan un amplio espectro de tareas que incluyen la comunicación, el análisis de requisitos, el modelado del diseño, la construcción del programa, la realización de pruebas y el soporte. Los métodos de la ingeniería del software se basan en un conjunto de principios básicos que gobiernan cada área de la

tecnología e incluye actividades de modelado y otras técnicas descriptivas.

» **Las herramientas:** las herramientas de la ingeniería en software proporcionan el soporte automatizado o semi-automatizado para el proceso y los métodos. Cuando las herramientas se integran de forma que la información que cree una de ellas pueda usarla otra, se dice que se ha establecido un sistema para el soporte del desarrollo del software.(5)

2.3.1. Proceso Unificado de Desarrollo de Software

El proceso unificado es un intento encaminado a reunir los mejores rasgos y características de modelos de procesos de software, pero los caracteriza de manera que implementa muchos de los principios del desarrollo ágil de software. Reconoce la importancia de la comunicación con el cliente y los métodos encaminados a describir el punto de vista de un cliente con respecto a un sistema (por ejemplo, el caso de uso).

El proceso unificado enfatiza el importante papel de la arquitectura del software, y ayuda al arquitecto a enfocarse en las metas correctas, como el entendimiento, el ajuste de los cambios futuros y la reutilización. Sugiere un flujo de proceso iterativo e incremental y que proporciona el sentido evolutivo esencial en el desarrollo del software moderno.(5)

2.3.1.1. Fases del Proceso Unificado

» **Fase de inicio:** la fase de inicio abarca la comunicación con el cliente y las actividades de planeación. Al colaborar con los clientes y

usuarios finales se identifican los requisitos de negocios para el software, se propone una arquitectura aproximada para el sistema y se desarrolla un plan para la naturaleza iterativa e incremental del sistema subsiguiente. Los requisitos fundamentales de negocios se describen a través de un conjunto preliminar de casos de uso que describen cuáles características y funciones son deseables para cada clase importante de usuarios.

» **Fase de elaboración:** abarca la comunicación con el cliente y las actividades de modelado del modelo genérico del proceso. La elaboración refina y expande los casos de usos preliminares que se desarrollaron como una parte de la fase de inicio; además, expande la representación arquitectónica para incluir cinco visiones diferentes del software: el modelo de caso de uso, el modelo de análisis, el modelo de diseño, el modelo de implementación y el modelo de despliegue.

» **Fase de construcción:** desarrolla o adquiere los componentes del software que harán que cada caso de uso sea operativo para los usuarios finales. Lograr esto requiere que los modelos de análisis y diseño iniciados durante la fase de elaboración se completen para reflejar la versión final del incremento del software.

» **Fase de transición:** el software se entrega a los usuarios finales para realizar pruebas beta, y la retroalimentación del usuario reporta tanto defectos como cambios necesarios. Además el equipo de software crea la información de soporte necesaria (por ejemplo, manuales de usuario, guías de resolución de problemas, procedimientos de instalación) para el lanzamiento.

» **Fase de Producción:** durante esta fase se monitorea el uso subsiguiente del software, se proporciona el soporte para el ambiente operativo (infraestructura), y se reciben y evalúan los informes de defectos y los requerimientos de cambios.

Es probable que mientras se realizan las fases de construcción, transición, y producción ya se hayan iniciado los trabajos para el siguiente incremento del software. Esto significa que las cinco fases del proceso unificado no suceden en una secuencia, sino en una concurrencia por etapas.(5)

2.3.2. Lenguaje Unificado de Construcción de Modelos (UML)

Es un sistema notacional (que, entre otras cosas, incluye el significado de sus notaciones) destinado a los sistemas de modelado que utilizan conceptos orientados a objetos. Se define como un lenguaje que permite especificar, visualizar y construir los artefactos de los sistemas de software.

La puerta más novedosa en el mundo del diseño de aplicaciones de software es el UML 2.0: la evolución de la programación hacia la ejecución y validación automática de modelos.(6)

2.3.2.1. Objetivos del UML 2.0

Al momento de desarrollar el nuevo estándar 2.0 del UML, la OMG se propuso, entre otros, dos objetivos que podrían considerarse principales debido a la influencia de éstos en la versión final del estándar. Estos objetivos son:

1. Hacer el lenguaje de modelado mucho más extensible de lo que era.
2. Permitir la validación y ejecución de modelos creados mediante el UML.

UML 2.0 se desarrolla sobre la base de estos dos objetivos, causando un quiebre respecto a versiones anteriores.

2.3.2.1.1. El UML y la Industria del Software

El UML se ha vuelto el estándar de facto (impuesto por la industria y los usuarios) para el modelado de aplicaciones de software. En los últimos años, su popularidad trascendió al desarrollo de software y, en la actualidad, el UML es utilizado para modelar muchos otros dominios, como por ejemplo el modelado de procesos de negocios.

2.3.2.1.2. Conceptos básicos sobre UML

UML son las siglas para *Unified Modeling Language*, que en castellano quiere decir: Lenguaje de Modelado Unificado.

- **Lenguaje:** el UML es, precisamente, un lenguaje. Lo que implica que éste cuenta con una sintaxis y una semántica. Por lo tanto, al modelar un concepto en UML, existen reglas sobre cómo deben agruparse los elementos del lenguaje y el significado de esta agrupación.

- **Modelado:** el UML es visual. Mediante su sintaxis se modelan distintos aspectos del mundo real, que permiten una mejor interpretación y entendimiento de éste.

» **Unificado:** unifica varias técnicas de modelado en una única.

2.3.2.1.3. OMG

La OMG (Object *Management Group*) es una asociación sin fines de lucro formada por grandes corporaciones, muchas de ellas de la industria del software, como por ejemplo: IBM, Apple Computer, Sun Microsystems Inc y Hewlett-Packard. Esta asociación se encarga de la definición y mantenimiento de estándares para aplicaciones de la industria de la computación. Otro de los estándares definidos por la OMG, además del UML, es CORBA, el cual permite interoperabilidad multiplataforma a nivel de objetos de negocio.

2.3.2.1.4. El Nuevo Enfoque del UML 2.0

En las versiones previas del UML, se hacía un fuerte hincapié en que UML no era un lenguaje de programación. Un modelo creado mediante UML no podía ejecutarse. En el UML 2.0, esta asunción cambió de manera drástica y se modificó el lenguaje, de manera tal que permitiera capturar mucho más comportamiento (*behavior*). De esta forma, se permitió la creación de herramientas que soporten la automatización y generación de código ejecutable, a partir de modelos UML.

2.3.2.1.5. Estándares que conforman el UML

» **Superestructura:** es la especificación donde se encuentran todos los diagramas que la mayoría de los desarrolladores conocen.

- **Infraestructura:** conceptos de bajo nivel. Meta-modelo que da soporte a la superestructura, entre otras.

- **OCL:** lenguaje de restricción. De utilidad para especificar conceptos ambiguos sobre los distintos elementos del diagrama.

- **XMI / Intercambio de diagramas:** permite compartir diagramas entre diferentes herramientas de modelado UML.

2.3.2.1.6. Superestructura

La superestructura del UML es la definición formal de los elementos del UML. Esta definición sola contiene más de 640 páginas. La superestructura es típicamente utilizada por los desarrolladores de aplicación. Es aquella sobre la que hablan los libros y la que la mayoría conoce de versiones anteriores del UML.

Es aquí donde se definen los diagramas y los elementos que los componen. La Superestructura se encuentra dividida en niveles. Estos niveles se conocen como:

- **Básico (L1):** contiene los elementos básicos del UML 2.0, entre ellos: diagramas de clases, de actividades, de interacciones y de casos de uso.

- **Intermedio (L2):** contiene los siguientes diagramas: diagramas de estado, perfiles, diagramas de componentes y de despliegue.

» **Completo (L3):** representa la especificación del UML 2.0 completa, como por ejemplo: las acciones, características avanzadas y *PowerTypes*, entre otros.

Es importante destacar que basta con que una herramienta implemente el nivel de conformidad básico (L1), para que se considere UML 2.0 compatible. Por eso, es normal ver una disparidad de características (*features*) bastante amplia entre dos herramientas distintas, aunque éstas sean UML 2.0 compatibles.

2.3.2.1.7. Organización de la Superestructura

El bloque de construcción básico del UML es un diagrama. La estructura de los diagramas UML está reflejado por el diagrama de la *Figura 2.1*, de acuerdo con la especificación del UML 2.0 del Object Management Group. Los detalles sobre estos diagramas específicos se organizan de acuerdo a esta estructura taxonómica, que da la perspectiva a los diagramas y a sus interrelaciones. Los diagramas de interacción comparten propiedades y atributos similares, como lo hacen los diagramas estructurales y de comportamiento.

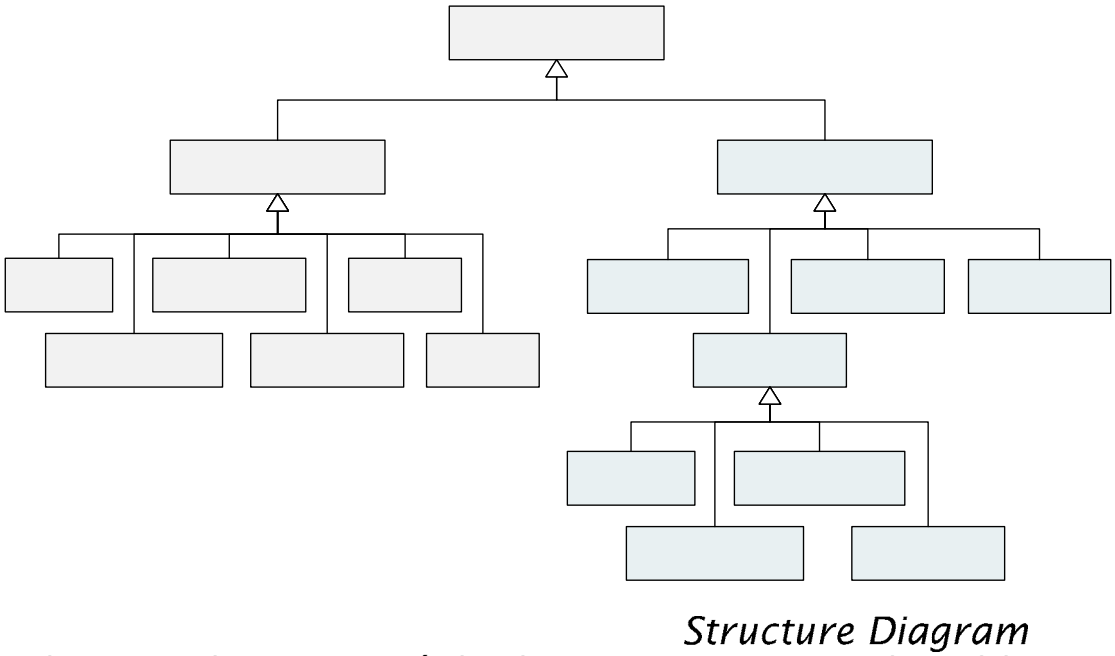


Figura 2.1. Diagrama taxonómico de estructura y comportamiento del UML 2.0

2.3.2.1.8. Diagramas de Estructura y Diagramas de Comportamiento

Los diagramas estructurales representan elementos y así componen un sistema o una función. Estos diagramas pueden reflejar las relaciones estáticas de una estructura, como lo hacen los diagramas de clases o de paquetes, o arquitecturas en tiempo de ejecución, tales como diagramas de objetos o de estructura de composición. Los diagramas de comportamiento representan las características de comportamiento de un sistema o proceso de negocios y, a su vez, incluyen a los diagramas de: actividades, casos de uso, máquinas de estados, tiempos, secuencias, repaso de interacciones y comunicaciones.

2.3.2.1.9. Breve descripción sobre los diagramas

En la *tabla 2.1*, se muestra la importancia que tiene, para un desarrollador, conocer cada una de las nuevas características del UML 2.0.(7)

Tabla 2.1. Diagramas del UML 2.0 (parte 1/2)

DIAGRAMA	DESCRIPCIÓN	PRIORIDAD
Diagrama de Clases	Muestra una colección de elementos de modelado declarativo (estáticos), tales como clases, tipos y sus contenidos y relaciones.	*Alta*
Diagrama de Componentes	Representa los componentes que componen una aplicación, sistema o empresa. Los componentes, sus relaciones, interacciones y sus interfaces públicas.	*Media*
Diagrama de Estructura de Composición	Representa la estructura interna de un clasificador (tal como una clase, un componente o un caso de uso), incluyendo los puntos de interacción de clasificador con otras partes del sistema.	*Baja*
Diagrama de Despliegue Físico	Un diagrama de despliegue físico muestra cómo y dónde se desplegará el sistema. Las máquinas físicas y los procesadores se representan como nodos y la construcción interna puede ser representada por nodos o artefactos embebidos. Como los artefactos se ubican en los nodos para modelar el despliegue del sistema, la ubicación es guiada por el uso de las especificaciones de despliegue.	*Media*
Diagrama de Objetos	Un diagrama que presenta los objetos y sus relaciones en un punto del tiempo. Un diagrama de objetos se puede considerar como un caso especial de un diagrama de clases o un diagrama de comunicaciones.	*Baja*
Diagrama de Paquetes	Un diagrama que presenta cómo se organizan los elementos de modelado en paquetes y las dependencias entre ellos, incluyendo importaciones y extensiones de paquetes.	*Baja*

Tabla 2.2. Diagramas del UML 2.0 (parte 2/2)

DIAGRAMA	DESCRIPCIÓN	PRIORIDAD
Diagrama de Actividades	Representa los procesos de negocios de alto nivel, incluidos el flujo de datos. También puede utilizarse para modelar lógica compleja y/o paralela dentro de un sistema.	*Alta*
Diagrama de Comunicaciones *(anteriormente: Diagrama de colaboraciones)*	Es un diagrama que enfoca la interacción entre líneas de vida, donde es central la arquitectura de la estructura interna y cómo ella se corresponde con el pasaje de mensajes. La secuencia de los mensajes se da a través de un esquema de numerado de la secuencia.	*Baja*
Diagrama de Revisión de la Interacción	Los Diagramas de Revisión de la Interacción enfocan la revisión del flujo de control, donde los nodos son Interacciones u Ocurrencias de Interacciones. Las Líneas de Vida los Mensajes no aparecen en este nivel de revisión	*Baja*
Diagrama de Secuencias	Un diagrama que representa una interacción, poniendo el foco en la secuencia de los mensajes que se intercambian, junto con sus correspondientes ocurrencias de eventos en las Líneas de Vida.	*Alta*
Diagrama de Máquinas de Estado	Un diagrama de Máquina de Estados ilustra cómo un elemento, muchas veces una clase, se puede mover entre estados que clasifican su comportamiento, de acuerdo con disparadores de transiciones, guardias de restricciones y otros aspectos de los diagramas de Máquinas de Estados, que representan y explican el movimiento y el comportamiento.	*Media*

DIAGRAMA	DESCRIPCIÓN	PRIORIDAD
Diagrama de Tiempos	El propósito primario del diagrama de tiempos es mostrar los cambios en el estado o la condición de una línea de vida (representando una Instancia de un Clasificador o un Rol de un clasificador) a lo largo del tiempo lineal. El uso más común es mostrar el cambio de estado de un objeto a lo largo del tiempo, en respuesta a los eventos o estímulos aceptados. Los eventos que se reciben se anotan, a medida que muestran cuándo se desea mostrar el evento que causa el cambio en la condición o en el estado.	*Baja*
Diagrama de Casos de Uso	Un diagrama que muestra las relaciones entre los actores y el sujeto (sistema), y los casos de uso.	*Media*

2.3.2.1.10. Poniendo UML a trabajar

Un modelo de UML proporciona a menudo apenas una vista de un sistema entre muchas vistas necesitadas para construir o para documentar realmente el sistema completo. Los usuarios nuevos a UML pueden caer en la trampa de intentar modelar todo sobre su sistema con un solo diagrama y terminar cayendo en la pérdida de la información crítica. O, en el otro extremo, pueden intentar incorporar cada diagrama posible de UML en su modelo, de tal modo que complican demasiado y crean una pesadilla del mantenimiento.

El llegar a ser perito con UML significa entender lo que tiene que ofrecer cada diagrama y saber cuándo aplicarlo. Habrá muchas veces en que un concepto se podría expresar usando cualquier número de diagramas; hay que escoger el diagrama o los diagramas que serán más significativos para la mayoría de los usuarios.(8)

2.3.2.1.11. Reglas para el UML

Mientras que UML proporciona un lenguaje común para la captura de funcionalidad e información del diseño, es deliberadamente ampliable permitir la flexibilidad necesaria para modelar diversos dominios. Hay varias reglas a tener presente al usar UML:

- **Casi todo en UML es opcional:** UML proporciona un lenguaje para la captura de información que varía grandemente dependiendo del dominio del problema. En hacer eso, hay a menudo partes de UML que no aplican a un problema particular o puedan no prestar cualquier cosa a la visión particular que se esté intentando representar. Es importante tener presente que no se necesita utilizar cada parte de UML en cada modelo que se crea. Posiblemente más importantemente, no se necesita utilizar cada símbolo disponible para un tipo de diagrama en cada diagrama que se cree. Se debe mostrar solamente lo que ayuda a clarificar el mensaje que se está intentando representar, y dejar lo que no se necesita. Hay ocasionalmente más que una forma para representar la misma información; hay que usar cuál es familiar a la audiencia.

- **Los modelos de UML son raramente completos:** como consecuencia de que todo es opcional, es común que un modelo de UML falten algunos detalles sobre un sistema. El truco es no perder los detalles claves que podrían afectar el diseño del sistema. Saber cuál es un detalle clave contra la información extraña viene con la experiencia; sin embargo, usar un proceso iterativo y la nueva revisión del modelo ayuda a dejar sólo lo que necesita estar allí.(8)

2.3.2.1.12. Infraestructura

En la infraestructura de UML se definen los conceptos centrales y de más bajo nivel. La infraestructura es un *meta-modelo* (un modelo de modelos), mediante el cual, se modela el resto de UML. No es empleada por usuarios finales de UML, pero ofrece la piedra fundamental sobre la cual se define la superestructura, que es la utilizada por el común de los usuarios. La infraestructura brinda, también, varios mecanismos de extensión que hacen de UML un lenguaje configurable.

2.3.2.1.13. OCL

Es la sigla en inglés de *Object Constraint Language*, o lenguaje de restricciones de objetos. Define un lenguaje simple para escribir restricciones y expresiones sobre elementos de un modelo. Suele ser útil cuando se está especificando un dominio particular mediante UML y es necesario restringir los valores permitidos para los objetos del dominio. OCL brinda la posibilidad de definir invariantes, pre-condiciones, post-condiciones y restricciones en los elementos de un diagrama. Fue incorporado a UML en la versión 1.1. Originalmente, fue especificado por IBM y es un ejemplo más de las muchas herramientas agregadas a UML.

2.3.2.1.14. XMI / Intercambio de diagramas

La especificación para el intercambio de diagramas fue escrita para poder compartir modelos realizados mediante UML entre diferentes herramientas de modelado.

En versiones anteriores de UML se utilizaba un *Schema XML* para capturar los elementos utilizados en el diagrama; pero este *schema* no decía nada acerca de cómo debía graficarse el modelo. Para solucionar este problema, la nueva especificación para el intercambio de diagramas fue desarrollada utilizando un nuevo *Schema XML*, que permite construir una representación SVG (*Scalable Vector Graphics*).

Esta especificación se denomina XMI (*XML Metadata Interchange*) o XML de intercambio de *metadata* (datos que representan datos). Típicamente, es utilizada sólo por quienes desarrollan herramientas de modelado UML.

2.3.2.2. Diagramas de casos de uso y de actividades

Explicar cómo se usan los diagramas es importante porque a través de esto se garantiza un claro entendimiento de lo que se está diagramando.

2.3.2.2.1. Descripción del diagrama de casos de uso

Debido a que el formato de descripción de los casos de uso es muy variante, se ha escogido el siguiente:

» **Actor principal.** Nombre para el rol del actor principal.

» **Alcance.** Puede ser alguno de los siguientes:

› *Alcance de empresa:* significa que se está discutiendo el comportamiento de la organización entera o la empresa en el cumplimiento de la meta del actor principal. Se etiqueta el campo del alcance del caso de uso con el nombre de la organización. Si se discute un departamento, se utiliza el nombre de departamento.

› *Alcance de sistema:* significa apenas la porción de software/hardware que se considera. Fuera están todas los piezas de hardware, software y de recursos humanos del sistema que interactúan con dicha porción.

› *Alcance de subsistema:* cuando se ha abierto el sistema principal y se está a punto de hablar de cómo una parte de él trabaja.

» **Nivel.** Puede ser uno de los siguientes:

› *Resumen:* contienen otros casos de uso de más bajo nivel.

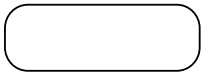
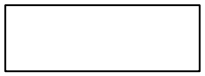
› *Meta de usuario:* la meta de usuario es la meta de mayor interés. Es la meta que el actor principal tiene al intentar completar una acción, o tiene al usar el sistema. Corresponde al "proceso elemental del negocio" en la literatura de la ingeniería de proceso del negocio.

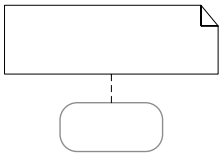
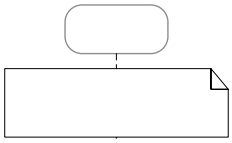
› *Sub-función:* comprende aquello requerido para realizar metas de usuario. Son necesarios en ocasiones para la legibilidad, o porque muchas otras metas los utilizan.(9)

2.3.2.2.2. Descripción del diagrama de actividades

En la *tabla 2.3* se presentan los elementos usados en los diagramas de actividades y sus descripciones.(10)

Tabla 2.3. Elementos del diagrama de actividades (parte 1/2)

ELEMENTO	NOTACIÓN O EJEMPLO	DESCRIPCIÓN
Acción		Asignación primitiva ejecutable o una computación que recibe un conjunto de valores de entrada y produce un cambio de estado y/o el retorno de valores de salida. Un cambio de estado de un objeto está reflejado por cambios para los valores de uno o más de sus atributos.
Objeto		Nodo que provee y acepta objetos y datos como fluyan en y por los comportamientos invocados en el contexto de la ejecución de una actividad.

ELEMENTO	NOTACIÓN O EJEMPLO	DESCRIPCIÓN
Pre-condición local		Condición que debe mantenerse verdadera cuando la ejecución de una acción empieza.
Post-condición local		Condición que debe mantenerse verdadera cuando la ejecución de una acción finaliza.
Pin		Elemento de modelado para una entrada o salida de una acción. Los valores de entrada para una acción vienen de un pin de entrada; y los valores de salida de una acción van a un pin de salida.
Nodo inicial		Está donde el flujo de control comienza cuando una actividad es invocada.
Nodo final		Termina todos los flujos dentro de la actividad y así termina la actividad misma.
Nodo de flujo final		Termina un flujo particular.

«localPrecondition»
Restricción

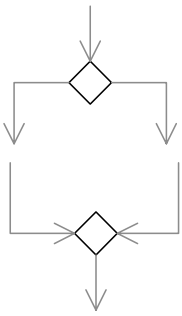
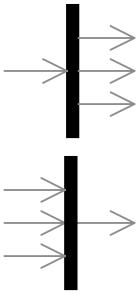
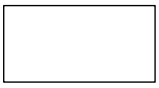
Nombre

Nombre

«localPostcondition»
Restricción

Parámetro 1 Nombre Parámetro 2

Tabla 2.4. Elementos del diagrama de actividades (parte 2/2)

ELEMENTO	NOTACIÓN O EJEMPLO	DESCRIPCIÓN
Nodo de decisión y nodo de asociación		Un nodo de decisión ofrece una elección entre dos o más bordes salientes de actividad, donde cada uno tiene una expresión booleana que debe resolver verdadero antes de que el camino asociado pueda ser tomado. Un nodo de asociación junta cursos alternos múltiples de control.
Nodo de ramificación y nodo de unión		Un nodo de ramificación desdobla un flujo en flujos concurrentes múltiples. Un nodo de unión sincroniza múltiples flujos de control.
Nodo de almacenamiento de datos		Un nodo *DataStore* representa un nodo para almacenamiento de información no-transitoria.
Llamado a una actividad		La llamada a una actividad es indicada poniendo un símbolo de tridente. El tridente se asemeja a una jerarquía miniatura, indicando que esta invocación comienza otra actividad que represente otra descomposición.

2.4. Bases de datos

Una base de datos es una colección de datos relacionados. Por *datos*, se quiere decir hechos conocidos que pueden registrarse y que tienen un significado implícito. Una base de datos tiene las siguientes propiedades implícitas:

- Una base de datos representa algunos aspectos del mundo real, en ocasiones denominado *mini-mundo* o Universo del Discurso (*UdD*). Los cambios en el *mini-mundo* se reflejan en la base de datos.

- Una base de datos es una colección coherente de datos con significados inherentes. Un conjunto aleatorio de datos no puede considerarse como una base de datos.

- Una base de datos se diseña, construye y puebla con datos para un propósito específico. Está destinada a un grupo de usuarios concreto y tiene algunas aplicaciones preconcebidas en las cuales están interesados dichos usuarios.(11)

2.4.1. Componentes principales de las bases de datos

- **Datos:** los datos son la base de datos propiamente dicha.

- **Hardware:** el hardware se refiere a los dispositivos de almacenamiento en donde reside la base de datos, así como a los

dispositivos periféricos (unidad de control, canales de comunicación, etc.) necesarios para su uso.

» **Software:** está constituido por un conjunto de programas que se conoce como Sistema Manejador de Base de Datos (*DMBS: Data Base Management System*). Este sistema maneja todas las solicitudes formuladas por los usuarios a la base de datos.

» **Usuarios:** existen tres clases de usuarios relacionados con una base de datos: El programador de aplicaciones, quien crea programas de aplicación que utilizan la base de datos. El usuario final, quien accede la base de datos por medio de un lenguaje de consulta o de programas de aplicación. El administrador de la base de datos (*DBA: Data Base Administrator*), quien se encarga del control general del sistema de base de datos.(11)

2.4.2. Clasificación de las bases de datos

Las bases de datos pueden clasificarse de varias maneras, de acuerdo al criterio elegido para su clasificación:

» Según la **variabilidad de los datos almacenados** existen bases de datos estáticas y bases de datos dinámicas.

» Según su **contenido** pueden ser bases de datos bibliográficas, bases de datos numéricas, bases de datos de texto completo, directorios, banco de imágenes (audio, vídeo, multimedia, etc.) y, por último, las bases de datos o "bibliotecas".

Además de la clasificación por la función de las bases de datos, éstas también se pueden clasificar de acuerdo a su modelo de datos:

- **Bases de datos jerárquicas:** estas son bases de datos que, como su nombre indica, almacenan su información en una estructura jerárquica. En este modelo los datos se organizan en una forma similar a un árbol (visto al revés), en donde un nodo padre de información puede tener varios hijos. El nodo que no tiene padres es llamado raíz, y a los nodos que no tienen hijos se los conoce como hojas.

- **Bases de datos de red:** éste es un modelo ligeramente distinto del jerárquico; su diferencia fundamental es la modificación del concepto de nodo: se permite que un mismo nodo tenga varios padres (posibilidad no permitida en el modelo jerárquico).

- **Bases de datos relacionales:** su artículo principal es el modelo relacional debido a que es el modelo más utilizado en la actualidad para modelar problemas reales y administrar datos dinámicamente. Estas relaciones podrían considerarse en forma lógica como conjuntos de datos llamados *tuplas*.

- **Bases de datos orientadas a objetos:** este modelo, bastante reciente, y propio de los modelos informáticos orientados a objetos, trata de almacenar en la base de datos los objetos completos (estado y comportamiento).

- **Bases de datos documentales:** permiten la indexación a texto completo, y en líneas generales realizar búsquedas más potentes.(11)

2.4.3. Modelo de datos

Una característica fundamental del enfoque de base datos es que proporciona cierto nivel de abstracción de los datos al ocultar detalles de almacenamiento que la mayoría de los usuarios no necesitan conocer. Un modelo de datos proporciona los medios necesarios para conseguir dicha abstracción. Con el concepto *estructura de una base de datos* se refiere a los tipos de datos, los vínculos y las restricciones que deben cumplirse para esos datos. La mayoría de los modelos de datos contienen un conjunto de operaciones básicas para especificar lecturas y actualizaciones de la base de datos.(11)

2.4.4. Sistema de Gestión de Base de Datos (SGBD)

Es una colección de programas que permiten a los usuarios crear y mantener una base de datos. El SGBD es por tanto un sistema software de propósito general que facilita los procesos de definición, construcción, y manipulación de base de datos para distintas aplicaciones. A continuación se describen cada uno de estos procesos:

- **Definición:** la definición de una base de datos consiste en especificar los tipos de de datos, las estructuras y restricciones para los datos que se van a almacenar en dicha base de datos.

- **La construcción:** es el proceso de almacenar los datos concretos sobre algún medio de almacenamiento controlado por el SGBD.

- **La manipulación:** incluye funciones tales como consultar la base de datos para recuperar datos específicos, actualizar la base de datos para reflejar los cambios ocurridos en el mini-mundo y generar informes a partir de los datos.(11)

2.4.4.1. Arquitectura de un SGBD

Una arquitectura para los sistemas de base de datos es la denominada arquitectura de tres esquemas. El objetivo de esta es separar las aplicaciones del usuario y la base de datos física. En esta arquitectura se definen esquemas en los tres siguientes niveles:

- El **nivel interno** tiene un esquema interno, que describe la estructura física de almacenamiento de la base de datos. El esquema interno emplea un modelo de datos físico y describe todos los detalles para su almacenamiento y acceso.

- El **nivel conceptual** tiene un esquema conceptual, que describe la estructura de la base de datos completa para una comunidad de usuarios. El esquema conceptual oculta los detalles de las estructuras físicas de almacenamiento y se concentra en describir entidades, tipos de datos, vínculos, operaciones de los usuarios y restricciones. En este nivel podemos usar un modelo de datos de alto nivel o uno de implementación.

- El **nivel externo** incluye varios esquemas externos o vistas de usuario. Cada esquema externo describe la parte de la base de datos que interesa a un grupo de usuarios determinado, y oculta a ese grupo el resto de la base de datos.(11)

2.4.4.2. Clasificación de los Sistemas de Gestión de Bases de Datos

El principal criterio que suele utilizarse para clasificar los SGBD es el modelo de datos en que se basan. Los modelos de datos empleados con mayor frecuencia en los SGBD comerciales actuales son el relacional, el de red y el jerárquico. A continuación se describen brevemente cada uno de estos:

- **El modelo de datos relacional:** representa una base de datos como una colección de tablas, cada una de las cuales se pueden almacenar en forma de archivo individual. Casi todas las bases de datos relacionales tienen lenguaje de consulta de alto nivel y manejan una forma limitada de vistas de usuarios.

- **El modelo de datos de red:** representan los datos como tipos de registros y también representa un tipo limitado de vínculos 1:N, llamado tipo de conjunto. Donde los tipos de registros aparecen como rectángulos y los tipos de conjuntos como flechas dirigidas rotuladas. Este modelo también tiene un lenguaje de registro por registro asociado que se debe incorporar en un lenguaje de programación anfitrión.

- **El modelo jerárquico:** representan los datos como estructura jerárquica árbol. Cada jerarquía representa varios registros relacionados entre sí. No existe un lenguaje estándar para el modelo

jerárquico, aunque la mayor parte de los SGBD jerárquicos cuentan de registro por registro.(11)

2.5. Tecnologías web

2.5.1. Internet

Término utilizado para referirse al agrupamiento más grande del mundo de redes interconectadas por *routers* y otros dispositivos que funciona (en general) como una sola red; conecta decenas de miles de redes de todo el mundo y con una cultura que se concentra en la investigación y estandarización basada en el uso real. Muchas tecnologías de avanzada provienen de la comunidad de la Internet.(12)

2.5.2. Lenguaje de etiquetas por hipertexto (HTML)

Formato simple de documentos en hipertexto que usa etiquetas para indicar cómo una aplicación de visualización, como por ejemplo un navegador de la Web, debe interpretar una parte determinada de un documento.(12)

2.5.3. servidor web

Un servidor web es un sistema informático conectado a una red, donde se almacenan las páginas, imágenes, etc. (que forman una aplicación web) disponibles para ser visitadas por los usuarios de la red.(13)

2.5.4. Protocolo de transferencia de hipertexto (http)

Uno de los protocolos más importantes de Internet. HTTP define como los navegadores y los servidores Web se comunican uno con otro. Está basado en texto y es transmitido sobre conexiones TCP.(13)

2.5.5. Aplicación web

Una aplicación web es un conjunto de páginas HTML que se transmiten por medio del protocolo HTTP de un servidor al cliente y viceversa, brindando distintas funcionalidades a un usuario final.(12)

2.6. Plataforma de desarrollo Microsoft .NET

Es una plataforma de desarrollo y ejecución de aplicaciones. Esto quiere decir que no sólo brinda todas las herramientas y servicios que se necesitan para desarrollar modernas aplicaciones empresariales y de misión crítica, sino que también provee de mecanismos robustos, seguros y eficientes para asegurar que la ejecución de las mismas sea óptima. Los componentes principales de la plataforma .NET son:

- Un entorno de ejecución de aplicaciones, también llamado *runtime*, que es un componente de software cuya función es la de ejecutar las aplicaciones .NET e interactuar con el sistema operativo ofreciendo sus servicios y recursos.

- Un conjunto de bibliotecas de funcionalidades y controles reutilizables, con una enorme cantidad de componentes ya programados y listos para ser consumidos por otras aplicaciones.

» Un conjunto de lenguajes de programación de alto nivel, junto con sus compiladores y *linkers*, que permitirán el desarrollo de aplicaciones sobre la plataforma .NET.

» Un conjunto de utilitarios y herramientas de desarrollo para simplificar las tareas más comunes del proceso de desarrollo de aplicaciones.

» Documentación y guías de arquitectura, que describen las mejores prácticas de diseño, organización, desarrollo, prueba e instalación de aplicaciones .NET.(14)

2.6.1. Principales características de la plataforma .NET

» Se dice que es una plataforma de **ejecución intermedia**, ya que las aplicaciones .NET no son ejecutadas directamente por el sistema operativo, como ocurre en el modelo tradicional de desarrollo. En su lugar, las aplicaciones .NET están diseñadas para ser ejecutadas contra un componente de software llamado Entorno de Ejecución (muchas veces también conocido como *runtime* o *máquina virtual*). Este componente es el encargado de manejar el ciclo de vida de cualquier aplicación .NET, iniciándola, deteniéndola, interactuando con el sistema operativo y proveyéndole servicios y recursos en tiempo de ejecución.

» Está completamente basada en el paradigma de **orientación a objetos**.

» Es **multi-lenguaje**: esto quiere decir que para poder codificar aplicaciones sobre esta plataforma no es necesario aprender un

único lenguaje específico de programación de alto nivel, sino que se puede elegir de una amplia lista de opciones: Microsoft Visual Basic .NET, Microsoft Visual C# .NET, Microsoft Visual J#.NET, Microsoft Visual C++.NET. Una aplicación escrita, por ejemplo, en Visual Basic .NET, puede incorporar sin problemas partes escritas en C# o C++ .NET.

- Es una plataforma que permite el desarrollo de **aplicaciones empresariales de misión crítica**, entendiéndose por esto que permite la creación y ejecución de aplicaciones de porte corporativo que sean críticas para la operación de tipos variados de organizaciones. Si bien también es muy atrayente para desarrolladores no profesionales, estudiantes y entusiastas, su verdadero poder radica en su capacidad para soportar las aplicaciones más grandes y complejas.

- Fue diseñado de manera tal de poder proveer un **único modelo de programación**, uniforme y consistente, para todo tipo de aplicaciones (ya sean de formularios Windows, de consola, aplicaciones Web, aplicaciones móviles, etc.) y para cualquier dispositivo de hardware (PC's, Pocket PC's, teléfonos celulares inteligentes, también llamados *SmartPhones*, Tablet PC's, etc.). Esto representa un gran cambio con respecto a las plataformas anteriores a .NET, las cuales tenían modelos de programación, bibliotecas, lenguajes y herramientas distintas según el tipo de aplicación y el dispositivo de hardware.

- Uno de los objetivos de diseño de .NET fue que tenga la posibilidad de interactuar e integrarse fácilmente con **aplicaciones desarrolladas**

en plataformas anteriores, particularmente en COM, ya que aún hoy existen una gran cantidad de aplicaciones desarrolladas sobre esa base.

» No sólo se integra fácilmente con aplicaciones desarrolladas en otras plataformas Microsoft, sino también con aquellas desarrolladas en **otras plataformas** de software, sistemas operativos o lenguajes de programación. Para esto hace un uso extensivo de numerosos estándares globales que son de uso extensivo en la industria. Algunos ejemplos de estos estándares son XML, HTTP, SOAP, WSDL y UDDI.(14)

2.7. Aplicaciones web: ASP.NET

ASP.NET es un “marco” (*framework*) para programar aplicaciones web, de un modo similar al que se programan las aplicaciones *windows*. El componente principal son los *WebForms* (formularios web) que permiten, entre otras cosas, separar la interfaz del usuario de la funcionalidad de la aplicación.(13)

2.7.1. Características de ASP.NET

» ASP.NET es el framework de programación web dentro de .NET.

» Permite desarrollar aplicaciones web con un modelo similar al utilizado para aplicaciones Windows.

» El componente fundamental de ASP.NET es el *WebForm*.

» Independencia del cliente (navegador, sistema operativo, dispositivo físico, etc.).

» Permite utilizar cualquier lenguaje .NET.

» Permite desarrollar Servicios Web XML.(13)

2.7.2. Ventajas de ASP.NET

» La “parte ejecutable” de una aplicación ASP.NET es compilada.

» Implementación y actualización de las aplicaciones sin reiniciar el servidor.

» Acceso a toda la *.NET Class Library*.

» Independiente del lenguaje de programación.

» Encapsulamiento de funcionalidad a través de controles de servidor y controles de usuario.

» Permite usar ADO.NET para acceso a datos.

» Soporta XML, hojas de estilo CSS, etc.

» Detección automática del navegador cliente, generando el lenguaje de marcas soportado por el mismo.

» Mecanismo de *caching* incorporado para páginas completas o partes de la misma frecuentemente solicitadas.(13)

2.7.3. Componentes de una aplicación ASP.NET

- *WebForms* (formularios web): uno o más archivos con extensión .aspx.

- Archivos *Code-Behind*: archivos asociados a formularios web que contienen código del lado del servidor (Ej. VB.NET, C#, etc.).

- Archivos de configuración con formato XML: un archivo de nombre *web.config* por cada aplicación. Y un único archivo *machine.config* por servidor.

- *Global.asax*: eventos a nivel de aplicación.

- Directorio *Bin*: contiene el *assembly* (ensamblado) de la aplicación (Ej.: MiAplicación.dll) y cero o más *assemblies* de terceros (componentes externos).

- Enlaces a *servicios web XML*: permiten a la aplicación ASP.NET enviar y recibir datos desde servicios web.

CAPÍTULO 3
FASE DE INICIO

3.1. Introducción

La *fase inicial* del Proceso Unificado permite definir el ámbito del sistema para justificar la factibilidad del proyecto. Este capítulo plantea un modelo conceptual de su arquitectura de acuerdo a las especificaciones funcionales que se obtengan de la identificación de los requisitos exigidos por el cliente. Igualmente, se efectúa su análisis, evaluación y definición de riesgos mediante el uso de las herramientas de UML (modelo de dominio, casos de uso, actividades, clases de diseño y paquetes).

En la fase de inicio se desarrollan las tres primeras iteraciones del flujo de trabajo normal. La intención es recopilar la mayor cantidad de requisitos.

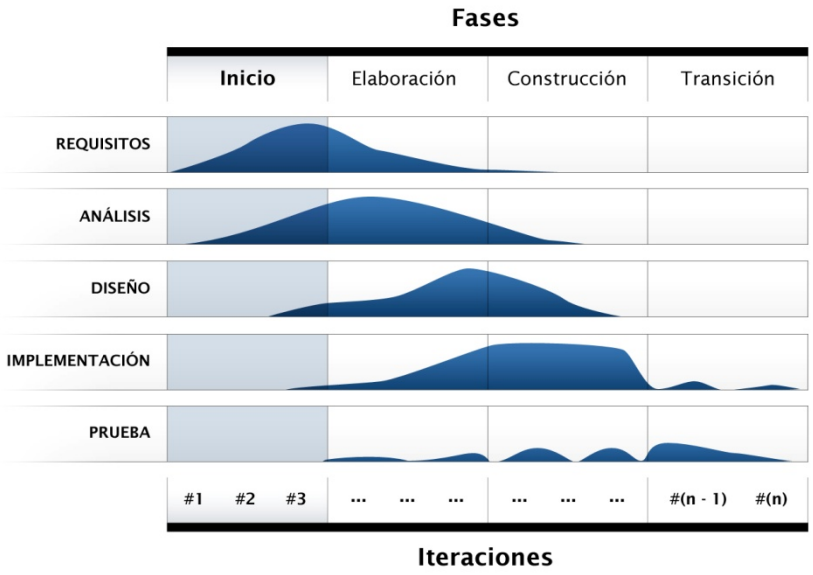


Figura 3.1. Diagrama de flujo de trabajo de las fases del proceso unificado, con identificación de la fase de inicio

3.2. Planificación de la fase de inicio

El desarrollo de la fase de inicio se plantea de manera progresiva. Parte con un listado de los requisitos y una breve descripción de estos, sigue con la definición del modelo de dominio en donde se describirán las principales clases que interactúan en el sistema, continuará con el desarrollo de los casos de usos para diagramar una estructura inicial según los requisitos listados. Una vez elaborados estos diagramas, se refinan y estructuran las funcionalidades a través de los diagramas de actividades, para comprender cómo se deben realizar los casos de uso. Y por último, se construirán los diagramas preliminares de clases y paquetes para encapsular los elementos que se van definiendo durante la realización del análisis del sistema.

3.3. Requisitos

La definición de los requisitos del sistema representa uno de los pasos más importantes en el cumplimiento de los objetivos del proyecto y en la satisfacción de las necesidades del cliente. Bien cita Fred Brooks:

> "*La parte más dura en la construcción de un sistema de software es decidir cómo construirlo… Ninguna otra parte del trabajo mutila el resultado del sistema si está hecho mal. Ninguna otra parte es más dificultosa para rectificarlo después."* (5)

A través de los requisitos se comenzarán a definir las funcionalidades que el sistema deberá tener.

3.3.1. Lista de requisitos o características

La lista presentada a continuación es producto de la recopilación de información necesaria para el entendimiento del problema y los procesos que se quieren automatizar y mejorar, a manera de generar un sistema adecuado. Según lo anterior el sistema debe comprender y permitir los siguientes requerimientos generales:

» Incluir componentes codificados que encapsulen las siguientes funcionalidades comunes de websites:

- Acceso de usuarios y seguridad.
- Controles de navegación, búsqueda y de contenido general.
- Galería multimedia.
- Listados de documentos y archivos.
- Canales de distribución de información (*mailer*, canales RSS).

» Implementar estos componentes en websites con el uso de lenguajes de marcado (XML y XHTML).

» Codificar los controles y componentes de forma totalmente compatible con los principales navegadores web existentes (Microsoft Internet Explorer versiones 6, 7 y 8, Mozilla Firefox versiones 2 y 3, Apple Safari versión 3, Opera versión 9.5 y Google Chrome).

- Establecer los valores predeterminados y parámetros con que iniciará la ejecución formal de un website o aplicación web.

- Desarrollar un *sistema administrador de contenidos* (CMS, por sus siglas en inglés) que a través de una interfaz clara y amigable le permita al usuario final administrar *toda la información manejada por los componentes implementados* en un website.

- Generar automáticamente toda la estructura tabular requerida en la base de datos de acuerdo a los parámetros configurados.

- Agregar o eliminar funcionalidades a un website puesto en producción, garantizando la mantenencia de los datos existentes.

- Utilizar AJAX y todas las herramientas requeridas para darle mayor agilidad a la visualización de los websites.

- Implementar opciones de seguridad que minimicen los riesgos de ataques o mal uso del sistema.

- Funcionar como una aplicación web, desarrollada con la tecnología ASP.NET 3.5, el lenguaje de programación C# y los manejadores de base de datos Microsoft SQL Server 2000 o superior y SQL Server Express 2005 o superior.

De forma específica se deben comprender y permitir los siguientes requerimientos para los componentes detallados del sistema:

» Incluir la opción para permitir a los internautas registrarse a través del sitio web para el caso de portales generadores de comunidad.

» Generar de forma dinámica los menús de acceso y mapa del sitio, a través del componente de *navegación*.

» Implementar el componente de *contenido general* de forma que permita la visualización de título, fecha, autor, contenido con formato enriquecido, fotografías y despieces, útiles para secciones de noticias, avisos y secciones del website que contengan básicamente texto.

» Visualización de fotografías, videos y audio, organizados en galerías a través del componente *galería multimedia*. Este componente incluirá la opción para ampliar las fotografías y videos, así como permitir varias disposiciones o diagramaciones para listar estos contenidos.

» Listar y cargar archivos y documentos en formatos populares (DOC, PDF, XLS, entre otros) por medio del componente *documentos y archivos*.

» Desarrollar el componente *mailer*, que servirá de base para el envío de correo electrónico (útil en secciones como contáctenos, sugerencias y soporte técnico)

» Desarrollar servicios web que permitan la distribución e interoperabilidad de la información de carácter público que se incluya en el sitio web; el más básico e importante será el componente para *canales RSS*.

3.3.2. Modelo de dominio

Para capturar los tipos más importantes de objetos en el contexto del sistema se presenta el modelo de domino. Los objetos o clases del dominio representan las “cosas” que existen o los eventos que suceden en el entorno en el que trabajará el sistema.

Su objetivo es contribuir a la comprensión del contexto del sistema, y por extensión de los requisitos que se desprenden de este contexto. Es decir, contribuye a una comprensión del problema que el sistema resolverá en relación a su contexto. El modo interno por el cual el sistema resolverá este problema se tratará en los siguientes flujos de trabajo. Las clases que se presentan en el modelo de domino son las siguientes:

- **Saetha**: representa la Dirección de Diseño y Sistemas de la organización Saetha Business Group, orientada al desarrollo de soluciones integrales en tecnología para Internet.

- **Website**: conjunto de páginas, imágenes y otros formatos multimedia, bases de datos y configuraciones que pueden ser accedidos y visualizados por internautas o clientes vía Internet. Se considera dinámico cuando ofrece opciones para que el cliente actualice de forma frecuente la información ofrecida por él.

- **Cliente**: cada uno de los clientes de Saetha BG que haya solicitado el desarrollo de al menos un website. En caso de que los websites adquiridos incluyan secciones dinámicas, el cliente deberá contar con

algún método para administrar sus contenidos y/o consultar las participaciones de los internautas.

» **Diseñador**: personal de Saetha BG capacitado para realizar el diseño gráfico de todas las páginas y componentes del website.

» **Programador**: personal de Saetha BG capacitado para llevar el diseño realizado del website al marcado XHTML, así como de asignarle funcionalidad correspondiente.

» **Internauta**: el usuario común de Internet que podrá visualizar el website.

» **Página**: cada pantalla de un website que organiza y visualiza los gráficos, textos y formularios con los que interactuarán los internautas.

» **Componente**: o también referido como *control*, representa cada bloques de elementos presentes en una página del sitio web con los que podrán interactuar los internautas en base a alguna función específica.

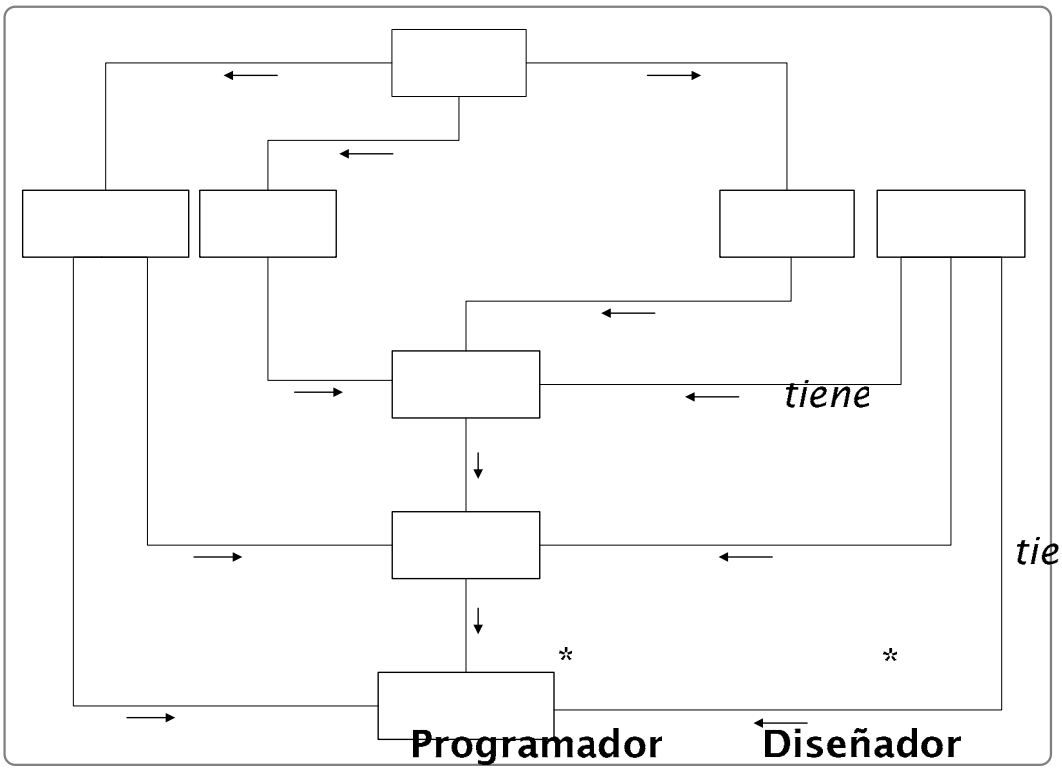


Figura 3.2. Modelo de dominio del sistema

3.3.3. Modelo de casos de uso

En esta sección se identifican los entes que interactúan y están involucrados con los procesos y los casos de uso necesarios para delimitar el sistema, el alcance del proyecto y detallar los más importantes o básicos.

3.3.3.1. Identificación de los actores

- **Analista**: personal programador o diseñador de Saetha BG capacitado para llevar el diseño realizado del website al marcado XHTML, así como de asignarle funcionalidad correspondiente.

- **Cliente**: personal perteneciente a una empresa cliente de Saetha BG, previamente designado y autorizado para realizar actualización o mantenimiento de los contenidos de un website desarrollado por Saetha BG para dicha empresa cliente.

- **Internauta**: cada usuario común de Internet que podrá visualizar algún website desarrollado por Saetha BG.

3.3.3.2. Identificación de casos de usos

De acuerdo al contexto y requisitos del sistema, se puede modelar una solución haciendo uso de dos subsistemas: uno público, accesible por todos los internautas, y otro, un CMS con el que sólo interactuarán los clientes y analistas. Ambos englobarán la totalidad de un mismo website, recordando que las herramientas que brindará este proyecto servirán para implementar la variedad de sitios web que Saetha BG desarrolle.

En la *figura 3.*3 se muestra el diagrama de casos de uso del sistema y en las *tablas 3.1* a *3.7* se describen cada uno de dichos casos de uso.

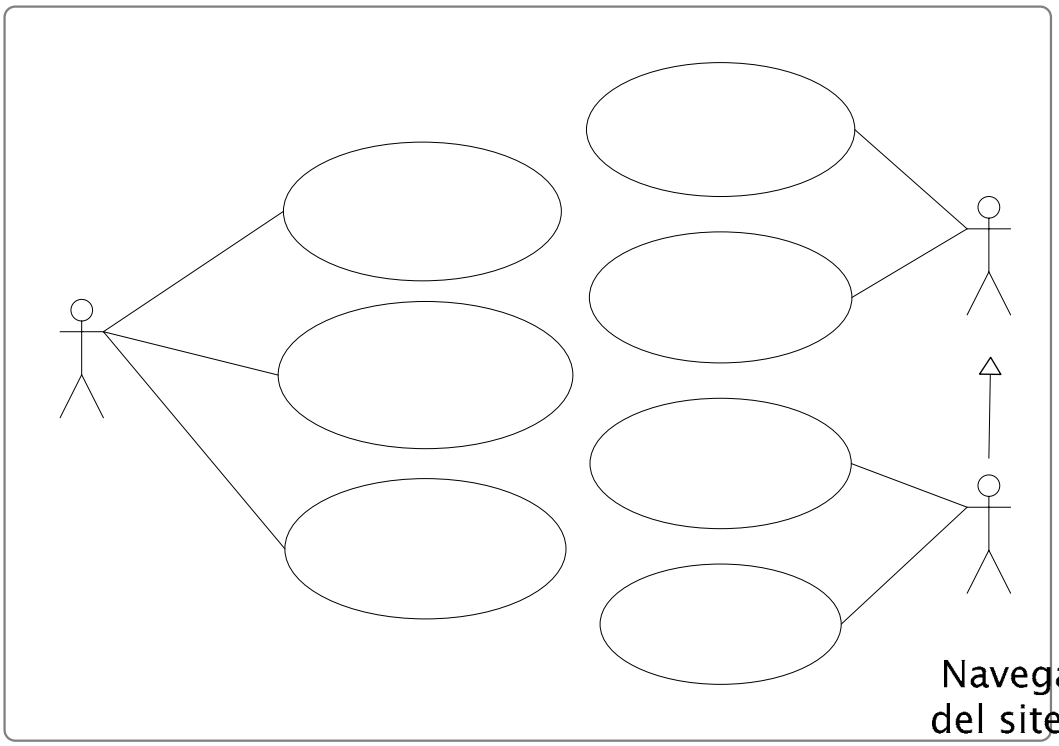


Figura 3.3. Diagrama de casos de uso

Tabla 3.1. Caso de uso Navegar en las secciones del site administradas por el sistema

ACTOR PRINCIPAL	Internauta
ALCANCE	Área pública
NIVEL	Sub-función
DESCRIPCIÓN	Cualquier internauta, al ingresar la dirección del sitio web publicado podrá acceder al diseño web realizado, teniendo menús y vínculos como herramientas para navegar entre todas las páginas y secciones públicas del website. De igual forma, se muestran las interfaces de todos los componentes implementados, de acuerdo a sus configuraciones.

Tabla 3.2. Interactuar con componentes implementados

ACTOR PRINCIPAL	Internauta
ALCANCE	Área pública
NIVEL	Sub-función
DESCRIPCIÓN	De acuerdo a las características de cada componente implementado, se le permitirán al usuario ciertas interacciones y funciones específicas.

Tabla 3.3. Caso de uso Registrarse como usuario para acceso a secciones protegidas

ACTOR PRINCIPAL	Internauta
ALCANCE	Área pública
NIVEL	Sub-función
DESCRIPCIÓN	Ciertos websites podrían ofrecer el registro libre de usuarios para dar acceso a funcionalidades que requieran de una identificación y/o seguimiento como usuario. De igual manera, aunque otros websites no ofrezcan el registro de forma libre para los usuarios, pueden proteger secciones de forma que sólo sean accesibles por los internautas a los que el cliente haya facilitado credenciales válidas.

Tabla 3.4. Caso de uso Validar credenciales de acceso

ACTOR PRINCIPAL	Cliente
ALCANCE	Sistema administrador de contenidos
NIVEL	Sub-función
DESCRIPCIÓN	Tanto analistas como clientes deberán ingresar credenciales válidas para tener acceso a las aplicaciones de gestión del CMS.

Tabla 3.5. Caso de uso Gestionar contenidos y parámetros

ACTOR PRINCIPAL	Cliente
ALCANCE	Sistema administrador de contenidos
NIVEL	Resumen
DESCRIPCIÓN	Los clientes podrán administrar los distintos componentes de contenido (por ejemplo, noticias, galerías fotográficas y documentos) así como establecer parámetros de los componentes estructurales.

Tabla 3.6. Caso de uso Configurar componentes en páginas

ACTOR PRINCIPAL	Analista

ALCANCE	Sistema administrador de contenidos
NIVEL	Resumen
DESCRIPCIÓN	Los analistas podrán configurar la forma en que se ensamblen los componentes en las páginas de un sitio, estableciendo los parámetros para que la implementación esté bien adaptada al diseño e ideas originales para dicho website.

Tabla 3.7. Caso de uso Gestionar credenciales de acceso

ACTOR PRINCIPAL	Analista
ALCANCE	Sistema administrador de contenidos
NIVEL	Sub-función
DESCRIPCIÓN	Los analistas podrán administrar las credenciales con que los clientes accederán al panel CMS de sus websites.

3.3.4. Identificación de los riesgos

La identificación de los riesgos es un intento sistemático para especificar las amenazas al plan del proyecto. Identificando los riesgos conocidos y predecibles, se da un paso adelante para evitarlos cuando sea posible y controlarlos cuando sea necesario.

Se pueden clasificar dos tipos diferenciados de riesgos: genéricos y específicos. Los riesgos genéricos son una amenaza potencial para todos los proyectos de software. Los específicos sólo los pueden identificar los que

tienen una clara visión de la tecnología, el personal y el entorno específico del proyecto en cuestión. Para identificar los riesgos específicos del producto se examinan el plan del proyecto y la declaración del ámbito del software y se desarrolla una respuesta a la siguiente pregunta: ¿Qué características especiales de este software pueden estar amenazadas por el plan del proyecto? (15)

Tanto los riesgos genéricos como los específicos se deberían identificar sistemáticamente. Un método para identificar riesgos es crear una lista de comprobación de elementos de riesgo. La lista de comprobación se enfoca en un subconjunto de riesgos conocidos y predecibles, que pueden ser organizados de diferentes maneras. Un formato de lista de comprobación de elementos de riesgo contiene simplemente las características relevantes para cada sub-categoría genérica. A continuación se definen cuáles de estos elementos genéricos podrían afectar la ejecución del proyecto, organizados de forma categorizada.

Tamaño del producto

Riesgos asociados con el tamaño general del software a construir.

- El software a desarrollar puede considerarse complejo en la medida que, para cumplir con los requerimientos de Saetha BG, se necesiten detallar muchas características y comportamientos distintos de los componentes, para que éstos no afecten en su implementación la originalidad y calidad de los diseños de la empresa.

Impacto en el negocio

Riesgos asociados con las limitaciones impuestas por la gestión o por el mercado.

- Aunque la tecnología ASP.NET 3.5 es una de las más innovadoras que hay en el mercado del desarrollo web, muchas empresas de hospedaje web aún no ofrecen soporte para la ejecución de esta versión. Esto, por la misma razón; el poco tiempo que está en el mercado.

- Hacer todos los componentes totalmente compatibles con la visualización en los principales navegadores existentes en el mercado puede tornarse engorroso.

Todos los riesgos presentados anteriormente serán solventados a medida que se vaya avanzando en las iteraciones del proceso unificado de las siguientes fases, para ir perfeccionando el diseño e implementación del sistema, siguiendo de manera adecuada la metodología, continuando la revisión de información pertinente y analizando todos los requisitos y funcionalidad del sistema.

3.4. Análisis

Se presenta a continuación un primer análisis de los requisitos anteriormente identificados.

3.4.1. Diagramas de actividades

Con los diagramas de actividades se representa el funcionamiento en forma general de las principales operaciones que realizará el sistema, la secuencia en que se ejecutan y cómo el sistema reacciona y se comporta en respuesta a las señales que emite el usuario durante los procesos de realización de los casos de uso.

También se incluyen en algunos diagramas ciertas actividades que poseen fondo gris; esto simboliza que dichas actividades son más complejas y podrían ser detalladas en el siguiente capítulo.

3.4.1.1. Diagrama de actividades del Área pública

Comprende las actividades del website público. Las señales son emitidas por los internautas que accedan la página web.

3.4.1.1.1. Navegar a una sección del website

Esta actividad se ejecuta cada vez que un internauta intenta acceder a una página del website. Se visualizarán todos los contenidos y componentes de la página, quedando abierta la posibilidad de que el usuario accione al seleccionar un vínculo o interactúe con algún control o componente.

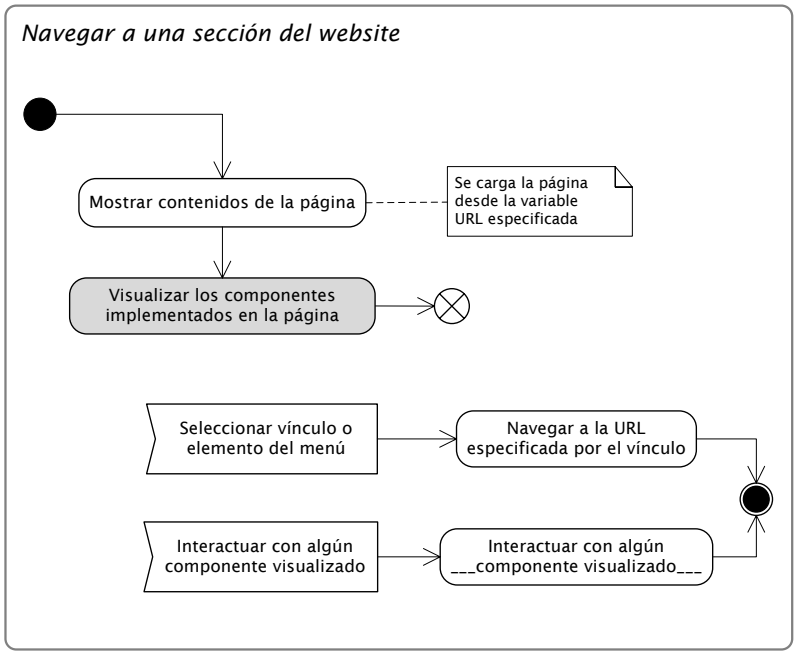


Figura 3.4. Diagrama de actividad Navegar a una sección del website

3.4.1.1.2. Registrar como usuario

Esta actividad se ejecuta cuando un internauta decide registrarse para acceder a secciones protegidas del website, introduciendo sus datos y siendo verificada la unicidad de su identificación.

3.4.1.1.3. Interactuar con componentes implementados

El usuario puede accionar cualquier elemento de dichos controles, no siendo restrictivo el orden en que se quiera interactuar con ellos. La forma en que cada componente procese y reaccione las señales del usuario serán detalladas en el próximo capítulo.

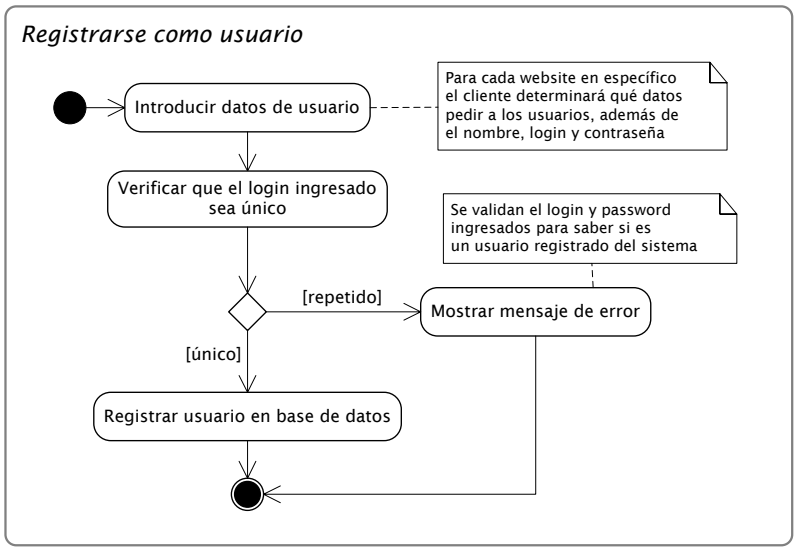


Figura 3.5. Diagrama de actividad Registrar como usuario

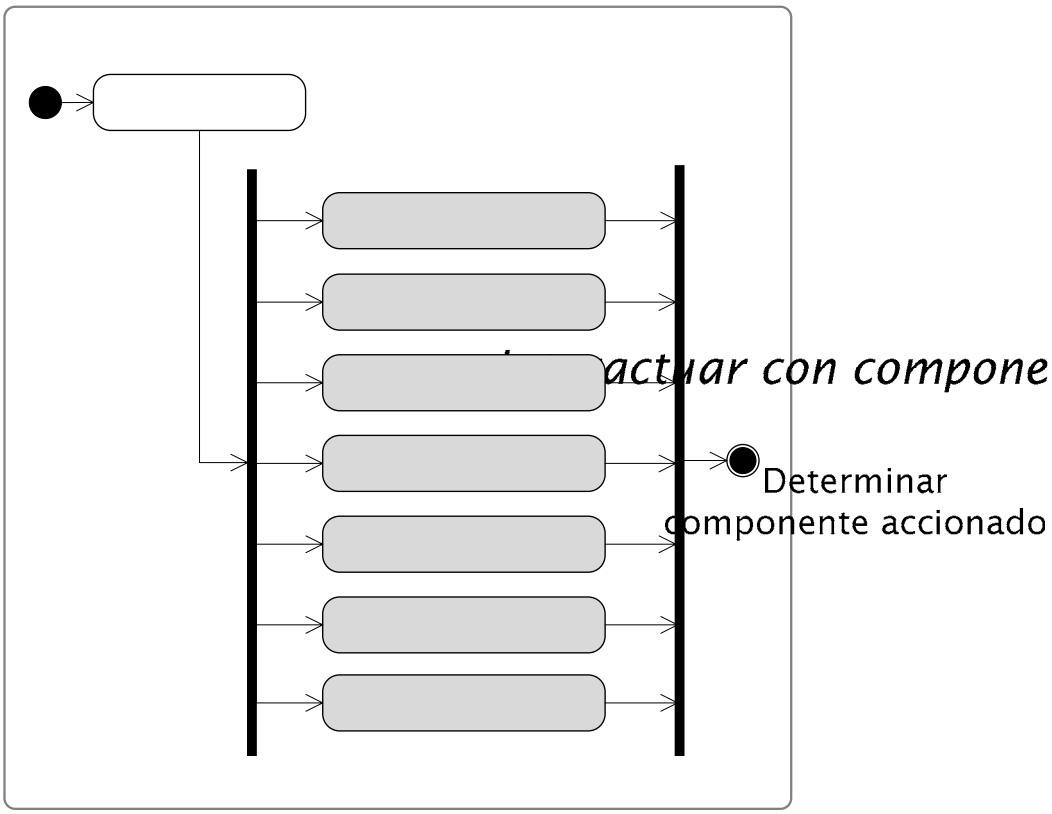


Figura 3.6. Diagrama de actividad Interactuar con componentes instalados

3.4.1.2. Diagrama de actividades del CMS

Comprende las actividades del CMS. Las señales son emitidas por los administradores del sitio web, así como por los analistas de Saetha.

3.4.1.2.1. Validar credenciales de acceso

Esta actividad se ejecuta cuando un administrador o analista de Saetha intentan acceder al CMS para administrar la información del sitio web. En primer lugar, se verifica que haya una sesión previamente iniciada; en caso

de que no, se muestra el panel para iniciar sesión, identificándose con su nombre de usuario y contraseña.

Al estar identificado el usuario, se determina cuál es su rol de usuario y se construye el menú de acceso de acuerdo a los permisos a aplicaciones correspondientes.

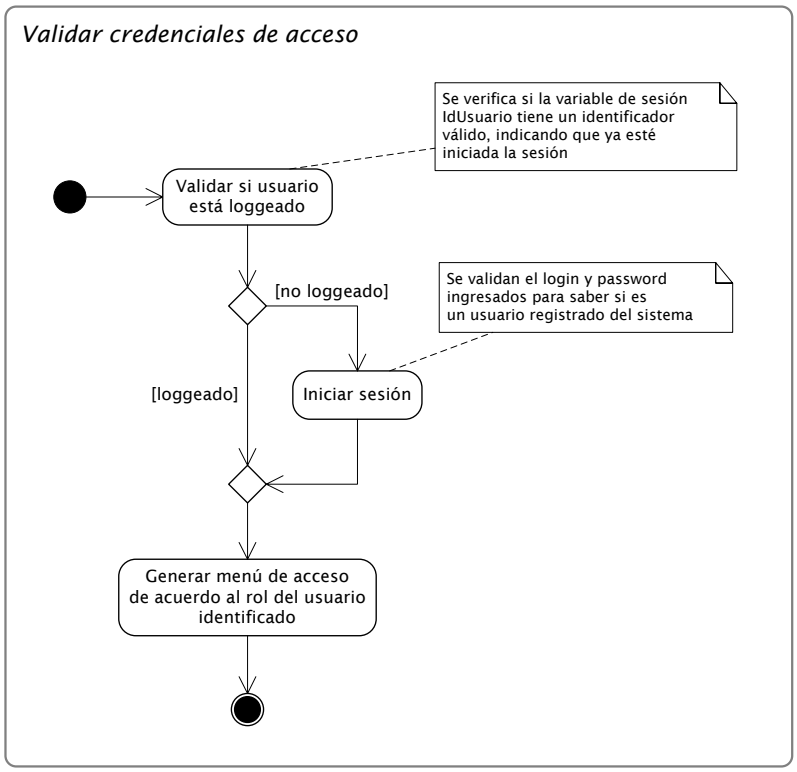


Figura 3.7. Diagrama de actividad Validar credenciales de acceso

3.4.1.2.2. Gestionar contenidos y parámetros

Esta actividad prosigue a la validación de credenciales de usuario. Al ser seleccionado algún elemento del menú de accesos permitidos para el

usuario, se visualiza la aplicación de configuración que corresponda. Cada una de estas aplicaciones será detallada en el próximo capítulo.

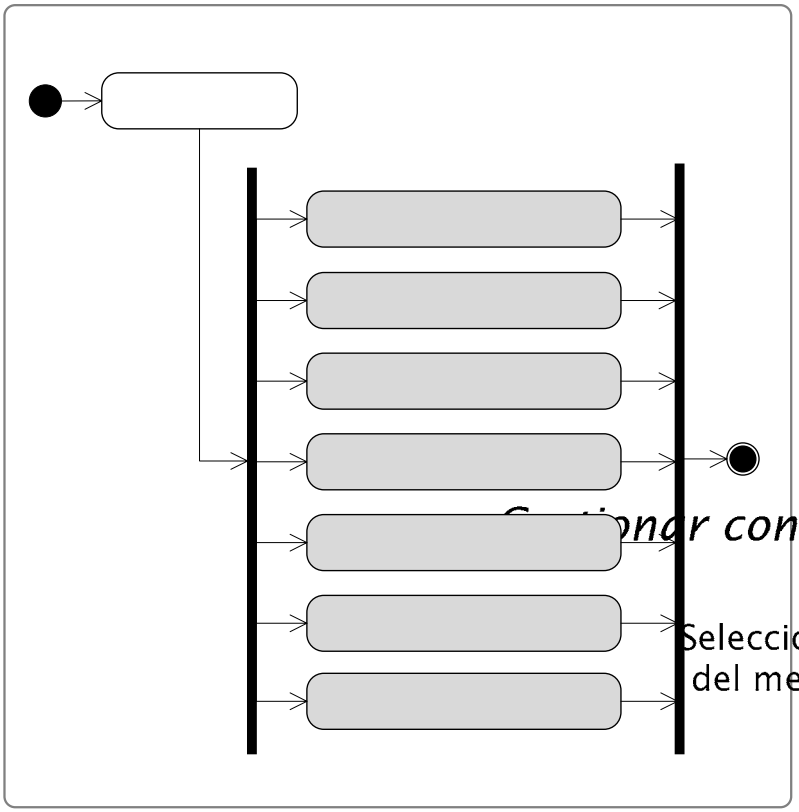


Figura 3.8. Diagrama de actividad Gestionar contenidos y parámetros

3.4.1.2.3. Configurar componentes en páginas

Esta actividad corresponde a la etapa de desarrollo del website, siendo los analistas de Saetha los únicos actores que podrán realizarla. El analista deberá abrir el código de cada página que quiera configurar, agregar el marcado de los componentes que quiera implementar y luego podrá ajustar el resto de los parámetros y propiedades que garantizarán la correcta composición entre diseño y funciones.

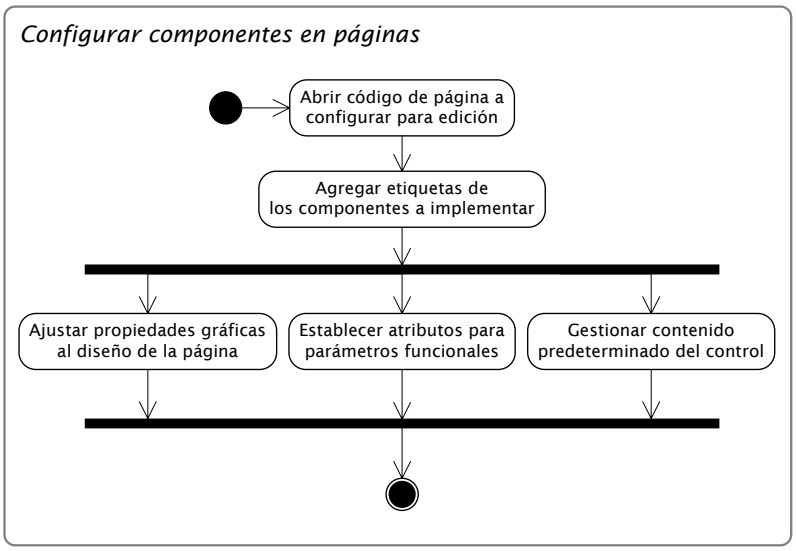


Figura 3.9. Diagrama de actividad Configurar componentes en páginas

3.4.1.2.4. Gestionar credenciales de acceso a usuarios

Los analistas de Saetha serán los actores de esta actividad, pudiendo definir las credenciales y roles para los administradores del website de parte del cliente.

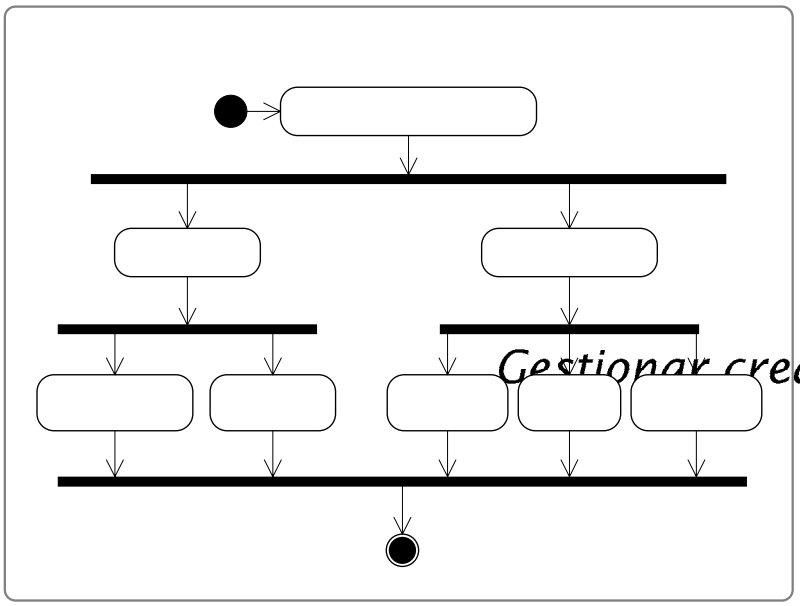


Figura 3.10. Diagrama de actividad Gestionar credenciales de acceso a usuarios

3.4.2. Diagramas preliminares de clases y paquetes

El modelado realizado en esta fase de inicio con los diagramas de caso de uso y actividades, ofrece una noción global de la forma en que se pueden establecer y organizar las clases para representar los aspectos estáticos del sistema.

Los diagramas preliminares de clases y paquetes que se presentan a continuación conforman las posibles clases y su organización agrupadas por los posibles paquetes, que serán evaluados y refinados en la siguiente fase.

3.4.2.1. Diagrama de paquetes

El diagrama de clases de más alto nivel es lógicamente un dibujo de los paquetes que componen el sistema. A su vez cada paquete tendrá un diagrama que muestra las clases del mismo.

3.4.2.1.1. Paquete SaethaWAM

El paquete *SaethaWAM* representa la totalidad del sistema a desarrollar y está formado por cinco paquetes distribuidos en dos capas: capa de *negocio* y de *presentación.* La *capa de negocio* contiene los cuatro paquetes que organizan la funcionalidad lógica de todos los componentes: *Seguridad, Estructura, Accesibilidad e Información*. La *capa de presentación* contiene un único paquete, *UI*. Este incluye los dos paquetes que organizarán la totalidad de elementos con que los distintos usuarios tendrán interacción directa: *Controles* y *CMS*.

- En el paquete *Seguridad* se encuentra la lógica necesaria para la administración de los usuarios y roles que accederán al sistema CMS, así como a las secciones protegidas.

- El paquete *Estructura* define las clases que organizarán al sitio web en las correspondientes secciones.

- Dentro de *Accesibilidad* ubicaremos las clases que faciliten el acceso a todas las áreas del website. En primera instancia, sólo formarán parte de él las clases que permitan realizar búsquedas en todo el website.

- El paquete *Información* incluye todas las clases que modelarán los elementos de información general (noticias, avances, galerías multimedia).

- Dentro de *UI* encontraremos las clases que modelarán los controles gráficamente visibles al usuario y que conectarán la interacción del usuario con las clases de la capa de negocio, así como las páginas que conformarán el sistema de administración de contenidos.

3.4.2.1.2. Paquete System

El resto de los elementos que el sistema va a usar son los que ofrece el marco de desarrollo .NET Framework 3.5. Se organizan principalmente en el paquete *System* del *.NET Class Library*. Los paquetes de este marco de desarrollo que se utilizarán son:

- **System.Data** y **System.Data.SqlClient**: Agrupan todos los elementos que proveerán acceso a datos y repositorios. Todo este segmento del *.NET Class Library* tiene el nombre de ADO.NET.

- **System.Web.UI y System.Web.UI.WebControls**: Todas las entidades que procesan la visualización e interacción con controles gráficos de usuario básicos se encuentran en este paquete. Entre las clases más comunes están *Page*, *MasterPage, Control* y los controles comunes (*TextBox, Label, CheckBox, RadioButton, Button, FileUpload,* entre otros).

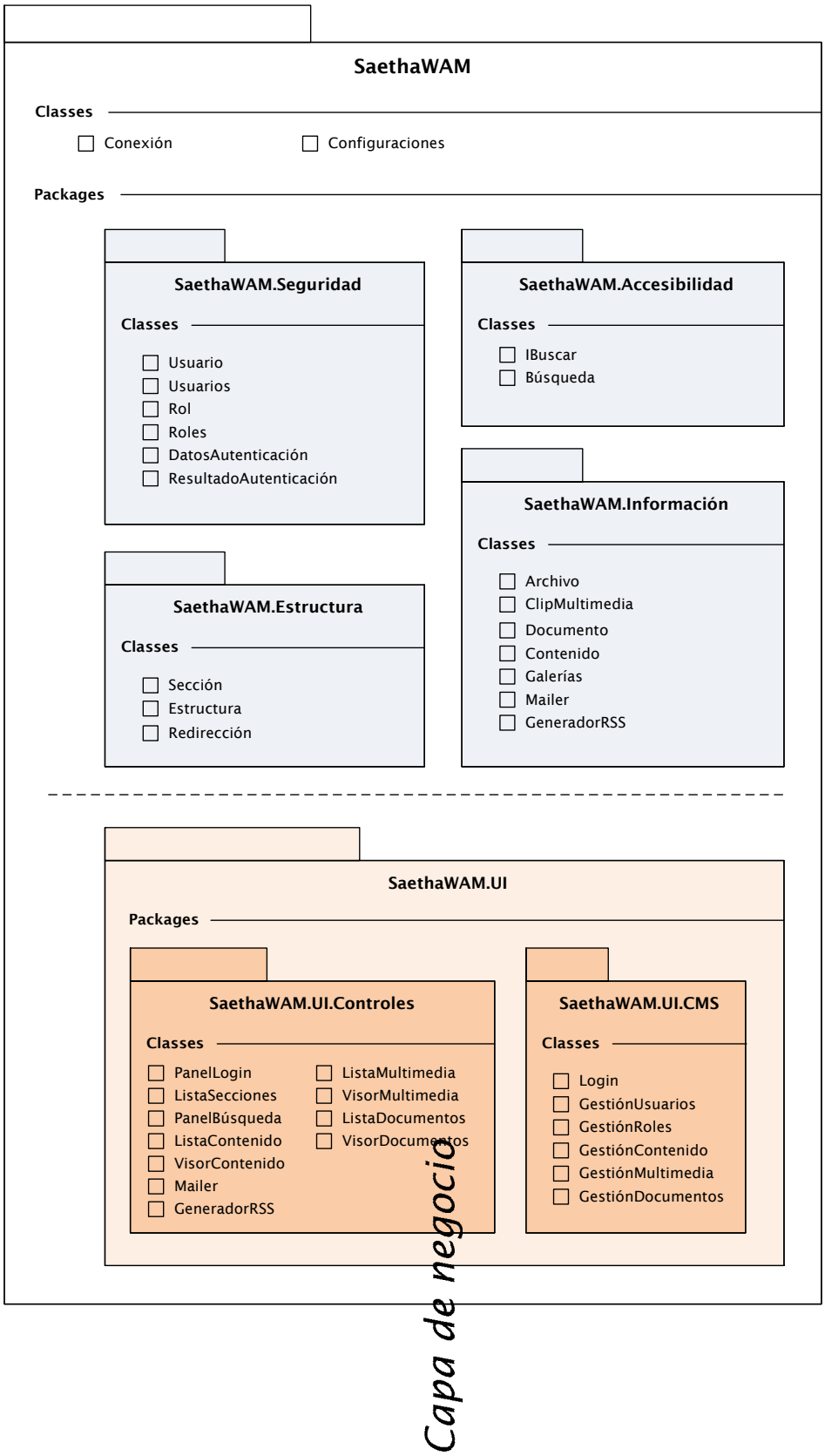

SaethaWAM
Classes
Conexión
Configuraciones
Packages
SaethaWAM.Seguridad
Classes
Usuario
Usuarios
Rol
Roles
DatosAutenticación
ResultadoAutenticación
SaethaWAM.Accesibilidad
Classes
IBuscar
Búsqueda
SaethaWAM.Información
Classes
Archivo
ClipMultimedia
Documento
Contenido
Galerías
Mailer
GeneradorRSS
SaethaWAM.Estructura
Classes
Sección
Estructura
Redirección
SaethaWAM.UI
Packages
SaethaWAM.UI.Controles
Classes
PanelLogin
ListaSecciones
PanelBúsqueda
ListaContenido
VisorContenido
Mailer
GeneradorRSS
ListaMultimedia
VisorMultimedia
ListaDocumentos
VisorDocumentos
SaethaWAM.UI.CMS
Classes
Login
GestiónUsuarios
GestiónRoles
GestiónContenido
GestiónMultimedia
GestiónDocumentos
Capa de negocio

Figura 3.11. Diagrama de paquetes SaethaWAM

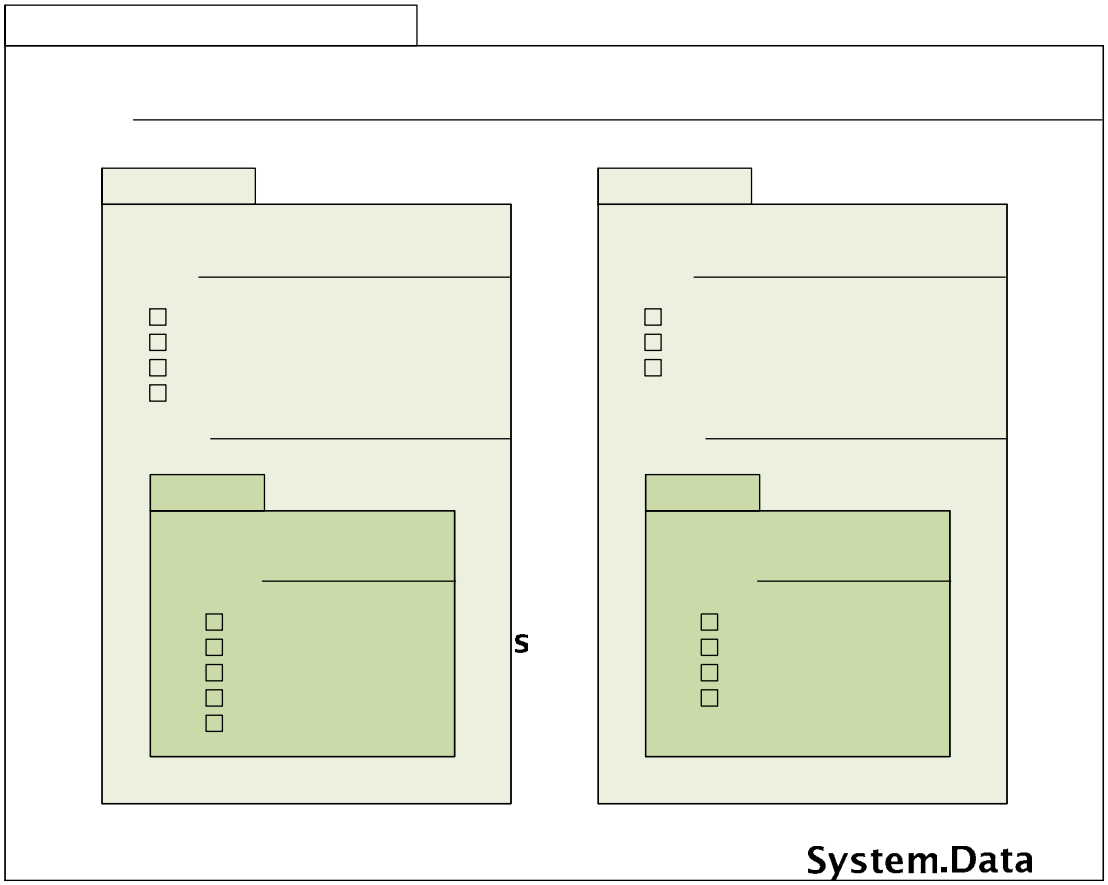


Figura 3.12. Diagrama de paquetes System

3.4.2.2. Clases del paquete SaethaWAM.Seguridad

En el paquete *SaethaWAM.Seguridad* nos conseguimos con dos clases estáticas, que son principales dentro de este paquete: *Usuarios* y *Roles*. La primera de ellas contendrá una representación de todos los usuarios

registrados en modo de colección, y ofrecerá métodos para la adición de nuevos usuarios y la autenticación de credenciales ya existentes.

Similarmente, *Roles* representará la colección de los distintos roles de usuario definidos para el acceso a las aplicaciones. Permitirá registrar nuevos esquemas de permisos, que representarán no más que el conjunto de aplicaciones y módulos que tendrán acceso garantizado o denegado para dichos esquemas.

Las clases *Usuario* y *Rol* (singulares) representan la definición de los datos que se manejarán respectivamente cada una de estas categorías. Adicionalmente, la clase *Rol* contendrá métodos para la definición de sus accesos a las aplicaciones. Para esto, se establece una relación con la clase *Sección* del paquete *SaethaWAM.Estructura*.

La ejecución del método de autenticación de usuarios previsto en la clase *Usuarios* retornará una clase del tipo *DatosAutenticación*, que incluye el resultado de la autenticación y la referencia al usuario correspondiente. El resultado de esta validación es del tipo *ResultadoAutenticación*, una enumeración de los tres retornos posibles.

3.4.2.3. Clases del paquete SaethaWAM.Estructura

Contendrá principalmente la clase estática *Estructura*, que organizará toda la jerarquía de secciones del website. La función y ubicación lógica de cada una de estas será modelada con el uso de la clase *Sección*.

La clase *Estructura* incluirá el nodo raíz para la jerarquía de secciones. Cada nodo tiene la posibilidad de agregar sub-secciones a él y generar así toda la estructura.

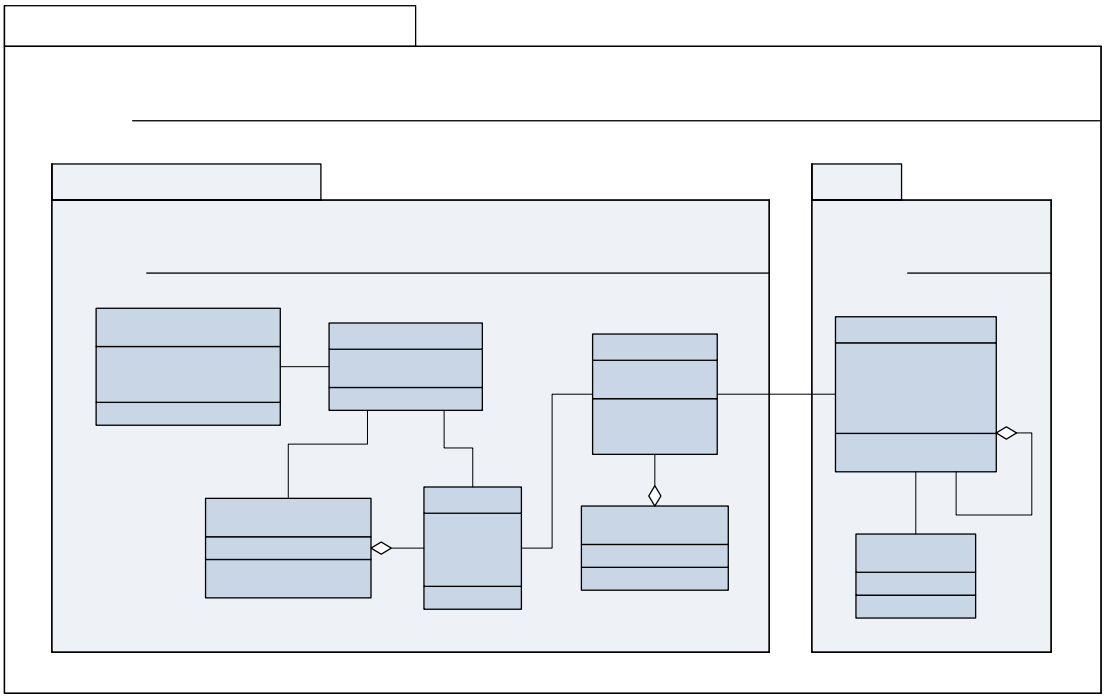


Figura 3.13. Diagrama de clases de diseño para los paquetes SaethaWAM.Seguridad y SaethaWAM.Estructura

3.4.2.4. Clases del paquete SaethaWAM.Accesibilidad

Esta clase ofrece la posibilidad de realizar búsquedas de contenido a través de la clase *Búsqueda*; inspeccionará todos los elementos y colecciones que implementen la interface *IBuscar*, que le añade a dicho contenido la suficiente metadata para acceder y presentar la información solicitada.

Entre la información que esta interface recogerá de las distintas colecciones están las *etiquetas*, que serán un conjunto de palabras clave que se podrán adjuntar a cada objeto de información (documentos, clips, galerías, contenidos). En base a estas etiquetas se podrán acertar búsquedas en las que los criterios introducidos por el usuario sean los elementos más relevantes.

Adicionalmente, esta interface permitirá a cada clase de información definir en qué tablas y campos de la base de datos se encuentra la información sensible a las búsquedas. También permitirá modelar la forma en que se presentará el resumen de la búsqueda, para que la visualización de los resultados por parte de todos los distintos tipos de contenido sea coherente y fácil de visualizar.

3.4.2.5. Clases del paquete SaethaWAM.Información

En este paquete se modelan los principales componentes que mostrarán información al internauta.

Cada elemento de contenido, definido por la clase homónima, podrá adjuntar ninguna o varias galerías ó clips multimedia, además de la información general y de texto (con y sin formato). Incluirá también la referencia al *padre* o *sección/categoría* dentro de la que figurará en la página web.

Las galerías, representadas con la clase *Galería,* incluirán tanto clips multimedia (video, audio, imagen) así como documentos. Podrán mostrar distintas diagramaciones para los elementos que la conforman.

Tanto los clips como los documentos heredarán de la clase *Archivo*, que incluye los atributos comunes a todos los elementos que requerirán almacenar archivos en el servidor.

Las clases estáticas *Mailer* y *GeneradorRSS* son utilitarias: la primera, para permitirle al internauta enviar correos a buzones predefinidos en el diseño del website (por ejemplo, en una sección *Buzón de sugerencias* o *Solicitud de servicio*); la segunda, para la distribución de canales RSS que listen información de cualquier instancia de los componentes de información, a los que pueda suscribirse el internauta.

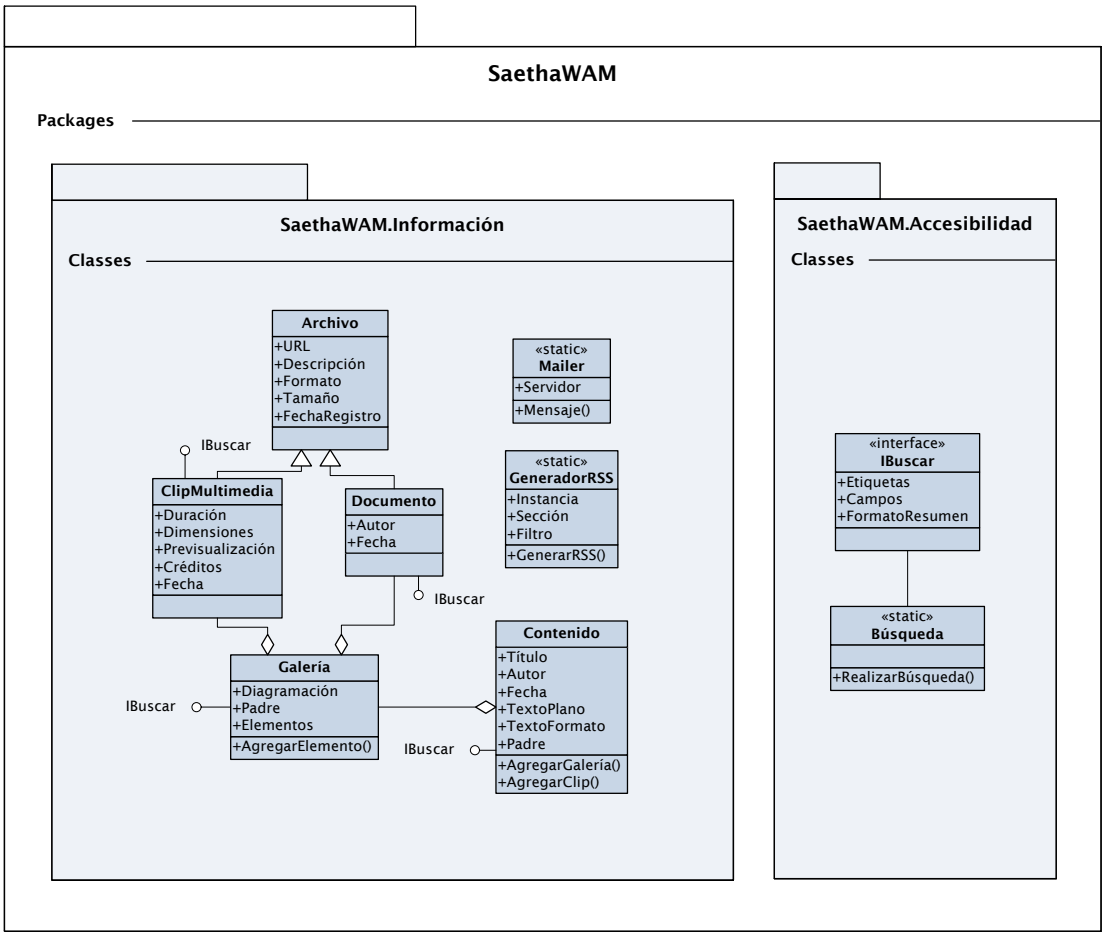


Figura 3.14. Diagrama de clases de diseño para los paquetes SaethaWAM.Información y SaethaWAM.Accesibilidad

3.4.2.6. Clases del paquete SaethaWAM.UI.Controles

Todas las clases de este paquete serán la representación gráfica de los componentes ya modelados en la capa de negocio. Incluirán la definición de los controles primitivos que darán funcionalidad a cada página del website, y se relacionarán con las clases que modelan de forma lógica los datos y comportamientos.

Cada clase será creada de forma que brinde facilidad al analista/diseñador de Saetha la posibilidad de configurar todos los parámetros con la menor cantidad de definiciones, sin comprometer la originalidad del diseño.

Adicional a los controles predefinidos en esta fase de inicio, podrán ser incluidos en la elaboración otros controles relacionados a ellos, más específicos, que granulen la forma de graficarlos y de hacerlos flexibles.

Todas estas clases heredarán de la clase *System.Web.UI.Control*, que es la clase base ofrecida por la plataforma ASP.NET para la creación de controles de servidor.

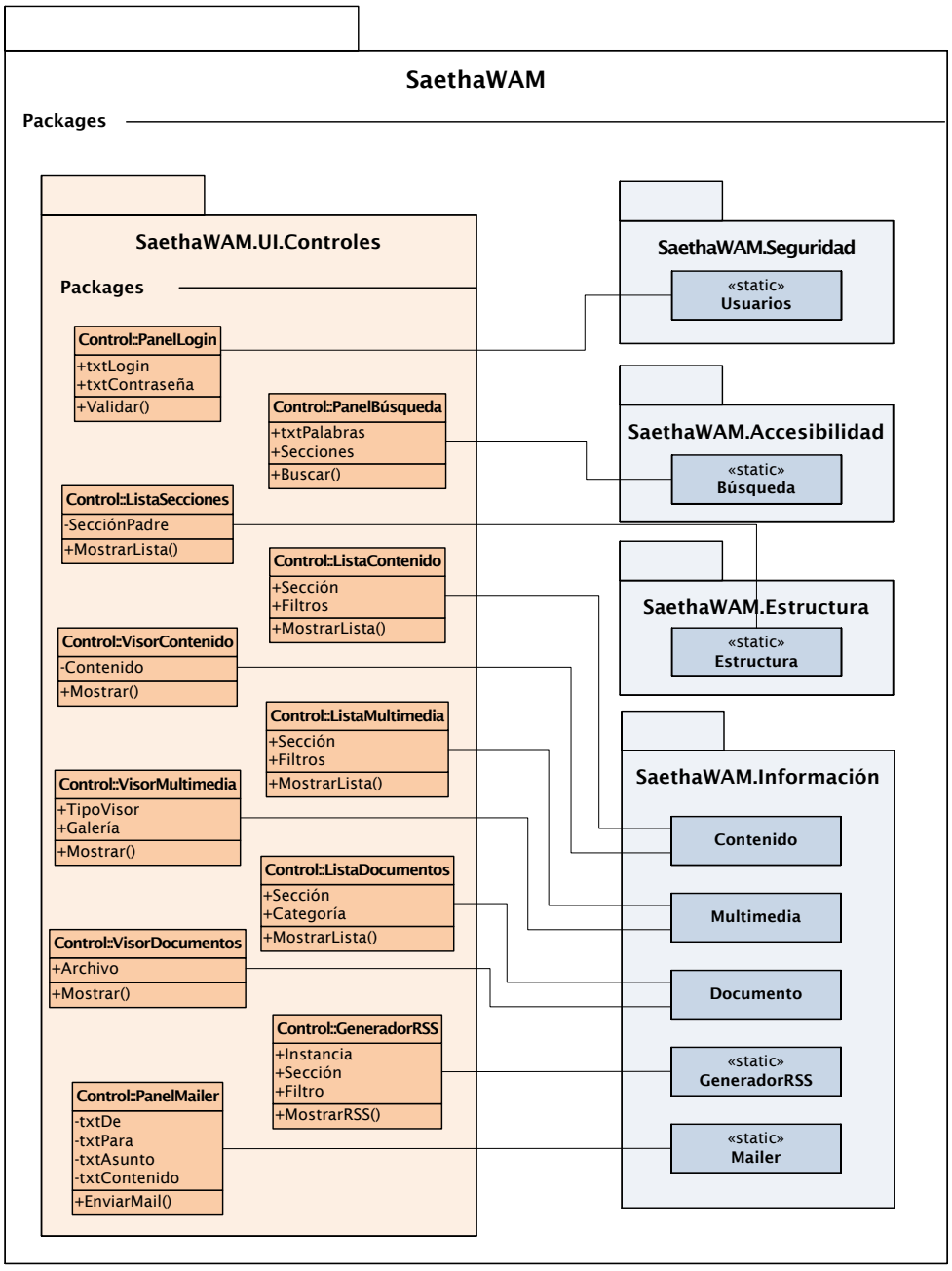


Figura 3.15. Diagrama de clases de diseño para el paquete SaethaWAM.UI.Controles

3.4.2.7. Clases del paquete SaethaWAM.UI.CMS

Este paquete definirá la aplicación administradora de contenido, con que se gestionará toda la información manejada por los componentes.

Todas sus clases heredan de *System.Web.UI.Page*, y representan el *code-behind* de los formularios ASP.NET (archivos *.aspx*) que conforman las interfaces gráficas del CMS.

Luego de la visualización del formulario *Login* y la correcta validación de usuario que allí se realice, se mostrará el acceso a todos los otros gestores que conformarán este módulo.

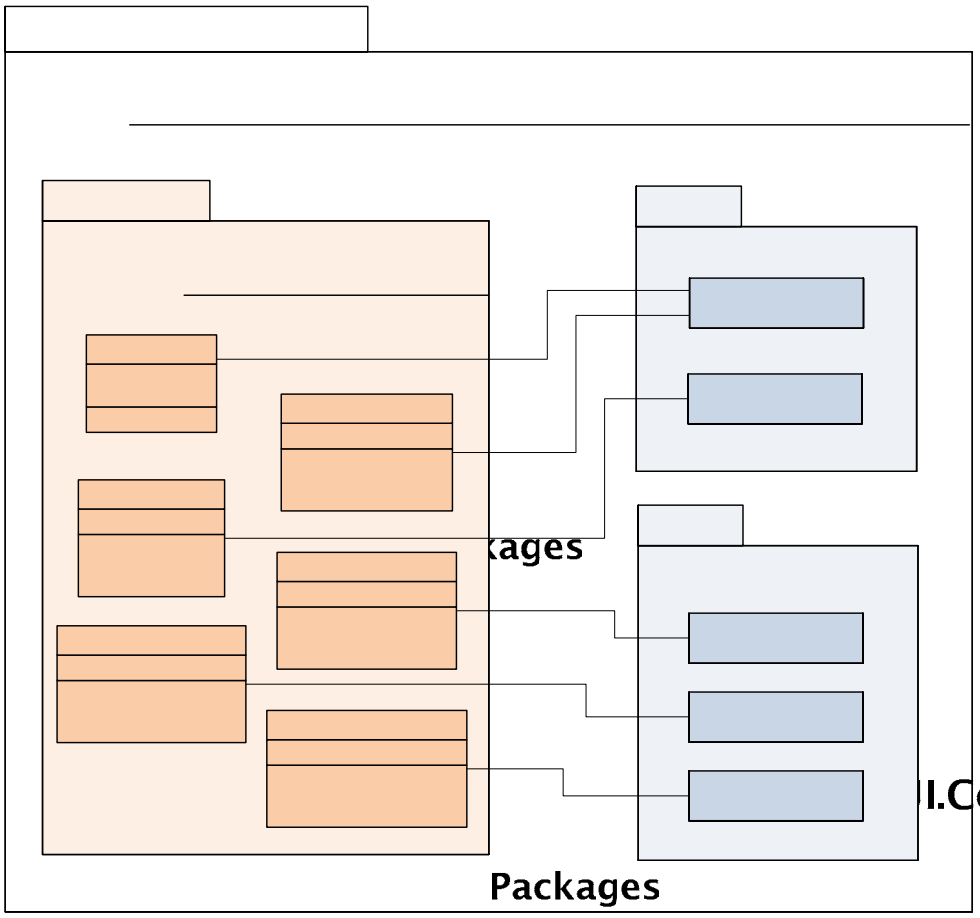


Figura 3.16. Diagrama de clases de diseño para el paquete SaethaWAM.UI.CMS

Page::Login
+txtLogin
+txtContraseña
+Validar()

Page::GestiónUsuarios
+ListaUsuarios
+Listar()
+Agregar()
+Modificar()

Page::GestiónRoles
+ListaRoles
+Listar()
+Agregar()
+Modificar()

Page::GestiónContenido
+ListaContenidos
+Listar()
+Agregar()
+Modificar()

Page::GestiónMultimedia
+ListaMultimedia
+Listar()
+Agregar()
+Modificar()

Page::GestiónDocumento
+ListaDocumentos

3.5. Evaluación de la fase de inicio

Con esta primera fase del proceso unificado se obtuvo una visión general de los requerimientos del proyecto, características claves y riesgos principales, lográndose esto mediante:

- La determinación de los requisitos generales del sistema, donde se trató de establecer el alcance o límite de forma precisa.

- La creación del modelo de dominio para definir el ámbito del sistema e identificar todas las posibles entidades y relaciones entre estas. Esto permitió determinar la división del sistema en dos áreas, una pública (accesible por todos los usuarios de Internet) y otra privada (para la administración del website) dirigido a los usuarios analistas de la empresa Saetha y a los webmaster de las empresas cliente.

- La definición de casos de uso principales para cada subsistema, que permitieron plasmar lo solicitado en los requisitos del sistema, siendo el nivel de definición de estos muy generales debido a que en las próximas fases se extenderá cada uno a un nivel de detalle superior.

- La identificación de los riesgos del sistema.

- El desarrollo de la iteración de análisis, en la cual se desarrollaron los diagramas de actividades para representar los flujos de ejecución de las operaciones generales determinadas en los casos de uso.

» Por último, se desarrollaron los diagramas preliminares de clases y paquetes en una iteración de diseño, mostrando las posibles clases del sistema y sus interrelaciones. Al mismo tiempo, se agruparon cada una de estas en los posibles paquetes de clases con los que se pueda construir el sistema.

CAPÍTULO 4
FASE DE ELABORACIÓN

4.1. Introducción

El propósito de la etapa de elaboración es crear la línea base de la arquitectura para así disponer de unos cimientos sólidos sobre los que se basarán el diseño e implementación durante la fase de construcción. La arquitectura evoluciona considerando los requisitos más significativos (aquellos que tienen un fuerte impacto en la arquitectura del sistema) y la evaluación de riesgos.

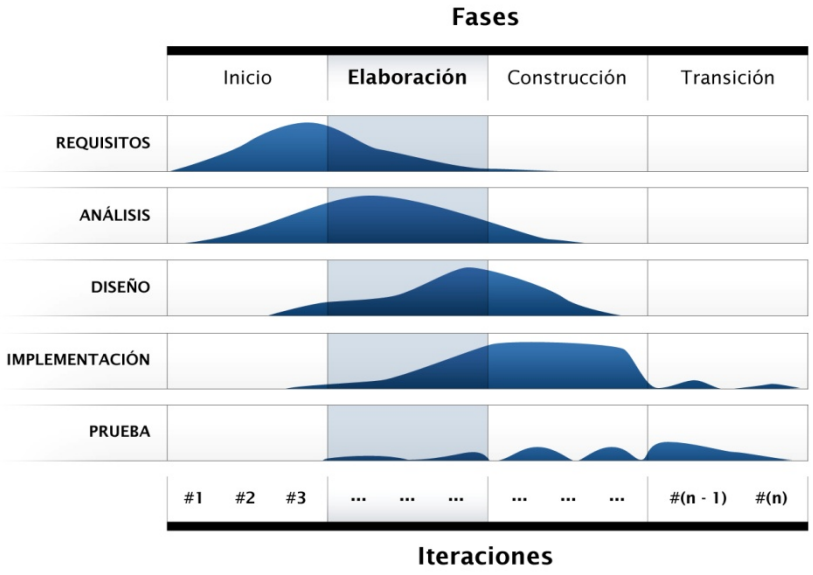


Figura 4.1. Diagrama de flujo de trabajo de las fases del proceso unificado, con identificación de la fase de elaboración

4.2. Planificación de la fase de elaboración

En la fase de elaboración se actualizarán todos los productos de la fase de inicio, comenzando por capturar la mayoría de los requisitos restantes: incluyendo la identificación y desarrollo de los casos de uso específicos, requisitos adicionales que capturan los requisitos no funcionales y cualquier requisito no asociado con un caso de uso específico.

En la iteración de análisis se refinarán las funcionalidades a través de los diagramas de clase de análisis. Luego, en la iteración de diseño, se especificarán los diagramas de clases y paquetes, se desarrollará el diagrama de base de datos y el diseño de prototipos de las interfaces de usuario; asimismo, se incluirán diagramas de secuencia para las principales actividades.

Por último, se realizará en la iteración de implementación el diagrama de despliegue de la aplicación, para combinar la arquitectura de hardware y la arquitectura de software necesaria, así como también se desarrollará un diagrama parcial de componentes.

4.3. Requisitos

En esta nueva iteración de la fase de elaboración se han captado nuevos requisitos para terminar de validar todas las funcionalidades del sistema.

4.3.1. Lista de requisitos o características

A la lista de requisitos identificados en la fase de inicio, se anexan la siguiente característica:

» Incluir controles para inicio de sesión en el componente de *acceso de usuarios y seguridad*, con el fin de limitar el acceso a las secciones que así lo requieran por parte de los internautas y garantizar así la seguridad del sitio web.

4.3.2. Modelo de casos de uso actualizado

En esta sección se presentan los casos de uso que conforman al sistema, incluyendo aquellos desarrollados durante la fase de inicio, sus modificaciones y el resto de los casos de uso específicos que completan los requisitos.

Los actores y casos de uso generales no varían con respecto a los actores y casos de uso generales identificados y descritos en la fase de inicio. Sin embargo, se incluyó un caso de uso en el diagrama de la figura 3.3, llamado *Validar acceso como usuario registrado.* Este nuevo caso de uso está relacionado con la adición del requisito descrito en esta fase de elaboración, e implica la validación de las credenciales de usuario del internauta al tratar de acceder a una página/componente que haya sido marcada como protegida.

En la *figura 4.2* se muestra el diagrama de casos de uso actualizado del sistema y en la *tabla 4.1* se describe el caso de uso añadido en esta iteración.

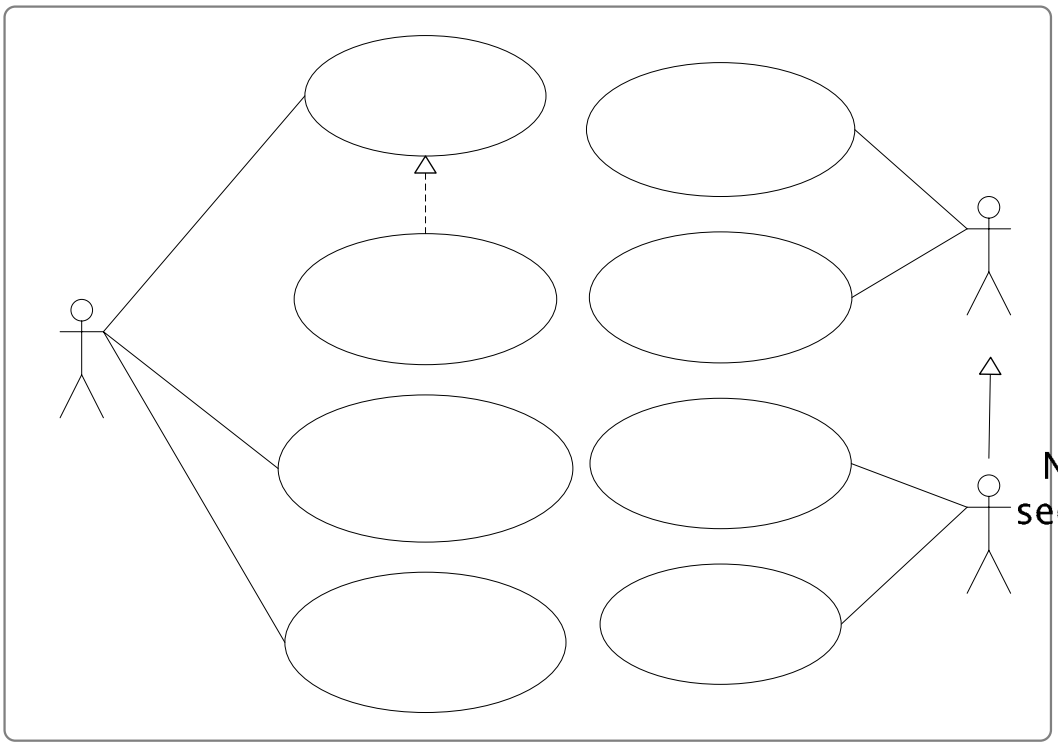


Figura 4.2. Diagrama de casos de uso actualizado

Tabla 4.1. Caso de uso Validar acceso como usuario registrado

ACTOR PRINCIPAL	Internauta
ALCANCE	Área pública
NIVEL	Sub-función
DESCRIPCIÓN	Si intenta acceder a una sección diseñada como protegida, deberá validar sus credenciales de acceso. Al iniciar la navegación de una página del website se verificará si esta página fue marcada como protegida y, en caso de que así fuese, solicitará la validación de credenciales.

4.3.3. Mitigación de los riesgos

Tras la identificación de los riesgos potenciales en la fase de inicio, se presenta a continuación una lista de las soluciones que se consideraron más adecuadas y oportunas, evaluando todos los factores necesarios, incluyendo los puntos que influyen o pudieran interferir directamente con el desarrollo del proyecto:

- Tras el riesgo en el crecimiento de la complejidad del sistema, se ha determinado una gran cantidad de similitudes en los componentes requeridos por Saetha BG, lo que permitirá generalizar muchas de las funcionalidades en los componentes previstos en la iteración de diseño de la fase de inicio.

- Respecto al uso de la tecnología ASP.NET 3.5, ya se ha visto el soporte de esta por parte de los principales proveedores de hospedaje con los que trabaja Saetha BG. Por esta razón, deja de ser un riesgo el uso de esta plataforma para la implementación de los websites producidos por la empresa.

- Tras las primeras iteraciones de construcción realizadas, se pudo notar lo beneficioso que es el uso del estándar XHTML en la codificación de la parte gráfica para la compatibilidad con la mayoría de los navegadores existentes. Todos estos aplicativos incluyen alta compatibilidad con el estándar mencionado y la visualización es uniforme en todos ellos.

4.4. Análisis

Se presenta a continuación un análisis de los requisitos anteriormente agregados o modificados.

4.4.1. Diagramas de actividades

Los aspectos agregados en la iteración de requisitos de esta fase sólo alteraron la representación de una actividad y adicionó una nueva.

4.4.1.1. Navegar a una sección del website

Esta actividad se ejecuta cada vez que un internauta intenta acceder a una página del website. Se verifica si dicha página es de acceso público o privado, siendo requerida en este último caso la validación de credenciales de usuario; posterior a ello se visualizarán todos los contenidos y componentes de la página, quedando abierta la posibilidad de que el usuario accione al seleccionar un vínculo o interactúe con algún control o componente.

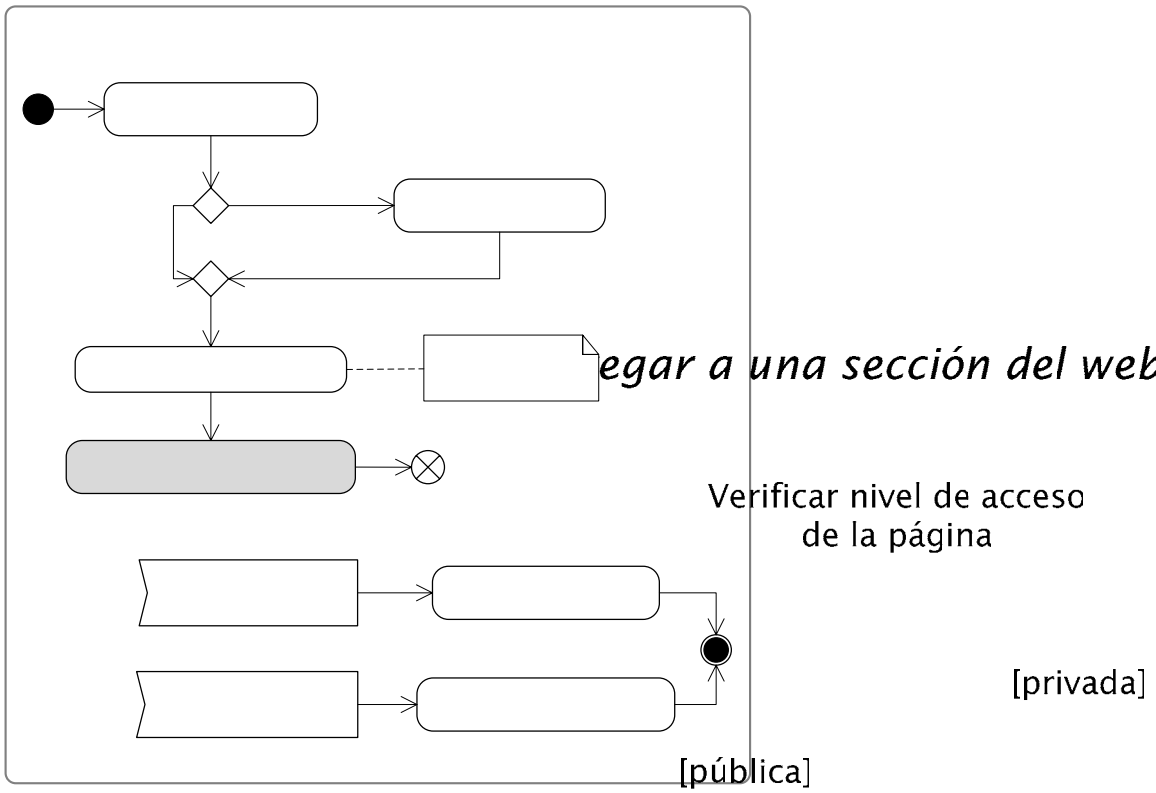


Figura 4.3. Diagrama de actividad Navegar a una sección del website

4.4.1.2. Validar acceso como usuario registrado

Esta validación de acceso se realiza sólo cuando una página que intenta ser accedida por un internauta es privada. Se evalúa en primera instancia si hay una sesión activa (el usuario ya se encuentra *loggeado*) y en caso de que no se le solicita iniciar sesión con su *login* y *password*.

Mostrar contenidos de la página

Visualizar los componentes implementados en la página

Seleccionar vínculo o elemento del menú

Interactuar con algún componente visualizado

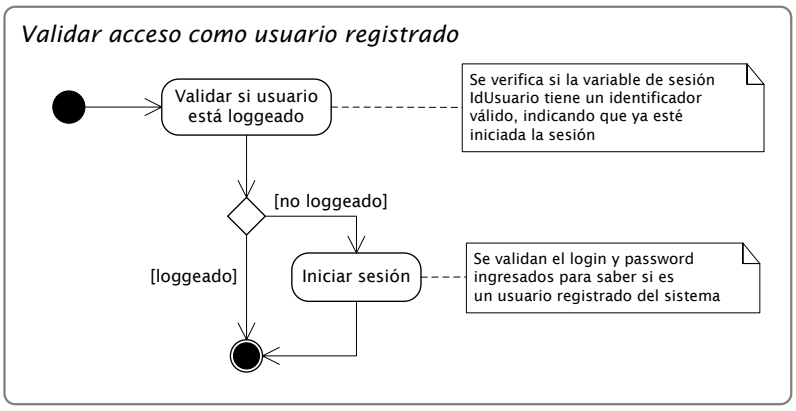


Figura 4.4. Diagrama de actividad Validar acceso como usuario registrado

4.5. Diseño

En esta iteración de la fase de elaboración se plantea un diseño detallado, dirigiéndose ya a un prototipo más evolucionado de la arquitectura del sistema.

4.5.1. Diagramas de clases de diseño y paquetes

Los diagramas de clases de diseño y paquetes representan la estructura estática del sistema.

En Saetha BG ya existía un marco de soporte y ejecución de aplicaciones web, que optimiza la uniformidad y accesibilidad en las interfaces del software y en el manejo de sesiones de usuario. Este marco ya finalizado será adicionado al proyecto bajo el paquete o espacio de nombres *SaethaWAM.Ejecución*. Un resumen de las características ofrecidas sería el siguiente:

- Organiza el almacenamiento de las variables generales de aplicación y específicas de sesión; dado que las correspondientes clases *Sistema* y *Sesión* pueden ser programadas con atributos de nombres y tipos específicos, se pueden mantener estas variables de forma más tipificada. Adicionalmente, heredan de la clase *DictionaryBase*, por lo que se puede almacenar una n-cantidad de objetos de todo tipo en ambos estratos.

» Segmenta de forma más segura y óptima el almacenamiento de variables de servidor para una aplicación o interfaz.

» La clase *Aplicación* representará la instancia de una aplicación en ejecución particular, pudiendo haber 10 de ellas abiertas en una misma sesión; esta clase hereda de *DictionaryBase* por lo que también posee funciones de almacenamiento sencillo.

» La clase *Pantalla* representará la instancia de una interfaz mostrada al usuario. Una aplicación puede apilar varias pantallas, mostrándose siempre la más superior y simulando de esa forma el comportamiento de *diálogos modales*. También hereda de *DictionaryBase.* Adicional a las variables que el programador codifique para su almacenamiento en este estrato, automáticamente se almacenarán los valores actuales de los controles de usuario, manteniéndose totalmente el *estado de vista* de las pantallas al apilarse unas sobre otras en una aplicación o, incluso, cambiar de aplicación. Cada pantalla referenciará la dirección url del *.aspx* que la describe, asociándose 1-a-1 con la clase *Página.*

» La clase *Página* hereda de *System.Web.UI.Page* y representará la clase base para los archivos *code-behind*, que acompañan a cada archivo *.aspx* (descriptor de interfaz). Esta clase implementa algunas propiedades muy útiles en la instancia de página (como lo serán las referencias a los estratos superiores, *Pantalla, Aplicación*, *Sesión* y *Configuraciones*), así como pre-establecer la ejecución de tareas comunes en el ciclo de vida de una interfaz de usuario dentro del marco (*CrearControles, CargarDatosControles, CargarDatosModificación,* entre otras).

Se pueden vislumbrar las ventajas del uso de este marco de desarrollo de Saetha BG; por lo tanto, el CMS tendrá como base esta plataforma.

4.5.1.1. Diagrama de paquetes SaethaWAM

Respecto al representado en la fase de inicio, se pueden notar grandes cambios y adiciones a este paquete.

En primer lugar, la adición del paquete *SaethaWAM.Configuración*, que incluirá una gran cantidad de clases que modelarán toda la configuración base del website, manteniéndose instanciado a nivel general del proceso de hospedaje y disminuyendo drásticamente el acceso a disco por concepto de lectura de archivos o de base de datos.

En segunda instancia, se puede observar la inclusión del paquete *SaethaWAM.Ejecución*, referido en la sección anterior.

También, se da la unión de los paquetes *Accesibilidad* y *Estructura* en un nuevo espacio de nombres *SaethaWAM.General*; aquí también se define a la clase *Archivo* que será de base para las cargas de imágenes, multimedia y documentos, y la clase *Categoría*.

Por último, la especificación de las clases del paquete *SaethaWAM.UI.CMS* se da de forma más específica en este diagrama.

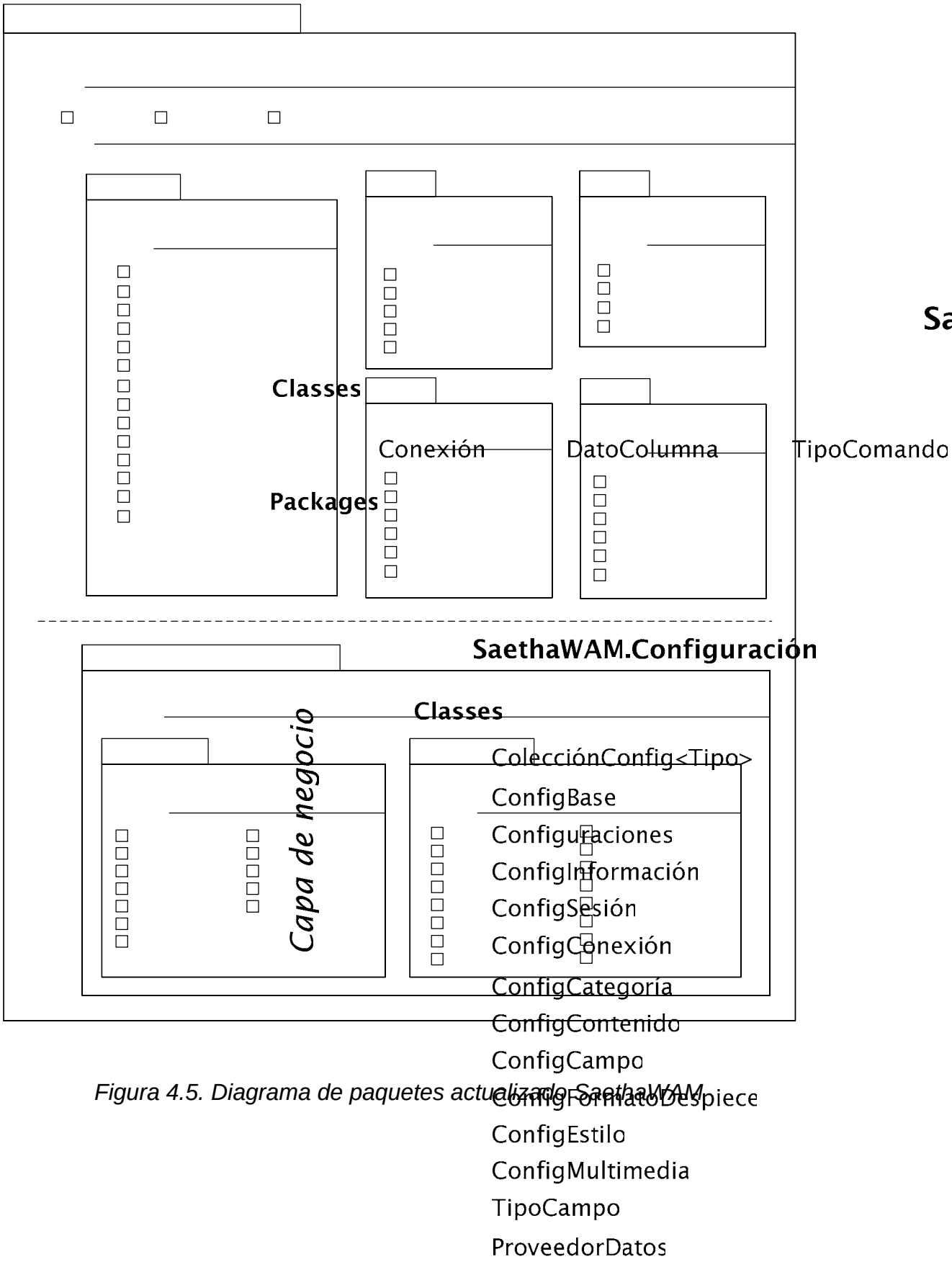


Figura 4.5. Diagrama de paquetes actualizado SaethaWAM

4.5.1.2. Clases del paquete SaethaWAM

A nivel general, dentro de este paquete sólo se encuentra la clase *Conexión* con dos entidades secundarias para su funcionamiento, *DatoColumna y TipoComando*. Al ubicarse en este paquete general, se busca hacer más asequible un recurso tan importante como el acceso a base de datos.

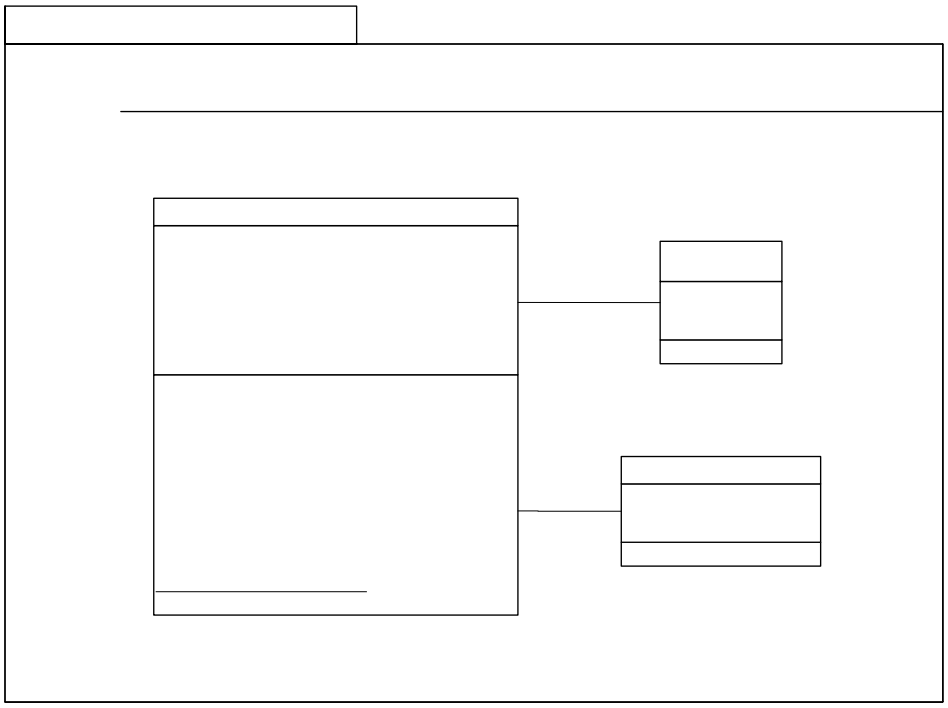


Figura 4.6. Diagrama de clases de diseño del paquete SaethaWAM

Sa

Classes

Conexión

+Comando : SqlCommand
+Lector : SqlDataReader
+Parámetros : SqlParameterCollection
+TextoComando : string
+Tipo : TipoComando
+NombreTabla : string
+Columnas : Dictionary<string,DatoCo
-Conn : SqlConnection
+Consultar(textoComando)

4.5.1.3. Clases del paquete SaethaWAM.Configuración

Las clases de este paquete modelan toda una estructura de configuraciones que será levantada en la inicialización del sistema, tomando en cuenta los parámetros establecidos en un archivo XML descriptor. La clase *Configuraciones* es el punto de partida; contiene los únicos métodos de inicialización del paquete y a partir de ahí instancia en la clase *ConfigInformación* y en sí misma el resto de las colecciones de configuración.

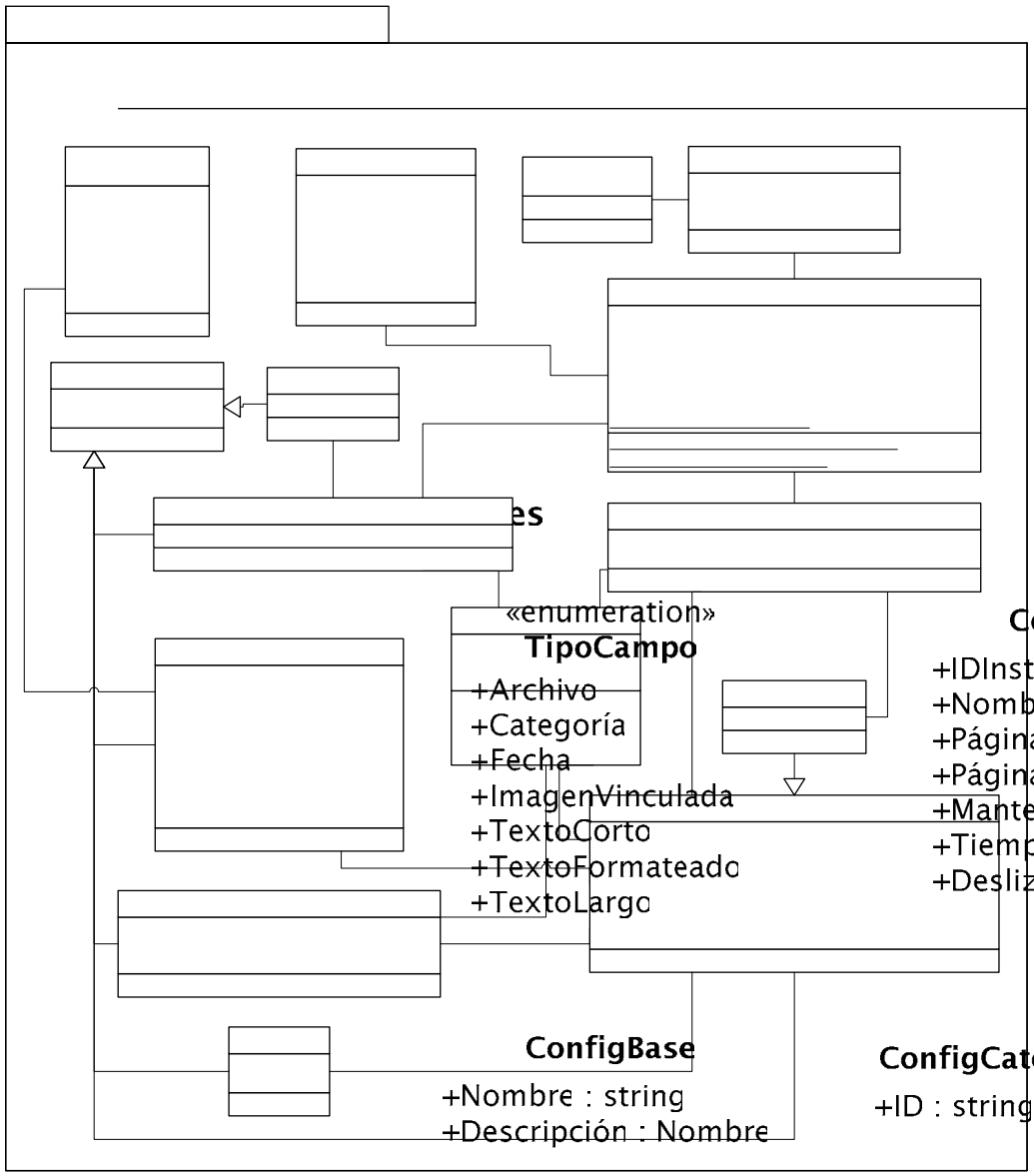


Figura 4.7. Diagrama de clases de diseño del paquete SaethaWAM.Configuración

4.5.1.4. Clases del paquete SaethaWAM.Ejecución

Este paquete es una adaptación de la plataforma de ejecución creada en Saetha BG.

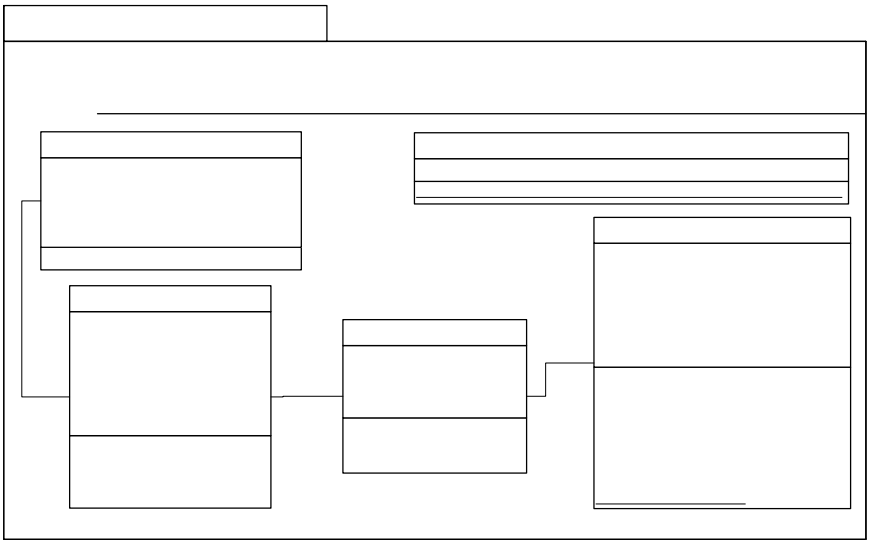


Figura 4.8. Diagrama de clases de diseño del paquete SaethaWAM.Ejecución

4.5.1.5. Clases del paquete SaethaWAM.Seguridad

Tras cambios al paquete modelado en la fase de inicio, fueron eliminadas las clases estáticas *Usuarios* y *Roles* para ubicar sus operaciones como métodos estáticos de las clases *Usuario* y *Rol*, respectivamente. Se adicionó, además, una clase estática *Acceso* que ofrece métodos para validar acceso, codificar y decodificar contraseñas, entre otras.

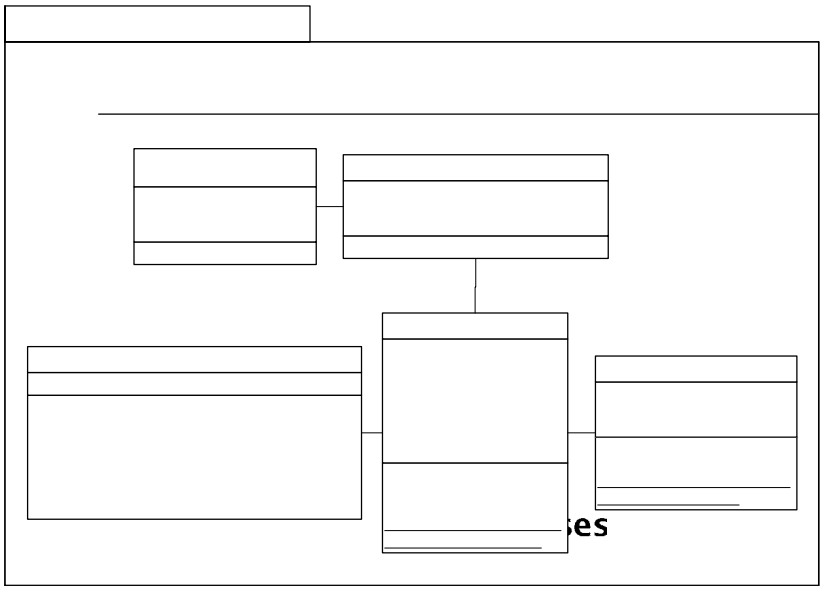


Figura 4.9. Diagrama de clases de diseño del paquete SaethaWAM.Seguridad

4.5.1.6. Clases del paquete SaethaWAM.General

En este paquete ahora se encuentran las definiciones de *Archivo* como utilitario para la carga de archivos en servidor (necesario para la inclusión de imágenes, multimedia y documentos). También, se define la clase *Categoría* que modelará la creación de listados de variables dinámicos, útiles como tipo de campo para todas las instancias de información.

Las clases *Búsqueda* (anteriormente en *Accesibilidad*) y *Sección* (anteriormente en *Estructura*) fueron movidas a este paquete, sin mayores cambios en su definición.

4.5.1.7. Clases del paquete SaethaWAM.Información

Se reestructuraron las clases principales de este paquete, centrando en la clase *InformaciónBase* la gran parte de las características de los componentes principales de información; las clases particulares para estos componentes (*ContenidoGeneral, Multimedia* y *Documento*) heredarán de *InformaciónBase*, especificando tan sólo algunas características —tales como el soporte para despieces que tiene *ContenidoGeneral*–.

Se incluye también la clase CanalRSS que genera el XML de un canal a partir de una instancia/sección de información y una cantidad máxima de notas que publicar.

4.5.1.8. Clases del paquete SaethaWAM.UI.Controles

Casi sin variaciones de estructura, las clases de este paquete están más detalladas en esta representación. Se incorporan herencias a partir de una nueva clase *Repetidor* y de la entidad *VisorContenido*.

4.5.1.9. Clases del paquete SaethaWAM.UI.CMS

Esta representación incluye un desglose mucho mayor del paquete. Cada clase representará el archivo de códigos controlador de cada interfaz de usuario. Las aplicaciones propiamente dichas mantienen una nomenclatura estándar *P#_nombreGestor_sufijo,* donde el *sufijo* puede ser “l” o “m”, de acuerdo a si es un listado de entrada o una página de modificación/registro, y el numeral agrupa las páginas que gestionan un mismo componente.

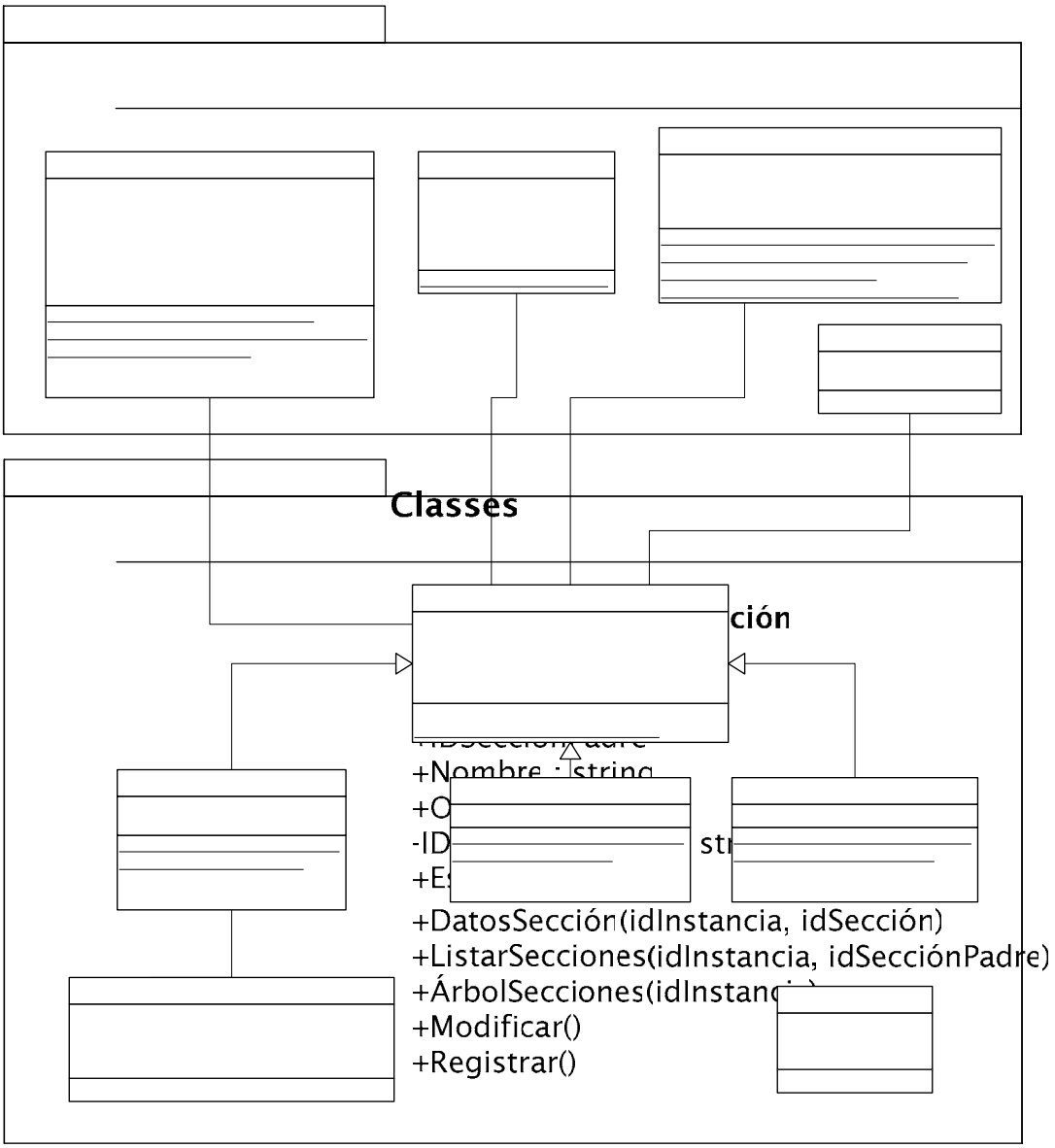


Figura 4.10. Diagrama de clases de diseño de los paquetes SaethaWAM.General y SaethaWAM.Información

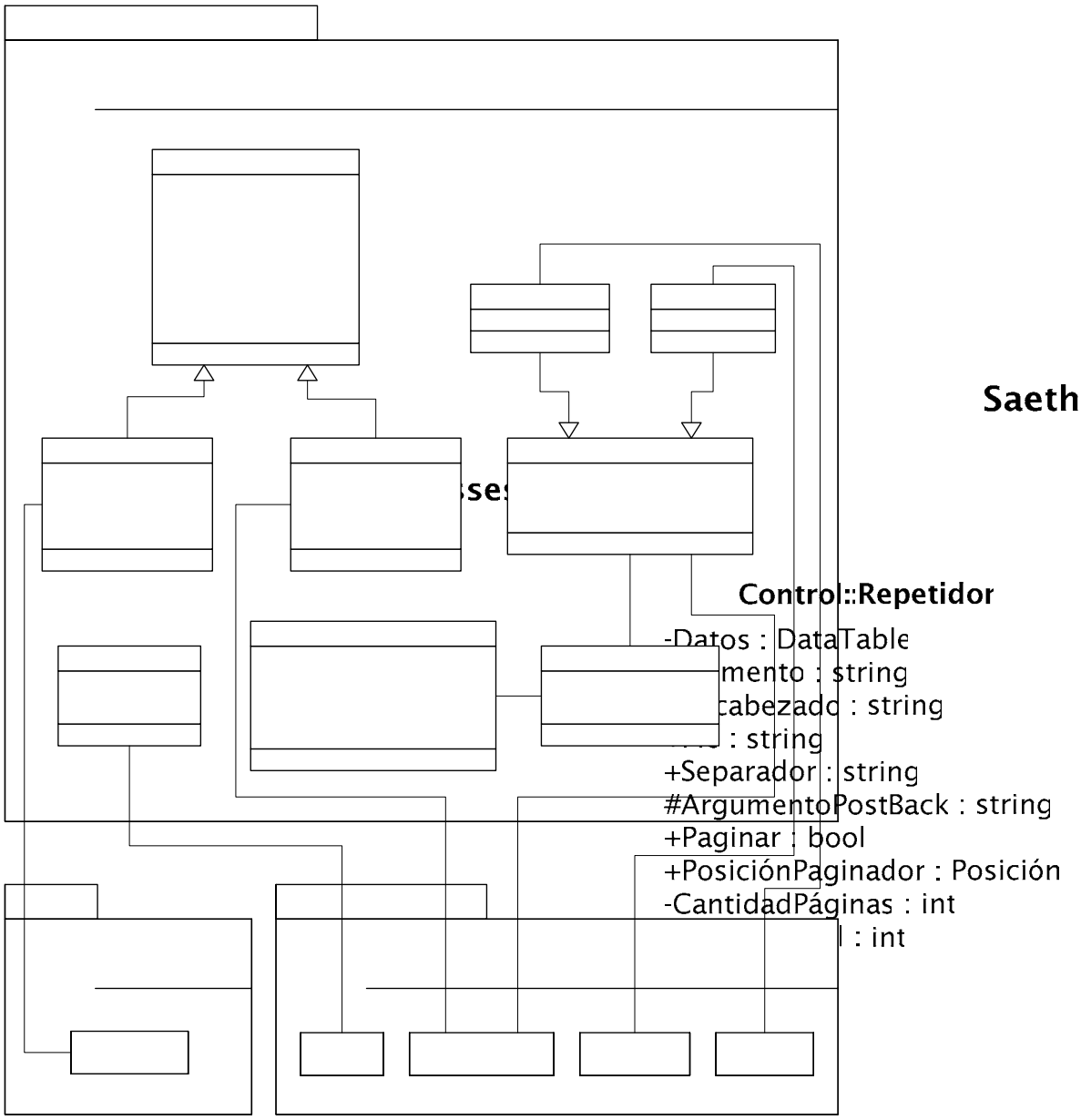


Figura 4.11. Diagrama de clases de diseño del paquete SaethaWAM.UI.Controles

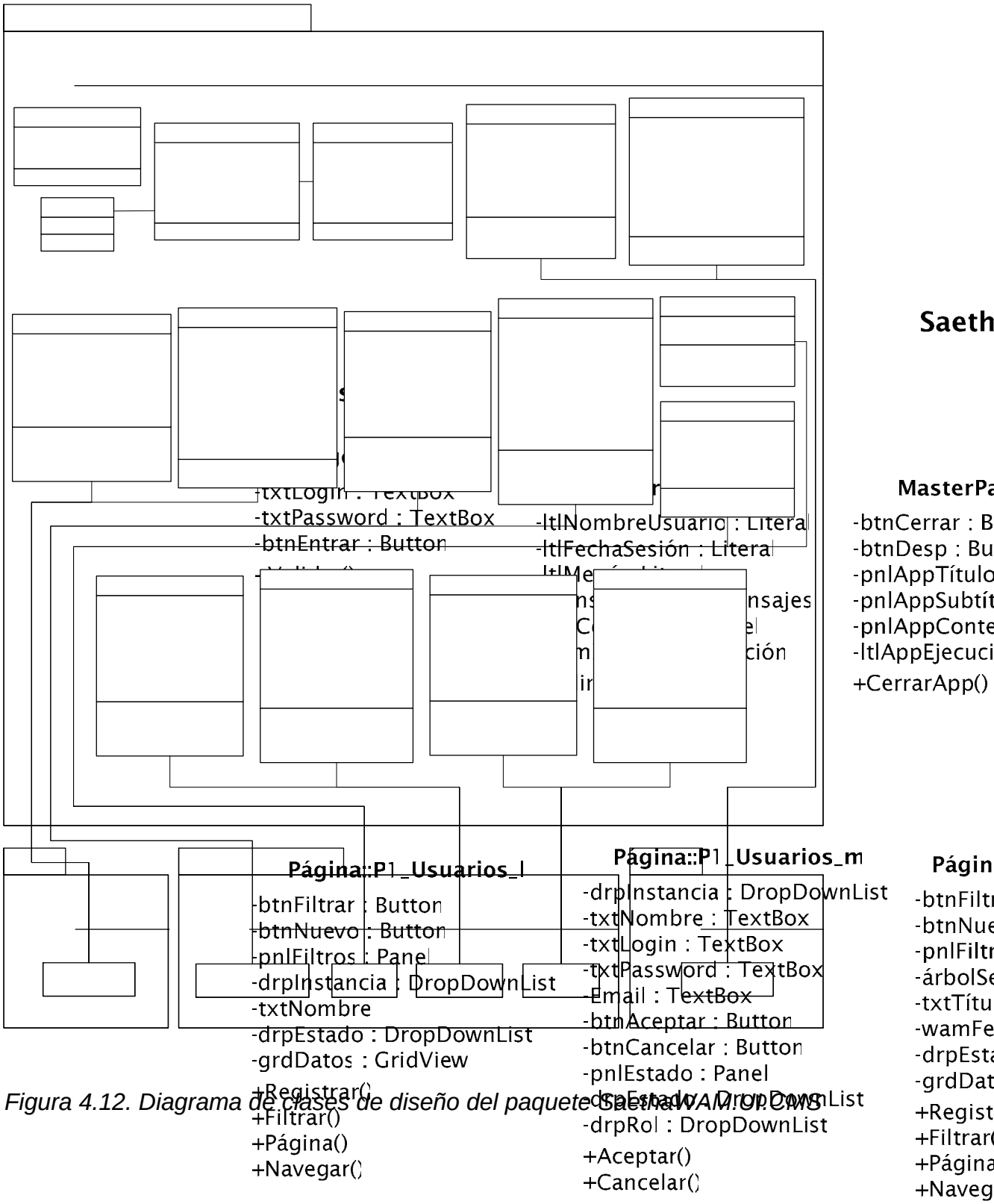


Figura 4.12. Diagrama de clases de diseño del paquete SaethaWAM.UI.CMS

4.5.2. Diagrama de capas

La arquitectura de los sistemas de cómputo suele estructurarse en especies de *estratos.* Cada uno de estos estratos, mejor llamados *capas*, incluirá paquetes funcionales relacionados con la naturaleza de la capa.

Este diagrama busca representar cómo se relacionan los distintos paquetes que conforman al sistema, estructurados en cuatro capas que serán organizadas desde la de más alto nivel hasta la de más bajo: *específica*, *general*, *intermedia* y *de software*.

Las dos primeras conformarán los paquetes desarrollados en el proyecto, diferenciadas por lo delimitado o genérico que sea el uso de cada uno de dichos paquetes; las capas intermedia y de software incluirán los paquetes producidos por terceros.

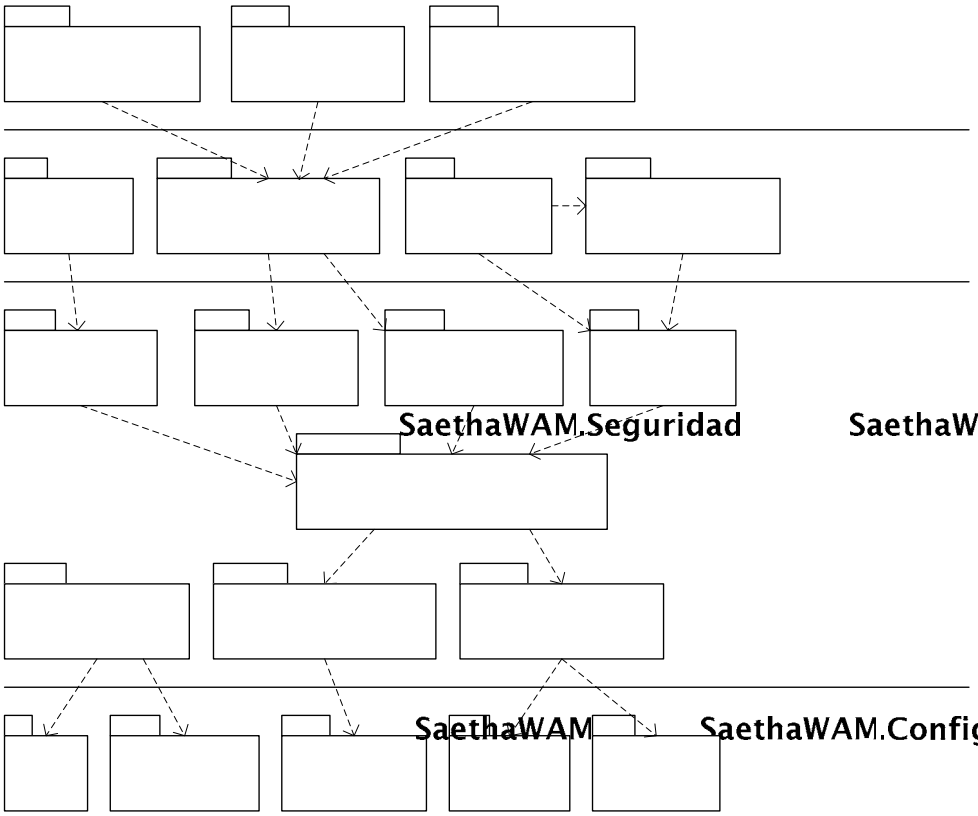


Figura 4.13. Diagrama de capas del sistema

Los paquetes incluidos en la **capa específica** no interactúan entre sí porque representan funcionalidades muy determinadas de distintos segmentos del sistema.

En la **capa general** fueron incluidos los paquetes cuyo uso es extendido en el sistema; de estos dependen los paquetes de la capa específica y se da una continua interacción entre estos estratos. SaethaWAM – uso extendido.

La **capa intermedia** incluye los paquetes más utilizados del *.NET Class Library*, que pueden ser ejecutados bajo dos entornos de ejecución: el *.NET*

Framework 3.5 CLR, entorno original de la plataforma y producido por Microsoft para sistemas basados en Windows; y *Mono 2.2 Project CLR*, proyecto de software libre patrocinado por Novell para el soporte multiplataforma de los proyectos basados en .NET. En la capa intermedia también se incluyen los distintos navegadores web que serán utilizados para la visualización e interacción con los websites y CMS; estos navegadores deben ser compatibles con XHTML 1.0.

En la **capa de software**, la de más bajo nivel, se incluyen únicamente las referencias a los sistemas operativos soportados para la ejecución a nivel de *servidor* (Windows, Linux y OS X); el protocolo de comunicación correspondiente (TCP/IP) y los sistemas operativos soportados por los navegadores web, tanto para computadores personales (distintas versiones de Windows, Linux, MacOS, OS X) como para dispositivos móviles (Symbian OS, BlackBerry OS, Palm OS, J2ME, Windows Mobile, entre otros).

4.5.3. Diagrama de base de datos

La base de datos relacional es el modelo de bases de datos actualmente más difundido, en gran medida debido a que ofrecen sistemas simples y eficaces para representar y manipular los datos.

Se basan en el modelo relacional, con sólidas bases teóricas. Entre estas bases, se da la relación a nivel de filas (tuplas o registros) y columnas (atributos o campos) dentro de una misma tabla (entidad); cada “instancia” de la entidad encontrará sitio en una tupla de la relación, mientras que los atributos de esta representarán las propiedades de la entidad. Adicionalmente, se da la relación entre entidades, cuando un atributo de una tabla hace referencia al atributo identificador de otra entidad.

Antes de realizar el diseño relacional de las tablas que conforman el sistema, se identificarán y explicarán cada una de ellas.

4.5.3.1. Identificación de las tablas

En esta etapa del diseño de la base de datos se han identificado un total de 11 tablas con las que contará el sistema. A continuación se muestran cada una de estas.

Tabla 4.2. Tablas de base de datos que conforman el sistema (parte 1/3)

TABLA (NOMBRE COMPLETO)	ENTIDAD	DESCRIPCIÓN

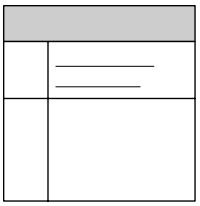
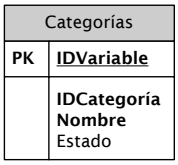
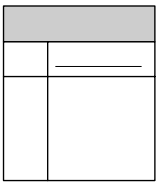
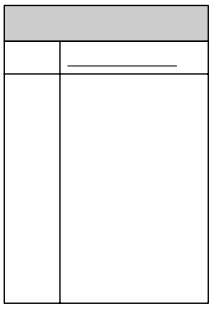
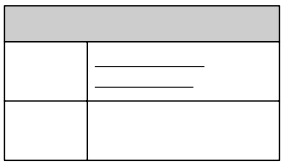
TABLA (NOMBRE COMPLETO)	ENTIDAD	DESCRIPCIÓN
Secciones		Esta tabla almacenará la estructura de secciones de todas las instancias (sean de contenido, multimedia o documentos). El campo *IDSecciónPadre* ayuda a establecer el orden jerárquico, valiendo *null* cuando es una sección raíz y almacenando el *IDSección* de otro registro cuando es "hijo" de este en el árbol.
Categorías		Aunque *IDVariable* es el único atributo marcado como clave primaria, *IDCategoría* filtrará las distintas variables que sean de una misma categoría. Los distintos valores de *IDCategoría* se configuran junto con las distintas instancias en la carga del sistema.

Tabla 4.3. Tablas de base de datos que conforman el sistema (parte 2/3)

Archivos		Todos los archivos cargados en los campos correspondientes, sean de imagen, audio, video, documentos u otros, almacenarán en esta tabla sus datos (url, tipo MIME, tamaño, entre otros).
Contenidos		En esta tabla se almacenarán todos los registros de contenido que se hagan en la vida del sistema. Los puntos suspensivos ("…") colocados en la definición de columnas indica el dinamismo con que el sistema creará nuevas columnas automáticamente para almacenar nuevos campos para las instancias, dadas las configuraciones iniciales establecidas para cada proyecto. Incluso, estos campos dinámicos pueden hacer referencia a las tablas *Categorías* y *Archivos* (representado con los campos ubicados entre los puntos suspensivos). El campo *TextoBúsqueda* estará incluido en un catálogo de búsqueda de texto completo y en él se almacenarán concatenados todos los valores de los campos de cadena que haya en cada registro, quitando las marcas de formato HTML de los textos formateados, mejorando así las capacidades de búsqueda del sistema.
Despieces de contenido		Los registros de contenido general pueden incrustar despieces en medio de los campos de texto formateado que se establezcan; estos despieces atenderán a uno de los distintos formatos establecidos en la configuración de cada proyecto y también incluirán un conjunto de campos dinámicos que se crearán automáticamente al cargar el sistema.

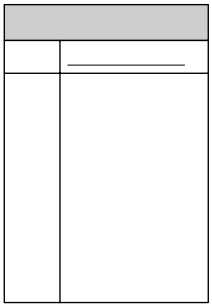
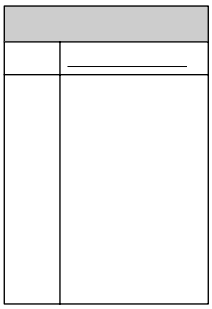
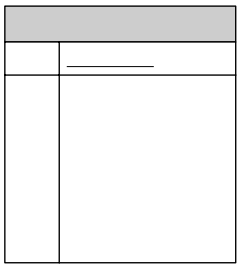
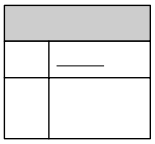
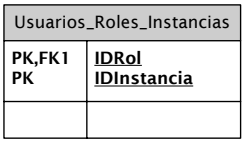
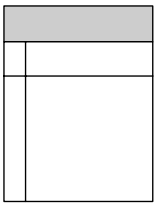
Multimedia		Aplican todas las acotaciones hechas a la entidad *Contenidos*, pero para el almacenamiento de entradas multimedia.

Tabla 4.4. Tablas de base de datos que conforman el sistema (parte 3/3)

Documentos		Aplican todas las acotaciones hechas a la entidad *Contenidos*, pero para el almacenamiento de documentos y publicaciones.
Usuarios		Aquí se almacenaran cada uno de los usuarios del sistema, tanto administradores como internautas que se registren para acceso a secciones privadas; el campo IDInstancia diferenciará las distintas listas de usuario que puedan usarse para proteger el acceso a secciones. El *IDRol* será establecido sólo si el campo *TipoAdministrador* indica que es un analista del sistema.
Roles de usuario		Se almacenarán los nombres de los distintos roles de usuario configurados para administrar los contenidos del website. El campo *Visible* determina si el rol podrá ser visualizado por un administrador cliente al configurar un usuario.

Asignación de instancias a los roles de usuario		En esta tabla se almacenarán con lógica invertida las instancias que podrán ser accedidas por los distintos roles de usuario para su administración. Si se incluye un registro que vincule al *IDRol* "x" con una *IDInstancia* "y", será porque los usuarios de rol "x" NO podrán administrar los contenidos de la instancia "y".
Registros (logs) de error en controles		Aquí se almacenarán todas las capturas de excepción ocurridas en los componentes de SaethaWAM; los detalles aquí registrados ayudarán a depurar a futuro cualquier inconveniente suscitado en la ejecución de los mismos.

4.5.3.2. Representación del diagrama relacional de base de datos

Luego de identificadas las tablas, se procedió a la elaboración del diagrama del modelo relacional.

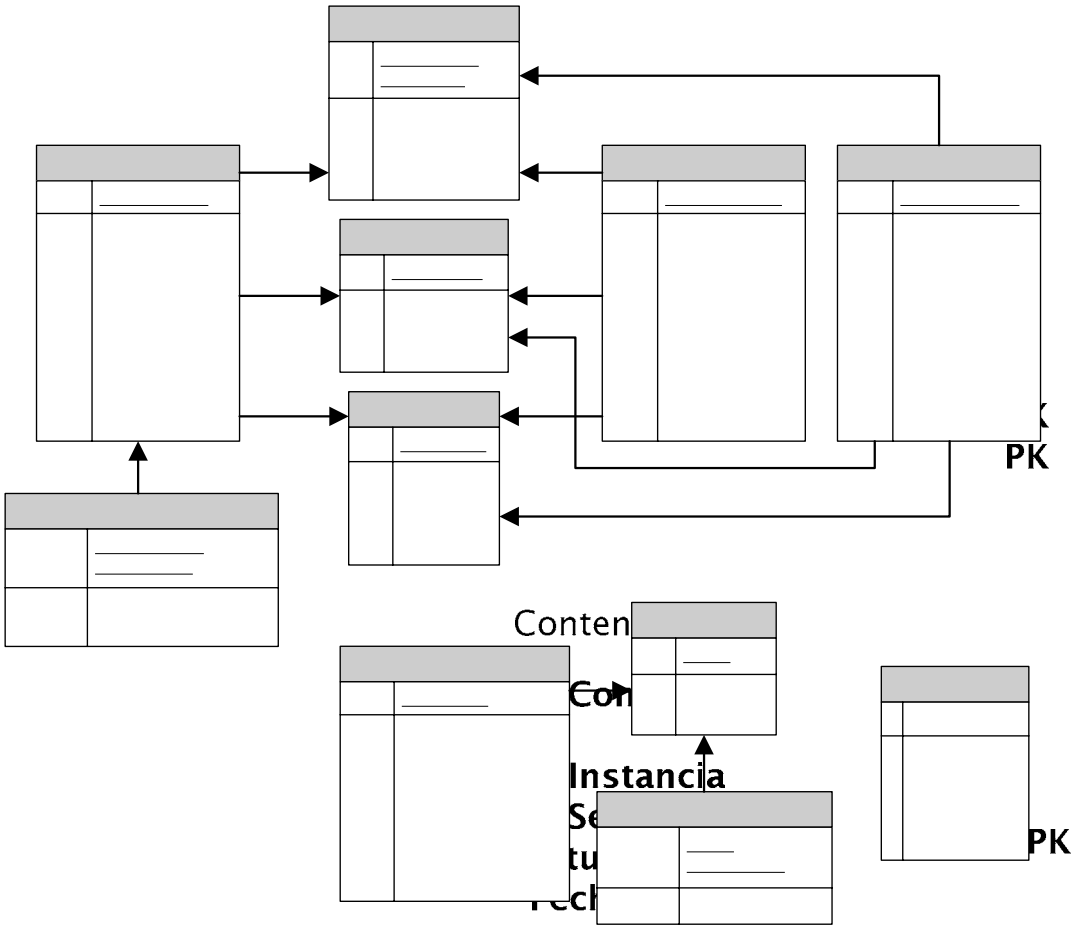


Figura 4.14. Diagrama de modelo relacional de la base de datos del sistema

4.5.4. Diseño de prototipos de interfaces de usuario

El objetivo del diseño de la interfaz es definir un conjunto de objetos y acciones de interfaz (y sus representaciones en la pantalla) que posibiliten al usuario llevar a cabo todas las tareas de administración de contenido definidas, de forma que cumplan todos los objetivos de usabilidad definidos por el sistema.

Una vez elaborados los casos de uso y definidas las operaciones que realizan, es el momento indicado para la esquematización de los prototipos de interfaces.

Dado que lo que se busca con los componentes a desarrollar es no limitar la representación gráfica de los mismos, de forma que cada website implementado pueda mantener un grado alto de originalidad en su imagen, el ambiente CMS será el único modelable en estos prototipos de interfaz de usuario.

Debido a que el entorno del sistema es Web, la ventana del explorador en el que se ejecute mostrará como titulo el nombre del sistema y la sección en la que se encuentre el usuario.

En todas las pantallas estará visible el logo del sistema (*SaethaWAM – website application manager*), el logo de Saetha BG y, opcionalmente, se podrá incluir el logo de la empresa cliente en un espacio reservado para ello.

Figura 4.15. Pantalla de autenticación de usuario para el sistema administrador de contenidos

Una vez iniciada la sesión, se visualiza la distribución estándar de la interfaz gráfica para las aplicaciones —mostrándose un mensaje de bienvenida en primera instancia– y desplegando igualmente el nombre completo del usuario autenticado, la fecha/hora de inicio de sesión y el menú de accesos.

Figura 4.16. Pantalla de bienvenida que muestra la aplicación al iniciar sesión o no tener abierta ninguna aplicación

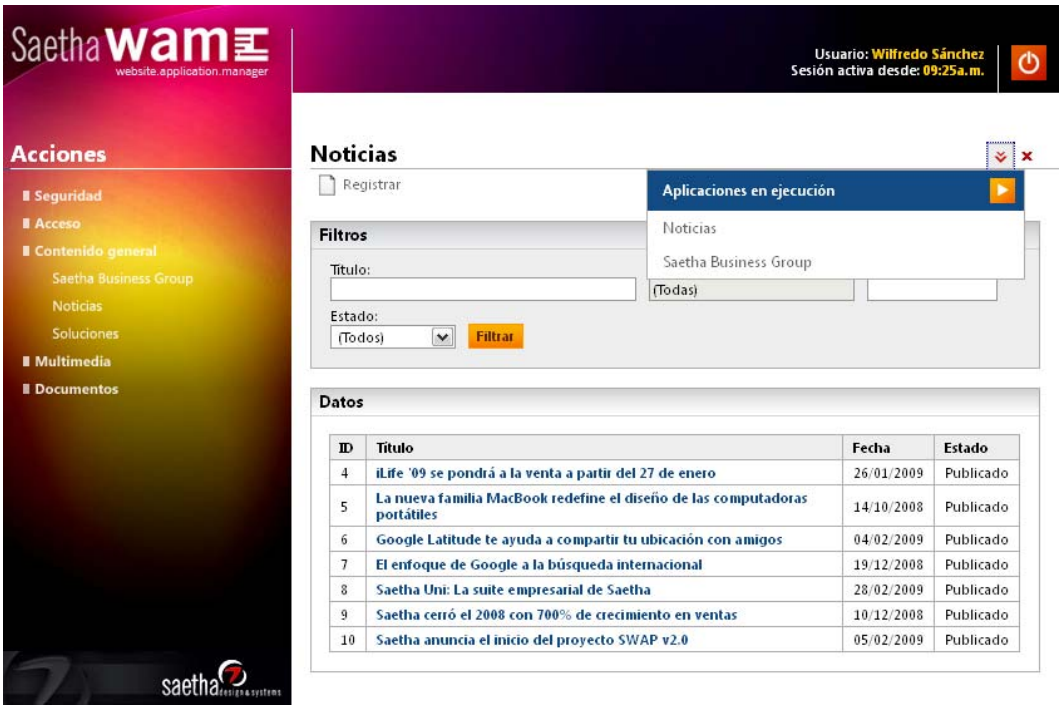


Figura 4.17. Pantalla actual de una sesión que tiene en ejecución dos aplicaciones

4.6. Implementación

Se realizó el diagrama de despliegue del sistema para mostrar su implementación y se codificaron los componentes para la configuración del proyecto web.

4.6.1. Diagrama de despliegue

Con el diagrama de despliegue se representa la arquitectura de ejecución del sistema, compuesta por nodos que pueden ser dispositivos de hardware o entornos de ejecución de software. Sobre los diferentes nodos de la

infraestructura de red se colocan, a modo de artefactos y componentes, los elementos de software.

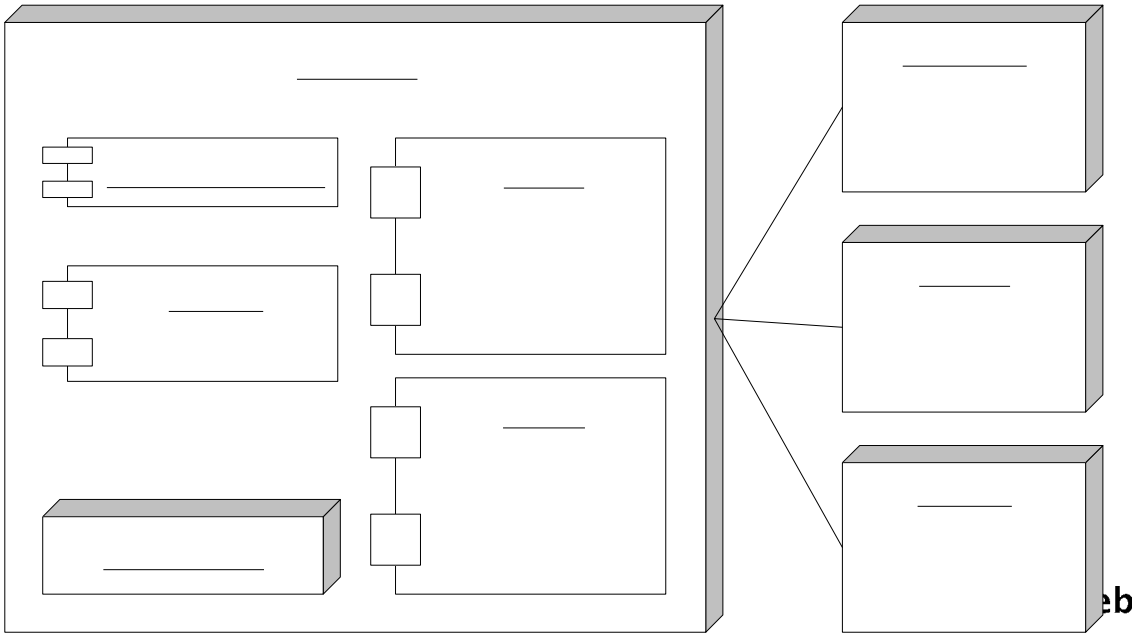


Figura 4.18. Diagrama de despliegue del sistema

4.6.1.1. Descripción de los nodos

- **Internet Information Server**: es el servidor web de Microsoft. Se ejecuta sobre plataformas Windows. Los servicios que ofrece son: FTP, SMTP, NNTP y HTTP/HTTPS.
- **.NET Framework 3.5**: es el componente fundamental de la plataforma Microsoft .NET, necesario tanto para poder desarrollar aplicaciones como para poder ejecutarlas luego en entornos de prueba o producción. Está formado por el entorno de ejecución de aplicaciones (CLR, *Common Language Runtime*), el conjunto de bibliotecas de

funcionalidad reutilizable (*.NET Framework Class Library*) y los compiladores y herramientas de desarrollo para los lenguajes .NET. Como parte del conjunto de la *.NET Framework Class Library* está ASP.NET, que constituye la tecnología dentro del .NET Framework para construir aplicaciones con interfaz de usuario web (es decir, aplicaciones cuya lógica se encuentra centralizada en uno o varios servidores y que los clientes pueden acceder usando un browser o navegador mediante una serie de protocolos y estándares como HTTP y HTML).

» **SQL Server 2005 Express Edition**: es un motor de bases de datos relacional de Microsoft, utilizado para la base de datos llamada SaethaWAM donde se almacenan los datos del sistema.

» **Website 1 y Website N**: son las aplicaciones web ASP.NET que conforman a los distintos sitios web desarrollados por Saetha para diferentes clientes. Se representaron de manera general los artefactos principales que las componen: clases, formularios web, librerías, descriptores de configuración y de aplicación.

» **PC Internauta, PC Cliente y PC Saetha**: representan los computadores con los que acceden los usuarios del sistema; es decir, los clientes y los empleados de Saetha (como usuarios de los websites y de los CMS de cada uno de ellos), y los internautas (como usuarios únicamente de la parte pública de los websites). Todos ellos requieren sólo de un navegador web para utilizarlos (por ejemplo, Internet Explorer o el popular Mozilla Firefox).

4.6.2. Diagrama de componentes

Para la fase de elaboración se ha estructurado en componentes al sistema, estableciendo las dependencias entre ellos, y se ha construido un adelanto de los componentes de configuración. Cabe recordar que el componente *Ejecución* ya había sido construido en Saetha BG y su desarrollo no forma parte como tal de este proyecto. A continuación se describen las dependencias entre los componentes generalizados del sistema, incluyendo los detalles del componente *Configuración*.

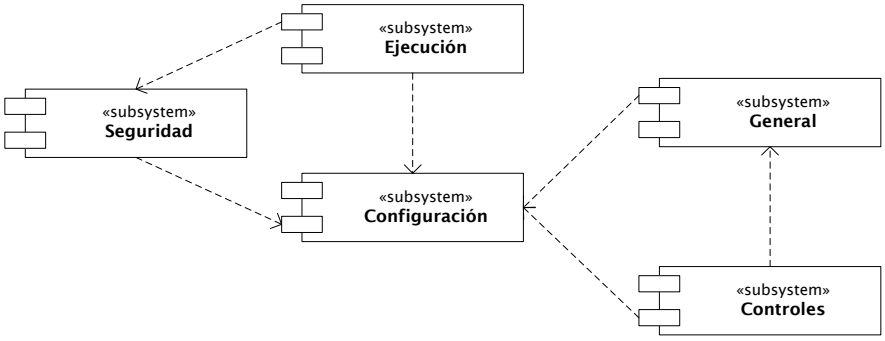


Figura 4.19. Diagrama de componentes generalizado

4.6.2.1. Componente Configuración

Está conformado a su vez por dos componentes internos: el *Descriptores de configuración* y el *Modelo de objetos de configuración*.

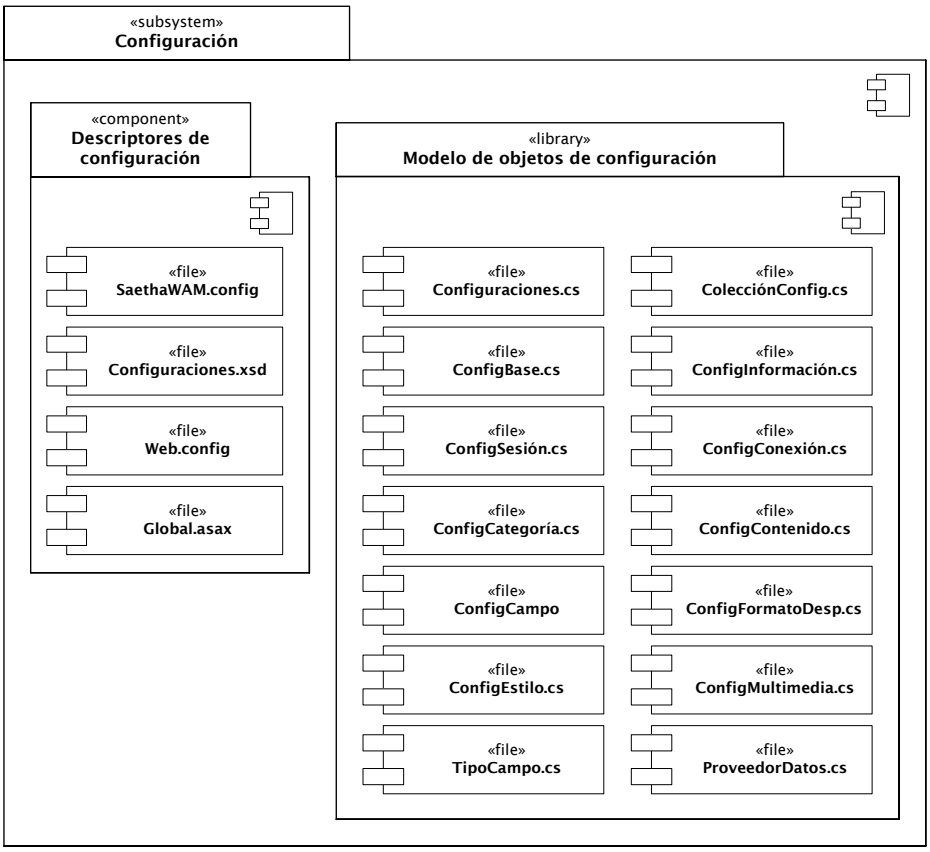


Figura 4.20. Diagrama del subsistema Configuración

» **Descriptores de configuración**: incluye el archivo *SaethaWAM.config*, un archivo XML que describe las características del sistema que serán aprovechadas en un website particular. También lo conforma el archivo *Configuraciones.xsd*, un archivo de definición de esquema XML (XSD, por sus siglas en inglés) que delimita la estructura esquemática que tendrá el archivo *SaethaWAM.config*. La existencia de este archivo de esquema es beneficioso para el analista de Saetha que incluirá las funcionalidades de SaethaWAM en un website, pues los entornos de desarrollo aprovechan este esquema para la sugerencia y validación

de etiquetas, atributos y valores. Este componente también incluye el archivo *Web.config*, que forma parte de la arquitectura de un website ASP.NET, al igual que el archivo *Global.asax*, que permite codificar los eventos generales de aplicación y sesión.

» **Modelo de objetos (clases) de configuración**: incluye todas las clases del paquete *SaethaWAM.Configuración*. Estas clases modelan en tiempo de ejecución todas las características configuradas del sistema para un website. La declaración estática del miembro *Actuales* de la clase *Configuraciones* se convierte en el punto de partida para este modelado, por lo que cada website tendrá sólo una declaración global de estas configuraciones que son tan frecuentemente accedidas desde el resto de los componentes. Esto optimiza el acceso a los datos al evitar acceso a base de datos y disco para la información estructural.

CÓDIGO DEL ARCHIVO **CONFIGURACIONES.XSD**

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.saetha.com/wam/config"
        xmlns:wam="http://www.saetha.com/wam/config"
        elementFormDefault="qualified"  xml:lang="es-es" >

   <xsd:element name="SaethaWAM">
      <xsd:annotation>
         <xsd:documentation>
            Configuraciones para la implementación de componentes del sistema SaethaWAM
         </xsd:documentation>
      </xsd:annotation>

      <xsd:complexType>
         <xsd:sequence>
            <xsd:element name="OrígenesDatos" minOccurs="1" maxOccurs="unbounded">
               <xsd:annotation>
                  <xsd:documentation>
                     Almacena la información de los orígenes de datos con que se establecerá
                     conexión en el sistema.
                  </xsd:documentation>
               </xsd:annotation>
               <xsd:complexType>
                  <xsd:sequence>
                     <xsd:element name="Conexión" minOccurs="1" maxOccurs="unbounded"
                        type="wam:Conexión" />
                  </xsd:sequence>
```

```
    </xsd:complexType>
</xsd:element>

<xsd:element name="Seguridad" minOccurs="0" maxOccurs="unbounded">
    <xsd:annotation>
        <xsd:documentation>
            Almacena la información de seguridad del sistema.
        </xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
        <xsd:sequence>
            <xsd:element name="Sesiones" minOccurs="0" maxOccurs="unbounded">
                <xsd:annotation>
                    <xsd:documentation>
                        Establece la información de distintos grupos de sesiones
                        que se necesitan autenticar para acceder a formularios del sistema.
                    </xsd:documentation>
                </xsd:annotation>
                <xsd:complexType>
                    <xsd:sequence>
                        <xsd:element name="Sesión" minOccurs="0" maxOccurs="unbounded"
                            type="wam:Sesión" />
                    </xsd:sequence>
                </xsd:complexType>
            </xsd:element>

            <xsd:element name="GruposUsuarios" minOccurs="0" maxOccurs="unbounded">
                <xsd:annotation>
                    <xsd:documentation>
                        Establece la información de distintos grupos de usuarios.
                    </xsd:documentation>
                </xsd:annotation>
                <xsd:complexType>
                    <xsd:sequence>
                        <xsd:element name="GrupoUsuarios" minOccurs="0" maxOccurs="unbounded"
                            type="wam:GrupoUsuarios" />
                    </xsd:sequence>
                </xsd:complexType>
            </xsd:element>
        </xsd:sequence>
    </xsd:complexType>
</xsd:element>

<xsd:element name="General" minOccurs="0" maxOccurs="unbounded">
    <xsd:complexType>
        <xsd:choice minOccurs="0" maxOccurs="unbounded">
            <xsd:element name="Categorías">
                <xsd:complexType>
                    <xsd:choice minOccurs="0" maxOccurs="unbounded">
                        <xsd:element name="Categoría">
                            <xsd:complexType>
                                <xsd:attribute name="IDCategoría" type="xsd:string"></xsd:attribute>
                                <xsd:attribute name="Nombre" type="xsd:string"></xsd:attribute>
                            </xsd:complexType>
                        </xsd:element>
                    </xsd:choice>
                </xsd:complexType>

            </xsd:element>
        </xsd:choice>
    </xsd:complexType>
</xsd:element>

<xsd:element name="Información" minOccurs="0" maxOccurs="1">
```

```
            <xsd:annotation>
               <xsd:documentation>
                  Agrupa las configuraciones para los componentes informativos
                  (contenido general, galerías multimedia, banners publicitarios)
               </xsd:documentation>
            </xsd:annotation>
            <xsd:complexType>
               <xsd:choice>
                  <xsd:element name="Contenido" minOccurs="0" maxOccurs="unbounded"
                     type="wam:Contenido" />
               </xsd:choice>
            </xsd:complexType>
         </xsd:element>

      </xsd:sequence>
      <xsd:attribute name="Nombre" use="required" type="xsd:string"></xsd:attribute>
      <xsd:attribute name="Versión" use="required" type="xsd:string"></xsd:attribute>
   </xsd:complexType>

</xsd:element>
<xsd:complexType name="ListaCampos">
   <xsd:annotation>
      <xsd:documentation>
         Establece los campos en que se organizará la información de esta instancia.
      </xsd:documentation>
   </xsd:annotation>
   <xsd:choice>
      <xsd:element name="Campo" minOccurs="0" maxOccurs="unbounded">
         <xsd:complexType>
            <xsd:attribute name="ID" use="required" type="xsd:string">
               <xsd:annotation>
                  <xsd:documentation>
                     Identificación del campo. Con él se identificará la información en la base de datos.
                     Se debe evitar el uso de caracteres especiales.
                  </xsd:documentation>
               </xsd:annotation>
            </xsd:attribute>
            <xsd:attribute name="Nombre" use="required" type="xsd:string">
               <xsd:annotation>
                  <xsd:documentation>
                     Nombre que identificará al campo en las interfaces gráficas de usuario (UI).
                  </xsd:documentation>
               </xsd:annotation>
            </xsd:attribute>
            <xsd:attribute name="Tipo" use="required">
               <xsd:simpleType>
                  <xsd:restriction base="xsd:string">
                     <xsd:enumeration value="TextoFormateado">
                        <xsd:annotation>
                           <xsd:documentation>
                              Aplicable si el texto excede los 4000 caracteres y poseerá
                              formato enriquecido.
                           </xsd:documentation>
                        </xsd:annotation>
                     </xsd:enumeration>
                     <xsd:enumeration value="TextoLargo">
                        <xsd:annotation>
                           <xsd:documentation>
                              Aplicable si el texto excede los 4000 caracteres.
                           </xsd:documentation>
                        </xsd:annotation>
                     </xsd:enumeration>
                     <xsd:enumeration value="TextoCorto">
                        <xsd:annotation>
```

```
                        <xsd:documentation>
                            Aplicable si el texto es igual o menor a 4000 caracteres.
                        </xsd:documentation>
                    </xsd:annotation>
                </xsd:enumeration>
                <xsd:enumeration value="Fecha">
                    <xsd:annotation>
                        <xsd:documentation>
                            Aplicable si es una fecha (sin especificar hora).
                        </xsd:documentation>
                    </xsd:annotation>
                </xsd:enumeration>
                <xsd:enumeration value="Categoría">
                    <xsd:annotation>
                        <xsd:documentation>
                            Aplicable si se quiere restringir la información a la de una categoría centralizada.
                        </xsd:documentation>
                    </xsd:annotation>
                </xsd:enumeration>
                <xsd:enumeration value="Archivo">
                    <xsd:annotation>
                        <xsd:documentation>
                            Aplicable si se requiere la carga de un archivo.
                        </xsd:documentation>
                    </xsd:annotation>
                </xsd:enumeration>
                <xsd:enumeration value="ImagenVinculada">
                    <xsd:annotation>
                        <xsd:documentation>
                            Aplicable si el campo no es para carga de un nuevo archivo de imagen
                            sino para el cambio de dimensiones de la imagen de otro campo Archivo.
                        </xsd:documentation>
                    </xsd:annotation>
                </xsd:enumeration>
            </xsd:restriction>
        </xsd:simpleType>
    </xsd:attribute>
    <xsd:attribute name="IDCategoría" use="optional" type="xsd:string">
        <xsd:annotation>
            <xsd:documentation>
                Identificación de la categoría cuyos datos restringirán los valores para este campo.
                Es aplicable sólo si el atributo 'Tipo' está establecido en 'Categoría'.
            </xsd:documentation>
        </xsd:annotation>
    </xsd:attribute>
    <xsd:attribute name="IncluirEnBúsqueda" default="true" type="xsd:boolean">
        <xsd:annotation>
            <xsd:documentation>
                Es 'true' si el contenido de este campo se utilizará al realizar
                búsquedas.
            </xsd:documentation>
        </xsd:annotation>
    </xsd:attribute>
    <xsd:attribute name="OrdenUI" use="optional" type="xsd:int">
        <xsd:annotation>
            <xsd:documentation>
                Establece un ordinal para la aparición organizada de los controles
                para los campos en las interfaces de usuario.
            </xsd:documentation>
        </xsd:annotation>
    </xsd:attribute>
    <xsd:attribute name="Opcional" use="optional" type="xsd:boolean" default="true">
        <xsd:annotation>
            <xsd:documentation>
```

```
                    Es 'true' si el campo no requiere ser rellenado para pasar la validación de registro.
                  </xsd:documentation>
                </xsd:annotation>
              </xsd:attribute>
              <xsd:attribute name="Imagen.Ancho" use="optional" type="xsd:int">
                <xsd:annotation>
                  <xsd:documentation>
                    Ancho del mapa de bits. Aplicable si el campo es de tipo Archivo (imagen) o
                    ImagenVinculada.
                  </xsd:documentation>
                </xsd:annotation>
              </xsd:attribute>
              <xsd:attribute name="Imagen.Alto" use="optional" type="xsd:int">
                <xsd:annotation>
                  <xsd:documentation>
                    Alto del mapa de bits. Aplicable si el campo es de tipo Archivo (imagen) o
                    ImagenVinculada.
                  </xsd:documentation>
                </xsd:annotation>
              </xsd:attribute>
              <xsd:attribute name="Imagen.IDCampoOriginal" use="optional" type="xsd:string">
                <xsd:annotation>
                  <xsd:documentation>
                    Campo de imagen que se tomará para crear la nueva versión redimensionada.
                    Aplicable si el tipo de campo es ImagenVinculada.
                  </xsd:documentation>
                </xsd:annotation>
              </xsd:attribute>
            </xsd:complexType>
          </xsd:element>
        </xsd:choice>
      </xsd:complexType>
      <xsd:complexType name="ÁrbolSecciones">
        <xsd:choice>
          <xsd:element name="Sección" minOccurs="0" maxOccurs="unbounded" type="wam:ÁrbolSecciones">
            <xsd:annotation>
              <xsd:documentation>
                Representa la información de una sección interna a una instancia.
              </xsd:documentation>
            </xsd:annotation>
          </xsd:element>
        </xsd:choice>
        <xsd:attribute name="ID" use="required" type="xsd:string">
          <xsd:annotation>
            <xsd:documentation>
              Cadena que identifique de forma única a la sección dentro de la instancia.
            </xsd:documentation>
          </xsd:annotation>
        </xsd:attribute>
        <xsd:attribute name="Nombre" use="required" type="xsd:string">
          <xsd:annotation>
            <xsd:documentation>
              Nombre completo de la sección, con que se identificará gráficamente en las interfaces de usuario.
            </xsd:documentation>
          </xsd:annotation>
        </xsd:attribute>
      </xsd:complexType>

      <xsd:complexType name="Conexión">
        <xsd:annotation>
          <xsd:documentation>
            Representa la información de un origen de datos.
          </xsd:documentation>
        </xsd:annotation>
```

```
    <xsd:attribute name="Nombre" type="xsd:string" use="required">
      <xsd:annotation>
        <xsd:documentation>
          Identificación del origen de datos.
        </xsd:documentation>
      </xsd:annotation>
    </xsd:attribute>

    <xsd:attribute name="Cadena" type="xsd:string" use="required">
      <xsd:annotation>
        <xsd:documentation>
          Cadena de conexión de ADO.NET para el origen de datos.
        </xsd:documentation>
      </xsd:annotation>
    </xsd:attribute>

    <xsd:attribute name="Proveedor" use="required">
      <xsd:annotation>
        <xsd:documentation>
          Especifica el tipo de manejador de base de datos que tiene el origen.
        </xsd:documentation>
      </xsd:annotation>
      <xsd:simpleType>
        <xsd:restriction base="xsd:string">
          <xsd:enumeration value="SQLServer">
            <xsd:annotation>
              <xsd:documentation>
                Servidor Microsoft SQL Server (compatible con las versiones 2000 y superior)
              </xsd:documentation>
            </xsd:annotation>
          </xsd:enumeration>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:attribute>
  </xsd:complexType>

  <xsd:complexType name="Sesión">
    <xsd:annotation>
      <xsd:documentation>
        Información para establecer un grupo de sesión autenticada de usuario.
      </xsd:documentation>
    </xsd:annotation>
    <xsd:attribute name="IDInstancia" use="required" type="xsd:ID">
      <xsd:annotation>
        <xsd:documentation>
          Identificación del grupo de sesión.
        </xsd:documentation>
      </xsd:annotation>
    </xsd:attribute>
    <xsd:attribute name="Nombre" use="required" type="xsd:string">
      <xsd:annotation>
        <xsd:documentation>
          Nombre que identificará la instancia en las interfaces de usuario.
        </xsd:documentation>
      </xsd:annotation>
    </xsd:attribute>
    <xsd:attribute name="MantenerCookie" use="optional" default="true" type="xsd:boolean">
      <xsd:annotation>
        <xsd:documentation>
          Si se establece en "true", al autenticar un usuario se almacenará una cookie
          en su equipo para mantener la información de autenticación. El valor predeterminado
          es "true".
        </xsd:documentation>
```

```
        </xsd:annotation>
    </xsd:attribute>
    <xsd:attribute name="TiempoSesión" use="optional" default="20" type="xsd:int">
        <xsd:annotation>
            <xsd:documentation>
                Si se va a mantener en cookie la autenticación, establece el tiempo en que se expirará
                el inicio de sesión.
            </xsd:documentation>
        </xsd:annotation>
    </xsd:attribute>
    <xsd:attribute name="DeslizarTiempo" use="optional" default="true" type="xsd:boolean">
        <xsd:annotation>
            <xsd:documentation>
                Si se establece en "true", tras cada interacción del usuario se reiniciará
                el conteo.
            </xsd:documentation>
        </xsd:annotation>
    </xsd:attribute>
    <xsd:attribute name="PáginaLogin" use="optional" default="~/Default.aspx" type="xsd:string">
        <xsd:annotation>
            <xsd:documentation>
                Página que se mostrará si se trata de acceder a una página de un grupo de sesión
                que aún no ha autenticado al usuario.
            </xsd:documentation>
        </xsd:annotation>
    </xsd:attribute>
    <xsd:attribute name="PáginaInicio" use="optional" default="~/Inicio.aspx" type="xsd:string">
        <xsd:annotation>
            <xsd:documentation>
                Página que se mostrará al loggear en un grupo de sesión.
            </xsd:documentation>
        </xsd:annotation>
    </xsd:attribute>
    <xsd:attribute name="IDGrupoUsuarios" use="required" type="xsd:IDREF">
        <xsd:annotation>
            <xsd:documentation>
                IDInstancia del grupo de usuarios con que se validará el inicio de
                sesión.
            </xsd:documentation>
        </xsd:annotation>
    </xsd:attribute>
</xsd:complexType>

<xsd:complexType name="GrupoUsuarios">
    <xsd:annotation>
        <xsd:documentation>
            Información del grupo de usuarios.
        </xsd:documentation>
    </xsd:annotation>
    <xsd:attribute name="IDInstancia" use="required" type="xsd:ID">
        <xsd:annotation>
            <xsd:documentation>
                Identificación del grupo de usuarios.
            </xsd:documentation>
        </xsd:annotation>
    </xsd:attribute>
    <xsd:attribute name="Nombre" use="required" type="xsd:string">
        <xsd:annotation>
            <xsd:documentation>
                Nombre que identificará la instancia en las interfaces de usuario.
            </xsd:documentation>
        </xsd:annotation>
    </xsd:attribute>
</xsd:complexType>
```

```
<xsd:complexType name="Estilos">
  <xsd:annotation>
    <xsd:documentation>
      Estilos que se podrán asignar a los distintos párrafos de los campos
      de texto formateado.
    </xsd:documentation>
  </xsd:annotation>
  <xsd:choice>
    <xsd:element name="Estilo" nillable="true" minOccurs="0" maxOccurs="unbounded">
      <xsd:annotation>
        <xsd:documentation>
          Definición de estilo para un párrafo.
        </xsd:documentation>
      </xsd:annotation>
      <xsd:complexType mixed="0">
        <xsd:attribute name="ID" type="xsd:ID" use="required">
          <xsd:annotation>
            <xsd:documentation>
              Identificación del estilo.
            </xsd:documentation>
          </xsd:annotation>
        </xsd:attribute>
        <xsd:attribute name="Nombre" type="xsd:string" use="required">
          <xsd:annotation>
            <xsd:documentation>
              Nombre con que se mostrará al usuario la identificación del estilo de párrafo.
            </xsd:documentation>
          </xsd:annotation>
        </xsd:attribute>
        <xsd:attribute name="CSS" type="xsd:string" use="required">
          <xsd:annotation>
            <xsd:documentation>
              Reglas de estilo CSS que se establecerán al párrafo.
            </xsd:documentation>
          </xsd:annotation>
        </xsd:attribute>
        <xsd:attribute name="Predeterminado" type="xsd:boolean" default="false" use="optional">
          <xsd:annotation>
            <xsd:documentation>
              Si se establece en 'true', será el estilo que mostrará de forma inicial el
              editor.
            </xsd:documentation>
          </xsd:annotation>
        </xsd:attribute>
      </xsd:complexType>
    </xsd:element>
  </xsd:choice>
</xsd:complexType>

<xsd:complexType name="Despieces">
  <xsd:annotation>
    <xsd:documentation>
      Aquí se especificarán los distintos formatos XHTML con los que se visualizarán
      los despieces
    </xsd:documentation>
  </xsd:annotation>
  <xsd:choice>
    <xsd:element name="Formato" minOccurs="0" maxOccurs="unbounded">
      <xsd:annotation>
        <xsd:documentation>
          Representa todas las propiedades de un formato de presentación de despieces.
        </xsd:documentation>
      </xsd:annotation>
```

```
        <xsd:complexType>
          <xsd:sequence>
            <xsd:element name="Campos" minOccurs="0" maxOccurs="1" type="wam:ListaCampos">
              <xsd:annotation>
                <xsd:documentation>
                  Establece los campos en que se organizará la información de este formato de despiece.
                </xsd:documentation>
              </xsd:annotation>
            </xsd:element>
            <xsd:element name="Plantilla" minOccurs="1" maxOccurs="1" type="xsd:string">
              <xsd:annotation>
                <xsd:documentation>
                  Representa la plantilla XHTML en base a la que se diagramará y visualizará el
                  despiece dentro del contenido.
                </xsd:documentation>
              </xsd:annotation>
            </xsd:element>
          </xsd:sequence>

          <xsd:attribute name="ID" use="required" type="xsd:string">
            <xsd:annotation>
              <xsd:documentation>
                Identificación del formato de despiece. Debe ser único para ésta instancia de contenido.
              </xsd:documentation>
            </xsd:annotation>
          </xsd:attribute>
          <xsd:attribute name="Descripción" use="optional" type="xsd:string">
            <xsd:annotation>
              <xsd:documentation>
                Texto descriptivo para el formato de despiece.
              </xsd:documentation>
            </xsd:annotation>
          </xsd:attribute>
        </xsd:complexType>
      </xsd:element>
    </xsd:choice>
  </xsd:complexType>

  <xsd:complexType name="Secciones">
    <xsd:annotation>
      <xsd:documentation>
        Predefine las secciones que organizarán a las instancia de contenido.
      </xsd:documentation>
    </xsd:annotation>
    <xsd:choice>
      <xsd:element name="Sección" minOccurs="0" maxOccurs="unbounded" type="wam:ÁrbolSecciones">
        <xsd:annotation>
          <xsd:documentation>
            Representa la información de una sección interna a una instancia.
          </xsd:documentation>
        </xsd:annotation>
      </xsd:element>
    </xsd:choice>
  </xsd:complexType>

  <xsd:complexType name="Contenido">
    <xsd:annotation>
      <xsd:documentation>
        Componente que muestra información general. Permite la visualización
        de datos separados por campos (título, antetítulo, sumario, contenido),
        así como elementos flotantes (fotografías, despieces) y listados de
        vínculos.
      </xsd:documentation>
    </xsd:annotation>
```

```
      <xsd:choice maxOccurs="unbounded">
        <xsd:element name="Campos" minOccurs="0" maxOccurs="1" type="wam:ListaCampos" />
        <xsd:element name="Estilos" minOccurs="0" maxOccurs="1" type="wam:Estilos" />
        <xsd:element name="Despieces" minOccurs="0" maxOccurs="1" type="wam:Despieces" />
        <xsd:element name="Accesos" minOccurs="0" maxOccurs="1"></xsd:element>
        <xsd:element name="Secciones" minOccurs="0" maxOccurs="1" type="wam:Secciones" />
      </xsd:choice>
      <xsd:attribute name="IDInstancia" use="required">
        <xsd:annotation>
          <xsd:documentation>
            Identificación de la instancia de elementos contenido.
          </xsd:documentation>
        </xsd:annotation>
        <xsd:simpleType>
          <xsd:restriction base="xsd:ID">
            <xsd:minLength value="1"></xsd:minLength>
            <xsd:maxLength value="50"></xsd:maxLength>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:attribute>
      <xsd:attribute name="Nombre" use="required" type="xsd:string">
        <xsd:annotation>
          <xsd:documentation>
            Nombre que identificará a la instancia en las interfaces con el usuario.
          </xsd:documentation>
        </xsd:annotation>
      </xsd:attribute>
      <xsd:attribute name="Descripción" default="" use="optional" type="xsd:string">
        <xsd:annotation>
          <xsd:documentation>
            Información más detallada que identificará a la instancia en las interfaces con el usuario.
          </xsd:documentation>
        </xsd:annotation>
      </xsd:attribute>
      <xsd:attribute name="IncluirCamposPorDefecto" default="true" type="xsd:boolean">
        <xsd:annotation>
          <xsd:documentation>
            Si se omite este campo o se especifica 'true', se incluirán los campos
            que por defecto se consideran para los componentes de contenido
            (antetítulo, autor, contenido). Aún omitiendo este atributo, se agregarán
            los campos obligados para la sección, título y fecha.
          </xsd:documentation>
        </xsd:annotation>
      </xsd:attribute>
      <xsd:attribute name="EstadoInicial" default="Publicado">
        <xsd:annotation>
          <xsd:documentation>
            Especifica el estado en el que nacen los contenidos registrados.
          </xsd:documentation>
        </xsd:annotation>
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:enumeration value="Publicado"></xsd:enumeration>
            <xsd:enumeration value="NoRevisado"></xsd:enumeration>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:attribute>
    </xsd:complexType>
</xsd:schema>
```

El archivo *Configuraciones.xsd* se mantendrá igual en cada website implementado; sin embargo, el contenido del archivo *SaethaWAM.config*

precisamente varía en cada implementación, de acuerdo a las instancias de controles que se quieran activar y personalizar. A continuación, se mostrará sólo un **ejemplo referencial** de cómo luciría el contenido de este archivo para un website sencillo.

EJEMPLO DE CÓDIGO DEL ARCHIVO **SAETHAWAM.CONFIG**

```
<?xml version="1.0" encoding="utf-8" ?>

<SaethaWAM Nombre="SaethaWAM" Versión="0.99" xmlns="http://www.saetha.com/wam/config">
  <OrígenesDatos>
    <Conexión Nombre="SaethaWAM" Cadena="Data Source=SDS026; Integrated Security=SSPI; Initial
Catalog=SaethaWAM;"
        Proveedor="SQLServer"></Conexión>
  </OrígenesDatos>
  <Seguridad>
    <Sesiones>
      <Sesión IDInstancia="CMS" Nombre="CMS | Sistema Administrador de Contenidos"
          MantenerCookie="true" DeslizarTiempo="true" IDGrupoUsuarios="Webmasters"
          PáginaInicio="~/CMS/Inicio.aspx" PáginaLogin="~/CMS/Default.aspx" />
    </Sesiones>
    <GruposUsuarios>
      <GrupoUsuarios IDInstancia="Webmasters" Nombre="Usuarios webmasters"/>
    </GruposUsuarios>
  </Seguridad>
  <General>
    <Categorías>
      <Categoría IDCategoría="Autor" Nombre="Autor"/>
    </Categorías>
  </General>
  <Información>
    <Contenido IDInstancia="Noticias" Nombre="Noticias" Descripción="Noticias y novedades"
IncluirCamposPorDefecto="false"
      EstadoInicial="Publicado">
      <Campos>
        <Campo ID="Título" Nombre="Título" Tipo="TextoCorto" OrdenUI="1"/>
        <Campo ID="Sumario" Nombre="Sumario" Tipo="TextoLargo" OrdenUI="2"/>
        <Campo ID="Contenido" Nombre="Contenido" Tipo="TextoFormateado" OrdenUI="3"/>
      </Campos>
      <Estilos>
        <Estilo ID="Subtítulo" Nombre="Subtítulo" CSS="font-family: Georgia; font-size: 16px; color: black;"/>
        <Estilo ID="Normal" Nombre="Nombre" CSS="font-family: Georgia; font-size: 12px; color: #808080;"
          Predeterminado="true"/>
      </Estilos>
    </Contenido>
    <Contenido IDInstancia="LaEmpresa" Nombre="Saetha Business Group" IncluirCamposPorDefecto="false"
      EstadoInicial="Publicado">
      <Campos>
        <Campo ID="Título" Nombre="Título" Tipo="TextoCorto" IncluirEnBúsqueda="true" />
        <Campo ID="Contenido" Nombre="Contenido" Tipo="TextoFormateado" IncluirEnBúsqueda="true" />
      </Campos>
    </Contenido>
    <Contenido IDInstancia="Productos" Nombre="Productos" IncluirCamposPorDefecto="false"
EstadoInicial="Publicado">
      <Campos>
        <Campo ID="Título" Nombre="Título" Tipo="TextoCorto" IncluirEnBúsqueda="true" />
        <Campo ID="Contenido" Nombre="Contenido" Tipo="TextoFormateado" IncluirEnBúsqueda="true" />
        <Campo ID="Imagen" Nombre="Imagen" Tipo="Archivo" />
```

```
      <Campo ID="ImagenPrev" Nombre="ImagenVinculada" Tipo="ImagenVinculada"
          Imagen.Ancho="300" Imagen.IDCampoOriginal="Imagen"/>
    </Campos>
    <Despieces>
     <Formato ID="Opiniones" Descripción="Caritas">
      <Campos>
       <Campo ID="Nombre" Nombre="Nombre" Tipo="TextoCorto" OrdenUI="1"/>
       <Campo ID="Cargo" Nombre="Cargo" Tipo="TextoCorto" OrdenUI="2"/>
       <Campo ID="Foto" Nombre="Foto" Tipo="Archivo" OrdenUI="3"/>
       <Campo ID="FotoPrev" Nombre="Previsualización" Tipo="ImagenVinculada"
Imagen.IDCampoOriginal="Foto"
           Imagen.Ancho="300" OrdenUI="4"/>
       <Campo ID="Opinión" Nombre="Opinión" Tipo="TextoLargo" OrdenUI="5"/>
      </Campos>
      <Plantilla>
       <![CDATA[
       <b><! Nombre ></b><br><i><! Cargo ></i><br><! Opinión >
       ]]>
      </Plantilla>
     </Formato>
    </Despieces>
   </Contenido>
  </Información>
</SaethaWAM>
```

El principal de los archivos del modelo de objetos es el archivo *Configuraciones.cs*.

CÓDIGO DEL ARCHIVO **CONFIGURACIONES.CS**

```
using System;
using System.Collections.Generic;
using System.IO;
using System.Linq;
using System.Security;
using System.Web;
using System.Xml;
using System.Xml.Linq;
using SaethaWAM.Información;


namespace SaethaWAM.Configuración
{
   /// <summary>
   /// Ubica métodos generales para la configuración del sistema
   /// </summary>
   public partial class Configuraciones
   {
      /// <summary>
      /// Construye un bloque de configuración para un sistema específico.
      /// </summary>
      /// <param name="nombreSistema">Nombre del sistema</param>
      /// <param name="versión">Versión del sistema</param>
      public Configuraciones(string nombreSistema, string versión)
      {
         this.nombreSistema = nombreSistema;
         this.versión = versión;
      }

      private string nombreSistema;
      /// <summary>
      /// Obtiene el nombre del sistema al que pertenece el bloque de configuración.
      /// </summary>
      public string NombreSistema { get { return nombreSistema; } }

      private string versión;
      /// <summary>
      /// Obtiene la versión del sistema al que pertenece el bloque de configuración.
      /// </summary>
      public string Versión { get { return versión; } }


      private ConfigInformación información = new ConfigInformación();

      /// <summary>
      /// Obtiene el bloque de configuraciones para los componentes de información.
      /// </summary>
      public ConfigInformación Información { get { return información; } }

      private ConfigGeneral general = new ConfigGeneral();
      /// <summary>
      /// Obtiene el bloque de configuraciones generales, aplicables a los distintos componentes.
```

```
/// </summary>
public ConfigGeneral General { get { return general; } }

/// <summary>
/// Obtiene la referencia a las configuraciones actuales del sistema. Ampliamente
/// útil desde toda capa del sistema (controles, clases de la capa de acceso o presentación),
/// por tratarse de una referencia estática.
/// </summary>
public static Configuraciones Actuales
{
   get
   {
      if (HttpContext.Current.Application == null ||
         HttpContext.Current.Application["Config"] == null)
      {
         throw new Exception("El sistema no se ha inicializado correctamente");
      }
      return (Configuraciones)HttpContext.Current.Application["Config"];
   }
}

private Dictionary<string, ConfigConexión> conexiones = new Dictionary<string,ConfigConexión>();
/// <summary>
/// Diccionario con las cadenas de conexión configuradas para el sistema.
/// </summary>
public Dictionary<string, ConfigConexión> Conexiones  { get { return conexiones; } }

private Dictionary<string, ConfigSesión> sesiones = new Dictionary<string, ConfigSesión>();
/// <summary>
/// Diccionario con los distintos bloques de sesión configurados para el sistema.
/// Cada bloque de sesión permitirá bloquear el acceso a algún conjunto de páginas, para así
/// controlar dentro del mismo dominio de configuración el acceso a distintos ambientes por
/// parte de distintos autores. De esta manera, si adicional al ambiente de administración CMS se
/// requiere un ambiente de administración técnica (para hacer visualización/limpieza de logs,
/// visualización de estadísticas técnicas o cosas similares) se puede configurar un bloque de sesión
/// distinto, de manera que se controle el acceso a ambos por separado, pero compartan el acceso a la
/// misma base de configuración.
/// </summary>
public Dictionary<string, ConfigSesión> Sesiones { get { return sesiones; } }
private static string urlArchivoConfig = "";
private static string directorioRaíz = "";
/// <summary>
/// Dirección física de la raíz del sistema.
/// </summary>
public static string DirectorioRaíz { get { return directorioRaíz; } }

/// <summary>
/// Realiza la carga de configuraciones del sistema de acuerdo
/// al archivo .config especificado.
/// </summary>
/// <param name="UrlArchivoConfig">Url del archivo de configuración.</param>
public static void CargarArchivoConfig(string UrlArchivoConfig)
{
   XmlDocument Config = new XmlDocument();

   if (!File.Exists(UrlArchivoConfig))
   {
      throw new ErrorAccesoDeArchivoException("El archivo de configuraciones '" +
          System.IO.Path.GetFileName(UrlArchivoConfig) + "' no existe.");
   }
   else
   {
      try
      {
```

```
            Config.Load(UrlArchivoConfig);
        }
        catch (XmlException ex)
        {
            throw new ErrorEnArchivoException("El archivo de configuraciones '" +
                System.IO.Path.GetFileName(UrlArchivoConfig) + "' no tiene un formato correcto.", ex);
        }
        catch (IOException ex)
        {
            throw new ErrorEnArchivoException("El archivo de configuraciones '" +
                System.IO.Path.GetFileName(UrlArchivoConfig) + "' no pudo ser abierto.", ex);
        }
        catch (UnauthorizedAccessException ex)
        {
            throw new ErrorAccesoDeArchivoException("El archivo de configuraciones '" +
                System.IO.Path.GetFileName(UrlArchivoConfig) + "' es de solo lectura, representa un directorio, no
es
                soportado por esta plataforma o no tiene el permiso requerido.", ex);
        }
        catch (SecurityException ex)
        {
            throw new ErrorAccesoDeArchivoException("El archivo de configuraciones '" +
                System.IO.Path.GetFileName(UrlArchivoConfig) + "' no tiene el permiso requerido.", ex);
        }
        catch (Exception ex)
        {
            throw new ErrorEnArchivoException("Hubo un problema con el archivo de configuraciones '" +
                System.IO.Path.GetFileName(UrlArchivoConfig) + "'.", ex);
        }
        directorioRaíz = Path.GetDirectoryName(UrlArchivoConfig);
        urlArchivoConfig = UrlArchivoConfig;


        try
        {
            Config.Schemas.Add("http://www.saetha.com/wam/config", directorioRaíz +
Path.DirectorySeparatorChar +
                "Esquemas" + Path.DirectorySeparatorChar + "Configuración.xsd");
            Config.Validate(new System.Xml.Schema.ValidationEventHandler(ErrorValidaciónXml),
Config.DocumentElement);
        }
        catch (Exception ex)
        {
            throw new ErrorEnArchivoException("El archivo de configuraciones '" +
                System.IO.Path.GetFileName(UrlArchivoConfig) + "' no tiene un formato correcto.", ex);
        }

        XmlReaderSettings Settings = new XmlReaderSettings();
        Settings.Schemas.Add("http://www.saetha.com/wam/config", directorioRaíz +
Path.DirectorySeparatorChar +
                "Esquemas" + Path.DirectorySeparatorChar + "Configuración.xsd");
        Settings.Schemas.Compile();
        Settings.ValidationType = ValidationType.Schema;
        Settings.ValidationFlags = System.Xml.Schema.XmlSchemaValidationFlags.ReportValidationWarnings;
        Settings.ConformanceLevel = ConformanceLevel.Document;

        XmlReader Reader = XmlReader.Create(UrlArchivoConfig, Settings);

        XElement XMLConfiguraciones = XElement.Load(Reader);


        Configuración.Configuraciones SistConfig = new Configuraciones(
            (string)XMLConfiguraciones.Attribute("Nombre"), (string)XMLConfiguraciones.Attribute("Versión"));
        HttpApplicationState Ambiente = HttpContext.Current.Application;
```

```
            Ambiente.Lock();
            Ambiente["Config"] = SistConfig;

            Ambiente.UnLock();
            XNamespace Ns = "http://www.saetha.com/wam/config";

            IEnumerable<XElement> ListaConexiones = from ItemConexiones in XMLConfiguraciones.Elements(Ns +
                "OrígenesDatos")  select ItemConexiones;
            IEnumerable<XElement> Conexiones = from ItemConexión in
ListaConexiones.Descendants<XElement>(Ns +
                "Conexión") select ItemConexión;
            foreach (XElement ItemConexión in Conexiones)
            {
                ConfigConexión Conexión = new ConfigConexión();
                Conexión.Nombre = (string)ItemConexión.Attribute("Nombre");
                Conexión.Cadena = (string)ItemConexión.Attribute("Cadena");
                Conexión.Proveedor = (ProveedorDatos)Enum.Parse(typeof(ProveedorDatos),
                    (string)ItemConexión.Attribute("Proveedor"));
                SistConfig.Conexiones.Add((string)ItemConexión.Attribute("Nombre"), Conexión);
            }

            IEnumerable<XElement> ListaSeguridad = from ItemSeguridad in XMLConfiguraciones.Elements(Ns +
"Seguridad")
                                                   select ItemSeguridad;
            IEnumerable<XElement> ListaSesiones = from ItemSesiones in
ListaSeguridad.Descendants<XElement>(Ns +
                "Sesiones") select ItemSesiones;
            IEnumerable<XElement> Sesiones = from ItemSesión in ListaSesiones.Descendants<XElement>(Ns +
"Sesión")
                                             select ItemSesión;
            foreach (XElement ItemSesión in Sesiones)
            {
                ConfigSesión Sesión = new ConfigSesión();
                Sesión.IDInstancia = (string)ItemSesión.Attribute("IDInstancia");
                Sesión.MantenerCookie = (bool)ItemSesión.Attribute("MantenerCookie");
                Sesión.TiempoSesión = (int)ItemSesión.Attribute("TiempoSesión");
                Sesión.DeslizarTiempo = (bool)ItemSesión.Attribute("DeslizarTiempo");
                Sesión.PáginaLogin = (string)ItemSesión.Attribute("PáginaLogin");
                Sesión.PáginaInicio = (string)ItemSesión.Attribute("PáginaInicio");
                Sesión.Nombre = (string)ItemSesión.Attribute("Nombre");
                SistConfig.Sesiones.Add((string)ItemSesión.Attribute("IDInstancia"), Sesión);
            }

            IEnumerable<XElement> Categorías = from ItemCategoría in XMLConfiguraciones.Elements(Ns +
                "General").Elements<XElement>(Ns + "Categorías").Elements<XElement>(Ns + "Categoría")
                select ItemCategoría;
            foreach (XElement ItemCategoría in Categorías)
            {
                ConfigCategoría Categoría = new ConfigCategoría();
                Categoría.ID = (string)ItemCategoría.Attribute("IDCategoría");
                Categoría.Nombre = (string)ItemCategoría.Attribute("Nombre");
                SistConfig.General.Categorías.Add((string)ItemCategoría.Attribute("IDCategoría"), Categoría);
            }

            IEnumerable<XElement> Informaciones = from ItemInformación in XMLConfiguraciones.Elements(Ns +
                "Información") select ItemInformación;
            foreach (XElement ItemInformación in Informaciones)
            {
                ConfigurarInformación(ItemInformación);
            }
        }
    }

    /// <summary>
```

```
/// Carga el modelo de objetos de configuración para los componentes
/// de información.
/// </summary>
/// <param name="Reader">XmlReader que contiene el subárbol de
/// configuraciones del archivo de configuración.</param>
private static void ConfigurarInformación(XElement Elemento)
{
   Configuraciones Config = Configuraciones.Actuales;
   XNamespace Ns = Elemento.GetDefaultNamespace();
   IEnumerable<XElement> Contenidos = from Contenido in Elemento.Elements(Ns + "Contenido")
                                      select Contenido;
   foreach (XElement ItemContenido in Contenidos)
   {
      string IDInstancia = (string)ItemContenido.Attribute("IDInstancia");
      ConfigContenido Contenido = new ConfigContenido();
      Contenido.IDInstancia = IDInstancia;
      Contenido.Nombre = (string)ItemContenido.Attribute("Nombre");
      Contenido.Descripción = (string)ItemContenido.Attribute("Descripción");
      Contenido.IncluirCamposPorDefecto = (bool)ItemContenido.Attribute("IncluirCamposPorDefecto");
      Contenido.EstadoInicial = (EstadoContenido)Enum.Parse(typeof(EstadoContenido),
            (string)ItemContenido.Attribute("EstadoInicial"));
      Config.Información.Contenidos[IDInstancia] = Contenido;

      ConfigCampo Campo;
      int OrdenUI = 1;

      Campo = new ConfigCampo();
      Campo.ID = "Título";
      Campo.Nombre = "Título";
      Campo.Tipo = TipoCampo.TextoCorto;
      Campo.IncluirEnBúsqueda = true;
      Campo.OrdenUI = OrdenUI++;
      Campo.Opcional = false;
      Contenido.Campos[Campo.ID] = Campo;

      Campo = new ConfigCampo();
      Campo.ID = "Fecha";
      Campo.Nombre = "Fecha";
      Campo.Tipo = TipoCampo.Fecha;
      Campo.IncluirEnBúsqueda = true;
      Campo.OrdenUI = OrdenUI++;
      Campo.Opcional = false;
      Contenido.Campos[Campo.ID] = Campo;

      if (Contenido.IncluirCamposPorDefecto)
      {
         Campo = new ConfigCampo();
         Campo.ID = "Antetítulo";
         Campo.Nombre = "Antetítulo";
         Campo.Tipo = TipoCampo.TextoCorto;
         Campo.IncluirEnBúsqueda = true;
         Campo.OrdenUI = OrdenUI++;
         Contenido.Campos["Antetítulo"] = Campo;

         Campo = new ConfigCampo();
         Campo.ID = "Contenido";
         Campo.Nombre = "Contenido";
         Campo.Tipo = TipoCampo.TextoFormateado;
         Campo.IncluirEnBúsqueda = false;
         Campo.OrdenUI = OrdenUI++;
         Campo.Opcional = false;
         Contenido.Campos["Contenido"] = Campo;
      }
```

```
            IEnumerable<XElement> ListaCampos = from ItemCampos in ItemContenido.Elements(Ns + "Campos")
                                                select ItemCampos;
            IEnumerable<XElement> Campos = from ItemCampo in ListaCampos.Descendants<XElement>(Ns +
"Campo")
                                           select ItemCampo;
            foreach (XElement ItemCampo in Campos)
            {
               string IDCampo = (string)ItemCampo.Attribute("ID");
               Campo = new ConfigCampo();
               Campo.ID = IDCampo;
               Campo.Nombre = (string)ItemCampo.Attribute("Nombre");
               Campo.Tipo = (TipoCampo)Enum.Parse(typeof(TipoCampo), (string)ItemCampo.Attribute("Tipo"));
               Campo.IncluirEnBúsqueda = (bool)ItemCampo.Attribute("IncluirEnBúsqueda");
               Campo.OrdenUI = OrdenUI++;
               if (Campo.Tipo == TipoCampo.Categoría)
               {
                  Campo.IDCategoría = (string)ItemCampo.Attribute("IDCategoría");
               }
               if (Campo.Tipo == TipoCampo.ImagenVinculada)
               {
                  Campo.ImagenIDCampoOriginal = (string)ItemCampo.Attribute("Imagen.IDCampoOriginal");
               }
               if ((Campo.Tipo == TipoCampo.Archivo || Campo.Tipo == TipoCampo.ImagenVinculada) &&
                  ItemCampo.Attribute("Imagen.Ancho") != null)
               {
                  Campo.ImagenAncho = (int)ItemCampo.Attribute("Imagen.Ancho");
               }
               if ((Campo.Tipo == TipoCampo.Archivo || Campo.Tipo == TipoCampo.ImagenVinculada) &&
                  ItemCampo.Attribute("Imagen.Alto") != null)
               {
                  Campo.ImagenAlto = (int)ItemCampo.Attribute("Imagen.Alto");
               }
               Campo.Opcional = (bool)ItemCampo.Attribute("Opcional");
               Contenido.Campos[IDCampo] = Campo;
            }

            IEnumerable<XElement> ListaEstilos = from ItemEstilos in ItemContenido.Elements(Ns + "Estilos")
                                                 select ItemEstilos;
            IEnumerable<XElement> Estilos = from ItemEstilo in ListaEstilos.Descendants<XElement>(Ns + "Estilo")
                                            select ItemEstilo;
            foreach (XElement ItemEstilo in Estilos)
            {
               string IDEstilo = (string)ItemEstilo.Attribute("ID");
               ConfigEstilo Estilo = new ConfigEstilo();
               Estilo.ID = IDEstilo;
               Estilo.Nombre = (string)ItemEstilo.Attribute("Nombre");
               Estilo.CSS = (string)ItemEstilo.Attribute("CSS");
               if ((bool)ItemEstilo.Attribute("Predeterminado"))
               {
                  Contenido.EstiloPorDefecto = Estilo.CSS;
               }
               Contenido.Estilos[IDEstilo] = Estilo;
            }

            IEnumerable<XElement> ListaDespieces = from ItemDespieces in ItemContenido.Elements(Ns +
"Despieces")
                                                   select ItemDespieces;
            IEnumerable<XElement> Formatos = from ItemFormato in ListaDespieces.Descendants<XElement>(Ns +
"Formato")
                                             select ItemFormato;
            foreach (XElement ItemFormato in Formatos)
            {
               string IDDespiece = (string)ItemFormato.Attribute("ID");
               ConfigFormatoDespiece FormatoDespiece = new ConfigFormatoDespiece();
```

```
                FormatoDespiece.ID = IDDespiece;
                FormatoDespiece.Descripción = (string)ItemFormato.Attribute("Descripción");
                Contenido.Despieces[IDDespiece] = FormatoDespiece;

                ListaCampos = from ItemCampos in ItemFormato.Elements(Ns + "Campos")
                              select ItemCampos;
                Campos = from ItemCampo in ListaCampos.Descendants<XElement>(Ns + "Campo")
                         select ItemCampo;

                OrdenUI = 1;
                foreach (XElement ItemCampo in Campos)
                {
                    string IDCampo = (string)ItemCampo.Attribute("ID");
                    Campo = new ConfigCampo();
                    Campo.ID = IDCampo;
                    Campo.Nombre = (string)ItemCampo.Attribute("Nombre");
                    Campo.Tipo = (TipoCampo)Enum.Parse(typeof(TipoCampo), (string)ItemCampo.Attribute("Tipo"));
                    Campo.IncluirEnBúsqueda = (bool)ItemCampo.Attribute("IncluirEnBúsqueda");
                    Campo.Opcional = (bool)ItemCampo.Attribute("Opcional");
                    Campo.OrdenUI = OrdenUI++;
                    if (Campo.Tipo == TipoCampo.Categoría)
                    {
                        Campo.IDCategoría = (string)ItemCampo.Attribute("IDCategoría");
                    }
                    if (Campo.Tipo == TipoCampo.ImagenVinculada)
                    {
                        Campo.ImagenIDCampoOriginal = (string)ItemCampo.Attribute("Imagen.IDCampoOriginal");
                    }
                    if ((Campo.Tipo == TipoCampo.Archivo || Campo.Tipo == TipoCampo.ImagenVinculada) &&
                        ItemCampo.Attribute("Imagen.Ancho") != null)
                    {
                        Campo.ImagenAncho = (int)ItemCampo.Attribute("Imagen.Ancho");
                    }
                    if ((Campo.Tipo == TipoCampo.Archivo || Campo.Tipo == TipoCampo.ImagenVinculada) &&
                        ItemCampo.Attribute("Imagen.Alto") != null)
                    {
                        Campo.ImagenAlto = (int)ItemCampo.Attribute("Imagen.Alto");
                    }
                    FormatoDespiece.Campos[IDCampo] = Campo;
                }

                FormatoDespiece.Plantilla = (string)ItemFormato.Element("Plantilla");
            }
        }
    }

    /// <summary>
    /// Función que se llama cuando hay error en la validación del esquema del archivo XML de configuración.
    /// </summary>
    public static void ErrorValidaciónXml(object sender, System.Xml.Schema.ValidationEventArgs e)
    {
        throw new ErrorEnArchivoException("El archivo de configuraciones '" +
                System.IO.Path.GetFileName(urlArchivoConfig) + "' no tiene un formato correcto.\n" + e.Message);
    }
  }
}
```

4.7. Evaluación de la fase de elaboración

En esta fase del proceso unificado se alcanzó un alto nivel de detalle en el análisis de requerimientos y diseño del sistema, así como un adelanto en la implementación, lográndose esto mediante:

- La determinación de un requisito adicional del sistema, estableciendo de forma definitiva el alcance del sistema.

- La adición de un caso de uso para representar el nuevo requisito, actualizando el diagrama de casos de uso correspondiente.

- El desarrollo de la iteración de análisis, en la que se realizó el diagrama de actividad para representar los flujos de ejecución del nuevo requisito.

- Realización de la iteración de diseño, en la que se especificaron los diagramas de paquetes y clases definitivos, así como la representación de la arquitectura del software en el diagrama de capas. Se realizó el diagrama de base de datos, detallando cada tabla junto con sus relaciones. También, se modelaron prototipos de interfaces de usuario.

- Una primera iteración de implementación. Se elaboró un diagrama de despliegue que muestra los dispositivos en que serían implementados los distintos componentes del sistema en la arquitectura cliente-servidor. También se realizó un diagrama generalizado de componentes, detallando sólo el subsistema *Configuración*, incluyendo la codificación de los archivos que forman parte de su definición y que garantizan el funcionamiento del mismo.

CAPÍTULO 5
FASE DE CONSTRUCCIÓN

5.1. Introducción

La finalidad principal de la fase de construcción es alcanzar la capacidad operacional del producto. Durante esta fase todos los componentes, características y requisitos identificados y detallados en las fases anteriores deben ser implementados, integrados y probados en su totalidad, obteniendo una versión aceptable del software.

En esta fase se hace hincapié en las iteraciones de implementación y prueba del flujo de trabajo normal, debido a que su meta es lograr el desarrollo del sistema con calidad de producción, mediante la implementación de toda la funcionalidad y realización de las pruebas.

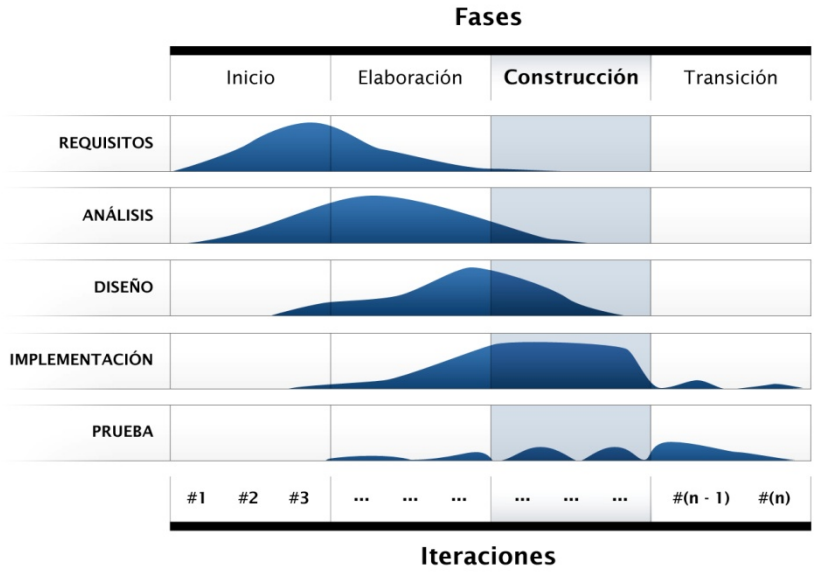


Figura 5.1. Diagrama de flujo de trabajo de las fases del proceso unificado, con identificación de la fase de construcción

5.2. Planificación de la fase de construcción

Para obtener la versión funcional o beta del sistema, se codificarán durante esta fase todas las secciones según el diseño obtenido en las fases anteriores.

En la iteración de implementación se diagramarán todos los componentes del software, realizándolos a un alto nivel de detalle y mostrando la codificación e interfaz gráfica de los principales componentes. Finalmente, se describirán las pruebas por unidad y las pruebas de integración durante la iteración de prueba.

5.3. Implementación

Durante la implementación se codificó la totalidad de secciones diseñadas para el sistema; sin embargo, por la gran cantidad de secciones y la extensión en páginas que se necesitaría, serán explicadas a detalle sólo las secciones relacionadas con *Contenido general*.

La explicación se basa en los componentes. Como punto de partida, se presenta un diagrama general de componentes en que se describen las dependencias de alto nivel entre subsistemas, seguido de otros diagramas que detallan la estructura de cada uno de estos.

Seguidamente, para los componentes relacionados con *Contenido general* se indicarán las clases que necesita, el código fuente y las interfaces gráficas finales de usuario relacionadas.

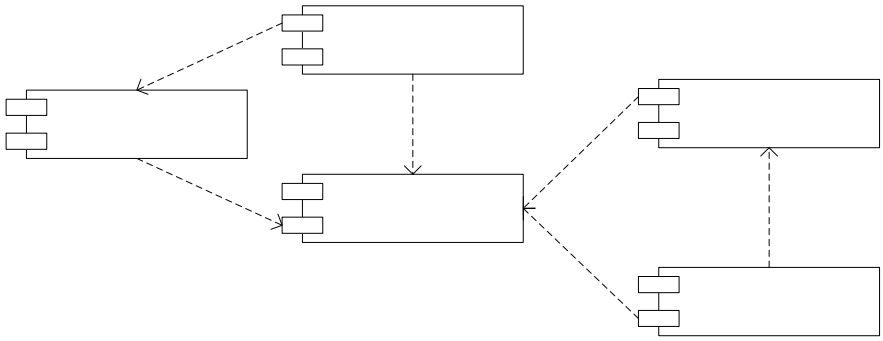


Figura 5.2. Diagrama de componentes generalizado

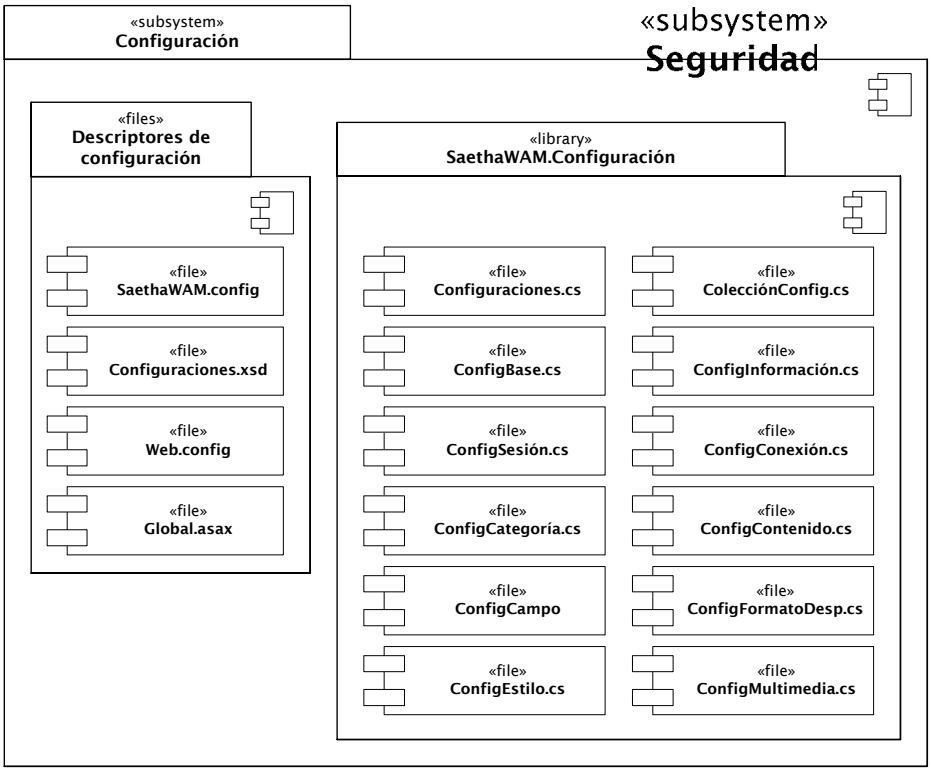
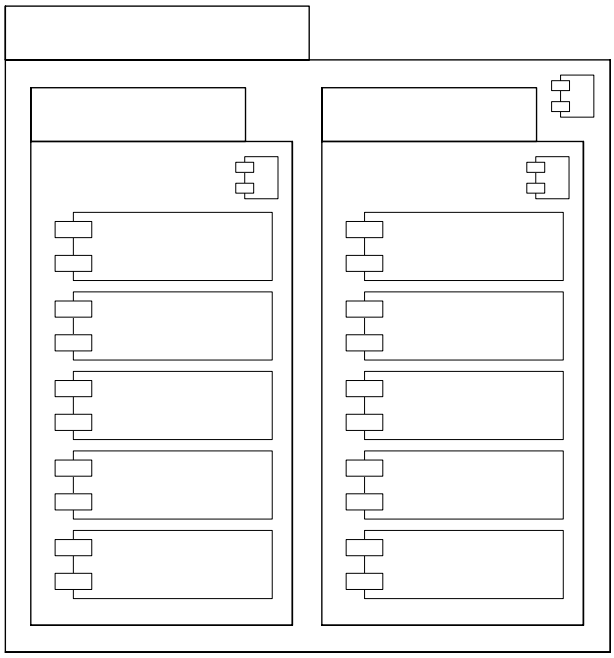


Figura 5.3. Diagrama de componentes del subsistema Configuración

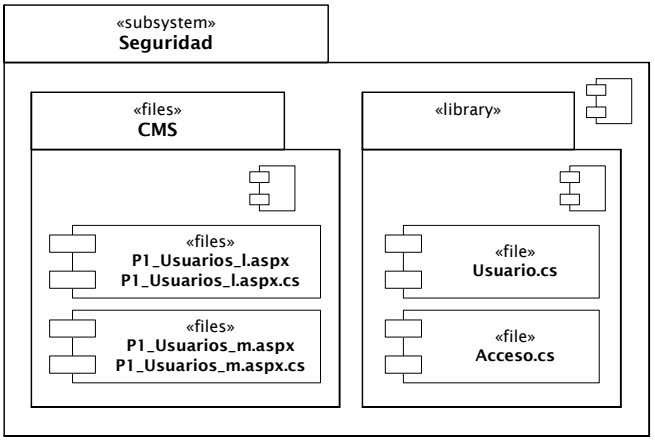


«subsystem»
Ejecución

«files»
CMS

Figura 5.4. Diagrama de componentes del subsistema Ejecución

«files»
Raiz.master
Raíz.master.cs



«files»
App.master
App.master.cs

«files»
Inicio.aspx
Inicio.aspx.cs

«files»
Archivos de estilos
CSS

Figura 5.5. Diagrama de componentes del subsistema Seguridad

«files»
Archivos de
imagen

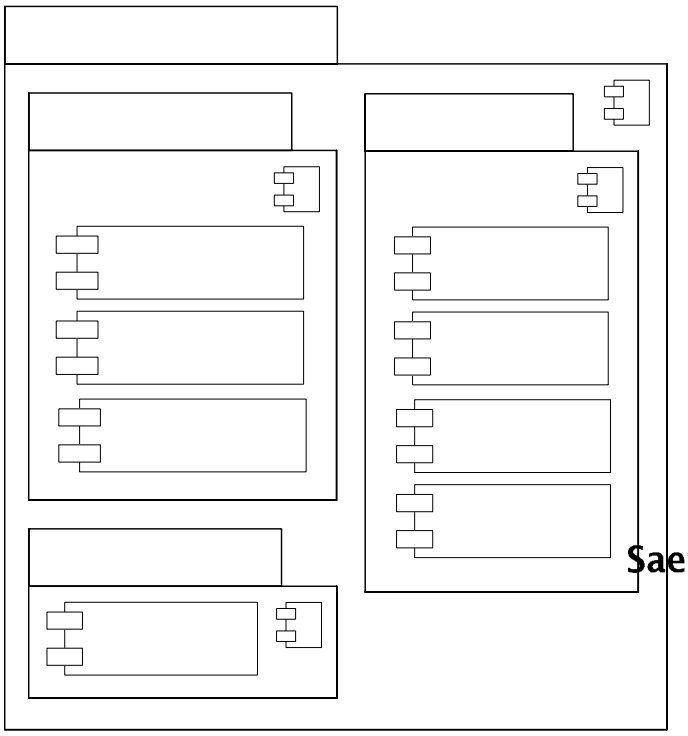


Figura 5.6. Diagrama de componentes del subsistema General

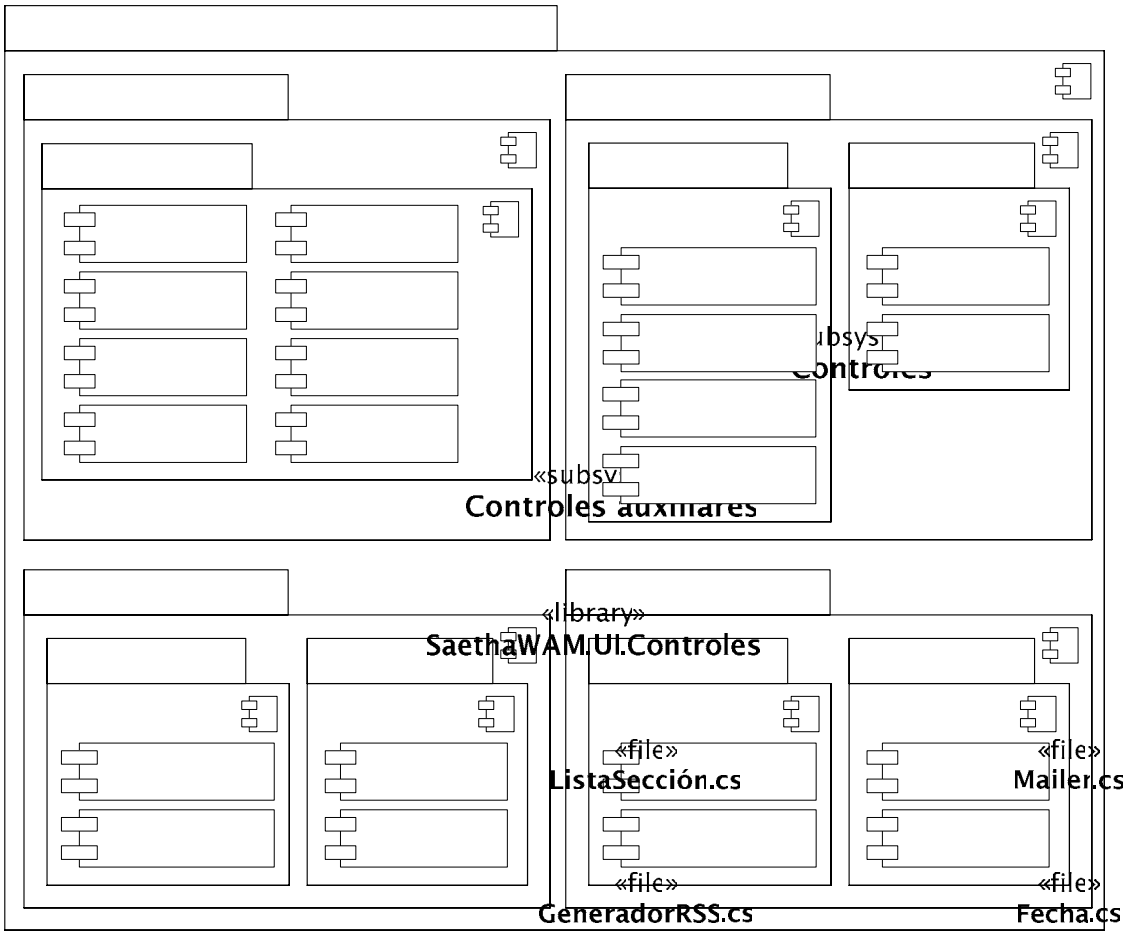


Figura 5.7. Diagrama de componentes del subsistema Controles

5.3.1. Implementación de funcionalidades de contenido general

Los subsistemas *Ejecución* y *Seguridad* ofrecen un marco o *framework* para la puesta en ejecución del sistema, controlando variables que son generales. Por su parte, el subsistema *Configuración* gana relevancia al incluir dentro de sí el modelo de cómo está configurada una instancia de este sistema y, por consiguiente, será requerido por casi la totalidad de controles. Todas estas configuraciones, sin embargo, se levantan con el uso de los descriptores sólo

cada vez que se inicia o reinicia la aplicación web (a nivel del servidor) y nunca genera interacción directa con el usuario en alguna interfaz gráfica.

Directamente relacionado con los controles, específicamente con el de *Contenido General*, se encuentran codificados los subsistemas *General* y *Controles*. Serán detallados los principales componentes/artefactos de estos subsistemas que están enlazados con la visualización de las funcionalidades de contenido general.

5.3.1.1. Subsistema General

En este subsistema se define lo que sería el *modelo de negocio*, ofreciendo abstracción para las propiedades y operaciones que le dan forma a cada funcionalidad. En sí mismos, no incluyen una interfaz gráfica para el usuario, sino que se constituyen en una pieza de código reutilizable que puede llevar a cabo las operaciones medulares y ser aprovechado por una cantidad variable de componentes de visualización e interfaz. Únicamente en la librería *SaethaWAM.Información* conseguimos las clases de este subsistema que están directamente relacionadas con el control de contenido.

5.3.1.1.1. Clases *ContenidoGeneral* y *Despiece*

Estas clases están implementadas en el artefacto *ContenidoGeneral.cs*.

CÓDIGO DEL ARCHIVO **CONTENIDOGENERAL.CS**

```
using System;
using System.Collections.Generic;
using System.Collections;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Data;
using SaethaWAM;
using SaethaWAM.Configuración;
using SaethaWAM.UI.Controles;
```

```
namespace SaethaWAM.Información
{
   public class ContenidoGeneral
   {
      public ContenidoGeneral()  { }
      public ContenidoGeneral(string IDInstancia, string IDSección, DateTime Fecha)
      {
         this.IDInstancia = IDInstancia;
         this.IDSección = IDSección;
         this.Fecha = Fecha;
      }

      private string idInstancia;
      public string IDInstancia
      {
         get { return idInstancia; }
         set
         {
            if (!Configuraciones.Actuales.Información.Contenidos.Contains(value))
               throw new ReferenciaNoExistenteException("La instancia de contenido general '" + value + "' no
existe.");
            idInstancia = value;
         }
      }
      private int idContenido;
      public int IDContenido { get { return idContenido; } }

      private string idSección;
      public string IDSección
      {
         get { return idSección; }
         set
         {
            Conexión Conn = new Conexión();
            Conn.Consultar("ExisteSección");
            Conn.AgregarParámetro("IDSección", value);
            if ((int)Conn.EjecutarEscalar() == 0)
               throw new ReferenciaNoExistenteException("La sección '" + value + "' no existe");
            if (!string.IsNullOrEmpty(idInstancia))
            {
               Conn.Consultar("ExisteSecciónDadaInstancia");
               Conn.AgregarParámetro("IDInstancia", idInstancia);
               if ((int)Conn.EjecutarEscalar() == 0)
                  throw new ReferenciaNoExistenteException("La sección '" + value + "' no existe en la instancia de
contenido '"
                        + idInstancia + "'.");
            }
            idSección = value;
         }
      }

      private DateTime fecha;
      public DateTime Fecha
      {
         get { return fecha; }
         set
         {
            if (value != null)
               fecha = value;
            else
               throw new ReferenciaNoVálidaException("La fecha del contenido no puede ser nula.");
         }
      }
```

```
        private EstadoContenido estado;
        public EstadoContenido Estado
        {
            get { return estado; }
            set
            {
                if ((byte)value >= 0 && (byte)value < 4)
                    estado = value;
                else
                    estado = EstadoContenido.Inactivo;
            }
        }

        private Dictionary<string, object> datosCampos;
        public Dictionary<string, object> DatosCampos
        {
            get
            {
                if (datosCampos == null) datosCampos = new Dictionary<string, object>();
                return datosCampos;
            }
        }
        public void AgregarDatoCampo(string Campo, object Valor)
        {
            DatosCampos.Add(Campo, Valor);
        }

        public static ContenidoGeneral DatosContenido(int idContenido)
        {
            ContenidoGeneral Contenido = new ContenidoGeneral();

            Conexión Conn = new Conexión();
            Conn.Consultar("DatosContenido");
            Conn.AgregarParámetro("IDContenido", idContenido);
            Conn.EjecutarLector();
            Conn.Lector.Read();
            string idInstancia = (string)Conn.Lector["IDInstancia"];
            Conn.CerrarLector();
            IEnumerable<ConfigCampo> CamposArchivo =
                                        from ConfigCampo Item in

Configuraciones.Actuales.Información.Contenidos[idInstancia].Campos.Values
                                        where Item.Tipo == TipoCampo.Archivo
                                        select Item;

            foreach (ConfigCampo archivo in CamposArchivo)
            {
                Conn.TextoComando = Conn.TextoComando.Replace("--a--", ", (select top 1 nombre from Archivos where
IDArchivo
                    = Contenidos." + archivo.ID + ") as URL" + archivo.ID + "\n--a--");
            }

            Conn.EjecutarLector();
            if (Conn.Lector.Read())
            {
                Contenido.idContenido = idContenido;
                Contenido.IDInstancia = (string)Conn.Lector["IDInstancia"];
                Contenido.IDSección = (string)Conn.Lector["IDSección"];
                Contenido.Fecha = (DateTime)Conn.Lector["Fecha"];
                Contenido.Estado = (EstadoContenido)(byte)Conn.Lector["Estado"];

                foreach (ConfigCampo Campo in
```

```
Configuraciones.Actuales.Información.Contenidos[(string)Conn.Lector["IDInstancia"]].Campos.Values)
            {
               if (Campo.Tipo == TipoCampo.Archivo)
                  Contenido.AgregarDatoCampo("URL" + Campo.ID, Conn.Lector["URL" + Campo.ID]);

               Contenido.AgregarDatoCampo(Campo.ID, Conn.Lector[Campo.ID]);
            }
            Conn.CerrarLector();
         }
         else
         {
            Conn.CerrarLector();
            throw new ReferenciaNoExistenteException("El ID de contenido '" + idContenido.ToString() + "' no es
válido.");
         }

        Conn.LimpiarParámetros();

        Conn.Consultar("ListarDespieces");
        Conn.AgregarParámetro("IDContenido", Contenido.IDContenido);
        Conn.EjecutarLector();

        Conexión Conn2 = new Conexión();

        while (Conn.Lector.Read())
        {
           Despiece despiece = new Despiece();
           despiece.IDContenido = (int)Conn.Lector["IDContenido"];
           despiece.IDDespiece = (int)Conn.Lector["IDDespiece"];
           despiece.IDFormatoDespiece = (string)Conn.Lector["IDFormatoDespiece"];

           foreach (ConfigCampo Campo in
                    Configuraciones.Actuales.Información.Contenidos[Contenido.IDInstancia].
                    Despieces[(string)Conn.Lector["IDFormatoDespiece"]].Campos.Values)
           {
              despiece.AgregarDatoCampo(Campo.ID, Conn.Lector[Campo.ID]);
           }

           Conn2.LimpiarParámetros();
           Conn2.Consultar("DatosDespiece");
           Conn2.AgregarParámetro("IDContenido", Contenido.IDContenido);
           Conn2.AgregarParámetro("IDDespiece", despiece.IDDespiece);

           CamposArchivo =
                      from ConfigCampo Item in
                           Configuraciones.Actuales.Información.Contenidos[Contenido.IDInstancia].
                           Despieces[despiece.IDFormatoDespiece].Campos.Values
                     where Item.Tipo == TipoCampo.Archivo
                     select Item;

           foreach (ConfigCampo archivo in CamposArchivo)
           {
              Conn2.TextoComando = Conn2.TextoComando.Replace("--a--", ", (select top 1 nombre from Archivos
where
                    IDArchivo = Contenidos_Despieces." + archivo.ID + ") as URL" + archivo.ID + "\n--a--");
           }

           Conn2.EjecutarLector();
           if (Conn2.Lector.Read())
           {
              foreach (ConfigCampo archivo in CamposArchivo)
              {
                 despiece.AgregarDatoCampo("URL" + archivo.ID, Conn2.Lector["URL" + archivo.ID]);
```

```
            }
        }
        Conn2.CerrarLector();

        Contenido.Despieces.Add(despiece);
    }
    Conn.CerrarLector();

    return Contenido;
}

public static DataTable ListarElementos(string sección)
{
    return new DataTable();
}

public static DataTable ListarContenido(string idInstancia, string título, DateTime? fecha, string sección,
        EstadoContenido? estado)
{
    return ListarContenido(idInstancia, título, fecha, sección, estado, "", null, int.MaxValue);
}

public static DataTable ListarContenido(string idInstancia, string título, DateTime? fecha, string sección,
        EstadoContenido? estado, string ordenar, IEnumerable<FiltroContenido> filtros, int cantidad)
{
    if (String.IsNullOrEmpty(idInstancia))
        throw new ArgumentException("El parámetro 'idInstancia' debe ser una instancia válida");

    if (!Configuraciones.Actuales.Información.Contenidos.Contains(idInstancia))
        throw new ArgumentException("El parámetro 'idInstancia' debe ser una instancia válida");

    Conexión Conn = new Conexión();
    Conn.Consultar("ListarContenidos");
    Conn.TextoComando = Conn.TextoComando.Replace("select ", "select top " + cantidad.ToString() + " ");
    Conn.AgregarParámetro("IDInstancia", idInstancia);

    if (!String.IsNullOrEmpty(título))
    {
        Conn.AgregarCondición("and Título like ?", "Título", título.Replace("*", "%") + "%");
    }
    if (fecha.HasValue)
    {
        Conn.AgregarCondición("and Fecha = ?", "Fecha", fecha.Value);
    }
    if (estado.HasValue)
    {
        Conn.AgregarCondición("and Estado = ?", "Estado", (byte)estado.Value);
    }
    if (!String.IsNullOrEmpty(sección))
    {
        Conn.AgregarCondición("and IDSección = ?", "IDSección", sección);
    }
    if (filtros != null)
    {
        foreach (FiltroContenido filtro in filtros)
        {
            if (filtro is FiltroCategoría)
            {
                FiltroCategoría categoría = (FiltroCategoría)filtro;
                ConfigCampo campo =
                        Configuraciones.Actuales.Información.Contenidos[idInstancia].Campos[categoría.IDCampo];
                if (campo.Tipo != TipoCampo.Categoría)
                    continue;
                string[] valores = categoría.Valor.Split(",".ToCharArray());
```

```
                string filtrosValores = "";
                foreach (string valor in valores)
                {
                  switch (categoría.TipoFiltro)
                  {
                    case FiltroTexto.CoincidenciaExacta:
                      filtrosValores += "or Nombre = '" + valor.Replace("'", "''") + "'";
                      break;
                    case FiltroTexto.Contiene:
                      filtrosValores += "or Nombre like '%" + valor.Replace("'", "''") + "%'";
                      break;
                    case FiltroTexto.IniciaCon:
                      filtrosValores += "or Nombre like '" + valor.Replace("'", "''") + "%'";
                      break;
                  }
                }
                Conn.AgregarCondición("and exists(select * from Categorías where IDVariable = Contenidos." +
                    categoría.IDCampo + " and IDCategoría = ? and (" + filtrosValores.Substring(3) + "))",
"IDCategoría",
                    campo.IDCategoría);
              }
              else if (filtro is FiltroFecha)
              {
                FiltroFecha ffecha = (FiltroFecha)filtro;
                string oper;
                switch (ffecha.TipoFiltro)
                {
                  case FiltroComparaciónFecha.IgualA:
                    oper = " = ";
                    break;
                  case FiltroComparaciónFecha.DistintoA:
                    oper = " <> ";
                    break;
                  case FiltroComparaciónFecha.MayorQue:
                    oper = " > ";
                    break;
                  case FiltroComparaciónFecha.MayorIgualQue:
                    oper = " >= ";
                    break;
                  case FiltroComparaciónFecha.MenorQue:
                    oper = " < ";
                    break;
                  case FiltroComparaciónFecha.MenorIgualQue:
                    oper = " <= ";
                    break;
                  default:
                    oper = " between ";
                    break;
                }
                Conn.AgregarCondición("and " + ffecha.IDCampo + oper + "?", "Fecha", ffecha.Valor);
                if (ffecha.TipoFiltro == FiltroComparaciónFecha.Entre)
                  Conn.AgregarCondición("and ?", "Fecha", ffecha.ValorAux);
              }
            }
          }
          if (!string.IsNullOrEmpty(ordenar))
          {
            Conn.TextoComando += " \n order by " + ordenar;
          }

          DataTable Datos;
          try
          {
            Datos = Conn.EjecutarTabla();
```

```
        }
        catch (Exception ex)
        {
            throw new Exception("Error al listar artículos de contenido.", ex);
        }

        Datos.FormatearFechas();
        Datos.FormatearEnum("Estado", typeof(EstadoContenido));
        return Datos;
    }

    public static bool ExisteCampoContenido(string idInstancia, string campo)
    {
        if (!Configuraciones.Actuales.Información.Contenidos[idInstancia].Campos.Contains(campo))
            return false;
        return true;
    }

    public static ContenidoGeneral RegistrarContenido(string idInstancia, string idSección, EstadoContenido
estado,
            IEnumerable<DatoColumna> datos)
    {
        Conexión Conn = new Conexión();
        ContenidoGeneral Nuevo = new ContenidoGeneral();
        Nuevo.IDInstancia = idInstancia;
        Nuevo.IDSección = idSección;
        Nuevo.idContenido = Conexión.SiguienteID("Contenidos", "IDContenido");
        Nuevo.Estado = estado;
        foreach (DatoColumna dato in datos)
        {
            Nuevo.AgregarDatoCampo(dato.NombreColumna, dato.Dato);
        }
        ((List<DatoColumna>)datos).Add(new DatoColumna("IDContenido", Nuevo.idContenido));
        ((List<DatoColumna>)datos).Add(new DatoColumna("IDInstancia", idInstancia));
        ((List<DatoColumna>)datos).Add(new DatoColumna("IDSección", idSección));
        ((List<DatoColumna>)datos).Add(new DatoColumna("Estado", (byte)estado));
        Conn.ConstruirInsertar("Contenidos", datos);

        try
        {
            Conn.EjecutarComando();
        }
        catch (Exception ex)
        {
            throw new Exception("No se pudo completar la inserción del contenido.", ex);
        }
        return Nuevo;
    }

    public static ContenidoGeneral ModificarContenido(int idContenido, string idSección, EstadoContenido estado,
        IEnumerable<DatoColumna> datos)
    {
        Conexión Conn = new Conexión();
        ContenidoGeneral Modificar = ContenidoGeneral.DatosContenido(idContenido);

        Modificar.IDSección = idSección;
        Modificar.Estado = estado;
        foreach (DatoColumna dato in datos)
        {
            Modificar.DatosCampos[dato.NombreColumna] = dato.Dato;
        }
        ((List<DatoColumna>)datos).Add(new DatoColumna("IDContenido", Modificar.IDContenido, true));
        ((List<DatoColumna>)datos).Add(new DatoColumna("IDSección", idSección));
        ((List<DatoColumna>)datos).Add(new DatoColumna("Estado", (byte)estado));
```

```
            Conn.ConstruirActualizar("Contenidos", datos);

            try
            {
                Conn.EjecutarComando();
            }
            catch (Exception ex)
            {
                throw new Exception("No se pudo completar la inserción del contenido.", ex);
            }
            return Modificar;
        }

        private List<Despiece> despieces;
        public List<Despiece> Despieces
        {
            get
            {
                if (despieces == null)
                    despieces = new List<Despiece>();
                return despieces;
            }
        }

        public void RegistrarDespieces(IEnumerable<Despiece> despieces)
        {
            Despieces.AddRange(despieces);
            RegistrarDespieces();
        }
        public void RegistrarDespieces()
        {
            if (idContenido != null)
            {
                Conexión Conn = new Conexión();
                Conn.Consultar("EliminarDespieces");
                Conn.AgregarParámetro("IDContenido", idContenido);
                Conn.EjecutarComando();
                Conn.LimpiarParámetros();

                foreach (Despiece despiece in Despieces)
                {
                    List<DatoColumna> datos = new List<DatoColumna>();
                    datos.Add(new DatoColumna("IDContenido", idContenido));
                    datos.Add(new DatoColumna("IDDespiece", despiece.IDDespiece));
                    datos.Add(new DatoColumna("IDFormatoDespiece", despiece.IDFormatoDespiece));
                    foreach (ConfigCampo campo in Configuraciones.Actuales.Información.Contenidos[this.IDInstancia].
                                Despieces[despiece.IDFormatoDespiece].Campos.Values)
                    {
                        datos.Add(new DatoColumna(campo.ID, despiece.DatosCampos[campo.ID]));
                    }
                    Conn.ConstruirInsertar("Contenidos_Despieces", datos);

                    try
                    {
                        Conn.EjecutarComando();
                    }
                    catch (Exception ex)
                    {
                        throw new Exception("No se pudo completar la inserción del despiece.", ex);
                    }
                }
            }
        }
    }
```

```
    public class Despiece
    {
        public Despiece()  { }

        private int idDespiece;
        public int IDDespiece { get { return idDespiece; } set { idDespiece = value; } }

        private int idContenido;
        public int IDContenido { get { return idContenido; } set { idContenido = value; } }

        private string idFormatoDespiece;
        public string IDFormatoDespiece { get { return idFormatoDespiece; } set { idFormatoDespiece = value; } }

        private Dictionary<string, object> datosCampos;
        public Dictionary<string, object> DatosCampos
        {
            get
            {
                if (datosCampos == null)
                    datosCampos = new Dictionary<string, object>();
                return datosCampos;
            }
        }

        public void AgregarDatoCampo(string Campo, object Valor)
        {
            DatosCampos.Add(Campo, Valor);
        }
    }

    public enum EstadoContenido : byte
    {
        Publicado = 1,
        NoRevisado = 0,
        Rechazado = 2,
        Inactivo = 3
    }
}
```

5.3.1.2. Subsistema Controles

En este subsistema sí se definen los componentes con los que habrá una interacción de usuario. Estos se ubicarían en dos ámbitos: el de *administración*, que serían archivos de formulario visibles dentro del CMS y que forman parte de la librería la librería *SaethaWAM.UI.CMS*, y el de *visualización,* que estaría conformado por las clases de la librería *SaethaWAM.UI.Controles*, implementables en el website con el uso de un marcado al estilo XML. Los formularios del CMS usarán, como convención para su nomenclatura, el prefijo “P” seguido de un ordinal incrementado por

cada componente, y sufijado con "_l" o "_m", representando si el formulario es de listado/acceso o de registro/edición, respectivamente.

5.3.1.2.1. Componente P2_Contenido_l - Listado de contenidos (CMS)

Este componente permitirá al usuario visualizar una rejilla de datos con todos los artículos de contenido registrados para una instancia particular. Se ofrecen opciones para filtrar y ubicar más fácilmente alguna información, así como para registrar nuevos artículos y modificar los ya existentes (presionando sobre el título de los mismos en la rejilla).

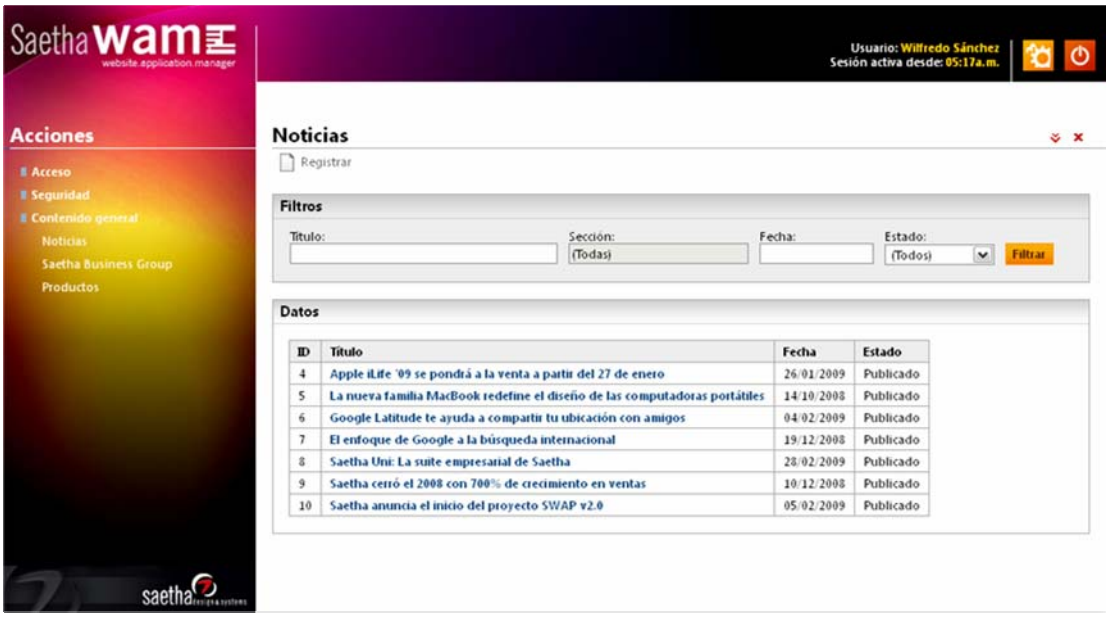


Figura 5.8. Interfaz del componente P2_Contenido_l

CÓDIGO DEL ARCHIVO **P2_CONTENIDO_L.ASPX**

```
<%@ Page Language="C#" MasterPageFile="~/CMS/App.master" AutoEventWireup="true"
CodeFile="2.Contenido_l.aspx.cs"
    Inherits="SaethaWAM.UI.CMS.P2_Contenido_l" Title="" %>
<%@ MasterType VirtualPath="~/CMS/App.master" %>
```

```
<asp:Content ID="Content1" ContentPlaceHolderID="AppContenido" Runat="Server">

<div class="appBarraHerramientas">
<asp:LinkButton runat="server" ID="btnNuevo" OnClick="Registrar" CssClass="botonBarraH"><img
src="Imgs/IconoNuevo.gif" width="25" height="25" border="0" class="izquierda" style="vertical-align: top" /><span
class="botonBarraHTexto izquierda">Registrar</span></asp:LinkButton>
</div>

<div class="appCuadroGris">
   <div class="appCuadroTitulo">Filtros</div>
   <div class="appCuadroContenido">

   <asp:Panel runat="server" ID="pnlFiltros" DefaultButton="btnFiltrar">
      <div class="lineac">
         <div class="control izquierda">
            <span>Título:</span>
            <div><asp:TextBox runat="server" ID="txtTítulo" CssClass="entrada ancho"></asp:TextBox></div>
         </div>
         <div class="control izquierda">
            <span>Sección:</span>
            <div><wam:ÁrbolDesplegable runat="server" ID="árbolSecciones" IDInstancia="Principal" Niveles="5"
                     TipoÁrbol="Secciones" CssClass="entrada mediano" TextoRaíz="(Todas)" /></div>
         </div>
         <div class="control izquierda">
            <span>Fecha:</span>
            <div><asp:TextBox runat="server" ID="txtFecha" CssClass="entrada angosto"></asp:TextBox>
               <ajaxToolkit:CalendarExtender ID="calFecha" runat="server" TargetControlID="txtFecha"
                  Format="dd/MM/yyyy" PopupPosition="BottomLeft" EnableViewState="true">
               </ajaxToolkit:CalendarExtender>
            </div>
         </div>
         <div class="control izquierda">
            <span>Estado:</span>
            <div><asp:DropDownList runat="server" ID="drpEstado" CssClass="entrada angosto">
               <asp:ListItem Text="(Todos)" Value="" />
               <asp:ListItem Text="Publicado" Value="1" />
               <asp:ListItem Text="No revisado" Value="0" />
               <asp:ListItem Text="Rechazado" Value="2" />
               <asp:ListItem Text="Inactivo" Value="3" />
            </asp:DropDownList></div>
         </div>
         <div class="controlb izquierda">
            <span class="botonf"><asp:LinkButton ID="btnFiltrar" Text="Filtrar" OnClick="Filtrar" CssClass="boton"
                  runat="server"></asp:LinkButton></span>
         </div>
      </div>
      <div class="clear"></div>
   </asp:Panel>
   </div>
</div>
<div class="appCuadroBlanco">
   <div class="appCuadroTitulo">Datos</div>
   <div class="appCuadroContenido">

      <asp:GridView ID="grdDatos" DataKeyNames="IDContenido" OnRowCommand="Navegar" runat="server"
         AutoGenerateColumns="false" BorderStyle="NotSet" CssClass="tablaDatos1"
         AllowPaging="true" PageSize="20" OnPageIndexChanging="Pagina"
         EmptyDataText="<div class='appCuadroNoData'>La consulta no devolvió datos</div>">
         <Columns>
            <asp:BoundField HeaderText="ID" DataField="IDContenido" />
            <asp:ButtonField HeaderText="Título" ButtonType="Link" DataTextField="Título"
CommandName="Navegar" />
            <asp:BoundField HeaderText="Fecha" DataField="FechaF" />
```

```
            <asp:BoundField HeaderText="Estado" DataField="EstadoF" />
        </Columns>
    </asp:GridView>
  </div>

</div>
</asp:Content>
```

CÓDIGO DEL ARCHIVO **P2_CONTENIDO_L.ASPX.CS**

```
using System;
using System.Data;
using System.Web.UI;
using System.Web.UI.WebControls;
using SaethaWAM.Configuración;
using SaethaWAM.Ejecución;
using SaethaWAM.Información;

namespace SaethaWAM.UI.CMS
{
  public partial class P2_Contenido_I : Página
  {
    protected override void CrearControles()
    {
      árbolSecciones.IDInstancia = ((ConfigContenido)Aplicación.InfoAplicación).IDInstancia;
      ControlesVS.AddRange(new Control[] { txtTítulo, árbolSecciones, txtFecha, drpEstado });
    }
    protected override void CargarDatosControles()
    {
      árbolSecciones.Valor = "";
    }

    protected void Registrar(object sender, EventArgs e)
    {
      Pantalla Nueva = Aplicación.NuevaPantalla("2.Contenido_m.aspx", TipoPantalla.Registro);
      Nueva.Abrir();
    }

    protected void Filtrar(object sender, EventArgs e)
    {
      FiltrarDatos();
      ConsultarDatos();
    }

    protected override void FiltrarDatos()
    {
      EstadoContenido? Estado = null;
      if (drpEstado.SelectedValue != "")
        Estado = (EstadoContenido?)Enum.Parse(typeof(EstadoContenido), drpEstado.SelectedValue);

      DateTime? Fecha = null;
      try
      {
        Fecha = DateTime.ParseExact(txtFecha.Text, calFecha.Format, null);
      }
      catch { }

      DataTable Datos =
ContenidoGeneral.ListarContenido(((ConfigContenido)Aplicación.InfoAplicación).IDInstancia,
          txtTítulo.Text, Fecha, árbolSecciones.Valor, Estado);
      Pantalla["Datos"] = Datos;
```

```
    }

    protected void Navegar(object sender, GridViewCommandEventArgs e)
    {
        if (e.CommandName == "Navegar")
        {
            ConsultarDatos();
            int index = Int32.Parse((string)e.CommandArgument);
            Pantalla Nueva = Aplicación.NuevaPantalla("2.Contenido_m.aspx", grdDatos.DataKeys[index].Value);
            Nueva.Abrir();
        }
    }

    protected override void ConsultarDatos()
    {
        grdDatos.DataSource = Pantalla["Datos"];
        grdDatos.DataBind();
    }

    protected void Pagina(object sender, GridViewPageEventArgs e)
    {
        grdDatos.PageIndex = e.NewPageIndex;
        ConsultarDatos();
    }
  }
}
```

5.3.1.2.2. Componente P2_Contenido_m - Edición de contenidos (CMS)

En este componente el administrador del website podrá registrar nuevos artículos de una instancia, así como modificar los ya registrados. Es importante resaltar que los controles visibles en esta interfaz son totalmente variables, de acuerdo a lo que se haya configurado en el descriptor de configuraciones. En la figura 5.9 se visualiza esta interfaz con campos de ejemplo, configurados en las primeras iteraciones de prueba.

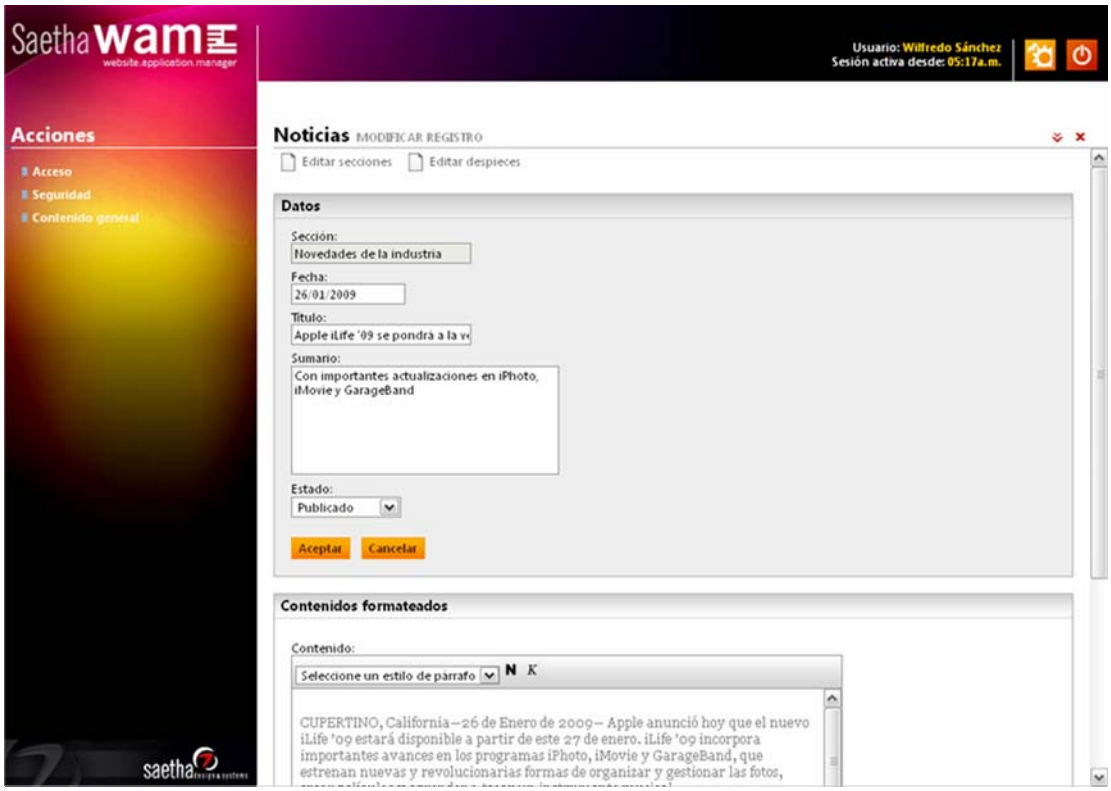


Figura 5.9. Interfaz del componente P2_Contenido_m

CÓDIGO DEL ARCHIVO **P2_CONTENIDO_M.ASPX**

```
<%@ Page Language="C#" MasterPageFile="~/CMS/App.master" ValidateRequest="false" AutoEventWireup="true"
      CodeFile="2.Contenido_m.aspx.cs" Inherits="SaethaWAM.UI.CMS.P2_Contenido_m" Title="" %>
<%@ MasterType VirtualPath="~/CMS/App.master" %>

<asp:Content ID="Content1" ContentPlaceHolderID="AppContenido" Runat="Server">

<div class="appBarraHerramientas">
<asp:LinkButton runat="server" ID="btnEditarSecciones" OnClick="EditarSecciones" CssClass="botonBarraH">
      <img src="Imgs/IconoNuevo.gif" width="25" height="25" border="0" class="izquierda" style="vertical-align: top"
/>
      <span class="botonBarraHTexto izquierda">Editar secciones</span>
</asp:LinkButton>
<asp:LinkButton runat="server" ID="btnEditarDespieces" OnClick="EditarDespieces" CssClass="botonBarraH">
      <img src="Imgs/IconoNuevo.gif" width="25" height="25" border="0" class="izquierda" style="vertical-align: top"
/>
      <span class="botonBarraHTexto izquierda">Editar despieces</span></asp:LinkButton>
</div>

<div class="appCuadroGris">
   <div class="appCuadroTitulo">Datos</div>
   <div class="appCuadroContenido">
      <asp:Panel runat="server" ID="pnlDatos" DefaultButton="btnAceptar">
         <asp:Panel runat="server" ID="pnlControles">
```

```
            <div class="lineac">
               <div class="control izquierda">
                  <span>Sección:</span>
                  <div><wam:ÁrbolDesplegable runat="server" ID="árbolSecciones" Niveles="5"
TipoÁrbol="Secciones"
                              CssClass="entrada mediano" TextoRaíz="(Seleccione)" /></div>
               </div>
            </div>
         </asp:Panel>
         <div class="lineac" runat="server" id="divEstado">
            <div class="control izquierda">
               <span>Estado:</span>
               <div><asp:DropDownList runat="server" ID="drpEstado" CssClass="entrada angosto">
                  <asp:ListItem Text="(Seleccione)" Value="" />
                  <asp:ListItem Text="Publicado" Value="1" />
                  <asp:ListItem Text="No revisado" Value="0" />
                  <asp:ListItem Text="Rechazado" Value="2" />
                  <asp:ListItem Text="Inactivo" Value="3" />
               </asp:DropDownList></div>
            </div>
         </div>
         <div class="lineac">
            <div class="controlb izquierda">
               <span class="botonf"><asp:LinkButton ID="btnAceptar" Text="Aceptar" OnClick="Aceptar"
CssClass="boton"
                     runat="server"></asp:LinkButton></span>
            </div>
            <div class="controlb izquierda">
               <span class="botonf"><asp:LinkButton ID="btnCancelar" Text="Cancelar" OnClick="Cancelar"
CssClass="boton"
                     runat="server"></asp:LinkButton></span>
            </div>
         </div>
      </asp:Panel>
      <div class="clear"></div>
   </div>
</div>
<div class="appCuadroBlanco">
   <div class="appCuadroTitulo">Contenidos formateados</div>
   <div class="appCuadroContenido">
      <asp:Panel runat="server" ID="pnlEditores"></asp:Panel>
      <div class="clear"></div>
   </div>
</div>
</asp:Content>
```

CÓDIGO DEL ARCHIVO **P2_CONTENIDO_M.ASPX.CS**

```
using System;
using System.Collections;
using System.Collections.Generic;
using System.Drawing;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.HtmlControls;
using System.Web.UI.WebControls;
using SaethaWAM.Configuración;
using SaethaWAM.Ejecución;
using SaethaWAM.Información;
using SaethaWAM.UI.Controles;
```

```
namespace SaethaWAM.UI.CMS
{
   public partial class P2_Contenido_m : Página
   {
      protected override string Subtítulo
      {
         get
         {
            if (Pantalla.TipoPantalla == TipoPantalla.Registro)
               return "Nuevo registro";
            else
               return "Modificar registro";
         }
      }
      protected override void CrearControles()
      {
         árbolSecciones.IDInstancia = ((ConfigContenido)Aplicación.InfoAplicación).IDInstancia;
         if (Pantalla.TipoPantalla == TipoPantalla.Registro)
         {
            divEstado.Style["display"] = "none";
            drpEstado.SelectedValue = ((byte)((ConfigContenido)Aplicación.InfoAplicación).EstadoInicial).ToString();
         }
         ControlesVS.AddRange(new Control[] { árbolSecciones });

         IEnumerable<ConfigCampo> campos = from ConfigCampo campo in
                                                ((ConfigContenido)Aplicación.InfoAplicación).Campos.Values
                                           orderby campo.OrdenUI
                                           select campo;
         foreach (ConfigCampo campo in campos)
         {
            if (FindControl("ctl" + campo.ID) != null)
               continue;
            switch (campo.Tipo)
            {
               case TipoCampo.TextoCorto:
                  Panel panelLínea = new Panel();
                  panelLínea.CssClass = "lineac";
                  Panel panelBloque = new Panel();
                  panelBloque.CssClass = "control izquierda";
                  Label etiqueta = new Label();
                  etiqueta.Text = campo.Nombre + ":";
                  panelBloque.Controls.Add(etiqueta);
                  Panel panelControl = new Panel();
                  TextBox Control = new TextBox();
                  Control.ID = "ctl" + campo.ID;
                  Control.CssClass = "entrada mediano";
                  panelControl.Controls.Add(Control);
                  panelBloque.Controls.Add(panelControl);
                  panelLínea.Controls.Add(panelBloque);
                  pnlControles.Controls.Add(panelLínea);
                  ControlesVS.Add(Control);
                  break;

               case TipoCampo.TextoLargo:
                  panelLínea = new Panel();
                  panelLínea.CssClass = "lineac";
                  panelBloque = new Panel();
                  panelBloque.CssClass = "control izquierda";
                  etiqueta = new Label();
                  etiqueta.Text = campo.Nombre + ":";
                  panelBloque.Controls.Add(etiqueta);
                  panelControl = new Panel();
```

```
                Control = new TextBox();
                Control.ID = "ctl" + campo.ID;
                Control.CssClass = "entrada ancho";
                Control.TextMode = TextBoxMode.MultiLine;
                Control.Rows = 6;
                panelControl.Controls.Add(Control);
                panelBloque.Controls.Add(panelControl);
                panelLínea.Controls.Add(panelBloque);
                pnlControles.Controls.Add(panelLínea);
                ControlesVS.Add(Control);
                break;
            case TipoCampo.Fecha:
                panelLínea = new Panel();
                panelLínea.CssClass = "lineac";
                panelBloque = new Panel();
                panelBloque.CssClass = "control izquierda";
                etiqueta = new Label();
                etiqueta.Text = campo.Nombre + ":";
                panelBloque.Controls.Add(etiqueta);
                panelControl = new Panel();
                Fecha ControlFecha = new Fecha();
                ControlFecha.ID = "ctl" + campo.ID;
                ControlFecha.CssClass = "entrada angosto";
                panelControl.Controls.Add(ControlFecha);
                panelBloque.Controls.Add(panelControl);
                panelLínea.Controls.Add(panelBloque);
                pnlControles.Controls.Add(panelLínea);
                ControlesVS.Add(ControlFecha);
                break;
            case TipoCampo.Categoría:
                panelLínea = new Panel();
                panelLínea.CssClass = "lineac";
                panelBloque = new Panel();
                panelBloque.CssClass = "control izquierda";
                etiqueta = new Label();
                etiqueta.Text = campo.Nombre + ":";
                panelBloque.Controls.Add(etiqueta);
                panelControl = new Panel();
                DropDownList ControlDrop = new DropDownList();
                ControlDrop.ID = "ctl" + campo.ID;
                ControlDrop.CssClass = "entrada mediano";
                ControlDrop.DataTextField = "Nombre";
                ControlDrop.DataValueField = "IDVariable";

                ControlDrop.DataSource = SaethaWAM.General.Categoría.ListarCategorías(campo.IDCategoría);
                ControlDrop.DataBind();
                ControlDrop.Items.Insert(0, new ListItem("(Seleccione)", "0"));
                panelControl.Controls.Add(ControlDrop);

                Control = new TextBox();
                Control.ID = "ctl" + campo.ID + "t";
                Control.CssClass = "entrada mediano";
                Control.Style["Display"] = "none";
                panelControl.Controls.Add(Control);

                HtmlGenericControl ControlLink = new HtmlGenericControl("a");
                ControlLink.ID = "ctl" + campo.ID + "l";
                ControlLink.InnerHtml = " › Agregar";
                panelControl.Controls.Add(ControlLink);

                HtmlGenericControl ControlLink2 = new HtmlGenericControl("a");
                ControlLink2.ID = "ctl" + campo.ID + "ll";
                ControlLink2.InnerHtml = " › Seleccionar";
                ControlLink2.Style["display"] = "none";
```

```
                panelControl.Controls.Add(ControlLink2);

                panelBloque.Controls.Add(panelControl);
                panelLínea.Controls.Add(panelBloque);
                pnlControles.Controls.Add(panelLínea);

                ControlLink.Attributes["class"] = "vínculo";
                ControlLink.Attributes["href"] = "#";
                ControlLink.Attributes["onClick"] = @"javascript: document.getElementById('" + ControlDrop.ClientID
+
                    "').style.display = 'none'; document.getElementById('" + Control.ClientID + "').style.display = '';
                    document.getElementById('" + ControlLink.ClientID + "').style.display = 'none';
document.getElementById('"
                    + ControlLink2.ClientID + "').style.display = ';';

                ControlLink2.Attributes["class"] = "vínculo";
                ControlLink2.Attributes["href"] = "#";
                ControlLink2.Attributes["onClick"] = @"javascript: document.getElementById('" + ControlDrop.ClientID
+
                    "').style.display = ''; document.getElementById('" + Control.ClientID + "').style.display = 'none';
                    document.getElementById('" + ControlLink.ClientID + "').style.display = '';
document.getElementById('" +
                    ControlLink2.ClientID + "').style.display = 'none'; document.getElementById('" + Control.ClientID
+
                    "').value = ''; ";

                ControlesVS.Add(ControlDrop);
                ControlesVS.Add(Control);
                ControlesVS.Add(ControlLink);
                ControlesVS.Add(ControlLink2);

                break;
            case TipoCampo.Archivo:
                panelLínea = new Panel();
                panelLínea.CssClass = "lineac";
                panelBloque = new Panel();
                panelBloque.CssClass = "control izquierda";
                etiqueta = new Label();
                etiqueta.Text = campo.Nombre + ":";
                panelBloque.Controls.Add(etiqueta);
                panelControl = new Panel();
                FileUpload ControlFile = new FileUpload();
                ControlFile.ID = "ctl" + campo.ID;
                ControlFile.CssClass = "entrada ancho";
                panelControl.Controls.Add(ControlFile);
                panelBloque.Controls.Add(panelControl);
                panelLínea.Controls.Add(panelBloque);
                pnlControles.Controls.Add(panelLínea);
                ControlesVS.Add(ControlFile);
                break;
            case TipoCampo.TextoFormateado:
                panelLínea = new Panel();
                panelLínea.CssClass = "lineac";
                panelBloque = new Panel();
                panelBloque.CssClass = "control izquierda";
                etiqueta = new Label();
                etiqueta.Text = campo.Nombre + ":";
                panelBloque.Controls.Add(etiqueta);
                panelControl = new Panel();
                Editor ControlEditor = new Editor();
                ControlEditor.ID = "ctl" + campo.ID;
                ControlEditor.EstiloCssBase = ((ConfigContenido)Aplicación.InfoAplicación).EstiloPorDefecto;
                ControlEditor.Ancho = new Unit(500, UnitType.Pixel);
                ControlEditor.Alto = new Unit(350, UnitType.Pixel);
```

```
                foreach (ConfigEstilo estilo in ((ConfigContenido)Aplicación.InfoAplicación).Estilos.Values)
                {
                    ControlEditor.Estilos.Add(new EstiloPárrafo(estilo.ID, estilo.Nombre, estilo.CSS));
                }

                panelControl.Controls.Add(ControlEditor);
                panelBloque.Controls.Add(panelControl);
                panelLínea.Controls.Add(panelBloque);
                pnlEditores.Controls.Add(panelLínea);
                ControlesVS.Add(ControlEditor);
                break;
            }
        }
    }
    protected override void CargarDatosControles()
    {
        árbolSecciones.Valor = "";
    }

    protected string IDControl(Control control)
    {
        return control.ID;
    }
    protected void Aceptar(object sender, EventArgs e)
   {
      Dictionary<string, SaethaWAM.General.Archivo> Archivos = new Dictionary<string,
SaethaWAM.General.Archivo>();
        if (Pantalla.TipoPantalla == TipoPantalla.Registro)
        {
            try
            {
                ConfigContenido contenido = (ConfigContenido)Aplicación.InfoAplicación;
                List<DatoColumna> datos = new List<DatoColumna>();
                foreach (ConfigCampo campo in contenido.Campos.Values)
                {
                    if (campo.Tipo == TipoCampo.Categoría && ((TextBox)BuscarControlVS("ctl" + campo.ID +
"t")).Text.Trim() != "")
                    {
                        if (SaethaWAM.General.Categoría.ExisteVariableCategoría(campo.IDCategoría,
                            ((TextBox)BuscarControlVS("ctl" + campo.ID + "t")).Text.Trim()))
                        {
                            object valor = SaethaWAM.General.Categoría.IDVariableCategoría(campo.IDCategoría,
                            ((TextBox)BuscarControlVS("ctl" + campo.ID + "t")).Text.Trim());
                            datos.Add(new DatoColumna(campo.ID, valor));
                        }
                        else
                        {
                            SaethaWAM.General.Categoría Nueva =
                                SaethaWAM.General.Categoría.RegistrarCategoría(campo.IDCategoría,
                                ((TextBox)BuscarControlVS("ctl" + campo.ID + "t")).Text.Trim(), Estado.Activo);
                            object valor = Nueva.IDVariable;
                            datos.Add(new DatoColumna(campo.ID, valor));
                        }
                    }
                    else if (campo.Tipo == TipoCampo.Archivo)
                    {
                        if (!((FileUpload)BuscarControlVS("ctl" + campo.ID)).HasFile)
                        {
                            if (!campo.Opcional)
                                throw new Exception("El archivo del campo '" + campo.Nombre + "' debe ser especificado.");
                        }
                        else
                        {
```

```
                        SaethaWAM.General.Archivo Cargado =
                            SaethaWAM.General.Archivo.RegistrarArchivo(((FileUpload)BuscarControlVS("ctl" +
                            campo.ID)).PostedFile);
                        object valor = Cargado.IDArchivo;
                        Archivos[campo.ID] = Cargado;
                        datos.Add(new DatoColumna(campo.ID, valor));
                    }
                }
                else if (campo.Tipo == TipoCampo.ImagenVinculada)
                {
                    if (!((FileUpload)BuscarControlVS("ctl" + campo.ImagenIDCampoOriginal)).HasFile)
                    {
                        if (!campo.Opcional)
                            throw new Exception("El archivo del campo vinculado '" + campo.Nombre + "' debe ser
especificado.");
                    }
                    else
                    {
                        if (Archivos[campo.ImagenIDCampoOriginal].Tipo == SaethaWAM.General.TipoArchivo.Imagen)
                        {
                            Bitmap original = new Bitmap(HttpContext.Current.Server.MapPath("~/Archivos/" +
                                Archivos[campo.ImagenIDCampoOriginal].Nombre));
                            Bitmap redimensión = new Bitmap(original);
                         AjustarImagen(campo, original, ref redimensión, Archivos);
                            SaethaWAM.General.Archivo Cargado =
SaethaWAM.General.Archivo.RegistrarArchivo(redimensión);
                            object valor = Cargado.IDArchivo;
                            datos.Add(new DatoColumna(campo.ID, valor));
                        }
                    }
                }
                else
                {
                    object valor = ValorControl(BuscarControlVS("ctl" + campo.ID));
                    datos.Add(new DatoColumna(campo.ID, valor));
                }
            }
            ContenidoGeneral contenidoRegistrado = ContenidoGeneral.RegistrarContenido(contenido.IDInstancia,
                    árbolSecciones.Valor, (EstadoContenido)Enum.Parse(typeof(EstadoContenido),
drpEstado.SelectedValue),
                    datos);
            if (Aplicación.Contains("VSDespieces") && Aplicación["VSDespieces"] is List<Despiece>)
            {
                contenidoRegistrado.RegistrarDespieces((IEnumerable<Despiece>)Aplicación["VSDespieces"]);
            }
            }
            catch (Exception ex)
            {
                Aplicación.Mensaje("Hubo un error en el registro del contenido", ex.Message, TipoMensaje.Error);
                return;
            }

        Aplicación.Mensaje("Registro correcto", "El contenido fue registrado sin problemas.",
TipoMensaje.Confirmación);

        }
        else if (Pantalla.TipoPantalla == TipoPantalla.Modificación)
        {
            try
            {
                ConfigContenido contenido = (ConfigContenido)Aplicación.InfoAplicación;
                List<DatoColumna> datos = new List<DatoColumna>();

                foreach (ConfigCampo campo in contenido.Campos.Values)
```

```
                {
                  if (campo.Tipo == TipoCampo.Categoría && ((TextBox)BuscarControlVS("ctl" + campo.ID +
"t")).Text.Trim() != "")
                {
                  if (SaethaWAM.General.Categoría.ExisteVariableCategoría(campo.IDCategoría,
                        ((TextBox)BuscarControlVS("ctl" + campo.ID + "t")).Text.Trim()))
                  {
                    object valor = SaethaWAM.General.Categoría.IDVariableCategoría(campo.IDCategoría,
                        ((TextBox)BuscarControlVS("ctl" + campo.ID + "t")).Text.Trim());
                    datos.Add(new DatoColumna(campo.ID, valor));
                  }
                  else
                  {
                    SaethaWAM.General.Categoría Nueva =
                        SaethaWAM.General.Categoría.RegistrarCategoría(campo.IDCategoría,
                      ((TextBox)BuscarControlVS("ctl" + campo.ID + "t")).Text.Trim(), Estado.Activo);
                    object valor = Nueva.IDVariable;
                    datos.Add(new DatoColumna(campo.ID, valor));
                  }
                }
                else if (campo.Tipo == TipoCampo.Archivo)
                {
                  if (!((FileUpload)BuscarControlVS("ctl" + campo.ID)).HasFile)
                  {
                    if (!campo.Opcional)
                      throw new Exception("El archivo del campo '" + campo.Nombre + "' debe ser especificado.");
                  }
                  else
                  {
                    SaethaWAM.General.Archivo Cargado =
                      SaethaWAM.General.Archivo.RegistrarArchivo(((FileUpload)BuscarControlVS("ctl" +
campo.ID)).PostedFile);
                    object valor = Cargado.IDArchivo;
                    Archivos[campo.ID] = Cargado;
                    datos.Add(new DatoColumna(campo.ID, valor));
                  }
                }
                else if (campo.Tipo == TipoCampo.ImagenVinculada)
                {
                  if (!((FileUpload)BuscarControlVS("ctl" + campo.ImagenIDCampoOriginal)).HasFile)
                  {
                    if (!campo.Opcional)
                      throw new Exception("El archivo del campo vinculado '" + campo.Nombre + "' debe ser
especificado.");
                  }
                  else
                  {
                    if (Archivos[campo.ImagenIDCampoOriginal].Tipo == SaethaWAM.General.TipoArchivo.Imagen)
                    {
                      Bitmap original = new Bitmap(HttpContext.Current.Server.MapPath("~/Archivos/" +
                          Archivos[campo.ImagenIDCampoOriginal].Nombre));
                      Bitmap redimensión = new Bitmap(original);
                      AjustarImagen(campo, original, ref redimensión, Archivos);
                      SaethaWAM.General.Archivo Cargado =
SaethaWAM.General.Archivo.RegistrarArchivo(redimensión);
                      object valor = Cargado.IDArchivo;
                      datos.Add(new DatoColumna(campo.ID, valor));
                    }
                  }
                }
                else
                {
                  object valor = ValorControl(BuscarControlVS("ctl" + campo.ID));
                  datos.Add(new DatoColumna(campo.ID, valor));
```

```
                }
                }
                ContenidoGeneral contenidoModificado = ContenidoGeneral.ModificarContenido((int)Pantalla["Clave"],
                  árbolSecciones.Valor, (EstadoContenido)Enum.Parse(typeof(EstadoContenido),
drpEstado.SelectedValue), datos);
            if (Aplicación.Contains("VSDespieces") && Aplicación["VSDespieces"] is List<Despiece>)
            {
              contenidoModificado.RegistrarDespieces((IEnumerable<Despiece>)Aplicación["VSDespieces"]);
            }
            }
            catch (Exception ex)
            {
              Aplicación.Mensaje("Hubo un error en la modificación del contenido", ex.Message, TipoMensaje.Error);
              return;
            }
            Aplicación.Mensaje("Modificación correcta", "El contenido fue actualizado sin problemas.",
              TipoMensaje.Confirmación);
        }
        Aplicación.Remove("VSDespieces");
        Pantalla.Cerrar();
      }

    protected void AjustarImagen(ConfigCampo campo, Bitmap original, ref Bitmap redimensión, Dictionary<string,
SaethaWAM.General.Archivo> Archivos)
    {
      if (campo.ImagenAncho != null && campo.ImagenAlto != null && campo.ImagenAncho > 0 &&
campo.ImagenAlto > 0)
      {
        if (campo.ImagenAncho * original.Height / original.Width <= campo.ImagenAlto)
        {
          redimensión = new Bitmap(original, campo.ImagenAncho, campo.ImagenAncho * original.Height /
original.Width);
          redimensión = new Bitmap(campo.ImagenAncho, campo.ImagenAncho * original.Height / original.Width);
          Graphics graph = Graphics.FromImage(redimensión);
          graph.InterpolationMode = System.Drawing.Drawing2D.InterpolationMode.High;
          graph.CompositingQuality = System.Drawing.Drawing2D.CompositingQuality.HighQuality;
          graph.SmoothingMode = System.Drawing.Drawing2D.SmoothingMode.AntiAlias;
          graph.DrawImage(original, new Rectangle(0, 0, campo.ImagenAncho, campo.ImagenAncho *
original.Height /
                  original.Width));
        }
        else
        {
          redimensión = new Bitmap(original, campo.ImagenAlto * original.Width / original.Height,
campo.ImagenAlto);
          redimensión = new Bitmap(campo.ImagenAlto * original.Width / original.Height, campo.ImagenAlto);
          Graphics graph = Graphics.FromImage(redimensión);
          graph.InterpolationMode = System.Drawing.Drawing2D.InterpolationMode.High;
          graph.CompositingQuality = System.Drawing.Drawing2D.CompositingQuality.HighQuality;
          graph.SmoothingMode = System.Drawing.Drawing2D.SmoothingMode.AntiAlias;
          graph.DrawImage(original, new Rectangle(0, 0, campo.ImagenAlto * original.Width / original.Height,
                  campo.ImagenAlto));
        }
        redimensión.Tag = new object[] {
                    Archivos[campo.ImagenIDCampoOriginal].Nombre,
                    "_" + campo.ImagenAncho.ToString() + "x" + campo.ImagenAlto.ToString(),
                    Archivos[campo.ImagenIDCampoOriginal]
                  };
      }
      else if (campo.ImagenAncho != null && campo.ImagenAncho > 0)
      {
        redimensión = new Bitmap(original, campo.ImagenAncho, campo.ImagenAncho * original.Height /
original.Width);
        redimensión = new Bitmap(campo.ImagenAncho, campo.ImagenAncho * original.Height / original.Width);
```

```
            Graphics graph = Graphics.FromImage(redimensión);
            graph.InterpolationMode = System.Drawing.Drawing2D.InterpolationMode.High;
            graph.CompositingQuality = System.Drawing.Drawing2D.CompositingQuality.HighQuality;
            graph.SmoothingMode = System.Drawing.Drawing2D.SmoothingMode.AntiAlias;
            graph.DrawImage(original, new Rectangle(0, 0, campo.ImagenAncho, campo.ImagenAncho * original.Height
/
                    original.Width));
            redimensión.Tag = new object[] {
                                Archivos[campo.ImagenIDCampoOriginal].Nombre,
                                "_w" + campo.ImagenAncho.ToString(),
                                Archivos[campo.ImagenIDCampoOriginal]
                            };
        }
        else if (campo.ImagenAlto != null && campo.ImagenAlto > 0)
        {
            redimensión = new Bitmap(original, campo.ImagenAlto * original.Width / original.Height, campo.ImagenAlto);
            redimensión = new Bitmap(campo.ImagenAlto * original.Width / original.Height, campo.ImagenAlto);
            Graphics graph = Graphics.FromImage(redimensión);
            graph.InterpolationMode = System.Drawing.Drawing2D.InterpolationMode.High;
            graph.CompositingQuality = System.Drawing.Drawing2D.CompositingQuality.HighQuality;
            graph.SmoothingMode = System.Drawing.Drawing2D.SmoothingMode.AntiAlias;
            graph.DrawImage(original, new Rectangle(0, 0, campo.ImagenAlto * original.Width / original.Height,
                    campo.ImagenAlto));
            redimensión.Tag = new object[] {
                                Archivos[campo.ImagenIDCampoOriginal].Nombre,
                                "_h" + campo.ImagenAncho.ToString(),
                                Archivos[campo.ImagenIDCampoOriginal]
                            };
        }
    }

    protected override void CargarDatosModificación()
    {
        if (Pantalla.Contains("Clave") && Pantalla["Clave"] is int)
        {
            ContenidoGeneral Datos = ContenidoGeneral.DatosContenido((int)Pantalla["Clave"]);
            árbolSecciones.Valor = Datos.IDSección;
            drpEstado.SelectedValue = ((byte)Datos.Estado).ToString();
            foreach (ConfigCampo Campo in ((ConfigContenido)Aplicación.InfoAplicación).Campos.Values)
            {
                if (BuscarControlVS("ctl" + Campo.ID) != null)
                {
                    switch (Campo.Tipo)
                    {
                        case TipoCampo.TextoCorto:
                            TextBox ControlTexto = (TextBox)BuscarControlVS("ctl" + Campo.ID);
                            ControlTexto.Text = (string)Datos.DatosCampos[Campo.ID];
                            break;
                        case TipoCampo.TextoLargo:
                            ControlTexto = (TextBox)BuscarControlVS("ctl" + Campo.ID);
                            ControlTexto.Text = (string)Datos.DatosCampos[Campo.ID];
                            break;
                        case TipoCampo.Fecha:
                            Fecha ControlFecha = (Fecha)BuscarControlVS("ctl" + Campo.ID);
                            ControlFecha.Valor =
((DateTime)Datos.DatosCampos[Campo.ID]).ToString(ControlFecha.FormatoFecha);
                            break;
                        case TipoCampo.TextoFormateado:
                            Editor ControlEditor = (Editor)BuscarControlVS("ctl" + Campo.ID);
                            ControlEditor.Valor = (string)Datos.DatosCampos[Campo.ID];
                            break;
                        case TipoCampo.Categoría:
                            DropDownList ControlDrop = (DropDownList)BuscarControlVS("ctl" + Campo.ID);
                            ControlDrop.ClearSelection();
```

```
                        if (!(Datos.DatosCampos[Campo.ID] is DBNull))
                        {
                            if (ControlDrop.Items.FindByValue(Datos.DatosCampos[Campo.ID].ToString()) != null)
                            {
                                ControlDrop.Items.FindByValue(Datos.DatosCampos[Campo.ID].ToString()).Selected =
true;
                            }
                            else
                            {
                                ControlDrop.Items[0].Selected = true;
                            }
                        }
                        break;
                    }
                }
            }
        }

        protected void EditarSecciones(object sender, EventArgs e)
        {
            Pantalla Nueva = Aplicación.NuevaPantalla("~/CMS/1.Secciones_I.aspx", TipoPantalla.Listado);
            Nueva.Abrir();
        }

        protected void EditarDespieces(object sender, EventArgs e)
        {
            if (!Aplicación.Contains("VSDespieces"))
            {
                Aplicación["VSDespieces"] = new List<Despiece>();
                if (Pantalla.TipoPantalla == TipoPantalla.Modificación)
                {
                    ContenidoGeneral contenido = ContenidoGeneral.DatosContenido((int)Pantalla["Clave"]);
                    ((List<Despiece>)Aplicación["VSDespieces"]).AddRange(contenido.Despieces);
                }
            }

            Pantalla Nueva = Aplicación.NuevaPantalla("~/CMS/2.Despiece_I.aspx", TipoPantalla.Listado);
            Nueva.Abrir();
        }

        protected void Cancelar(object sender, EventArgs e)
        {
          Aplicación.Remove("VSDespieces");
            Pantalla.Cerrar();
        }
    }
}
```

5.3.1.2.3. Componente ListaContenido - Control de listado de contenidos (Website)

Este componente se hace visible cada vez que sea implementado en un archivo público de formulario dentro del website. Tanto los datos a mostrar

como el diseño con que se visualizarán están en función de las configuraciones.

Los datos disponibles para visualizar en el control dependerán de los campos definidos para la instancia que se listará en el control.

En el caso del diseño, este será maquetado como plantilla HTML dentro de cada implementación del control (enlazándose a los campos de datos con el uso de cadenas con el formato "*<! nombreCampo >"*), ofreciéndose también la posibilidad de vincular el contenido de un campo a la propiedad de otro control.

Para visualizar el detalle de un contenido listado, se ofrece tanto la posibilidad de hacer un vínculo a otra página y enviar el identificador vía *queryString*, como de ubicar en la misma página el control VisorContenido y actualizarlo con los detalles, sin necesidad de cambiar de página.

Todas estas consideraciones fueron tomadas para **maximizar el grado de flexibilidad** con que contarán los diseñadores y maquetadores al realizar la producción de los websites de los clientes.

La mayoría de las propiedades del control se establecen en la etiqueta del mismo. Algunos datos más complejos, como las plantillas de elemento, separador, encabezado y pie, así como los filtros aplicables, serán detallados como etiquetas internas al control. De esta forma, se hace más legible la visualización y comprensión del código, facilitándose los posteriores mantenimientos que se le pueda dar al código del website. Por otra parte, para el funcionamiento estándar del control no se requiere ninguna

codificación adicional en el archivo *code-behind* de la página, lográndose una abstracción efectiva de las operaciones del control.

Habiéndose explicado la naturaleza configurable del control, se mostrará la codificación del mismo, así como una interfaz que ejemplifica una implementación del control y el fragmento de código con que se configura (resaltado en negritas el etiquetado de los controles).

Figura 5.10. Interfaz de muestra del control ListaContenido, instanciado dos veces (para listar notas de prensa y novedades por separado) en esta página del website de prueba

FRAGMENTO DE CÓDIGO DEL ARCHIVO **NOTICIAS.ASPX**

```
<div class="título"><img src="Imgs/Titulo_Noticias.gif" border="0" /></div>
<div style="width: 430px; float: left; padding-top: 8px">
    <div style="font-family: Segoe UI, Tahoma, Verdana, Arial, Helvetica, sans-serif; font-weight: bold; font-size: 14px; color:
        #7a5d00; border-bottom: 1px solid #999999; padding-bottom: 4px; margin-bottom: 12px;">Notas de prensa
»</div>

    <wam:ListaContenido Ordenar="Fecha DESC" ID="wamLista" IDInstancia="Noticias"
ArchivoDestino="NoticiasV.aspx"
```

```
      IDSección="Prensa" runat="server">
      <Elemento>
         <div class="elemento">
            <span class="tituloElemento"><a href="<! LINK >;" class="tituloElemento"><! Titulo ></a></span>
            <span class="fechaElemento"><! Fecha ></span>
            <span class="resumenElemento"><! Sumario ></span>
         </div>
      </Elemento>
   </wam:ListaContenido>
</div>
<div style="width: 180px; float: left; margin-left: 19px; background-color: white; padding: 8px;">
   <div style="font-family: Segoe UI, Tahoma, Verdana, Arial, Helvetica, sans-serif; font-weight: bold; font-size: 14px;
color:
         #7a5d00; border-bottom: 1px solid #999999; margin-bottom: 12px;">Novedades tecnológicas »</div>
   <wam:ListaContenido ID="ListaContenido1" IDInstancia="Noticias" IDSección="Novedades" runat="server">
      <Elemento>
         <div class="elemento">
            <span class="fechaElemento"><! Fecha >  </span>
            <span class="títuloElemento2"><a href="<! PB >;" class="títuloElemento2"><! Título ></a></span>
         </div>
      </Elemento>
   </wam:ListaContenido>
</div>
```

CÓDIGO DEL ARCHIVO **LISTACONTENIDO.CS**

```
using System;
using System.Collections;
using System.Collections.Generic;
using System.Collections.Specialized;
using System.Data;
using System.Text.RegularExpressions;
using System.Web.UI;
using SaethaWAM.Configuración;
using SaethaWAM.Ejecución;
using SaethaWAM.Información;

namespace SaethaWAM.UI.Controles
{
   [ToolboxData("<{0}:ListaContenido runat=server></{0}:ListaContenido>")]
   [ParseChildren(true)]
   public class ListaContenido : Repetidor, IControlSección, IPostBackDataHandler
   {
      protected override void OnLoad(EventArgs e)
      {
         base.OnLoad(e);
      }
      protected override void OnInit(EventArgs e)
      {
         Page.RegisterRequiresControlState(this);
         Page.RegisterRequiresPostBack(this);
         Page.RegisterRequiresRaiseEvent(this);

         base.OnInit(e);
      }

      public bool LoadPostData(string postDataKey, System.Collections.Specialized.NameValueCollection
postCollection)
      {
         if (postCollection["__EVENTTARGET"] == postDataKey)
         {
```

```
            RaisePostBackEvent(postCollection["__EVENTARGUMENT"]);
            return true;
        }
        return false;
    }

    public void RaisePostDataChangedEvent() { }

    protected override string ArgumentoPostBack
    {
        get { return Page.ClientScript.GetPostBackClientHyperlink(this,"<! IDContenido >", true); }
    }

    public override void RaisePostBackEvent(string eventArgument)
    {
        int prueba;
        if (!string.IsNullOrEmpty(idControl) && !string.IsNullOrEmpty(eventArgument) && Page != null
            && Page.FindControl(idControl) != null && Page.FindControl(idControl) is VisorContenido
            && Int32.TryParse(eventArgument, out prueba))
        {
            try
            {
                int IDContenido = Int32.Parse(eventArgument);
                VisorContenido Visor = (VisorContenido)Page.FindControl(idControl);
                Visor.IDContenido = IDContenido;
            }
            catch
            {
                Sistema.RegistrarErrorControl(typeof(ReferenciaNoVálidaException), this,
                    "No se pudo procesar el postback del evento");
            }
        }
        base.RaisePostBackEvent(eventArgument);
    }

    private string idInstancia;
    public string IDInstancia
    {
        get
        {
            return idInstancia;
        }
        set
        {
            if (!Configuraciones.Actuales.Información.Contenidos.Contains(value))
                Sistema.RegistrarErrorControl(typeof(ReferenciaNoExistenteException), (Control)this,
                    "La instancia de contenido general '" + value + "' no existe.");
            idInstancia = value;
        }
    }

    private string idSección;
    public string IDSección
    {
        get
        {
            return idSección;
        }
        set
        {
            Conexión Conn = new Conexión();
            Conn.Consultar("ExisteSección");
            Conn.AgregarParámetro("IDSección", value);
```

```
            if ((int)Conn.EjecutarEscalar() == 0)
            {
                Sistema.RegistrarErrorControl(typeof(ReferenciaNoExistenteException), (Control)this, "La sección '" +
value +
                    "' no existe");
            }

            if (!string.IsNullOrEmpty(idInstancia))
            {
                Conn.Consultar("ExisteSecciónDadaInstancia");
                Conn.AgregarParámetro("IDInstancia", idInstancia);
                if ((int)Conn.EjecutarEscalar() == 0)
                {
                    Sistema.RegistrarErrorControl(typeof(ReferenciaNoExistenteException), (Control)this, "La sección '"
+ value
                        + "' no existe en la instancia de contenido '" + idInstancia + "'.");
                }
            }
            idSección = value;
        }
    }

    protected override void OnPreRender(EventArgs e)
    {
        VerificarDatos();
    }

    private void VerificarDatos()
    {
        if (cargarQS)
        {
            if (Page != null && Page.Request != null && Page.Request.QueryString != null &&
                !string.IsNullOrEmpty(Page.Request.QueryString["wam"]))
            {
                if (información == null) información = new NameValueCollection();
                información.Clear();
                información.DecodificarDatosXml(Page.Request.QueryString["wam"].DescomprimirGZipUrl());
                if (!string.IsNullOrEmpty(información["IDInstancia"]))
                {
                    IDInstancia = información["IDInstancia"];
                }
                if (!string.IsNullOrEmpty(información["IDSección"]))
                {
                    IDSección = información["IDSección"];
                }
            }
        }
        if (!string.IsNullOrEmpty(idInstancia))
        {
            datos = ContenidoGeneral.ListarContenido(idInstancia, "", null, idSección, EstadoContenido.Publicado,
Ordenar,
                    Filtros, cantidadFilas);
        }
    }

    private int cantidadFilas = int.MaxValue;
    public int CantidadFilas
    {
        get
        {
            return cantidadFilas;
        }
        set
```

```
    {
      cantidadFilas = value;
    }
  }

  private string ordenar;
  public string Ordenar
  {
    get { return ordenar; }
    set { ordenar = value; }
  }

   private string idControl;
   public string IDControl
   {
      get
      {
         return idControl;
      }
      set
      {
         idControl = value;
      }
   }

   private bool cargarQS = true;
   public bool CargarQS
   {
      get
      {
         return cargarQS;
      }
      set
      {
         cargarQS = value;
      }
   }

   private string archivoDestino;
   public string ArchivoDestino
   {
      get
      {
         if (string.IsNullOrEmpty(archivoDestino))
            archivoDestino = Page.Request.Url.AbsolutePath;
         return archivoDestino;
      }
      set
      {
         if (!string.IsNullOrEmpty(value))
            archivoDestino = value;
      }
   }

   protected NameValueCollection información;
   protected override string ArgumentoVínculo(string archivo, NameValueCollection información)
   {
      string Codificado = información.CodificarDatosXml();
      string Comprimido = Codificado.ComprimirGZipUrl();

      string vínculo = archivo + "?wam=" + Comprimido;
      return vínculo;
   }
```

```
    protected override void Render(HtmlTextWriter writer)
    {
        bool primero = true;
        writer.BeginRender();
        writer.Write(encabezado);
        if (datos != null && datos is DataTable && datos.Rows.Count > 0 && !string.IsNullOrEmpty(elemento))
        {
            ArrayList Selección = new ArrayList();
            if (paginar && tamañoPágina > 0)
            {
                for (int i = páginaActual * tamañoPágina, j = 0; j < tamañoPágina && i < datos.Rows.Count; j++, i++)
                {
                    Selección.Add(datos.Rows[i]);
                }
            }
            else
            {
                Selección.AddRange(datos.Rows);
            }

            foreach (DataRow Fila in Selección)
            {
                string Item = elemento;
                string Expresión = @"<!\s*(?<Item>\bPB\b)\s*>";
                Regex Reg = new Regex(Expresión, RegexOptions.Compiled | RegexOptions.ExplicitCapture |
                    RegexOptions.IgnoreCase);
                MatchCollection Campos = Reg.Matches(Item, 0);
                foreach (Match Campo in Campos)
                {
                    Item = Item.Replace(Campo.Value, FunciónPostBack(Fila));
                }

                Expresión = @"<!\s*(?<Item>\bLINK\b)\s*>";
                información = new NameValueCollection();
                información["IDContenido"] = Fila["IDContenido"].ToString();
                Reg = new Regex(Expresión, RegexOptions.Compiled | RegexOptions.ExplicitCapture |
RegexOptions.IgnoreCase);
                Campos = Reg.Matches(Item, 0);
                foreach (Match Campo in Campos)
                {
                    Item = Item.Replace(Campo.Value, ArgumentoVínculo(archivoDestino, información));
                }

                Expresión = @"<!\s*(?<Item>\b\w*\b)\s*>";
                Reg = new Regex(Expresión, RegexOptions.Compiled | RegexOptions.ExplicitCapture |
RegexOptions.IgnoreCase);
                Campos = Reg.Matches(Item, 0);
                foreach (Match Campo in Campos)
                {
                    string NombreCampo = Campo.Groups["Item"].Value;
                    if (Fila.Table.Columns.Contains(NombreCampo + "F") && Fila[NombreCampo + "F"] != null &&
                        !(Fila[NombreCampo + "F"] is DBNull))
                        Item = Item.Replace(Campo.Value, Fila[NombreCampo + "F"].ToString());
                    else if (Fila.Table.Columns.Contains(NombreCampo) && Fila[NombreCampo] != null &&
!(Fila[NombreCampo]
                            is DBNull))
                        Item = Item.Replace(Campo.Value, Fila[NombreCampo].ToString());
                }
                writer.Write(Item);

                if (primero)
                {
```

```
                    writer.Write(separador);
                    primero = false;
                }
            }
        }
        writer.Write(pie);
        writer.EndRender();
    }

    private List<FiltroContenido> filtros;
    [PersistenceMode(PersistenceMode.InnerProperty)]
    public List<FiltroContenido> Filtros
    { get { if (filtros == null) filtros = new List<FiltroContenido>(); return filtros; } }
}

public abstract class FiltroContenido
{
    private string idCampo;
    public string IDCampo { get { return idCampo; } set { idCampo = value; } }
}

public class FiltroCategoría : FiltroContenido
{
    private string valor;
    public string Valor { get { return valor; } set { valor = value; } }
    private FiltroTexto tipoFiltro;
    public FiltroTexto TipoFiltro { get { return tipoFiltro; } set { tipoFiltro = value; } }
}

public enum FiltroTexto
{
    CoincidenciaExacta,
    IniciaCon,
    Contiene
}
public enum FiltroComparaciónFecha
{
    IgualA,
    MenorQue,
    MayorQue,
    MenorIgualQue,
    MayorIgualQue,
    DistintoA,
    Entre
}
public class FiltroFecha : FiltroContenido
{
    private DateTime valor;
    public DateTime Valor { get { return valor; } set { valor = value; } }
    private DateTime valorAux;
    public DateTime ValorAux { get { return valorAux; } set { valorAux = value; } }
    private FiltroComparaciónFecha tipoFiltro;
    public FiltroComparaciónFecha TipoFiltro { get { return tipoFiltro; } set { tipoFiltro = value; } }
}
}
```

5.3.1.2.4. Componente VisorContenido - Visualización de artículos (Website)

Muy similar al anterior en cuanto a las capacidades de adaptabilidad gráfica y uso de plantillas para el maquetado, ofrece la posibilidad de visualizar todo el detalle de un artículo en particular, seleccionado desde el listado correspondiente.

Igualmente se mostrará un ejemplo de interfaz con el fragmento de código que lo implementa, así como la codificación del control como tal.

Figura 5.11. Interfaz de muestra del control VisorContenido

FRAGMENTO DE CÓDIGO DEL ARCHIVO **NOTICIASV.ASPX**

```
<div id="info">
   <div class="título"><img src="Imgs/Titulo_Noticias.gif" border="0"/></div>
   <wam:VisorContenido runat="server" ID="wamVisor">
      <Formato>
         <div class="tituloElemento"><b><! Título ></b></div>
         <br />
         <div class="resumenElemento"><! Contenido ></div>
      </Formato>
   </wam:VisorContenido>
</div>
```

CÓDIGO DEL ARCHIVO **VISORCONTENIDO.CS**

```
using System;
using System.Collections;
using System.Collections.Generic;
using System.Collections.Specialized;
using System.Text.RegularExpressions;
using System.Web.UI;
using System.Web.UI.WebControls;
using SaethaWAM.Ejecución;
using SaethaWAM.Información;

namespace SaethaWAM.UI.Controles
{
   [ToolboxData("<{0}:VisorContenido runat=server></{0}:VisorContenido>")]
   [ParseChildren(true)]
   public class VisorContenido : Control
   {
      protected override void OnLoad(EventArgs e)
      {
         base.OnLoad(e);
         CargarDatos();
      }
      protected override void OnInit(EventArgs e)
      {
         Page.RegisterRequiresControlState(this);
         base.OnInit(e);
      }

      private ContenidoGeneral Contenido;
      private int idContenido;
      public int IDContenido
      {
       get { return idContenido; }
         set
         {
            if (value > 0)
            {
               try
               {
                  Contenido = ContenidoGeneral.DatosContenido(value);
               }
               catch (Exception ex)
               {
                  Sistema.RegistrarErrorControl(ex.GetType(), (Control)this, ex.Message);
               }
               idContenido = value;
            }
            else
            {
               Sistema.RegistrarErrorControl(typeof(ReferenciaNoVálidaException), (Control)this,
                     "El ID del contenido debe ser mayor a cero.");
               return;
            }
         }
      }

      protected void CargarDatos()
      {
         MostrarDatos();
      }

      protected void MostrarDatos()
      {
```

```
        NameValueCollection información;
        if (cargarQS)
        {
            if (Page != null && Page.Request != null && Page.Request.QueryString != null &&
                !string.IsNullOrEmpty(Page.Request.QueryString["wam"]))
            {
                información = new NameValueCollection();
                información.Clear();
                información.DecodificarDatosXml(Page.Request.QueryString["wam"].DescomprimirGZipUrl());
                if (!string.IsNullOrEmpty(información["IDContenido"]))
                {
                    IDContenido = int.Parse(información["IDContenido"]);
                }
            }
        }
        if (Contenido != null)
        {
            foreach(AsigControl Asignación in ListaControles)
            {
                if (!ContenidoGeneral.ExisteCampoContenido(Contenido.IDInstancia, Asignación.Campo))
                {
                    Sistema.RegistrarErrorControl(typeof(ElementoRequeridoException),(Control)this, "El campo '" +
                            Asignación.Campo + "' especificado en el control no está registrado para la instancia '" +
                            Contenido.IDInstancia + "'");
                }
                if (this.Page.FindControl(Asignación.IDControl) == null)
                {
                    Sistema.RegistrarErrorControl(typeof(ElementoRequeridoException), (Control)this, "El ID '" +
                            Asignación.IDControl + "' especificado no corresponde al de ningún otro control de la
página");
                }
                Control Control = this.Page.FindControl(Asignación.IDControl);
                if (!string.IsNullOrEmpty(Asignación.MiembroControl))
                {
                    if (Control.GetType().GetProperty(Asignación.MiembroControl) == null ||
                        !Control.GetType().GetProperty(Asignación.MiembroControl).CanWrite ||
                        Control.GetType().GetProperty(Asignación.MiembroControl).PropertyType !=
                            Contenido.DatosCampos[Asignación.Campo].GetType())
                    {
                        Sistema.RegistrarErrorControl(typeof(ReferenciaNoVálidaException), (Control)this, "El miembro '"
+
                            Asignación.MiembroControl + "' del control '" + Asignación.IDControl + "' no existe o no es
de un
                            tipo de datos válido para el campo '" + Asignación.Campo + "'.");
                    }
                    Control.GetType().GetProperty(Asignación.MiembroControl).SetValue(Control,
                            Contenido.DatosCampos[Asignación.Campo], null);
                    continue;
                }
                else
                {
                    if (Control is ListControl)
                    {
                        ListControl ControlLista = (ListControl)Control;
                        if (ControlLista.Items.FindByValue(Contenido.DatosCampos[Asignación.Campo].ToString()) !=
null)
                        {
                            ControlLista.ClearSelection();
                            ControlLista.SelectedValue = Contenido.DatosCampos[Asignación.Campo].ToString();
                        }
                        return;
                    }
```

```
                bool flag = false;
                string[] Propiedades = new string[] { "Text", "SelectedDate", "Value" };
                foreach (string Propiedad in Propiedades)
                {
                    if (Control.GetType().GetProperty(Propiedad) != null &&
                        Control.GetType().GetProperty(Propiedad).PropertyType ==
                            Contenido.DatosCampos[Asignación.Campo].GetType())
                    {
                        Control.GetType().GetProperty(Propiedad).SetValue(Control,
                            Contenido.DatosCampos[Asignación.Campo], null);
                        flag = true;
                        break;
                    }
                }
                if (flag) continue;

                flag = false;
                Propiedades = new string[] { "Text", "Value" };
                foreach (string Propiedad in Propiedades)
                {
                    if (Control.GetType().GetProperty(Propiedad) != null &&
                        Control.GetType().GetProperty(Propiedad).PropertyType == typeof(String))
                    {
                        Control.GetType().GetProperty(Propiedad).SetValue(Control,
                            Contenido.DatosCampos[Asignación.Campo].ToString(), null);
                        flag = true;
                        break;
                    }
                }
                if (flag) continue;
            }
        }
    }
    else
    {
        Sistema.RegistrarErrorControl(typeof(ElementoRequeridoException),(Control)this, "No se pueden mostrar
los
            datos de un contenido no especificado");
    }
}

protected override void LoadControlState(object savedState)
{
    base.LoadControlState(((ArrayList)savedState)[0]);
    this.IDContenido = (int)((ArrayList)savedState)[1];
}

protected override object SaveControlState()
{
    object anterior = base.SaveControlState();
    ArrayList nuevo = new ArrayList();
    nuevo.Add(anterior);
    nuevo.Add(idContenido);
    return nuevo;
}

private List<AsigControl> listaControles;

[PersistenceMode(PersistenceMode.InnerProperty)]
public List<AsigControl> ListaControles
{
    get
    {
```

```
            if (listaControles == null)
                listaControles = new List<AsigControl>();
            return listaControles;
        }
    }

    protected override void OnPreRender(EventArgs e)
    {
        MostrarDatos();
        base.OnPreRender(e);
    }

    private bool cargarQS = true;
    public bool CargarQS
    {
        get { return cargarQS; }
        set { cargarQS = value; }
    }

    protected string formato;
    [PersistenceMode(PersistenceMode.InnerProperty)]
    public string Formato
    {
        get { return formato; }
        set { formato = value; }
    }


    protected override void Render(HtmlTextWriter writer)
    {
        writer.BeginRender();
        if (Contenido != null)
        {
            string Item;
            string Expresión;
            Regex Reg;
            MatchCollection Campos;

            Item = formato;

            Expresión = @"<!\s*(?<Item>\b\w*\b)\s*>";
            Reg = new Regex(Expresión, RegexOptions.Compiled | RegexOptions.ExplicitCapture |
RegexOptions.IgnoreCase);
            Campos = Reg.Matches(Item, 0);
            foreach (Match Campo in Campos)
            {
                string NombreCampo = Campo.Groups["Item"].Value;
                Item = Item.Replace(Campo.Value, Contenido.DatosCampos[NombreCampo].ToString());
            }
            writer.Write(Item);
        }
        writer.EndRender();
    }
}

public class AsigControl
{
    private string idControl;
    public string IDControl
    {
        get { return idControl; }
        set { idControl = value; }
    }
```

```
        private string miembroControl;
        public string MiembroControl
        {
            get { return miembroControl; }
            set { miembroControl = value; }
        }

        private string campo;
        public string Campo
        {
            get { return campo; }
            set { campo = value; }
        }
    }
}
```

5.4. Pruebas

Las pruebas son el instrumento que permite validar y verificar el software, es decir, son los procesos que determinan si el software satisface los requisitos y funciona de la manera establecida. Por lo tanto, es necesario definir y aplicar un método para asegurar la calidad del software.

Las pruebas aplicadas al sistema se realizaron durante la codificación, de manera que los detalles se corregían a medida que se identificaban y permitieron avanzar con el desarrollo de una manera adecuada. También se realizó un repaso de las pruebas luego de terminar la codificación del sistema, para realizar ajustes y corregir los detalles que comúnmente se escapan al programador en el momento de la codificación.

5.4.1. Pruebas por unidad

Las pruebas por unidad se aplicaron mediante la prueba de la caja negra sobre los diversos componentes del sistema. Para realizar este tipo de pruebas se identifican un conjunto de valores que pueden ser introducidos por un actor y se expresaron como clases de equivalencia para poder abarcar la totalidad de las ocurrencias de un evento de inserción de datos.

A continuación se representan las clases de equivalencia del componente *P2Contenido_m*. Este será probado de acuerdo a los campos configurados en el componente *SaethaWAM.config* para la instancia de contenido “Noticias y novedades”, tal como se muestra a continuación:

```
<Contenido IDInstancia="Noticias" Nombre="Noticias" Descripción="Noticias y novedades"
  IncluirCamposPorDefecto="false" EstadoInicial="Publicado">
 <Campos>
  <Campo ID="Título" Nombre="Título" Tipo="TextoCorto"/>
```

```
    <Campo ID="Sumario" Nombre="Sumario" Tipo="TextoLargo" Opcional="true"/>
    <Campo ID="Contenido" Nombre="Contenido" Tipo="TextoFormateado"/>
    <Campo ID="FechaApertura" Nombre="Fecha de apertura del evento" Tipo="Fecha" Opcional="true"/>
    <Campo ID="FechaCierre" Nombre="Fecha de cierre del evento" Tipo="Fecha" Opcional="true"/>
    <Campo ID="Fotografía" Nombre="Fotografía" Opcional="true" Tipo="Archivo"/>
    <Campo ID="TipoNota" Nombre="Tipo de noticia" Tipo="Categoría" IDCategoría="TipoNota"/>
  </Campos>
  <Estilos>
    <Estilo ID="Subtítulo" Nombre="Subtítulo" CSS="font-family: Georgia; font-size: 16px; color: black;"/>
    <Estilo ID="Normal" Nombre="Nombre" CSS="font-family: Georgia; font-size: 12px; color: #808080;"
      Predeterminado="true"/>
  </Estilos>
</Contenido>
```

Se pueden distinguir los distintos tipos de campo especificados (textos cortos, largos y formateados, así como fecha, archivo y categoría), algunos de ellos marcados como opcionales (cuando no se especifica, el campo de forma predeterminada se establece como requerido).

Tabla 5.1. Clase de equivalencia para registrar un artículo en la instancia "Noticias y novedades" (parte 1/2)

Nº	DATO	CLASE DE EQUIVALENCIA	VÁLIDO	INVÁLIDO
1	***Sección***	Valor predeterminado *(Seleccione)*		**x**
2	***Sección***	Cualquier otro nodo del árbol	**x**	
3	***Fecha***	Cadena vacía		**x**
4	***Fecha***	Carácter alfanumérico (sin formato fecha)		**x**
5	***Fecha***	Carácter (con formato *dd/mm/yyyy*)	**x**	
6	***Título***	Cadena vacía		**x**
7	***Título***	Carácter alfanumérico	**x**	
8	***Título***	Longitud de caracteres > 2000		**x**
9	***Título***	1 <= Longitud de caracteres <= 2000	**x**	
10	***Sumario***	Cadena vacía (campo opcional)	**x**	
11	***Sumario***	Carácter alfanumérico	**x**	
12	***Sumario***	Longitud de caracteres > 2^{30} - 1		**x**
13	***Sumario***	0 <= Longitud de caracteres <= 2^{30} - 1	**x**	
14	***Fecha de apertura***	Cadena vacía (campo opcional)	**x**	

Tabla 5.2. Clase de equivalencia para registrar un artículo en la instancia "Noticias y novedades" (parte 2/2)

Nº	DATO	CLASE DE EQUIVALENCIA	VÁLIDO	INVÁLIDO
15	***Fecha de apertura***	Carácter alfanumérico (sin formato fecha)		**x**
16	***Fecha de apertura***	Carácter (con formato *dd/mm/yyyy*)	**x**	
17	***Fecha de cierre***	Cadena vacía (campo opcional)	**x**	
18	***Fecha de cierre***	Carácter alfanumérico (sin formato fecha)		**x**
19	***Fecha de cierre***	Carácter (con formato *dd/mm/yyyy*)	**x**	
20	***Fotografía***	Cadena vacía (campo opcional)	**x**	
21	***Fotografía***	Tamaño de archivo > 10MB		**x**
22	***Fotografía***	0 MB < Tamaño de archivo <= 10MB	**x**	
23	***Tipo de noticia***	Valor predeterminado *(Seleccione)*		**x**
24	***Tipo de noticia***	Cualquier otro valor cargado	**x**	
25	***Tipo de noticia***	Cadena vacía (modo adición)		**x**
26	***Tipo de noticia***	Carácter alfanumérico	**x**	
27	***Tipo de noticia***	Longitud de caracteres > 1000		**x**
28	***Tipo de noticia***	1 <= Longitud de caracteres <= 1000	**x**	
29	***Contenido***	Cadena vacía		**x**
30	***Contenido***	Longitud de caracteres > 2^{30} - 1		**x**
31	***Contenido***	1 <= Longitud de caracteres <= 2^{30} – 1	**x**	

En la *tabla 5.2* se presentan algunos casos de prueba, en los que se verifica que la salida sea la esperada de acuerdo a las clases de equivalencia determinadas en la *tabla 5.1*. Si se cumplen todas las clases involucradas, la salida será válida.

Tabla 5.3. Casos de prueba evaluados en la instancia "Noticias y novedades" (parte 1/2)

DATO	CASO DE PRUEBA	SALIDA	CLASE CUBIERTA
Sección	*(Seleccione)*	INVÁLIDO	1
Sección	*Novedades de la industria*	VÁLIDO	2
Fecha	*(Cadena vacía)*	INVÁLIDO	3
Fecha	*123456*	INVÁLIDO	4
Fecha	*10 de marzo de 2009*	INVÁLIDO	4
Fecha	*10/03/2009*	VÁLIDO	5
Título	*(Cadena vacía)*	INVÁLIDO	6
Título	*Saetha lanza el sistema SWAP v2.0*	VÁLIDO	7, 9
Título	*123456*	VÁLIDO	7, 9
Sumario	*(Cadena vacía)*	VÁLIDO	10
Sumario	*Cargado de creativas soluciones*	VÁLIDO	11, 13
Sumario	*123456*	VÁLIDO	11, 13
Fecha de apertura	*(Cadena vacía)*	VÁLIDO	14
Fecha de apertura	*123456*	INVÁLIDO	15
Fecha de apertura	*10 de marzo de 2009*	INVÁLIDO	15
Fecha de apertura	*10/03/2009*	VÁLIDO	16
Fecha de cierre	*(Cadena vacía)*	VÁLIDO	17
Fecha de cierre	*123456*	INVÁLIDO	18
Fecha de cierre	*10 de marzo de 2009*	INVÁLIDO	18
Fecha de cierre	*10/03/2009*	VÁLIDO	19
Fotografía	*Gigantografía.jpg (13,4MB)*	INVÁLIDO	21
Fotografía	*(Ningún archivo seleccionado)*	VÁLIDO	20
Fotografía	*ImagenPortada.jpg (640KB)*	VÁLIDO	22

Tabla 5.4. Casos de prueba evaluados en la instancia "Noticias y novedades" (parte 2/2)

DATO	CASO DE PRUEBA	SALIDA	CLASE CUBIERTA
Tipo de noticia	*(Seleccione)*	**INVÁLIDO**	23
Tipo de noticia	*Noticia de última hora (valor precargado)*	**VÁLIDO**	24
Tipo de noticia	*(Cadena vacía en modo adición)*	**INVÁLIDO**	25
Tipo de noticia	*Tecnologías de software (modo adición)*	**VÁLIDO**	26, 28
Contenido	*(Cadena vacía)*	**INVÁLIDO**	29
Contenido	***Subtítulo:** texto de prueba (con formato)*	**VÁLIDO**	31

5.4.2. Pruebas de integración

El objetivo principal de las pruebas de integración es detectar las fallas de interacción entre las distintas clases que componen al sistema. Debido a que cada clase probada por separado se inserta de manera progresiva dentro de la estructura, las pruebas de integración se convierten en el mecanismo que comprueba el correcto ensamblaje del sistema completo.

Las pruebas de integración se realizaron para todas las secciones del sistema; sin embargo, se explicará el proceso de estas pruebas sólo a través de un componente representativo, indicando los pasos seguidos. La funcionalidad a evaluar será la de *Contenido general*.

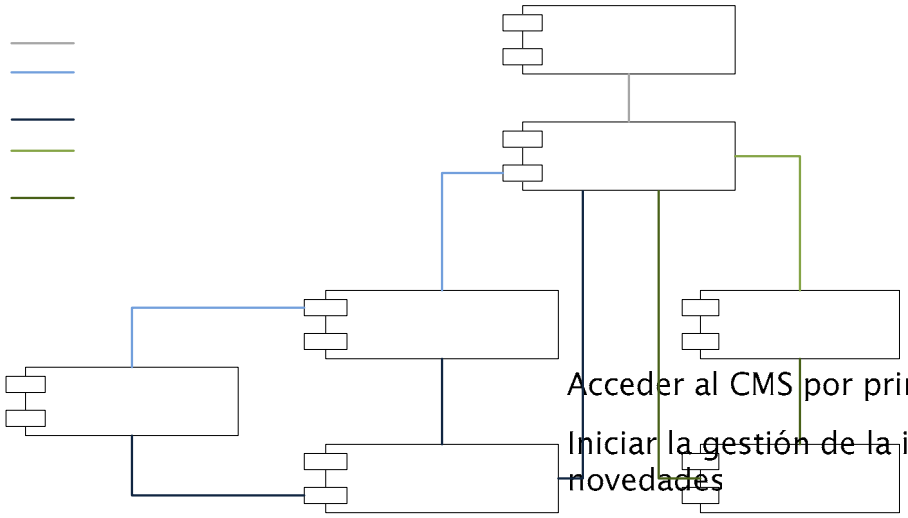


Figura 5.12. Diagrama de implementación con la secuencia de las pruebas de integración relacionadas con la funcionalidad de contenido general

5.4.2.1. Pasos de prueba

Paso 1

Abrimos el archivo *SaethaWAM.config* con algún editor de XML y comprobamos la etiqueta para la instancia de contenido general *Noticias y novedades,* usada en las pruebas por unidad. Revisamos que sus campos incluyan el título, sumario, contenido, fechas de apertura y cierre, el archivo de fotografía y el tipo de nota.

P2_C
P2_Co
«subsystem»
Ejecución
P2_Co
P2_Cor

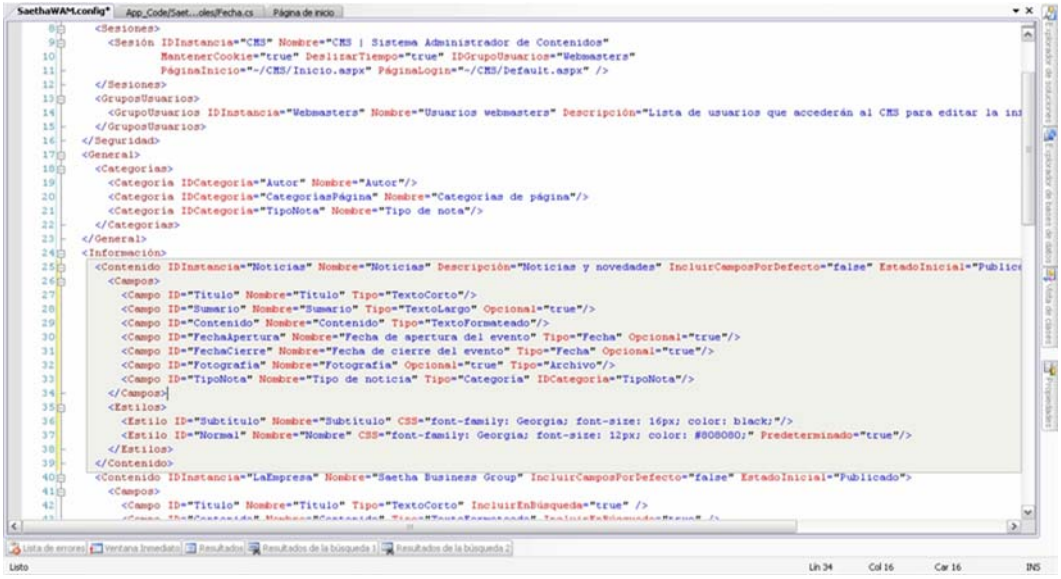


Figura 5.13. Captura de pantalla, contenido del archivo SaethaWAM.config

Paso 2

Acceder por primera vez el CMS. El servidor web recibe la primera solicitud de acceso y se ejecuta automáticamente el evento *Application_OnInit* declarado en el artefacto *global.asax*, ejecutándose la inicialización del modelo de objetos del componente *Configuración*. El mostrarse la pantalla de inicio de sesión por primera vez sin recibir ninguna notificación de error nos indica que todo el modelo de objetos de configuración fue levantado a memoria sin problema alguno.

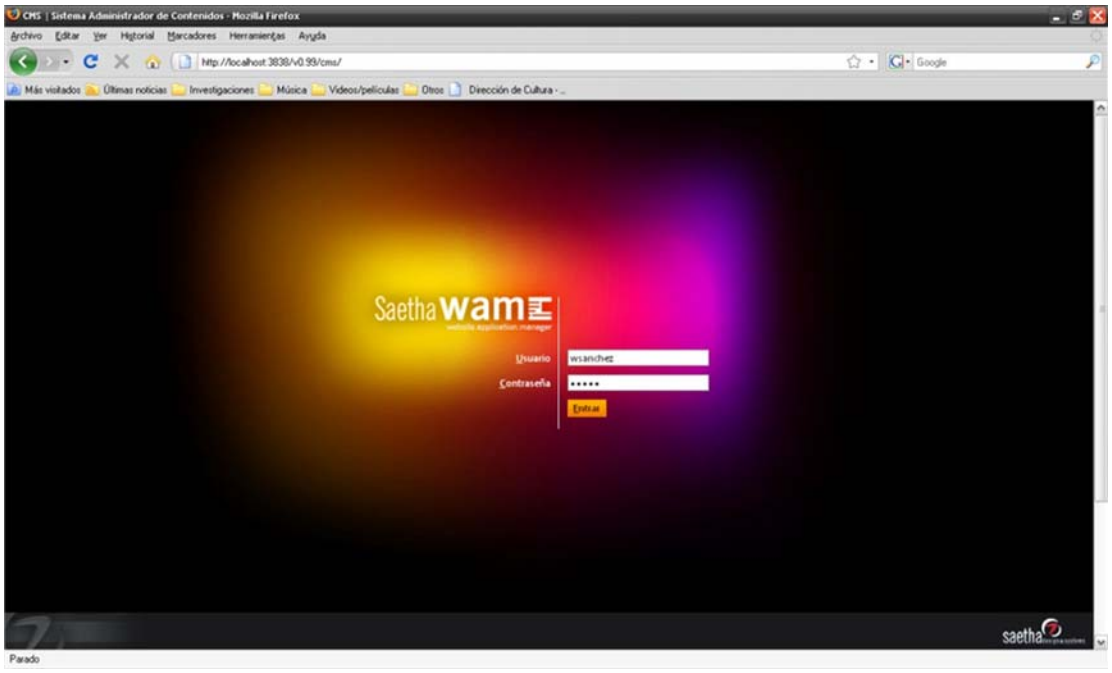


Figura 5.14. Pantalla de inicio de sesión, luego de haberse levantado el modelo de objetos de configuración

Paso 3

Una vez iniciada la sesión, ejecutamos la instancia *Noticias y novedades*, mostrándose el esquema de ejecución de aplicaciones de acuerdo al subsistema *Ejecución*. La pantalla que inicialmente se abre (*P2_Contenido_I*) muestra la lista de noticias previamente registradas en la instancia, de acuerdo a los datos proporcionados por la clase *ContenidoGeneral*.

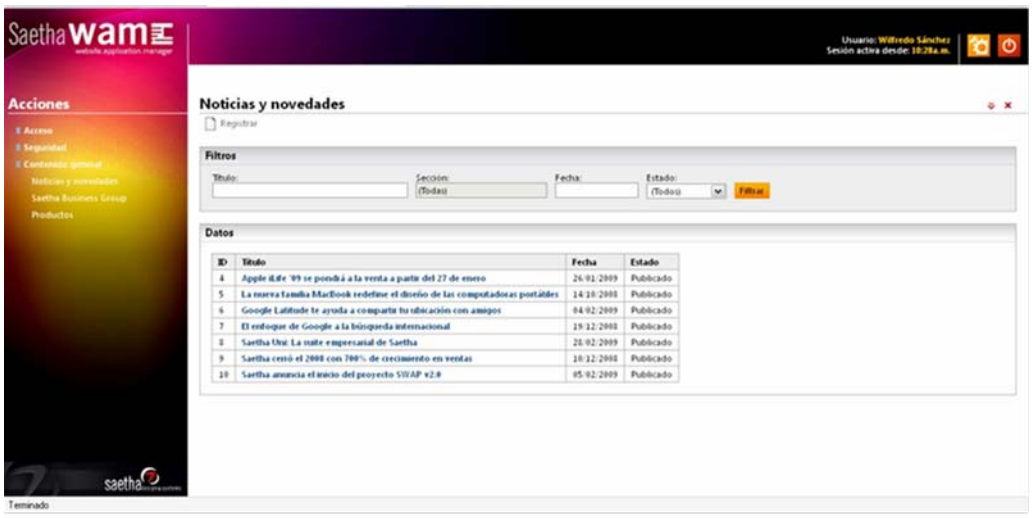


Figura 5.15. Lista de la instancia Noticias y novedades

Paso 4

Activamos la opción para registrar una nueva noticia. Se solicitarán, además de la sección y la fecha de registro, los restantes campos configurados en *SaethaWAM.config*. De esta forma se verifica la correcta integración entre los subsistemas *Configuración* y *General* (que incluye el modelo de negocios de todos los componentes de control).

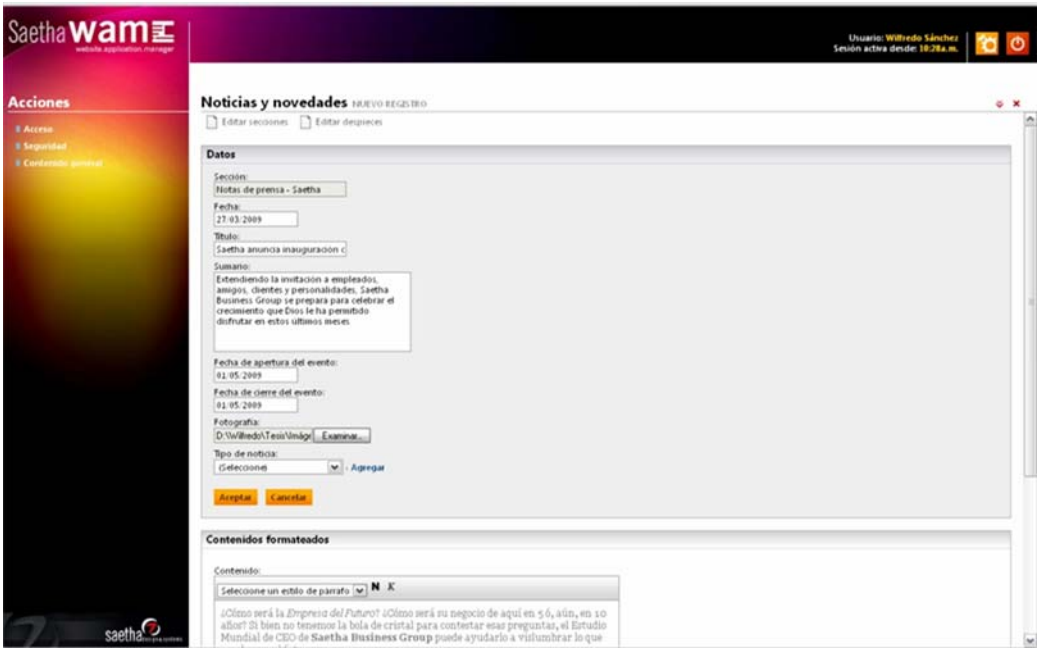


Figura 5.16. Pantalla de registro de un artículo para la instancia Noticias y novedades

Paso 5

Se puede verificar en la interfaz gráfica del RBDMS la correcta interacción de la aplicación con el manejador de base datos, visualizando en la tabla *Contenidos* la fila con los datos registrados en el *Paso 4*.

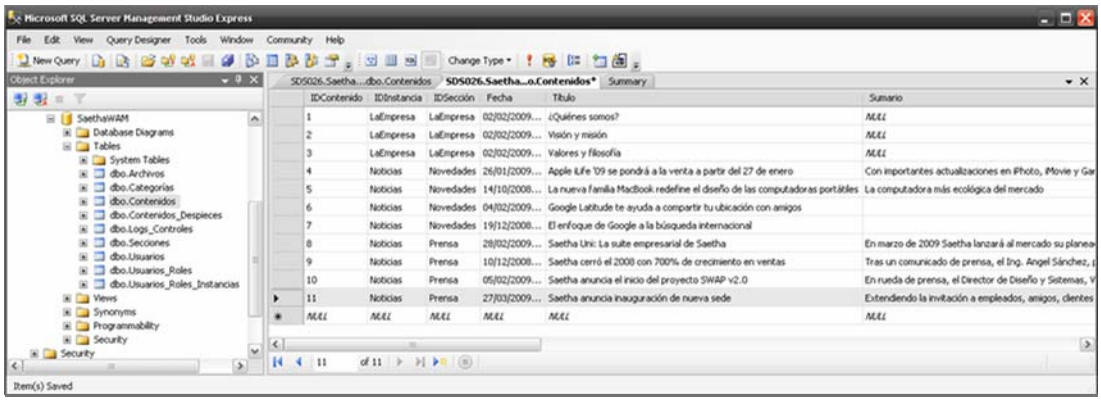


Figura 5.17. Visualización de la tabla de base de datos, con resalte de la fila recién registrada

Paso 6

Luego de haber registrado correctamente el artículo, pasamos a visualizar el ambiente público del website, en la página que implementó el control *ListaContenido*.

Figura 5.18. Página del área pública que lista las noticias y novedades registradas por medio de Saetha WAM, con el uso del control de servidor ListaContenido

Paso 7

Acceder al detalle del artículo recién registrado, navegando en el vínculo del título.

Figura 5.19. Página de detalle del artículo, configurada en el área pública con el uso del control VisorContenido

5.5. Evaluación de la fase de construcción

Durante la implementación se realizó la codificación de todos los componentes del sistema y en la iteración de prueba se efectuaron las pruebas respectivas, las cuales permitieron mejorar la calidad del software.

Las pruebas de unidad se aplicaron mediante el método de caja negra. Adicionalmente, las pruebas de integración se aplicaron para detectar y corregir fallas en el funcionamiento de los componentes y en la forma de acoplarse unos a otros.

Con la culminación de la fase de construcción, se ha obtenido como resultado un producto estable y maduro, es decir, la versión beta lista para ser entregada a la comunidad de usuarios finales y ser probada.

CAPÍTULO 6

FASE DE TRANSICIÓN

6.1. Introducción

La etapa de transición permite probar la calidad del software sobre la plataforma del entorno de operación. Es donde los usuarios empiezan a interactuar con el sistema y el desarrollador comienza a corregir defectos encontrados.

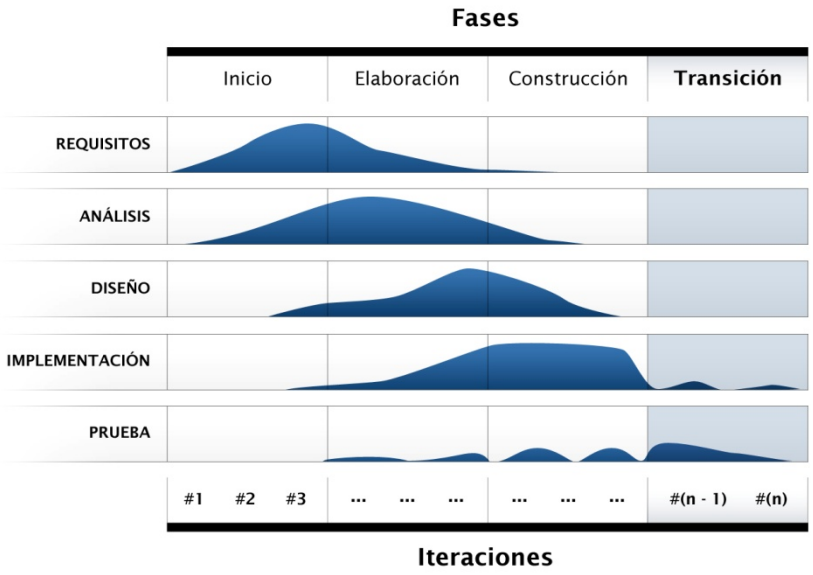


Figura 6.1. Diagrama de flujo de trabajo de las fases del proceso unificado, con identificación de la fase de transición

6.2. Planificación de la fase de transición

La atención se enfocará en asegurar que el software está disponible y operativo para los usuarios finales; se pondrá en funcionamiento para estos usuarios la versión beta del sistema, obtenida como resultado de completar la fase de construcción. Se realizarán las pruebas del producto como parte de su preparación para ser entregado, se entrenará al personal y se harán los ajustes menores que sean necesarios y surjan en respuesta a la retroalimentación recibida de dichos usuarios. Estos ajustes serán específicos y de corto alcance al producto, los cuales se resolverán junto a otros temas que puedan presentarse como configuración, instalación, y usabilidad, debido a que los ajustes estructurales mayores han sido solucionados durante las fases previas.

6.3. Evaluación de la fase de transición

La evaluación de esta fase dependerá de la corrección de defectos después de pruebas beta, modificaciones mínimas del software para solventar problemas no previstos que pudieran presentarse, y de la aceptación por parte del usuario de los resultados obtenidos. Se estima y garantiza que el producto final cumplirá completamente con los requisitos establecidos, funcionará de acuerdo a lo estipulado y superará las expectativas de los usuarios finales, quienes lo aceptarán y se adaptarán rápidamente a la funcionalidad y beneficios que el uso del sistema comprende.

CONCLUSIONES

- Se consideró apropiado diseñar con una alta complejidad el subsistema de configuración del sistema, obteniendo una gran flexibilidad en el uso del mismo.

- La codificación de los componentes de contenido como controles de servidor ASP.NET facilita en gran medida la adaptabilidad gráfica de sus implantaciones.

- El uso del subsistema *Ejecución* existente en Saetha Business Group ofrece una intuitiva y efectiva experiencia con el usuario que administrará los contenidos del sitio web.

- La codificación de funciones para la creación automática de las tablas de la base de datos (requiriendo tan sólo la configuración de la cadena de conexión a la misma) facilita en gran medida la puesta en ejecución del website, pues tanto las estructuras estáticas como los campos dinámicos de los controles se consideran al crearse automáticamente las tablas.

- La unificación en un mismo directorio virtual de las funcionalidades públicas del website así como del CMS para su administración mejoran la sincronía entre ambos ambientes para el acceso a datos de configuración, carga de archivos y configuración de la aplicación en servidores de hospedaje.

» La organización del código tanto a nivel físico (con el uso de estructura de directorios) como a nivel lógico (con el uso de *espacios de nombre*) y el aprovechamiento de las características fundamentales de la programación orientada a objetos facilita la reutilización del código y la adición de nuevos controles y componentes.

» El uso de los estándares XML y XSD para el modelado de las configuraciones facilita la comprensión de la misma y la validación de su contenido.

» Dado el nivel de análisis aplicado en la fase de inicio del *Proceso Unificado* sobre casi la totalidad de requisitos, así como la elaboración del diseño preliminar de las estructuras estáticas, permitieron tener una clara visión de todo el sistema, manteniéndose clara y poco alterada en el transcurso del proceso.

» La elaboración detallada de los diagramas de estructura estática en la fase de elaboración del *Proceso Unificado* facilitó la codificación de los componentes del software.

» La codificación de un componente (subsistema *Configuración)* antes de entrar a la fase de construcción ayudó a esclarecer el uso de las técnicas y herramientas, así como a aplicarlas correctamente en el desarrollo del resto de los componentes.

» Al finalizar la implementación de la totalidad de componentes, se pudo evidenciar el alto nivel de utilidad que, como herramienta, este trabajo representará dentro del marco de producción web de la empresa desarrolladora de software.

RECOMENDACIONES

» Considerando que el sistema ya se encuentra en funcionamiento bajo un período de *transición*, se recomienda codificar otros componentes funcionales bajo la misma plataforma de ejecución y actualización de contenidos. Estos podrían enfocarse en implementar funciones de blog, foros y encuestas, entre otras que puedan ser requeridas dentro de la producción de un website.

» Implantar el software en un website de administración interna a la empresa, para depurar cualquier detalle de usabilidad que se pueda optimizar.

» Mantener actualizados los documentos de diseño y manuales de usuario, con respecto a cualquier nuevo componente funcional, así como frente a modificaciones o actualizaciones realizadas al sistema. Llevar un registro de estos.

» Codificar nuevos servicios web XML's para facilitar la integración de contenidos con plataformas multimedia como Adobe Flash y Microsoft Silverlight, así como con sistemas elaborados en otros lenguajes o plataformas, como Java. Estos servicios web podrían ser codificados bajo el estándar SOAP para maximizar su interoperabilidad con otros sistemas y ambientes.

BIBLIOGRAFÍA

1. **Tenías Belmonte, Juan Carlos.** *Desarrollo de un software basado en aplicaciones web para el monitoreo de dispositivos de la plataforma de telecomunicaciones de PDVSA-GAS, Tesis.* 2007.

2. **Salazar Guzmán, Pedro Emilio.** *Desarrollo de un software para la automatización de reportes y consultas de archivos históricos del tráfico de conexiones de red realizada por la Superintendencia de Seguridad Lógica de una empresa petrolera utilizando tecnología web, Tesis.* 2007.

3. **Bilbao, Aivett y Arriojas, Edgar.** *Desarrollo de un sistema de soporte y atención al cliente para una empresa desarrolladora de software, Tesis.* 2007.

4. **Archer, T.** *A fondo C#.* Madrid, España : Editorial McGraw-Hill Interamericana, 2001.

5. **Pressman, Roger S.** *Ingeniería del Software: un enfoque práctico.* Quinta edición. Madrid : McGraw-Hill, 2002.

6. **Epidata Consulting SRL.** Epidata Consulting SRL. [En línea] http://www.epidataconsulting.com/tikiwiki/tiki-read_article.php?articleId=15.

7. **Object Management Group.** *UML Superstructure Specification, v2.1.1.* EEUU : s.n., 2007.

8. **Pilone, D. y Pitman, N.** *UML 2.0 in a Nutshell.* EEUU : Editorial O'Reilly, 2005.

9. **Cockburn, A.** *Writing Effective Use Cases.* Primera edición. s.l. : Editorial Addison-Wesley, 2001.

10. **Kendall, S.** *Fast Track UML 2.0.* Primera edición. s.l. : Editorial Apress, 2004.

11. **Elmasri, R. y Navathe, S.** *Fundamentos de Sistemas de Bases de Datos.* Tercera edición. Madrid : Editorial Pearson Educación, 2002.

12. **Cisco Systems, Inc.** *Programa de la Academia Networking Cisco.* Módulo 2. 2003.

13. **Microsoft Corporation.** Sitio web oficial de ASP.NET. [En línea] http://www.asp.net.

14. —. Sitio web oficial de documentación de Microsoft Visual Studio. [En línea] http://msdn.microsoft.com.

15. **Digital Project Laboratory, S.L.** *Wikilearning: un proyecto de aprendizaje colaborativo online.* [En línea] [Citado el: 24 de Agosto de 2008.] http://www.wikilearning.com/curso_gratis/gestion_de_riesgos_en_ingenieria_del_software-identificacion_del_riesgo/3620-3.

METADATOS PARA TRABAJOS DE GRADO, TESIS Y ASCENSO:

TÍTULO	"DESARROLLO DE UNA HERRAMIENTA DE SOFTWARE QUE DISMINUYA LOS TIEMPOS DE CREACIÓN Y ADMINISTRACIÓN DE CONTENIDO DINÁMICO EN WEBSITES Y APLICACIONES WEB PARA UNA EMPRESA DE DESARROLLO DE SOFTWARE"
SUBTÍTULO	

AUTOR (ES):

APELLIDOS Y NOMBRES	CÓDIGO CULAC / E MAIL
Sánchez V., Wilfredo E.	**CVLAC: V-17.262.710** **E MAIL: wsanchez@saetha.com**
	CVLAC: **E MAIL:**
	CVLAC: **E MAIL:**
	CVLAC: **E MAIL:**

PALÁBRAS O FRASES CLAVES:

Desarrollo web, Sistemas de administración de contenidos dinámicos

_____

_____

_____

_____

_____

_____

METADATOS PARA TRABAJOS DE GRADO, TESIS Y ASCENSO:

ÀREA	SUBÀREA
Ingeniería y Ciencias Aplicadas	Ingeniería en Computación

RESUMEN (ABSTRACT):

La presente investigación consiste en el desarrollo de un sistema web para una empresa desarrolladora de software que permitirá a los analistas del departamento de diseño web implantar funcionalidad de contenido dinámico a los sitios web diseñados, sin necesidad de escribir código imperativo. Para lograr esto sólo se requiere establecer en un archivo XML de configuraciones las estructuras de las secciones y de los datos a almacenar en cada una de ellas, y hacer referencia a las instancias de contenido desde los formularios web con el uso de etiquetas XML.

El núcleo del sistema analiza el archivo de configuración y genera automáticamente el *Sistema administrador de contenidos* (*CMS*, por sus siglas en inglés), así como la estructura de la base de datos. Se pueden registrar usuarios cliente que accederán al CMS y establecer las secciones y aplicaciones específicas que estos podrán actualizar.

El diseño y desarrollo del proyecto se realizó según las fases del Proceso Unificado de Desarrollo de Software y utilizando el Lenguaje Unificado de Modelado UML 2.0. La aplicación se construyó e implementó utilizando la tecnología ASP.NET 3.5, el lenguaje de programación C# 3.0 y el manejador de base de datos SQL Server 2005 Express Edition._____

METADATOS PARA TRABAJOS DE GRADO, TESIS Y ASCENSO:

CONTRIBUIDORES:

APELLIDOS Y NOMBRES	ROL / CÓDIGO CVLAC / E_MAIL				
García R., Zulirais A.	ROL	CA	AS X	TU	JU
	CVLAC:	V-10.299.576			
	E_MAIL	zzzuliii@hotmail.com			
	E_MAIL				
Cortínez N., Claudio A.	ROL	CA	AS	TU	JU X
	CVLAC:	V-12.155.334			
	E_MAIL	cl_cortinez@cantv.net			
	E_MAIL				
Rodríguez M., Rhonald E.	ROL	CA	AS	TU	JU X
	CVLAC:	V-14.077.185			
	E_MAIL	rerodriguez@anz.udo.edu.ve			
	E_MAIL				
	ROL	CA	AS	TU	JU
	CVLAC:				
	E_MAIL				
	E_MAIL				

FECHA DE DISCUSIÓN Y APROBACIÓN:

2009	05	13
AÑO	MES	DÍA

LENGUAJE. SPA

METADATOS PARA TRABAJOS DE GRADO, TESIS Y ASCENSO:

ARCHIVO (S):

NOMBRE DE ARCHIVO	TIPO MIME
TESIS.SaethaWAM.doc	Application/msword

CARACTERES EN LOS NOMBRES DE LOS ARCHIVOS: A B C D E F G H I J K L M N O P Q R S T U V W X Y Z. a b c d e f g h i j k l m n o p q r s t u v w x y z. 0 1 2 3 4 5 6 7 8 9.

ALCANCE

ESPACIAL: _____ **(OPCIONAL)**

TEMPORAL: _____ **(OPCIONAL)**

TÍTULO O GRADO ASOCIADO CON EL TRABAJO:

Ingeniero en Computación

NIVEL ASOCIADO CON EL TRABAJO:

Pre-Grado

ÁREA DE ESTUDIO:

Departamento de Computación y Sistemas

INSTITUCIÓN:

Universidad de Oriente – Núcleo de Anzoátegui

METADATOS PARA TRABAJOS DE GRADO, TESIS Y ASCENSO:

DERECHOS

De acuerdo con el artículo 44 del reglamento de trabajo de grado:

"Los trabajos de grado son de exclusiva propiedad de la Universidad de Oriente y sólo podrán ser utilizados a otros fines con el consentimiento del consejo de núcleo respectivo, quien lo participará al Consejo Universitario".

AUTOR **AUTOR** **AUTOR**

TUTOR **JURADO** **JURADO**

POR LA SUBCOMISIÓN DE TESIS