**UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS**
**FACULTAD 3**


# Servicio de asistencia al proceso de consulta popular en la Facultad de Derecho de la Universidad de la Habana

---

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

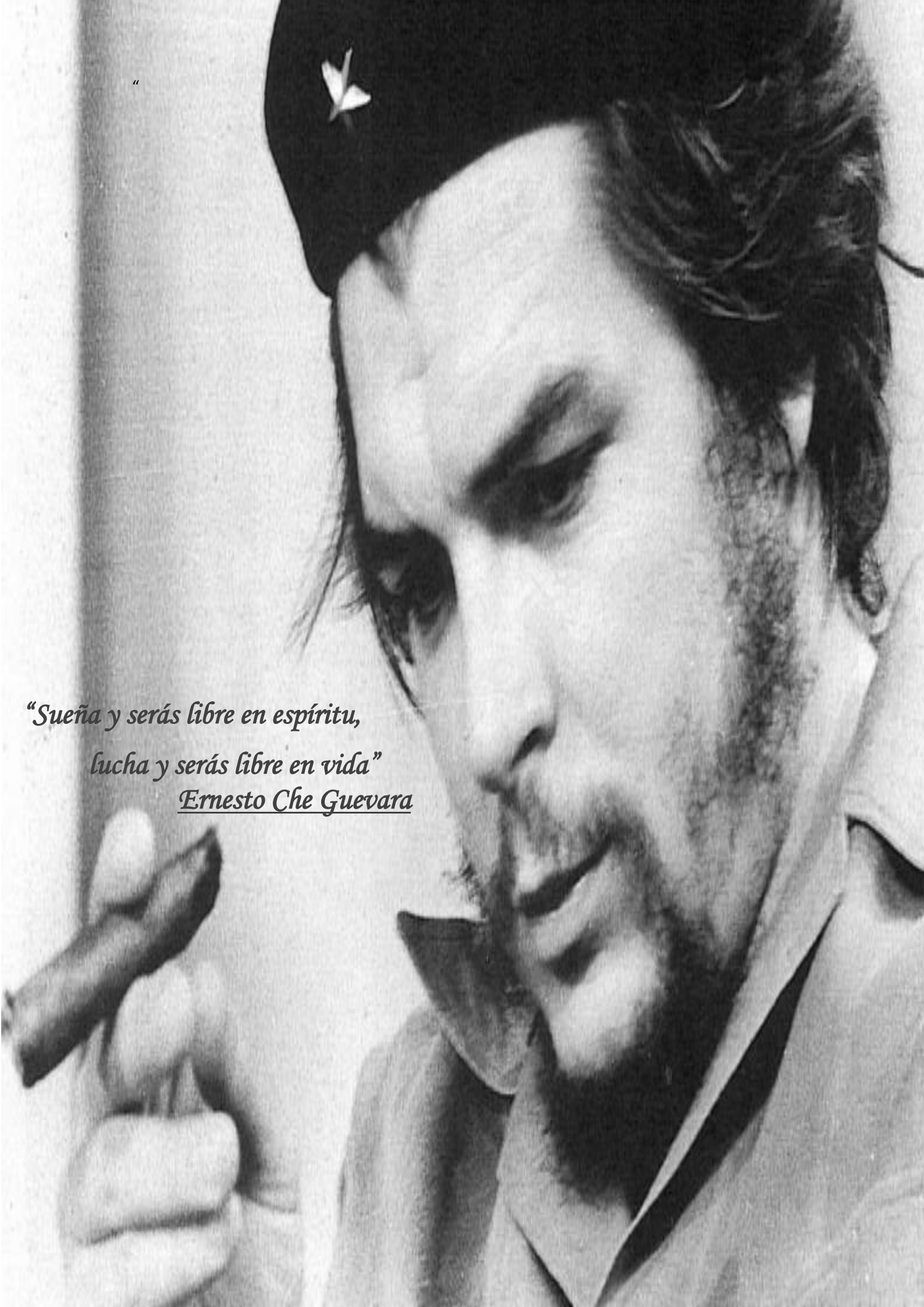
---

**Autores:**
Alejandro Miguel Pedraza Fernández
Gabriel Alejandro González Olivera

**Tutor:**
MSc. Yarina Amoroso Fernández.

**Co-Tutor:**
Ing. Daniel Varona Cordero.

**Ciudad de la Habana, Junio del 2014.**
**"Año 56 de la Revolución"**



*"Sueña y serás libre en espíritu,*

*lucha y serás libre en vida"*

*Ernesto Che Guevara*

Declaramos que somos los únicos autores del trabajo “Servicio de asistencia al proceso de consulta popular en la Facultad de Derecho de la Universidad de la Habana” y autorizamos a la Facultad 3 de la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio. Para que así conste firmamos la presente a los ____ días del mes de _____ del año _____.

_____

Gabriel Alejandro González Olivera.

Autor

_____

Alejandro Miguel Pedraza Fernández.

Autor

_____

MSc. Yarina Amoroso Fernández.

Tutor

_____

Ing. Daniel Varona Cordero.

Co-Tutor

*A la gracia de Dios por permitir cumplir mis sueños.*

*A la bella mujer que nunca ha sabido negarme su mano, la mujer que le debo la vida y todo lo que soy, ella que ha jugado diversos roles en su vida solo con el fin de formarme, de hacerme grande, tú, que has sido mi amiga, mi confidente, mi madre, a ti, que eres mi luz, mi ejemplo hoy y siempre. Gracias, Gracias mamá.*

*A mis abuelos, siempre alegres y buscándole solución a todo, los amo, Gracias por todo el cariño que desde los primeros días de vida impregnaron en mí. Por nunca detenerse a la hora de complacerme o de hacerme sentir bien, ahora les digo que me toca complacerlos a ustedes.*

*A mi padre bailarín y exigente en todo momento, gracias por todo tu apoyo, por tus consejos y charlas.*

*A Ania por sus atenciones, preocupaciones y por las veces que me hizo reír con sus inigualables pasos de salsa.*

*A ese hermano mayor que aunque se encuentra muy lejos quisiera mostrarle toda mi gratitud y decirle ¨Te quiero hermano¨.*

*A Maire, mi compañera en el amor, en el estudio, a ti mil gracias por estar ahí en esos momentos difíciles que solo tú logras entender, por tus consejos, tu apoyo incondicional, decirte que siempre te tendré presente, que en este corazón siempre habrá un espacio reservado para ti.*

*A mi hermano y amigo, Adrian Peña, gracias miles pues sin ti este logro no habría sido igual. Gracias por estar siempre disponible y nunca decir no cuando necesitaba tu ayuda.*

*A todos aquellos amigos de la infancia y del barrio, esos que nunca se olvidan y llegan para quedarse, Eniel, Luis Ángel, El Daro, El Yely, Corona, Aliek, Raulito, Pedrito y Laura, Sureny, Ari, Yoan, la Yaile ,Jachero y Leidys.*

*Agradecer además a esos primeras amistades que se hacen a penas se pisa suelo universitario como mi negrito querido el chocolatico Duvergel, Al chulin Abelito , a Fernan , Treto quien jugó un papel fundamental en la realización de este trabajo, Erick Al Triple, a los dos Carlos, el habanero y el que se hizo habanero a la fuerza ,a Camilo, el Titi ,Yoan ,Eduardo, Raúl, Adrian, Celso, Alejandro, Héctor, Mailenis el manguito*

*de Ciego, Gelsys, Danaray, Greter, Neny, Dianelis, Yaidelis, Aroldo, Osiel, Carola, Gemita, Ariel, Noraivis, Pedro Romero, Chaian, Adonis la voz de oro, Dayán, Lázara la tía, Lisvany, mi mamita de universidad.*

*A todo el piquete de los Dj que me permitieron vivir divertidos momentos con todos los chistes y bromas que a cada rato nos lanzábamos.*

*A mi segundo y último grupo gracias a todos, Mi otro amigazo el Félix, Samuel Juanito, Rey, Allen, Leo mi niña linda la señorita Dayana, Mirisleidys, Leidys, Rosana, Nailyn, Carlos, Yoendry, Gracias a todo el grupo en general.*

*A todos esos profesores que dejaron su huella y ejemplo en mí como Dariela, Pedro Nogales, Magdanis, Hayde, Rainer, Sandor y todos con los que tuve la oportunidad de relacionarme.*

*Agradecimientos además para mi gran compañero de Tesis que a pesar de las peleas cariñosas y jocosas k teníamos, supimos mantener una magnífica relación y logramos enfocarnos completamente en la meta trazada, ser todo un profesional.*

*A los tutores que fueron de gran ayuda en todo el transcurso de la investigación.*

*-Para finalizar, con la idea de mostrar mi más sincera gratitud a todos sin que se quede nadie-*

*"A los maravillosos amigos de todos los tiempos aquí en la UCI, a todas esas personas con que tuve la oportunidad de compartir al menos un instante…. Gracias a todos"*

*Gabriel Alejandro González Olivera*

*A mis padres por su apoyo incondicional y confianza, hoy hemos alcanzado una meta de las tantas que alcanzaremos en un futuro.*

*A mis tutores por ser nuestros guías en el cumplimiento de este sueño que nos trazamos al inicio de la Universidad.*

*A mis amistades del aula, Aliesky, Leandro, Ernesto, Allens, en fin a todo el grupo, gracias por pasar estos 5 años juntos.*

*A mis nuevas amistades, Maire y Peña.*

*A mi compañero de tesis por soportarme estos 10 meses, y si tuviera que volver a hacer la tesis, pues volveríamos a ser compañeros, aunque sigamos peleando el uno con el otro a la hora de trabajar.*

*A mi novia por ser mi apoyo estos 5años de la carrera, gracias por estar siempre ahí para mí. Te amo.*

*Alejandro Miguel Pedraza Fernández*

*A mi hermosa madre*

*A mis abuelos*

*A mi padre*

*A esa encantadora familia que siempre confió en mí, que siempre esperaron la victoria tras el reto trazado.*

*Gabriel Alejandro González Olivera*

*A mi familia por ser mi fuerza, especialmente a mi hermano y a ese pequeño que viene en camino. Espero ser un ejemplo a seguir algún día.*

*Alejandro Miguel Pedraza Fernández*

# RESUMEN

Por la necesidad de mejorar el actual proceso de consulta popular a partir de su integración al proyecto de rediseño del sitio web de la Facultad de Derecho en la Universidad de la Habana, se realiza un estudio para seleccionar las herramientas informáticas más factibles, teniendo en cuenta las tendencias del desarrollo web, se analizan algunos de los sistemas similares definiendo posibles funcionalidades para la solución propuesta, agregándole las necesidades del cliente. El desarrollo se divide en las fases propuestas por la metodología seleccionada, documentando cada paso y cada artefacto generado. Para verificar el cumplimiento de las funcionalidades propuestas se realizan las pruebas de aceptación, pruebas de usabilidad y pruebas unitarias. Como resultado se obtiene un producto funcional con una documentación que sirve de base para futuras investigaciones o modificaciones a la propuesta de solución. El resultado de la investigación deja constancia documental de la metodología empleada, que permitió cumplir el objetivo general propuesto y la satisfacción de las necesidades del cliente.

**Palabras claves:** consulta popular, portal web.

# ÍNDICE

# ÍNDICE DE FIGURAS

# ÍNDICE DE TABLAS

# INTRODUCCIÓN

Cuba se encuentra inmersa en un proceso de rediseño de su modelo económico y social. Debido a ello, recientemente se han estado llevando a cabo un conjunto de acciones encaminadas a la actualización del cuerpo jurídico para su adecuación a las características actuales y entorno en que se desenvuelve el ciudadano cubano; entiéndase lineamientos que rigen la política económica y social del Partido y la Revolución, así como el nuevo Código de Trabajo.

Ambas experiencias movilizaron al pueblo en un proceso de consulta en ejercicio de participación ciudadana, para que estos anteproyectos de Ley resultasen enriquecidos tras un número de modificaciones, como se notará a continuación.

Primeramente para el caso de los lineamientos de la política social y económica, el proceso se comportó de la siguiente forma: en 3 rondas durante las cuales de 291 lineamientos originales, tras 283 modificaciones propuestas, resultaron 313 lineamientos aprobados, para ver detalles de lo antes planteado ver Anexo 1.

Mientras que para el caso del nuevo Código de Trabajo, tras una minuciosa revisión, el escrutinio motivó la modificación de más de 100 artículos, la inclusión de 28 nuevas normativas, y la total reelaboración del Capítulo II -referente a las organizaciones sindicales-, del artículo 2, sobre los principios que rigen el derecho de trabajo, y el cambio parcial del Capítulo XV, dedicado a las autoridades laborales.

*Per sé* el proceso de consulta tiene alusión en la Constitución de la República de Cuba en el segundo punto de su artículo 75, donde expone entre las atribuciones de la Asamblea del Poder Popular: “aprobar, modificar o derogar las leyes y someterlas previamente a la consulta popular cuando lo estime procedente en atención a la índole de la legislación de que se trate”

En la práctica ambas experiencias demostraron que es un proceso de participación masiva, que requiere de considerable contacto con las masas ciudadanas en cada centro laboral o de estudio, en los Comités de Defensa de la Revolución CDR, etc… y que cada opinión es importante. El cotejo, catalogación, y resumen de cada parte resultante de esas reuniones fue llevado a cabo de forma manual y el proceso de cotejo general puede asumirse que requirió un esfuerzo considerable.

Por otro lado, la Facultad de Derecho de la Universidad de la Habana, en completa concordancia con las políticas generales de informatización de la sociedad, está determinada a actualizar sus potencialidades de interacción en las redes mediante la modernización de su sitio web. En un intento por llevar al mundo virtual la mayor cantidad de comportamientos sociales en el mundo fuera de lo virtual y parecerse más a sus estudiantes y profesores.

Luego, bajo la asunción de que seguirán sucediéndose nuevas experiencias modificadoras del cuerpo jurídico, tras el impacto en el pueblo de las mencionadas anteriormente, y el análisis de las carencias del proceso de consulta popular como proceso manual en extremo engorroso; puede identificarse la actualización del sitio de la Facultad de Derecho de la Universidad de la Habana como una oportunidad de informatización de dicho proceso, aprovechando que la comunidad de usuarios de esta institución posee un conocimiento peculiar sobre los temas legislativos.

Debido a la problemática anteriormente expuesta, es posible identificar el siguiente **problema a resolver:** ¿Cómo mejorar el actual proceso de consulta popular a partir de su integración al proyecto de rediseño del sitio web de la Facultad de Derecho en la Universidad de la Habana?

Con vista a la solución del problema se define como **objeto de estudio:** Proceso de consulta popular. La investigación se encuentra enmarcada en el **campo de acción:** Sistema de asistencia al proceso de consulta popular, trazándose como **objetivo general:** Desarrollar el servicio de asistencia a la consulta popular integrado al portal web de la Facultad de Derecho de la Universidad de la Habana para mejorar el proceso de consulta popular como ejercicio de participación ciudadana, definiendo como **idea a defender:** El desarrollo del servicio de asistencia a la consulta popular integrado al nuevo portal web de la Facultad de Derecho de la Universidad de la Habana, constituye una mejora al proceso de consulta popular como ejercicio de participación ciudadana.

Para alcanzar los resultados esperados en la investigación se definieron las siguientes tareas investigativas:

1. Estudio de los principales conceptos relacionados al proceso de consulta popular.
2. Análisis y caracterización de sistemas homólogos a la solución.
3. Estudio de las características fundamentales y herramientas tecnológicas utilizadas para desarrollar la solución.
4. Levantamiento y especificación de requisitos que debe comprender la solución.
5. Elaboración de los artefactos que se generan según los requisitos adquiridos.
6. Diseño de los prototipos de interfaz de usuario.
7. Diseño del modelo de datos conveniente a la solución.
8. Implementación de los requisitos funcionales identificados.
9. Validación de la solución mediante los métodos y técnicas de pruebas previamente seleccionadas.

El presente estudio se desarrolló con el empleo de los **métodos científicos de investigación**.

Como métodos teóricos se utilizaron:

**Método histórico lógico**: el uso de este método permitió la realización del estado del arte de la parte de la ciencia que está siendo objeto de investigación, las tecnologías, lenguajes, metodologías y tendencias actuales para el desarrollo de portales web.

**Método analítico-sintético:** permitió el análisis y esencia de los fenómenos estudiados para la elaboración de la teoría que sustenta la investigación.

**Modelación:** se utilizó con el objetivo de realizar una abstracción para dar una explicación detallada de la estructura en términos de diseño de la solución propuesta.

Como métodos empíricos se empleó:

**Entrevista (fundamentalmente a los clientes):** permitió conocer y delimitar las funcionalidades a desarrollar.

**Observación**: se utiliza para determinar la evolución de los procesos, específicamente en las validaciones de las funcionalidades de la solución.

La siguiente investigación está estructurada en 3 capítulos. A continuación se muestra una breve descripción de los temas a tratar por cada capítulo.

**Capítulo 1**. Fundamentación teórica: en este capítulo se abordan los aspectos teóricos que dan sustento a la investigación y que corresponden al análisis detallado de las metodologías, herramientas y tecnologías a utilizar en el proceso de desarrollo de la aplicación. Se exponen definiciones claves para una fácil comprensión del trabajo de diploma.

**Capítulo 2**. Propuesta de solución: en este capítulo se presenta el diseño de la propuesta de solución al problema planteado, que contará con la determinación de los requisitos funcionales y no funcionales de la aplicación, así como diseño y estructura de la misma.

**Capítulo 3**. Implementación y prueba: en este capítulo se explica cómo se construyó la aplicación. Se describen, además, las pruebas realizadas con el objetivo de validar el correcto funcionamiento de la solución.

# CAPÍTULO 1. FUNDAMENTACIÓN TEÓRICA.

## 1.1. Introducción

En este capítulo se describen los elementos principales que fundamentan el contenido de la investigación. Se realiza un análisis valorativo de sistemas informáticos existentes asociados al desarrollo del servicio de asistencia al proceso de consulta popular. También se realiza una breve descripción de la metodología, tecnologías y herramientas que fueron utilizadas para realizar el diseño y la implementación de la solución.

## 1.2. Marco conceptual

Primeramente para el desarrollo de este sistema de gestión de información se hace énfasis en los elementos conceptuales que enmarcan la investigación que se presenta; como lo pueden ser: la consulta popular, los portales web como una tendencia en este tipo de problemáticas.

De manera general, y por ser el objeto primario de la aplicación que se propone se hace necesario determinar lo que se entenderá en lo adelante por Proyecto legislativo.

### 1.2.1 Proyecto legislativo

Los proyectos legislativos, en cualquiera de sus clases: Ley, Resolución, Comunicación, Declaración o Decreto –por solo mencionar algunos- son la herramienta principal con la que cuenta el legislador para expresar o promover las decisiones propias; aquellas atinentes al cumplimiento de sus funciones**.**

La forma, tramitación, aprobación o legitimación para presentar un Proyecto legislativo dependerá del ordenamiento jurídico de cada estado y, en particular, de lo dispuesto en su Constitución (1). Para el caso de la presente investigación el ordenamiento es el greco-romano y entre sus métodos relacionados se encuentra la consulta popular.

### 1.2.2 Consulta popular

En gran parte de Latinoamérica se convoca a una consulta popular a la hora de tomar decisiones respecto a un tema de interés para la sociedad. En la Ley Federal de Consulta Popular de la republica de México en el artículo 4, se define consulta popular como un mecanismo de participación por el cual los ciudadanos ejercen su derecho a través del voto emitido, mediante el cual expresan su opinión respecto a uno o varios temas de trascendencia nacional.

En la Constitución de la República de Cuba en el segundo punto de su artículo 75, se expone que entre las atribuciones de la Asamblea del Poder Popular están: “aprobar, modificar o derogar leyes y someterlas previamente a la consulta popular cuando lo estime procedente en atención a la índole de la legislación de que se trate”. (2)

Las consultas populares, o sufragios populares, en derecho constitucional y en la historia constitucional, son deliberaciones públicas tomadas por el país (toma de decisiones) como cuerpo electoral y cuerpo de legislación. Existen distintos tipos de consultas que se toman en el ejercicio de una forma de participación política, y cada vez el pueblo llega a la decisión de forma directa sobre algo sometido a su voluntad, tanto los órganos del estado como los ciudadanos ejercen una forma de democracia directa.

Decisiones sobre la base de consultas populares

- ✓ Decisión de la votación, acerca de los electos, regulada por la constitución y las leyes que determina el sistema electoral.
- ✓ Decisión de la iniciativa legislativa, acerca de un proyecto de Ley.
- ✓ Decisión de una petición, acerca de una solicitud.
- ✓ Decisión de un plebiscito, acerca de una propuesta o una controversia política.
- ✓ Decisión de la revocatoria del mandato o referéndum revocatorio, acerca de un funcionario electo.
- ✓ Decisión del referéndum, acerca de una propuesta de ley específica.

Dado el desarrollo de las tecnologías de la información y las telecomunicaciones el espectro de aplicación de las TICs ha ido modificando y delimitando a su vez cada esfera de la sociedad. Existe una rama dentro de esta orientada a modelar interacciones sociales en entornos en-Línea, y de manera general, determinada por el área de aplicación, a la informática jurídica.

### 1.2.3 Informática Jurídica

La informática jurídica surge en el año 1949, en los Estados Unidos de América y tiene como propósito la aplicación de la informática para la recuperación de información jurídica, así como la elaboración y aprovechamiento de los instrumentos de análisis y tratamiento de dicha información, necesarios para una toma de decisión con repercusiones jurídicas.

La Informática jurídica se puede clasificar de la siguiente forma atendiendo a sus núcleos polémicos:

- ✓ Documental: relacionada con el almacenamiento y recuperación de documentos.
- ✓ De control y gestión: relacionado con el desarrollo de actividades jurídico-adjetivas.
- ✓ Meta documental: vinculada al apoyo en la decisión, educación, investigación, redacción y previsión del derecho.(3)

El desarrollo de la primera de estas clasificaciones, se debe, en gran medida, al aumento considerable de la documentación, que hacen ver obsoletos a los métodos tradicionales de búsqueda. Esto hace que actualmente se creen sistemas de Informática jurídica que posean un

banco de datos jurídicos, con el objetivo de interrogarlo teniendo en cuenta criterios propios en base a esa información y su relevancia jurídica. La segunda clasificación se usa sobre todo para agilizar el proceso administrativo-legislativo, brindándole mayor rapidez y eficiencia, mientras que la tercera se vincula más bien con el tema de la investigación y resultados de esta.

De manera transversal a estos núcleos están los enfocados al tratamiento de la Calidad y al desarrollo de tecnología para el operador jurídico. Englobando este último todas aquellas herramientas, métodos, modelos, etc… que brindan una aproximación a determinado problema desde una arista informática. Específicamente para el caso que se presenta, y por la adaptación al contexto de análisis la aproximación que se llevará a cabo debe no menos que estar contemplada en un ambiente web.

### 1.2.4 Portal web

Un portal web o sitio web tiene como objetivo ofrecer al usuario, de forma fácil e integrada, el acceso a una serie de recursos y de servicios, entre los que suelen encontrarse buscadores, foros, documentos, aplicaciones, compra electrónica, etc. Principalmente están dirigidos a resolver necesidades específicas de un grupo de personas o de acceso a la información y servicios de una institución pública o privada. (4)

Los portales pueden clasificarse en horizontales, verticales y diagonales como puede verse a continuación.

**Horizontales:** son los llamados portales masivos, se dirigen a los usuarios en general e incluso a los usuarios corporativos, tratando de llegar a toda la multitud con muchos objetos.

**Verticales**: son los que ofrecen contenidos específicos por temas como salud, finanzas, música y deportes.

**Diagonales:** son una mezcla de las dos anteriores.

Otra definición es la del autor Juan Carlos García Gómez (5) , que define un portal web como: “ un punto de entrada a internet donde se organizan sus contenidos, ayudando al usuario y concentrando servicios y productos, de forma que le permitan realizar cuanto necesite hacer en la red diariamente, o al menos que pueda encontrar allí todo cuanto utiliza cotidianamente sin necesidad de salir de dicho sitio”.

La implantación de servicios asociados al proceso de consulta popular enmarcado en un espacio web que cuente con una comunidad de usuarios con un conocimiento peculiar sobre los temas legislativos mejora al actual proceso de consulta popular. Por ello es necesario estudiar sistemas homólogos que aporten soluciones al problema anteriormente planteado.

## 1.3. Estado del arte

Se presenta un análisis de los diferentes sistemas estudiados para resolver el problema antes planteado, los cuales permitirán definir un mecanismo para realizar la verificación y toma de decisiones en el momento de realizar un nuevo Proyecto legislativo.

### 1.3.1. Sistemas existentes a nivel internacional.

La siguiente figura muestra las soluciones analizadas.

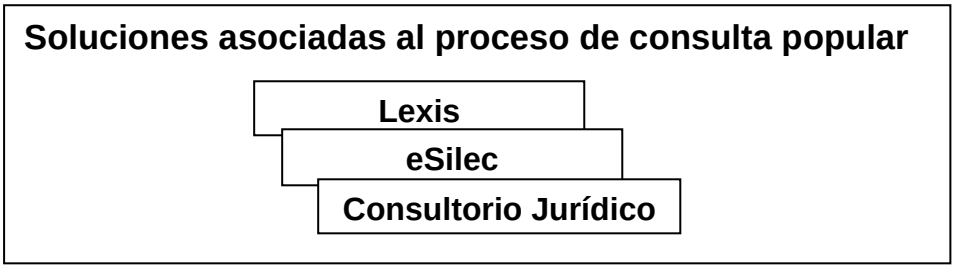



**Figura 1.** Conjunto de soluciones analizadas. (Elaboración propia)

#### 1.3.1.1. Lexis

Empresa pionera en Latinoamérica en la producción de sistemas de información jurídica documental en plataformas informáticas. Con una extensa experiencia en la captura, recopilación y sistematización de más de 350.000 documentos legales, y el desarrollo continúo del software especializado para su administración, mantenimiento y consulta, la misma cuenta con diferentes servicios de apoyo a temas legislativos a través de herramientas como el Sistema Integrado de la Legislación Ecuatoriana (eSilec).


**Figura 2.** Sitio web Lexis (www.lexis.com.ec).

#### 1.3.1.2. eSilec

El Sistema Integrado de la Legislación Ecuatoriana eSilec, es la más completa, eficiente y confiable herramienta de investigación jurídica del Ecuador; que almacena y organiza todos los actos normativos y administrativos, y toda la jurisprudencia de última instancia que se ha publicado en instrumentos oficiales desde inicios de la República.

Cada norma legal, de un total de más de 265.000, ha sido registrada y catalogada desde su fecha de promulgación, y verificada diariamente para determinar su período de vigencia, registrándose los datos de todas las reformas que cada norma ha sufrido a lo largo de su vida útil.

La aplicación es 100% compatible con los navegadores más utilizados en el mercado: Microsoft Explorer, Firefox, Safari, Chrome, Opera, etc.


**Figura 3.** Servicio eSilec (www.lexis.com.ec/website/content/servicio/esilec.aspx).

#### 1.3.1.3. Consultorio jurídico

El Consultorio jurídico es un servicio social gratuito dirigido a la comunidad de escasos recursos, donde los estudiantes de últimos semestres de la Facultad de Derecho y Ciencias Humanas de la Fundación Universitaria Luis Amigó, prestan asistencia como abogados de pobres, conforme a la ley y en desarrollo de la misión-visión de la institución".

En la opción consultas en línea se utiliza un foro para consultar, de manera anónima si se desea, ubicando la inquietud en una de las siguientes temáticas: civil, comercial, laboral, familiar, penal y administrativo donde están descritas para mejor ubicación, en caso de no tener claridad sobre la temática, se ingresa al tema general.


**Figura 4.** Sitio web Consultorio jurídico (www.funlam.edu.co/virtualjuridico).

Tras el análisis de las soluciones que se presentaron se identifican y resaltan de estas los elementos que deben regir el diseño de la propuesta a la cual arribar. De cada uno de estos sistemas se extraen las principales funcionalidades que realizan para el intercambio entre usuarios y la interactividad con el sistema, logrando establecer distintos puntos de vista que permiten el desarrollo del proceso de consulta llevado a la web.

## 1.4 Necesidad de un sistema informático en Cuba

A partir del estudio de los sistemas existentes en el mundo y de analizar sus características e imposibilidad de adoptar una de ellas como solución al problema que se enfrenta, ya sea por el hecho de estar concebidos bajo estatutos legislativos del país al que pertenecen y resultar muy difícil adecuarlos al sistema cubano se plantea la necesidad de implementar un sistema informático que mejore el actual proceso de consulta popular a raíz de no identificar sistemas que respondan a dicho proceso en Cuba. Es por ello que es necesario estudiar las metodologías de desarrollo de software y las tecnologías con el fin de escoger la más adecuada para el correcto desarrollo de un sistema que satisfaga las necesidades del cliente.

## 1.5 Metodologías de desarrollo de software

Las metodologías de desarrollo de software son un conjunto de procedimientos, herramientas, técnicas y soporte a la documentación a la hora de desarrollar un producto de software. Son muchas

las ventajas que puede aportar el uso de una metodología. Estas pueden ser agrupadas desde diferentes perspectivas, desde el punto de vista de gestión permite facilitar la tarea de planificación, la tarea del control y el seguimiento de un proyecto, desde el punto de vista de los ingenieros del software posibilita ayudar a la comprensión del problema, optimizar el conjunto, facilitar el mantenimiento del producto final y permitir la reutilización de partes del mismo, y desde el punto de vista del cliente o usuario garantiza un determinado nivel de calidad en el producto final así como la confianza en los plazos de tiempo fijados en la definición de estos. (6)

Existen dos grupos de metodologías de desarrollo del software, las metodologías robustas o tradicionales y las metodologías ágiles.

### 1.5.1. Metodologías tradicionales

Entre las metodologías robustas se encuentran, RUP (por sus siglas en inglés, Rational Unified Process), MSF (por sus siglas en inglés, Microsoft Solution Framework), entre otras. Estas son denominadas metodologías pesadas, se centran en la definición detallada de los procesos y tareas a realizar, herramientas a utilizar, y requieren una extensa documentación, ya que pretenden prever todo de antemano. Estas son más eficaces y necesarias cuanto mayor sea el proyecto a realizar respecto a tiempo y recursos.

No se escogen estas metodologías para el desarrollo de la aplicación debido a que poseen cierta resistencia al cambio durante el desarrollo. Además, al no demandar un contacto sistemático con el cliente son descartadas en la presente investigación y se propone el empleo de una metodología ágil.

### 1.5.2. Metodologías ágiles

Las metodologías ágiles promueven generalmente un proceso de gestión de proyectos que fomenta el trabajo en equipo, la organización y responsabilidad propia, un conjunto de mejores prácticas de ingeniería que permiten la entrega rápida de software de alta calidad, y un enfoque de negocio que alinea el desarrollo con las necesidades del cliente y los objetivos de la compañía. Entre las metodologías ágiles más notables están Scrum, Crystal Clear, XP (Extreme Programming), ASD (Adaptive Software Development), Feature Driven Development y DSDM (Dynamic Systems Development method).

#### 1.5.2.1. Scrum

Es una metodología ágil que se puede usar para gestionar y controlar desarrollos complejos de software y productos usando prácticas iterativas e incrementales. Dicha metodología posee desventajas que no le permiten ser una opción factible para la propuesta de solución tales como: plantea un problema si el desarrollo está restringido por una fecha de entrega, presupone que los requerimientos cambian, pero no de forma que el cliente acepte un diseño funcional/técnico, se requiere de un experto en la metodología que monitorice su cumplimiento.

#### 1.5.2.2. Extreme programming (XP)

La programación extrema (XP) es un enfoque de la ingeniería del software formulado por Kent Beck. La metodología ágil XP está centrada en potenciar las relaciones interpersonales como clave para el éxito en el desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores, y propiciando un buen clima de trabajo, se basa en cuatro principios fundamentales, la realimentación continua entre el cliente y el equipo de desarrollo, la comunicación fluida entre todos los participantes, la simplicidad en las soluciones implementadas y el coraje para enfrentar los cambios. También se define como adecuada para proyectos con requisitos imprecisos y donde existe un alto riesgo técnico.

Esta metodología consta de doce prácticas fundamentales destacando entregas pequeñas, diseño simple, programación en parejas, pruebas e integración continua donde el mayor beneficio de las prácticas se consigue con su aplicación conjunta y equilibrada puesto que se apoyan unas en otras, además se realizan pruebas continuas durante el proyecto y es mejor utilizada en la implementación de nuevas tecnologías donde los requerimientos cambian rápidamente y el cliente tiene el control sobre las prioridades. (7)

XP cuenta con las siguientes seis fases: **exploración**, en la cual los clientes plantean sus necesidades y el equipo de desarrollo revisa las tecnologías, prácticas y herramientas que van a ser utilizadas durante el proyecto, **planificación,** donde los programadores y clientes se ponen de acuerdo para priorizar las historias de usuario y el alcance de la primera versión, **desarrollo o iteración hacia la primera versión**, aquí el cliente decide las historias de usuario (HU) que se realizarán dividiéndose en tareas que serán desarrolladas en la siguiente versión, en la fase de **producción** se deciden los errores, las nuevas funcionalidades o las modificaciones, la de **mantenimiento** implica iteraciones más largas, incorpora nuevos desarrolladores e incluso reestructura el equipo de desarrollo y la última fase denominada **cierre del proyecto** finaliza cuando el cliente no tenga más historias de usuario para ser incluidas en el sistema. En la figura 5 se evidencian las fases del ciclo de vida de la metodología de desarrollo XP. (7)

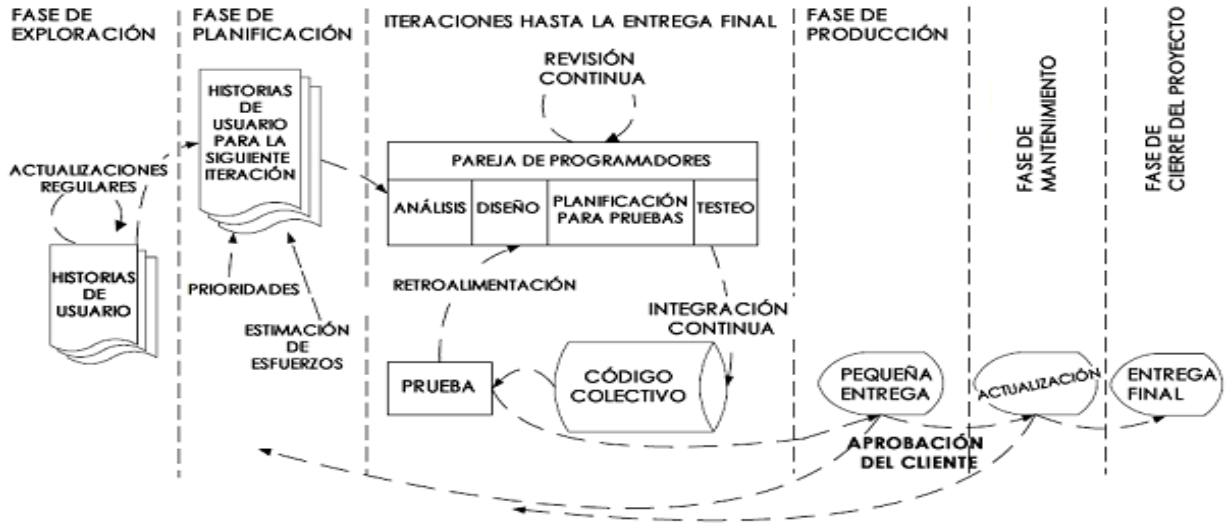



**Figura 5.** Fases del ciclo de vida de la metodología XP.

Además según Billy Reinoso (6) en una conferencia sobre metodologías ágiles de desarrollo de software, plantea que XP ha demostrado ser la metodología ágil que más estudios dispone y sobre la que mayor número de artículos se han escritos.

Dada las condiciones que brinda y la idea de desarrollo que se tiene de la solución, se decide que las características de la metodología de desarrollo de software XP se asocian más al tipo de proyecto que se lleva a cabo durante la elaboración de la solución informática, ya que es empleada para proyectos de corto plazo con un equipo de trabajo pequeño, propone una realimentación continua entre el cliente y el equipo de desarrollo, define un modelo de 40 horas semanales para no trabajar horas extras, entre otros aspectos mencionados con anterioridad.

Después de seleccionada la metodología de desarrollo de software a continuación se presentan los lenguajes, herramientas y tecnologías utilizadas para el desarrollo de la solución que se propone.

## 1.6 Lenguajes, herramientas y tecnologías utilizadas

### 1.6.1 Lenguajes de modelado

Se denomina lenguaje de modelado de objetos al conjunto estandarizado de símbolos para modelar un diseño de software. Para la modelación de la solución se propone como lenguaje de modelado el siguiente:

**UML**

Lenguaje Unificado de Modelado en lo adelante UML[1]. Lenguaje que captura decisiones y conocimiento sobre los sistemas que se deben construir. Se usa para entender, diseñar, hojear, configurar, mantener, y controlar la información sobre tales sistemas. Está pensado para usarse con

[1] **UML:** es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad. Es un lenguaje de modelado visual que se usa para especificar, visualizar, construir y documentar artefactos de un sistema de software.

todos los métodos de desarrollo, etapas del ciclo de vida, dominios de aplicación y medios. UML incluye conceptos semánticos, notación, y principios generales. Tiene partes estáticas, dinámicas, de entorno y organizativas. Está pensado para ser utilizado en herramientas interactivas de modelado visual que tengan generadores de código así como generadores de informes.

Es importante resaltar que UML es un "lenguaje" para especificar y no para describir métodos o procesos. Se utiliza para definir un sistema de software, para detallar los artefactos en el sistema y para documentar y construir. En otras palabras, es el lenguaje en el que está descrito el modelo. Se puede aplicar en una gran variedad de formas para dar soporte a una metodología de desarrollo de software, pero no especifica en sí mismo qué metodología o proceso usar.

Debido a las características antes expuestas hacen propicia la elección de UML en su versión 8.0 como el lenguaje de modelado para la realización del diseño del portal web que se va a implementar. Para un mejor resultado en la aplicación de UML son utilizadas con frecuencia las llamadas herramientas CASE. (8)

## 1.7 Herramientas CASE

Las herramientas CASE (Computer Aided Software Engineering, Ingeniería de Software Asistida por Ordenador) son diversas aplicaciones informáticas destinadas a aumentar la productividad en el desarrollo de software reduciendo el coste de las mismas en términos de tiempo y de dinero. Estas herramientas pueden ayudar en todos los aspectos del ciclo de vida de desarrollo del software en tareas como el proceso de realizar un diseño del proyecto, cálculo de costes, implementación de parte del código automáticamente con el diseño dado, compilación automática, documentación o detección de errores entre otras. Existen muchísimas herramientas CASE que responden a diferentes necesidades de los clientes que las usan; entre ellas podemos encontrar Visual Paradigm, Erwin, EasyCASE, Oracle Designer, System Architect por solo citar algunas.

### 1.7.1 Visual Paradigm

La ingeniería de software recomienda el uso de herramientas CASE[2] para llevar a cabo el análisis y diseño de cualquier sistema de información, en este caso las de soporte a UML. Visual Paradigm es una herramienta de modelado visual para todos los tipos de diagramas UML. Es compatible con una amplia gestión de casos de uso, diagramas, requerimiento, diseño de base de datos y proporciona medidas más eficaces en el análisis y diseño de sistemas. Está diseñada para una amplia gama de usuarios, incluyendo los ingenieros de software, analistas de sistemas, analistas de negocios y arquitectos de sistemas, o para cualquier persona que esté interesada en la construcción de forma fiable a gran escala de sistemas de software con un enfoque orientado a objetos. (8)

[2] **CASE:** es un conjunto de programas y ayudas que dan asistencia a los analistas, ingenieros de software y desarrolladores, durante todos los pasos del Ciclo de Vida de desarrollo de un sistema informático. (9)

Se seleccionó Visual Paradigm en su versión 8.0 como herramienta CASE por contar con soporte para resolver varios problemas de modelado. (9)

## 1.8 Sistemas Gestores de Contenido(CMS)

En los últimos años se ha desarrollado el concepto de sistemas de gestión de contenidos (Content Management Systems o CMS). También son conocidos como gestores de contenido web (Web Content Management o WCM). Estas herramientas permiten crear y mantener páginas Web con facilidad, confiriendo al usuario o al autor la autonomía y permisos necesarios para realizar trabajos que hasta hace poco estaba en manos de los administradores de los servidores Web.

Los CMS ofrecen posibilidades de personalización muy altas. Todos ofrecen diferentes plantillas para personalizar la presentación, y los genéricos, además, ofrecen numerosas opciones para añadir funcionalidades al sistema. Incluso en aquellos CMS donde no se ofrecen excesivas funcionalidades, no es extraño que ofrezcan un conjunto de funciones pensadas para que el usuario pueda modificar completamente el aspecto o pueda añadir funcionalidades creando plugins. Además proporcionan un entorno que posibilita la actualización, mantenimiento y ampliación de la web con la colaboración de múltiples usuarios.

**Las características fundamentales de los CMS son:**

- ✓ Intuitivo: fácil de entender y utilizar.
- ✓ Flexible en la personalización
- ✓ Extensible a través de plugins y módulos.
- ✓ Optimizado para un alto rendimiento y velocidad.
- ✓ Que ofrezca una seguridad.
- ✓ Documentación y apoyo comunitario.
- ✓ Énfasis en estándares web y buenas prácticas.

### 1.8.1. CMS más usados en el desarrollo web

En la presente investigación se profundiza en los CMS WordPress, Joomla y Drupal por ser de código abierto y contar con una relevancia y seguimiento importante que garantiza su continuidad y apoyo por parte de usuarios y desarrolladores, además de una gran cantidad de documentación.

**WordPress**

WordPress es una avanzada plataforma semántica de publicación personal orientada a la estética, los estándares web y la usabilidad, es libre y al mismo tiempo gratuito. Es el sistema que utilizas cuando se desea trabajar con una herramienta de publicación.

WordPress también puede operar como un gran sistema de administración de contenidos. Dispone de un excelente editor de texto, un sistema robusto de plantillas y cientos de plugins.

Para lograr que los usuarios puedan utilizar WordPress como un CMS completo, se incluye acciones como:

- ✓ Los contenidos se ingresan en páginas.
- ✓ La navegación se implementará automáticamente.
- ✓ Los enlaces internos entre páginas, entradas y categorías se simplificarán y serán absolutamente seguros.
- ✓ El sistema será tolerante ante errores humanos.
- ✓ Personas con un entrenamiento mínimo podrán organizar esos sitios web.
- ✓ Administrar sitios web equipados con WordPress Multilingüe no demandará ningún esfuerzo. (10)

**Drupal**

Drupal es otro CMS de código abierto bastante poderoso que puede ser utilizado para todo, desde sitios corporativos hasta sitios de comercio electrónico o redes sociales. La interfaz es muy simple, con vínculos lógicamente organizados para crear y/o editar nuevos contenidos o administrar usuarios y permisos. Se usa para crear sitios web. Es modular, se utiliza en el marco de gestión de contenidos con un énfasis en la colaboración. Es extensible, compatible con los estándares, y se esfuerza por limpiar el código y una pequeña huella. Presenta funciones adicionales que se obtienen al activar módulos integrados o de terceros. Drupal está diseñado para ser personalizado, pero la personalización se realiza reemplazando el núcleo o mediante la adición de módulos, no mediante la modificación del código en el núcleo.

Se pone como punto negativo que Drupal es complejo y tiene una curva de aprendizaje alta, es menos usable y el costo de mantenimiento en las aplicaciones desarrolladas en este CMS es más alto. (11)

**Joomla**

Joomla es actualmente considerado uno de los CMS de código abierto más populares. La interfaz es relativamente simple y directa, con distintas secciones para manejar artículos, bloques, lista de vínculos, multimedia y contenidos. Es un CMS bastante poderoso, por lo que es un CMS que no conviene para sitios simples, en los que tendría un exceso de funcionalidades. Incluye un número de provisiones para hacer que las páginas carguen más rápido, incluyendo compresión cache.

**Dentro de las características principales de Joomla están:**

- ✓ Software libre (Licencias GNU5/GPL), ampliable al disponer el código fuente.
- ✓ Completa y fácil administración por la web
- ✓ Creación y administración rápida de una comunidad en línea.
- ✓ Creación de la web por inserción de módulos y componentes independientes.
- ✓ Creación y actualización dinámica de secciones, subsecciones y contenidos (públicos y privados).
- ✓ Creación de perfiles y privilegios con niveles jerárquicos para diferentes niveles de usuarios
- ✓ Administrador, gerente, editor y usuario registrado).
- ✓ Plantillas para transformar el diseño visual de la web de forma automática en pocos minutos.
- ✓ Estadísticas de acceso a los contenidos. (12)

Después de haber descrito las principales características de Drupal, WordPress y Joomla existe un difícil dilema a la hora de escoger cuál de estos sistemas gestores de contenidos es más apropiado para el desarrollo de la solución. Son CMS de código abierto y hay cientos de desarrolladores independientes y centros de desarrollo en todo el mundo para atender el desarrollo cada vez mayor en la web.

Pero a diferencia de Drupal y WordPress, Joomla está diseñado para los sitios web donde las personas no necesitan gran conocimiento de la tecnología, es más fácil de configurar y poner en marcha con una curva de aprendizaje relativamente baja, es significativamente potente y flexible además de poseer un rendimiento y escalabilidad aceptable para aplicaciones no muy grandes; las funcionalidades de edición y construcción del contenido son excelentes y la consola de administración es mucho más fácil para organizar y encontrar contenido. Joomla posee además una interfaz amigable en cuanto a diseño y aspecto estético, con una variedad de temas, plantillas muy sólidas y profesionales que mejoran el ambiente durante la navegación, tiene un extenso directorio de plugins, componentes y su condición de ser orientado a objetos facilita el desarrollo. Por todas las características anteriormente expuestas, el CMS Joomla, en su versión 2.5, satisface las necesidades para la construcción del portal de la Facultad de Derecho de la UH.

## 1.9. Lenguajes de desarrollo

Un lenguaje de programación es un idioma artificial diseñado para expresar computaciones que pueden ser llevadas a cabo por máquinas como las computadoras. Pueden usarse para crear programas que controlen el comportamiento físico y lógico de una máquina, para expresar algoritmos con precisión, o como modo de comunicación humana.(13)

### 1.9.1 Lenguajes de desarrollo del lado del cliente

La programación del lado del cliente tiene como principal ventaja que la ejecución de la aplicación se

delega al cliente, con lo cual se evita recargar al servidor de trabajo. El servidor solo envía el código, y es tarea del buscador interpretarlo.

#### 1.9.1.1 Javascript

Javascript[3] fue creado por Brendan Eich en la empresa Netscape Communications. Utilizado principalmente en páginas web. Es similar a Java, aunque no es un lenguaje orientado a objetos, el mismo no dispone de herencias. Es un lenguaje interpretado, no requiere compilación. La mayoría de los navegadores en sus últimas versiones interpretan código Javascript, realiza acciones dentro del ámbito de una página web. Su compatibilidad con la mayoría de los navegadores modernos, lo posiciona como el lenguaje de programación del lado del cliente más utilizado.

Podemos crear efectos especiales en las páginas y definir interacción con el usuario. El navegador (browser) del cliente es el encargado de interpretar las instrucciones Javascript y ejecutarlas para realizar estos efectos e interactividades, de modo que el mayor recurso y tal vez el único con que cuenta este lenguaje es el propio navegador.

Además soporta el Modelo de Objetos de Documento (DOM, Document Object Model). El DOM es el conjunto de objetos predefinidos que nos permite acceder a todos los elementos de una página y a ciertas características específicas del navegador.

**Ventajas**

- ✓ Lenguaje de scripting seguro y fiable.
- ✓ Los script tienen capacidades limitadas, por razones de seguridad.
- ✓ El código JavaScript se ejecuta en el cliente.

**Desventajas**

- ✓ Código visible por cualquier usuario.
- ✓ El código debe descargarse completamente. (14)

#### 1.9.1.2 XHTML

Lenguaje Extensible de Marcado de Hipertexto, XHTML por sus siglas en inglés (extensible HyperText Markup Language), se crea con el objetivo de sustituir al Lenguaje de Marcado de Hipertexto (HTML por sus siglas en inglés, HyperText Markup Language). XHTML mantiene casi todas las etiquetas y características del HTML, pero añade algunas restricciones y elementos propios de XML, es el lenguaje de marcado pensado para sustituir a HTML como estándar para las páginas web. XHTML es la versión XML de HTML, por lo que tiene básicamente las mismas funcionalidades, pero cumple las especificaciones más estrictas de XML.

[3] **Javascript:** es un lenguaje de programación que se utiliza principalmente para crear páginas web dinámicas.

El principal inconveniente que presenta es el aumento del tamaño del documento, por lo que en general se utilizan etiquetas con nombres cortos. (15)

#### 1.9.1.3 CSS

Las Hojas de Estilo en Cascada, CSS por sus siglas en inglés, Cascading Style Sheets es un lenguaje de hojas de estilos creado para controlar el aspecto o presentación de los documentos electrónicos definidos con HTML y XHTML. Es la mejor forma de separar los contenidos y su presentación y es imprescindible para crear páginas web complejas. Al crear una página web, se utiliza en primer lugar el lenguaje HTML/XHTML para marcar los contenidos, es decir, para designar la función de cada elemento dentro de la página: párrafo, titular, texto destacado, tabla, lista de elementos, entre otras. Una vez creados los contenidos, se utiliza el lenguaje CSS para definir el aspecto de cada elemento: color, tamaño y tipo de letra del texto, separación horizontal y vertical entre elementos, posición de cada elemento dentro de la página. (16)

### 1.9.2 Lenguajes de desarrollo del lado del servidor

Un lenguaje del lado del servidor es aquel que se ejecuta en el servidor web, justo antes de que se envíe la página al cliente. Las páginas que se ejecutan en el servidor pueden realizar accesos a bases de datos, conexiones en red y otras tareas para crear la página final que verá el cliente. Son reconocidos, ejecutados e interpretados por el propio servidor y que se envían al cliente en un formato comprensible para él. Por tanto, son independientes del navegador, y no necesitarán plugins especiales para visualizar correctamente cualquier página.

#### 1.9.2.1 PHP

PHP es un acrónimo recursivo que significa “PHP Hypertext Pre-processor”, (inicialmente se llamó Personal Home Page). Surgió en 1995, desarrollado por PHP Group, es un lenguaje de script interpretado en el lado del servidor utilizado para la generación de páginas web dinámicas, embebidas en páginas HTML y ejecutadas en el servidor. PHP no necesita ser compilado para ejecutarse. Para su funcionamiento necesita tener instalado Apache con las librerías de PHP. La mayor parte de su sintaxis ha sido tomada de C++ y Java con algunas características específicas. Los archivos cuentan con la extensión (php).

**Ventajas:**

- ✓ Se caracteriza por ser un lenguaje muy rápido.
- ✓ Soporta en cierta medida la orientación a objeto. Clases y herencia.
- ✓ Es un lenguaje multiplataforma: Linux, Windows, entre otros.
- ✓ Capacidad de conexión con la mayoría de los manejadores de base de datos: MySQL, PostgreSQL, Oracle, MS SQL Server, entre otras.

- ✓ Capacidad de expandir su potencial utilizando módulos.
- ✓ Posee documentación en su página oficial la cual incluye descripción y ejemplos de cada una de sus funciones.
- ✓ Es libre, por lo que se presenta como una alternativa de fácil acceso para todos.
- ✓ Incluye gran cantidad de funciones.
- ✓ No requiere definición de tipos de variables ni manejo detallado del bajo nivel.

**Desventajas:**

- ✓ Todo el trabajo lo realiza el servidor y no delega al cliente. Por tanto puede ser más ineficiente a medida que las solicitudes aumenten de número.
- ✓ La legibilidad del código puede verse afectada al mezclar sentencias HTML y PHP.
- ✓ Dificulta la organización por capas de la aplicación. (17)

## 1.10 Entorno de Desarrollo Integrado

Los IDEs son un conjunto de herramientas para el programador, que suelen incluir en una misma suite, un buen editor de código, administrador de proyectos, archivos, enlace transparente a compiladores e integración con sistemas controladores de versiones o repositorios. Estas series de herramientas las utilizan los programadores para desarrollar código y están pensada para su utilización con un único lenguaje de programación o bien puede dar cabida a varios de estos.

Existen disimiles alternativas para elegir un IDE, pero para realizar una buena selección existen varios factores fundamentales como es la proporción (precio – potencialidad). En la actualidad existen diferentes IDE, ejemplos: Eclipse, MS Visual Studio.NET, Dreamweaver, JDeveloper, JBuilder, Delphi, NetBeans y Zend Studio.

A continuación se presenta el IDE NetBeans que se utilizará en el desarrollo de la propuesta de solución.

### 1.10.1 NetBeans 7.4

NetBeans es una plataforma independiente de código abierto que soporta una amplia variedad de lenguajes, incluyendo PHP. Permite la codificación de programas en C++ y otros aunque está pensado para Java. Es gratuito y tiene una gran comunidad de usuarios y desarrolladores de todo el mundo. Permite desarrollar de manera rápida y fácil Java de escritorio, para móviles y aplicaciones web, mientras que también proporciona una gran herramienta para PHP y C / C + + desarrolladores.

Es un IDE modular basado en estándares y una plataforma para aplicaciones de cliente enriquecidas que se puede utilizar como marco genérico para crear cualquier tipo de aplicación. Además, se le

pueden agregar numerosas librerías y extensiones, las cuales brindan opciones al momento de programar. (18)

#### 1.10.1.1 Fundamentación del IDE seleccionado

NetBeans utiliza plantillas para proyectos PHP, que se pueden ajustar con los plugins y herramientas necesarias en el servidor de integración. Además de su velocidad y las capacidades de refactorización, proporciona el autocompletado con el apoyo sugerencia para funciones y clases de todo el proyecto.

## 1.11 Sistemas Gestores de Base de Datos

Un Sistema Gestor de Bases de Datos (SGBD) o Data Base Management System (DBMS, por sus siglas en inglés) es una colección de programas cuyo objetivo es servir de interfaz entre la base de datos, el usuario y las aplicaciones. Se compone de un lenguaje de definición de datos, de manipulación de datos y consulta. Un SGBD permite definir los datos a distintos niveles de abstracción y manipular dichos datos, garantizando la seguridad e integridad de los mismos. (19)

Un SGBD debe permitir:

- ✓ Definir una base de datos: especificar tipos, estructuras y restricciones de datos.
- ✓ Construir la base de datos: guardar los datos en algún medio controlado por el mismo SGBD.
- ✓ Manipular la base de datos: realizar consultas, actualizarla, generar informes.

### 1.11.1 MySQL 5.6

MySQL es un SGBD relacional, multihilo, multiplataforma y multiusuario. Es una idea originaria de la empresa Open Source MySQL AB fundada en 1995, que pasó a manos de Sun Microsystems [4] en 2008 cuando adquirió la empresa, luego en 2010 Sun Microsystems fue adquirida por la empresa Oracle Corporation, lo que justifica el desarrollo de MySQL como software libre en un esquema de licenciamiento dual. Por un lado se ofrece bajo la GNU GPL [5] para cualquier uso compatible con esta licencia, pero para aquellas empresas que quieran incorporarlo en productos privativos deben comprar a la empresa una licencia específica que les permita este uso.

EL objetivo que persigue MySQL es cumplir el estándar SQL, pero sin sacrificar velocidad, fiabilidad o usabilidad. En las últimas versiones se pueden destacar las siguientes características principales:

- ✓ El propósito general es lograr mayor velocidad y robustez.
- ✓ Soporta gran cantidad de tipos de datos para las columnas.

**Sun Microsystems:** es una empresa informática adquirida por Oracle Corporation, anteriormente parte de Silicon Valley, fabricante de semiconductores y software. Sun Microsystems es la creadora de la popular suite ofimática OpenOffice.org y el lenguaje de programación Java.

**GNU GPL:** licencia creada por la Free Software Foundation en 1989 (la primera versión), y está orientada principalmente a proteger la libre distribución, modificación y uso de Software. Su propósito es declarar que el software cubierto por esta licencia es Software libre y protegerlo de intentos de apropiación que restrinjan esas libertades a los usuarios.

- ✓ Gran portabilidad entre sistemas, puede trabajar en distintos sistemas operativos.
- ✓ Cada base de datos cuenta con 3 archivos: Uno de estructura, uno de datos y uno de índice y soporta hasta 32 índices por tabla.

**Ventajas:**

- ✓ Escalabilidad y flexibilidad: posibilidad de manipular bases de datos desde un megabyte hasta almacenes de datos enormes y está diseñado para soporte multiplataforma.
- ✓ Alto rendimiento: posee una arquitectura única de motores de bases de datos que permite a los profesionales configurar el servidor MySQL para aplicaciones específicas, dando como resultado un rendimiento espectacular.
- ✓ Alta disponibilidad: solidez y disponibilidad constante ofreciendo una variedad de soluciones desde replicación a servidores de clúster especializados, u ofertas de terceros.
- ✓ Robusto soporte transaccional: posee un soporte completo de ACID (atómica, consistente, aislada, duradera), bloqueo a nivel de filas, posibilidad de transacciones distribuidas, y soporte de transacciones con múltiples versiones.
- ✓ Fortaleza en la Web: posee un motor de consultas de alto rendimiento, con posibilidad de insertar datos a gran velocidad y un buen soporte para funciones web especializadas como las búsquedas de texto completo.
- ✓ Fuerte protección de datos: ofrece características de seguridad que aseguran una protección absoluta de los datos. (6)

## 1.12 Servidor web

Un servidor web es un programa que se ejecuta continuamente en un computador, manteniéndose a la espera de peticiones de ejecución que le hará un cliente o un usuario de Internet. El servidor web se encarga de contestar a estas peticiones de forma adecuada, entregando como resultado una página web o información de todo tipo de acuerdo a los comandos solicitados.

En la presente investigación se analiza el servidor Apache por ser el más usado en el desarrollo con Joomla.

### 1.12.1 Apache 2.4

Apache es programa de servidor HTTP Web de código abierto. Fue desarrollado en 1995 y actualmente es uno de los servidores web más utilizados en la red. La primera versión apareció en enero de 1996, el Apache 1.0. Está estructurado en módulos. La configuración de cada módulo se hace mediante la configuración de las directivas que están contenidas dentro del módulo. (20)

Entre sus principales características se encuentran las siguientes:

- ✓ Es multiplataforma.
- ✓ Es una tecnología gratuita de código fuente abierta.

- ✓ Es un servidor altamente configurable de diseño modular.
- ✓ Trabaja con gran cantidad de lenguajes como Perl, PHP y script.
- ✓ Permite personalizar la respuesta ante los posibles errores que se puedan dar en el servidor.(20)

### 1.12.2 Fundamentación del servidor web seleccionado

El servidor web que se selecciona para el desarrollo de las funcionalidades del Portal web es Apache, por ser multiplataforma, incluyendo varias versiones de UNIX, Windows y MacOS. Es un servidor eficiente, y consume menos recursos del sistema en comparación a otros servidores. Su arquitectura modular permite personalizarlo y ampliarlo. Descartando Internet Information Server que solo funciona bajo servidores Microsoft, debe ser usado bajo licencia.

## 1.13 Conclusiones del capítulo

- ✓ Una consulta popular se constituye a partir de deliberaciones públicas promovidas por los gobiernos respecto al cuerpo electoral o de legislación; un ejercicio de participación ciudadana donde cada vez el pueblo llega a la decisión de forma directa como expresión de democracia sobre proyectos legislativos; entiéndase Ley, Resolución, Comunicación, Declaración o Decreto, entre otros, y que encuentran soporte en la Informática Jurídica.
- ✓ El análisis realizado a sistemas informáticos que presentan alguna solución para satisfacer los servicios brindados por el proceso de consulta popular, permitió definir que la utilización de un portal web como plataforma es una de las mejores soluciones para realizar dicho proceso.
- ✓ Entre las tecnologías, herramientas, lenguajes de modelado y desarrollo escogidos se encuentran: como sistema gestor de contenido Joomla 2.5, como herramienta case Visual Paradigm 8.0, como entorno de desarrollo integrado Net Beans 7.4, como sistema gestor de base de datos MySQL 5.6, como servidor web Apache 2.4, como lenguaje de modelado UML, como lenguajes de desarrollo por parte del cliente CCS, Javascrip, XHTML y como lenguaje de desarrollo por parte del servidor PHP.

# CAPÍTULO 2. PROPUESTA DE SOLUCIÓN.

## 2.1. Introducción

El presente capítulo detalla las fases de exploración y planeación definidas en el ciclo de vida de la metodología de desarrollo XP que posibilita describir la solución propuesta. De igual forma se muestran los principios, prácticas y técnicas que sirven de guía para los desarrolladores, entre las que se destacan: Historias de usuarios HU, Plan de iteraciones, Plan de entrega. Contiene a su vez los roles de usuarios definidos para el acceso a la información y la seguridad en la aplicación, así como la propuesta de interfaz de usuario diseñada.

## 2.2. Propuesta de solución

Una de las buenas prácticas de la metodología XP es definir una metáfora para el sistema. En XP no se enfatiza la definición temprana de una arquitectura estable para el sistema. Dicha arquitectura se asume evolutiva y los posibles inconvenientes que se generarían por no contar con ella explícitamente en el comienzo del proyecto se solventan con la existencia de una metáfora. El sistema es definido mediante una metáfora o un conjunto de metáforas compartidas por el cliente y el equipo de desarrollo. Una metáfora es una historia compartida que describe cómo debería funcionar el sistema. Por consiguiente, la metáfora que se propone de la solución que se quiere alcanzar es: "un espacio donde se 'recolecte' información relevante a un tema de discusión para ayudar a la toma de decisiones por parte de los expertos en dicho tema."

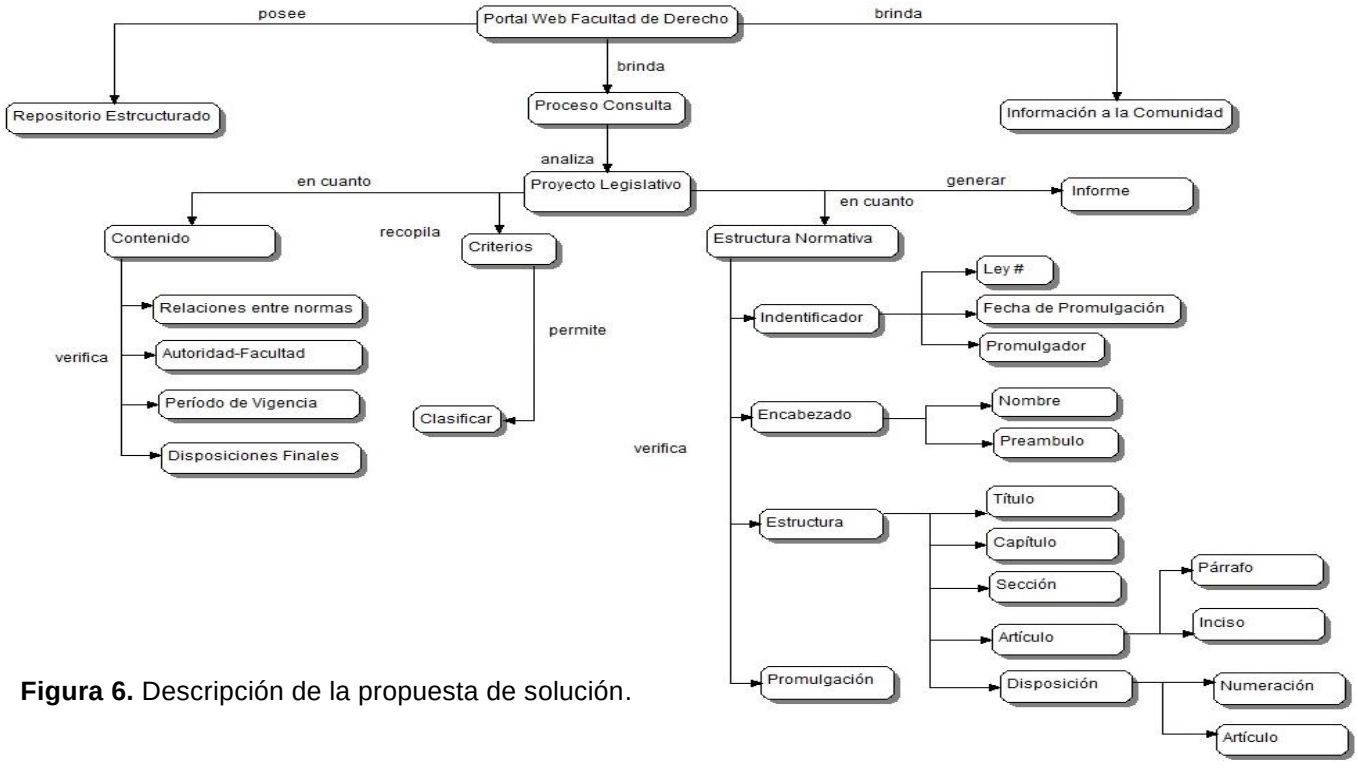



**Figura 6.** Descripción de la propuesta de solución.

La figura 6 describe un modelo de dominio a partir de la desagregación estructural/funcional de la propuesta de solución, y su integración con el proyecto de rediseño del Portal web de la Facultad de Derecho.

Los autores consideran oportuno ahondar sobre la representación de la rama "*Proyecto* legislativo *en cuanto a contenido*" de la figura 6 ya que su representación gráfica, por sí misma, no es lo suficientemente explicativa.

**Relaciones entre normas:** el sistema permite verificar la relación que se establece entre las normas que se mencionan en los por cuantos de derecho [6] del Proyecto legislativo que se está analizando.

**Autoridad-facultad:** el sistema permite verificar si la institución que emite el nuevo Proyecto legislativo está facultada y representada para expresar o promover las decisiones propias; aquellas atinentes al cumplimiento de sus funciones**.**

**Período de vigencia:** el sistema emite la fecha de entrada en vigencia del nuevo Proyecto legislativo en cuestión y el período restante para el mismo.

**Disposiciones finales:** el sistema permite establecer una correspondencia entre las disposiciones finales y los por cuantos que se enuncian en el preámbulo del nuevo Proyecto legislativo.

Para llevar a cabo la solución planteada y además garantizar quien tiene privilegios de acceso en su totalidad a la misma o a algunos componentes de ella, se hace necesario establecer el sistema de control de acceso basado en roles, delimitando de esta forma los usuarios que pueden o no realizar una determinada acción.

### 2.3. Sistemas de control de acceso

Uno de los elementos esenciales en el desarrollo de la solución es definir el control de acceso de los usuarios a la aplicación, para determinar si el usuario tiene permitido el acceso a los recursos. Existen modelos de control de acceso como el control de acceso discrecional (DAC, por sus siglas en Inglés), el control de acceso obligatorio (MAC, por sus siglas en Inglés) y el control de acceso basado en roles (RBAC, por sus siglas en Inglés), donde los controles de acceso DAC y MAC por sí solos son inadecuados para cubrir las necesidades de la mayor parte de las organizaciones.

El modelo DAC es demasiado débil para controlar el acceso a los recursos de información de forma efectiva, en tanto que el MAC es demasiado rígido. El modelo RBAC, es un intento de unificar los modelos anteriores, consiguiendo un sistema que impone el control de acceso, pero sin las

**Por cuantos de derecho:** son aquellos por cuantos que hacen alusión a terceras normas en su propio contenido. Los por cuantos se clasifican en por cuantos de derecho y hecho, este último no referencia normas.

restricciones rígidas impuestas por las etiquetas de seguridad, para esta investigación se definió el RBAC.

**RBAC**

El Sistema de Control de Acceso Basado en Roles (RBAC) básicamente consiste en la creación de roles para los trabajos o funciones que se realizan en la organización, su arquitectura está ideada para asignar permisos a operaciones más que a objetos concretos, siendo por tanto su principal ventaja el permitir alinear la infraestructura de seguridad de la información con los objetivos de negocio de una forma natural.

RBAC recopila las capacidades de súper usuario (usuario root) en perfiles de derechos, los mismos se asignan a cuentas de usuario especiales denominadas roles. Luego, un usuario puede asumir un rol para realizar un trabajo que requiere algunas de las capacidades de súper usuario. (6)

RBAC actualmente es considerado uno de los modelos más generales, debido a su neutralidad respecto a las políticas de control de acceso y a su flexibilidad. A causa de la existencia de múltiples variantes de modelos e implementaciones del mismo se destacan las siguientes características:

- ✓ Administración de autorizaciones
  - La asignación de permisos a usuarios tiene dos partes: asociar usuarios a roles y asignar permisos para objetos a roles.
  - Si un usuario cambia de tareas, solo basta con cambiarle el rol.
- ✓ Jerarquía de roles
  -Los roles también poseen relaciones de jerarquía.
  - Pueden heredar privilegios y permisos de otros roles de menor jerarquía, simplificando la administración de las autorizaciones.
- ✓ Menor privilegio
  -Permite implementar la política del menor privilegio posible la cual consiste en que un usuario dispone exactamente de la cantidad de privilegios necesaria para realizar un trabajo.
  -Si una tarea no va a ser ejecutada por un usuario, entonces su rol no tendrá los permisos para hacerla, de esta manera se minimizan riesgos de daños.
- ✓ Separación de responsabilidades
  -Se basa en el principio de que ningún usuario tenga suficientes privilegios para usar el sistema en su propio beneficio.

### 2.3.1. Roles de usuario

Una de las premisas fundamentales a tener en cuenta cuando se comienza el desarrollo de cualquier sistema informático, la constituye el delimitar la audiencia a la cual va dirigido el mismo, teniendo en cuenta que esta puede estar dividida a su vez en grupos atendiendo a sus competencias. Dentro de

la audiencia antes mencionada se incluyen, como personal relacionado al sistema, al administrador que será el encargado de gestionar toda la información, el usuario anónimo y el usuario autenticado.

Los usuarios de la aplicación serán aquellos que obtengan un resultado de la ejecución de uno o varios procesos del sistema. La tabla 1, a continuación, muestra un listado con los roles de usuarios definidos para esta aplicación.

**Tabla 1.** Roles de usuario del Sistema.

| Roles | Descripción |
|---|---|
| Administrador | Encargado de gestionar toda la información tanto para los usuarios autenticados como para los invitados. Establece los permisos pertinentes para los distintos tipos de usuarios. |
| Usuario Autenticado | Tiene acceso a todo lo publicado en el sitio de forma pública definido por el administrador del sistema. Tendrá privilegios extras en dependencia de los permisos que le sean asignados por el administrador. |
| Usuario Anónimo | Cualquier otra persona que acceda a la aplicación. Su acceso a la misma es definido por el administrador funcional, así como su nivel de visualización de la información. |

## 2.4. Fase de exploración

En esta fase, los clientes plantean a grandes rasgos las historias de usuarios que son de interés para la primera entrega del producto. Al mismo tiempo, el equipo de desarrollo se familiariza con las herramientas, tecnologías y prácticas que se utilizarán en el proyecto.
En esta fase, el cliente define lo que necesita mediante la redacción de historias de usuarios. Los programadores estiman los tiempos de desarrollo sobre la base de esta información.

### 2.4.1. Historias de usuarios (HU)

Las HU son la técnica utilizada en XP para especificar los requisitos del software. Se trata de tarjetas en las cuales el cliente describe brevemente las características que el sistema debe poseer, sean requisitos funcionales o no funcionales. El tratamiento de las HU es dinámico y flexible, en cualquier momento las mismas pueden romperse, reemplazarse por otras más específicas o generales, añadirse nuevas o ser modificadas.

Cada HU es lo suficientemente comprensible y delimitada para que los programadores puedan implementarla en unas semanas. XP propone fichas para registrar las mismas y también tareas

asociadas a ellas. La Tabla 2, a continuación, muestra el formato general de una HU propuesto por R. Jeffries, A. Anderson y C. Hendrick (21); mientras que las Tablas 3-5, debajo, exhiben ejemplos correspondientes a las HU#13, 14, y 15 respectivamente; las más significativas a consideración de los autores.

**Tabla 2.** Formato general de las HU.

| **Historia de usuario** | |
|---|---|
| **Número :** número consecutivo a partir del 1 | **Nombre:** identifica la HU. |
| **Usuario:** ¿quién ejecuta la HU? | |
| **Prioridad en el negocio:** define la relevancia e impacto de la historia de usuario para el negocio de acuerdo con las necesidades del usuario. | **Riesgo de desarrollo:** define la dificultad técnica que supone desarrollar la historia de usuario desde el punto de vista del programador. |
| **Puntos estimados:** permite estimar la duración de la implementación, representando con "1" una semana de trabajo. | **Iteración:** precisa la iteración en la que será desarrollada la HU. |
| **Descripción:** | Explica en qué consiste la HU, teniendo en cuenta las acciones realizadas por el usuario y la respuesta brindada por el sistema. |
| **Observaciones:** | Brinda información extra que se estime agregar para hacer más comprensible la HU. Por ejemplo: conceptos, post-condiciones, relación con otros requisitos, entre otras. |

**Tabla 3.**Historia de usuario 13.

| **Historia de usuario** | |
|---|---|
| **Número** : 13 | **Nombre:** Verificar contenido |
| **Usuario:** usuario autenticado | |
| **Prioridad en el negocio:** alta | **Riesgo de desarrollo:** alta |
| **Puntos estimados:** 2 | **Iteración:** 2 |
| **Descripción:** | El usuario podrá verificar el contenido implícito en el nuevo Proyecto legislativo que está en fase de aprobación, en cuanto a relaciones entre normas, autoridad-facultad, período de vigencia y disposiciones finales. |

**Tabla 4.**Historia de usuario 14.

| Historia de usuario | |
|---|---|
| **Número :** 14 | **Nombre:** Verificar estructura normativa |
| **Usuario:** usuario autenticado | |
| **Prioridad en el negocio:** alta | **Riesgo de desarrollo:** alta |
| **Puntos estimados:** 1 | **Iteración:** 2 |
| **Descripción:** | El usuario podrá verificar si la estructura del documento que representa el nuevo Proyecto legislativo está acorde a lo establecido. |

**Tabla 5.** Historia de usuario 15.

| Historia de usuario | |
|---|---|
| **Número :** 15 | **Nombre:** Recopilar criterio |
| **Usuario:** usuario autenticado | |
| **Prioridad en el negocio:** alta | **Riesgo de desarrollo:** alta |
| **Puntos estimados**: 1.5 | **Iteración:** 3 |
| **Descripción:** | El usuario podrá recopilar los criterio de la comunidad para poder llegar a una conclusión sobre el nuevo Proyecto legislativo en cuestión. |
| **Observaciones:** | Deben de haberse realizado las HU 13 y HU 14. |

Estas HU integradas conforman el proceso consulta popular en general, ya que una vez verificado el contenido, la estructura normativa y recopilados los criterios se cumple el objetivo de dicho proceso.

## 2.5. Requisitos

Un requisito se puede describir como: una condición o capacidad que debe ser poseída por un sistema o componente del mismo para satisfacer un contrato, estándar, especificación, u otro documento formalmente impuesto o una representación documentada de una condición o capacidad como las descritas en los incisos a) o b) (IEEE 610.12-1990). En caso de la solución que se presentan están expresadas en las tareas de ingenierías que se exhiben más adelante en el acápite 3.6.

### 2.5.1. Requisitos no funcionales

Los requisitos no funcionales son restricciones de los servicios o funciones ofrecidas por el sistema. Incluyen restricciones de tiempo, sobre el proceso de desarrollo y estándares. Estas propiedades reflejan las características que hacen al producto atractivo, usable, rápido y confiable. (22)

Los que se listan a continuación inciden directamente en la solución que se propone:

### 2.5.2. Requisito no funcional de software

Se necesitan como requerimientos mínimos:

- ✓ **Para el cliente:** sistema Operativo Windows XP o superior y Linux, Navegador web Mozilla Firefox 3.0 o Internet Explorer 5 o versiones superiores.
- ✓ **Para el servidor:** sistema Operativo Windows Server, Linux (cualquier distribución), Servidor web: Apache 2, Lenguaje de Programación: PHP 5.3.
- ✓ **Para la Base de Datos:** MySQL 4.0 o superior.

#### 2.5.2.1 Requisito no funcional de hardware

Se necesitan como requerimientos mínimos:

- ✓ **Para el cliente:** procesador Pentium III, 128 MB de RAM, 100 MB de disco duro.
- ✓ **Para el servidor:** procesador Pentium IV a 3.0 GHz, 1 GB de RAM, 10 GB de disco duro.

#### 2.5.2.2 Requisito no funcional de apariencia o interfaz externa

- ✓ Diseño perfectamente encuadrado para resoluciones de 1024x768, pero preparado para verse en otras resoluciones.
- ✓ Cumple con los estándares de la W3C[7].

La aplicación contará con una interfaz amigable, páginas no cargadas de mucha información y colores suaves que representan seriedad, además de estar en correspondencia con los colores de la comunidad.

#### 2.5.2.3 Requisito no funcional de seguridad

- ✓ **Confidencialidad:** existencia de distintos roles que establezcan que la información sólo sea vista por aquellos usuarios que posean los privilegios suficientes; restringir la ejecución de acciones a usuarios sin credenciales que intenten acceder a las mismas y la verificación de que el usuario esté autentificado antes de realizar alguna acción.

- ✓ **Integridad:** validación de los datos en el servidor para evitar estados inconsistentes. La información manejada por el sistema estará protegida del acceso y divulgación no autorizada. Se debe realizar la confirmación sobre acciones irreversibles como eliminaciones.

- ✓ **Disponibilidad:** el sistema estará disponible las 24 horas del día a los usuarios autorizados, garantizando el acceso a la información en cualquier momento. Los mecanismos utilizados para lograr la seguridad no obstruyen el acceso a la información. Se realizarán salvas

**W3C**: El World Wide Web Consortium, abreviado W3C, es un consorcio internacional que produce recomendaciones para la World Wide Web.

periódicas en otros dispositivos para evitar pérdidas de información en caso de que haya fallos en el sistema.

#### 2.5.2.4 Requisito no funcional de diseño e implementación

Para el diseño y documentación de la aplicación se utiliza la metodología XP, usando el lenguaje de modelado UML 8.0, utilizando para el modelado visual paradigm 5.0 y el CMS Joomla como base para el desarrollo. El lenguaje empleado es PHP.

#### 2.5.2.5 Requisito no funcional de soporte

Garantía de instalación y prueba del sistema, además de un breve entrenamiento a los futuros usuarios, se contará con documentación sobre el tema.

#### 2.5.2.6 Requisito no funcional de portabilidad

El sistema podrá ser instalado en el ambiente especificado en los requisitos tecnológicos para servidores.

Después de identificar los requisitos no funcionales, una manera de presentar al cliente una visión futura de la aplicación es a través de un prototipo de interfaz de usuario no funcional, el cual se visualiza en el siguiente acápite.

## 2.6. Prototipo de interfaz de usuario

Un prototipo es una visión preliminar del sistema futuro, es un modelo operable, fácilmente ampliable y modificable, que tiene todas las características que hasta el momento debe tener el sistema.

Las ventajas principales de los prototipos son:

- ✓ Posibilidad de cambiar el modelo.
- ✓ Oportunidad para suspender el desarrollo del modelo si no es funcional.
- ✓ Posibilidad de crear un nuevo modelo que se ajuste mejor a las necesidades y expectativas del usuario.

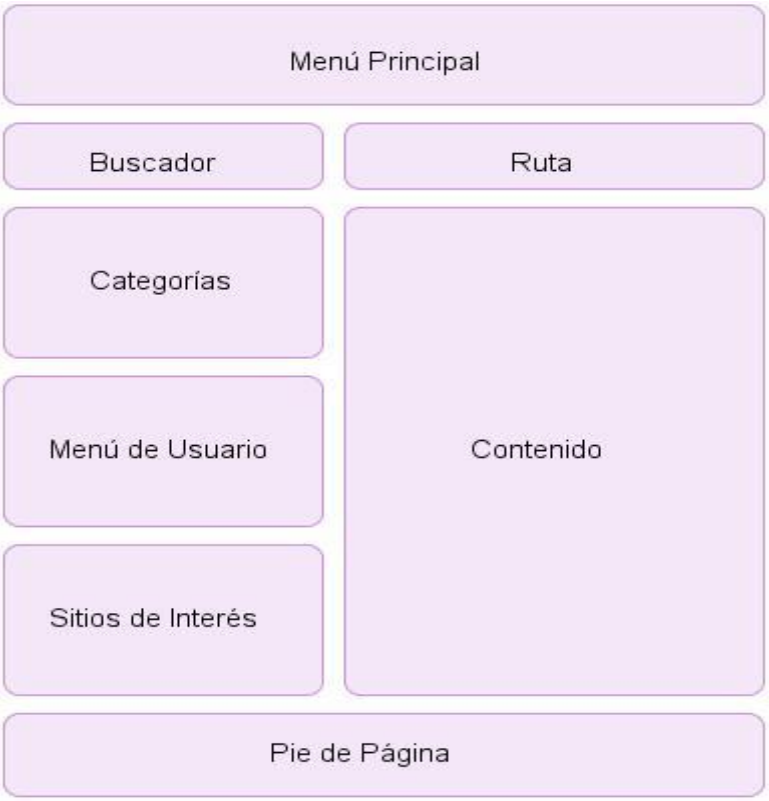



**Figura 7.** Prototipo de interfaz de usuario.

Luego que los clientes definan a grandes rasgos las historias de usuario y los programadores se familiaricen con las herramientas, tecnologías y prácticas que se utilizarán para el desarrollo de la aplicación se culmina la fase de exploración de la metodología seleccionada dando paso a la siguiente fase, Planificación de la entrega.

## 2.7. Planificación de la entrega

En esta fase el cliente establece la prioridad de cada historia de usuario, y correspondientemente, los programadores realizan una estimación del esfuerzo necesario de cada una de ellas. Se toman acuerdos sobre el contenido de la primera entrega y se determina un cronograma en conjunto con el cliente. Esta fase dura unos pocos días, identificándose además, el número y tamaño de las iteraciones, al igual que se plantean ajustes necesarios a la metodología según las características del proyecto. (23)

### 2.7.1. Estimación de esfuerzos por HU

La estimación de esfuerzos es establecida por los programadores utilizando las semanas como medida. Las HU deben ser programadas en un tiempo estimado hasta 3 semanas. Si la estimación supera las 3 semanas la HU deberá ser dividida hasta que pueda ser desarrollada en un tiempo factible. En caso de que el esfuerzo sea menor que una semana la HU será combinada con otra. La siguiente tabla muestra la estimación de esfuerzos por HU de la solución.

**Tabla 6.** Estimación de esfuerzos por HU.

| No | Historia de Usuario | Estimación (semanas) |
|---|---|---|
| 1 | Buscar información | 1 |
| 2 | Gestionar usuario | |
| 3 | Autenticar usuario | 1 |
| 4 | Registrar usuario | |
| 5 | Gestionar noticia | 1 |
| 6 | Gestionar enlace de interés | |
| 7 | Gestionar información | 1 |
| 8 | Gestionar aviso | |
| 9 | Gestionar comentario | 1 |
| 10 | Descargar documentación | |
| 11 | Gestionar documentación | 1 |
| 12 | Buscar documentación | 1 |
| 13 | Verificar contenido | 2 |
| 14 | Verificar estructura normativa | 1 |
| 15 | Recopilar criterio | 2 |
| 16 | Generar informe | 1 |

Según el análisis de los programadores y la estimación de esfuerzos por HU se concluye que se necesitan trece semanas para la implementación de la aplicación, estableciendo 4 iteraciones para la entrega del producto.

Después que los programadores realizan la estimación del esfuerzo necesario para cada una de las HU y se tomen acuerdos sobre el contenido de la primera entrega, se concluye la fase de Planificación de la entrega, dando paso a la siguiente fase, Iteraciones.

## 2.8. Iteraciones

Esta fase incluye varias iteraciones sobre el sistema antes de ser entregado. El plan de entrega está compuesto por iteraciones de no más de tres semanas. En la primera iteración se puede intentar establecer una arquitectura del sistema que pueda ser utilizada durante el resto del proyecto (como valor agregado al negocio). Esto se logra escogiendo las historias que condicionen la creación de esta arquitectura. Esto no siempre es posible ya que es el cliente quien decide qué historias se implementarán en cada iteración. Al final de la última iteración el sistema estará listo para entrar en producción.

### 2.8.1. Plan de iteraciones

El plan de iteración se compone de las historias de usuario definidas en el pan de entrega, más las pruebas de aceptación con resultados insatisfactorios en el ciclo anterior los cuales son analizados para evaluar su corrección, así como para prever que no vuelvan a ocurrir, dichas iteraciones poseen una duración de 1 a 3 semanas y no permiten realizar ninguna tarea no programada para la iteración. A continuación se procede a exponer las iteraciones definidas por los autores.

**Iteración 1:** en la primera iteración se entregarán algunas de las funcionalidades que tienen prioridad alta para el desarrollo de la aplicación, correspondiendo a las HU:

- ✓ HU 2. Gestionar Usuario.
- ✓ HU 3. Autenticar usuario.
- ✓ HU 4. Registrar usuario.
- ✓ HU 5. Gestionar noticia.
- ✓ HU 7. Gestionar información.
- ✓ HU 9. Gestionar comentario.

**Iteración 2:** en esta iteración se realizarán las HU con prioridad alta para el cliente siendo estas:

- ✓ HU 13. Verificar contenido.
- ✓ HU 14. Verificar estructura normativa.

**Iteración 3:** en esta iteración se realizarán las restantes HU que son importantes para el cliente siendo estas:

- ✓ HU 1. Buscar información.
- ✓ HU 15. Recopilar criterio.
- ✓ HU 16.Generar informe.

**Iteración 4:** en esta iteración se implementan las HU de baja prioridad para el cliente pero no menos importante que las anteriores para los desarrolladores, las que se definen por:

- ✓ HU 6.Gestionar enlace de interés.
- ✓ HU 8.Gestionar aviso.
- ✓ HU 10.Descargar documentación.
- ✓ HU 11.Gestionar documentación.
- ✓ HU 12. Buscar documentación.

### 2.8.2. Plan de duración de las iteraciones

El plan de duración de las iteraciones se realiza luego de tener el estimado en días que demora implementar cada HU. Se tiene en cuenta la prioridad que el cliente le asigna a cada historia y el nivel de complejidad que estas poseen. La siguiente tabla demuestra lo anteriormente expuesto.

**Tabla 7.** Plan de duración de las iteraciones.

| Iteraciones | Historias de usuarios | Fecha inicio | Fecha final |
|---|---|---|---|
| **Iteración 1** | HU2. Gestionar Usuario.<br>HU 3. Autenticar usuario.<br>HU 4. Registrar usuario.<br>HU 5. Gestionar noticia.<br>HU 7. Gestionar información.<br>HU 9.Gestionar comentario. | **1/2/2014** | **22/22014** |
| **Solución de no conformidades** | | **23/22014** | **25/2/2014** |
| **Iteración 2** | HU 13. Verificar contenido.<br>HU14.Verificar estructura normativa. | **26/2/2014** | **19/3/2014** |
| **Solución de no conformidades** | | **20/3/2014** | **23/3/2014** |
| **Iteración 3** | HU 1. Buscar información.<br>HU 15. Recopilar criterio.<br>HU 16.Generar informe. | **24/3/2014** | **13/4/2014** |
| **Solución de no conformidades** | | **14/4/2014** | **17/4/2014** |
| **Iteración 4** | HU 6.Gestionar enlace de interés.<br>HU 8.Gestionar aviso.<br>HU 10.Descargar documentación.<br>HU 11.Gestionar documentación.<br>HU 12. Buscar documentación. | **18/4/2014** | **8/5/2014** |
| **Solución de no conformidades** | | **9/5/2014** | **12/5/2014** |

Se concluye con el análisis de la tabla anterior que el cumplimiento del tiempo estimado para cada iteración así como la solución de las no conformidades, garantiza que la fecha de entrega de la solución se aproxime a las trece semanas de desarrollo.

### 2.8.3. Arquitectura del sistema

Según Kruchten Philippe (24) la arquitectura de software tiene que ver con el diseño y la implementación de estructuras de software de alto nivel. Es el resultado de ensamblar un cierto número de elementos arquitectónicos de forma adecuada.
La arquitectura en tres niveles es la especialización de la arquitectura cliente-servidor donde la carga se divide en tres capas: una capa para la presentación (interfaz de usuario), otra para el cálculo (donde se encuentra la lógica de negocio) y otra para el almacenamiento (persistencia de datos). Una capa solamente tiene relación con la siguiente y el desarrollo se puede llevar a cabo en varios niveles.

En Joomla la arquitectura en tres niveles está formada por las capas: extensiones, aplicación y marco de trabajo. En la figura 8 se representan las diferentes unidades pertenecientes a cada capa de la arquitectura.


**Figura 8.** Representación de la arquitectura del sistema.

**Capa de Extensión**: está formada por las extensiones del CMS: módulos, componentes y plantillas del sistema. Se trata de programas con una función específica, desarrollados total o parcialmente con el marco de trabajo de Joomla para ejecutarse sobre alguna de las aplicaciones de la capa inferior.

**Capa de Aplicación**: esta capa se comunica con la capa de extensión para recibir las solicitudes y presentar los resultados, y con la capa marco de trabajo para solicitar al gestor de base de datos almacenar o recuperar datos. En esta aparecen tres aplicaciones básicas que extienden de la clase JApplication, estas proporcionan una serie de funcionalidades para la gestión de contenido y una estructura sobre la que se ejecutan extensiones.

**Capa de Marco de trabajo**: en esta capa residen los datos y es la encargada de acceder a los mismos. Se encarga del encapsulamiento de la lógica de acceso a datos. Está formada por bibliotecas externas que proporcionan funcionalidades específicas que ya han sido desarrolladas por terceros, el marco de trabajo propiamente dicho y una serie de complementos que permiten extender sus funcionalidades.

La definición de esta arquitectura es un resultado más que se obtiene a lo largo del desarrollo de cada iteración, la misma que generalmente es definida en la iteración inicial permite el desarrollo estable del proyecto conjuntamente con diferentes artefactos que se irán obteniendo posteriormente y que se encuentran reflejados en el plan de entrega.

### 2.8.4. Plan de entrega

El plan de entrega es un plan dividido en varios planes de iteración donde las entregas van determinadas por los artefactos resultantes en cada iteración. En los planes de iteración, como se ha dicho, el usuario selecciona las HU en cada iteración y las pruebas funcionales de estas historias son validadas al final de la misma, además el cronograma de entregas establece qué HU serán agrupadas para conformar una entrega, y su orden; este cronograma será el resultado de una reunión entre todos los actores del grupo de desarrollo incluyendo el cliente permitiendo alcanzar un mayor entendimiento en la implementación del sistema.

**Tabla 8.** Plan de entrega.

| Historia de Usuario | Primera iteración | Segunda iteración | Tercera iteración | Cuarta iteración |
|---|---|---|---|---|
| Gestionar Usuario | V1.0 | Finalizado | Finalizado | Finalizado |
| Autenticar usuario | V1.0 | Finalizado | Finalizado | Finalizado |
| Registrar usuario | V1.0 | Finalizado | Finalizado | Finalizado |
| Gestionar noticia | V1.0 | Finalizado | Finalizado | Finalizado |

| | | | | |
|---|---|---|---|---|
| Gestionar comentario | V1.0 | Finalizado | Finalizado | Finalizado |
| Gestionar información | V1.0 | Finalizado | Finalizado | Finalizado |
| Verificar contenido | - | V1.0 | Finalizado | Finalizado |
| Verificar estructura normativa | - | V1.0 | Finalizado | Finalizado |
| Recopilar criterio | - | - | V1.0 | Finalizado |
| Generar informe | - | - | V1.0 | Finalizado |
| Buscar información | - | - | V1.0 | Finalizado |
| Gestionar documentación | - | - | - | V1.0 |
| Buscar documentación. | - | - | - | V1.0 |
| Descargar documentación. | - | - | - | V1.0 |
| Gestionar enlace de interés | - | - | - | V1.0 |
| Gestionar aviso | - | - | - | V1.0 |

La tabla 8, exhibe en cada columna la sucesión de iteraciones, y en cada fila las historias de usuario; todo ello para visualizar dada una intersección, con la nomenclatura del versionado, la entrega del artefacto correspondiente al momento de análisis.

## 2.1. Conclusiones del capítulo

En el presente capítulo se abordaron los elementos fundamentales correspondientes a la fase de Exploración, Planificación e Iteraciones, generando los artefactos correspondientes por fase según propone la metodología XP.

- ✓ En la fase de Exploración se definen: 16 historias de usuario que contemplan 37 funcionalidades expresadas en tareas de ingeniería, y el modelo conceptual a partir de la desagregación estructural/funcional de la propuesta de solución; que integrada al prototipo del sitio de la Facultad de Derecho de la UH resultará parte del resultado del proyecto de su rediseño.

- ✓ En la fase de Planificación se realiza la estimación del esfuerzo necesario para cada una de las HU previendo un total de trece semanas para la implementación de la aplicación; habiéndose determinado para HU de complejidad alta y media hasta dos y hasta una semana respectivamente.

- ✓ En la fase de Iteraciones se define la arquitectura en 3 capas para el sistema y se conforma el Plan de entrega que incluye cuatro iteraciones con el objetivo de planificar el trabajo del equipo de desarrollo.

# CAPÍTULO 3. IMPLEMENTACIÓN Y PRUEBA

## 3.1 Introducción

El presente capítulo basado en la metodología XP plantea que la implementación de un software debe realizase de forma iterativa e incremental, obteniéndose al final de cada iteración un producto probado y mostrado al cliente, permitiendo una constante retroalimentación (desarrolladores-cliente). También se detallan los patrones utilizados y las cuatro iteraciones llevadas a cabo durante la etapa de construcción del sistema, exponiéndose las tareas generadas por cada historia de usuario, así como las pruebas realizadas al sistema.

## 3.2 Patrones utilizados en la aplicación

### 3.2.1 Patrón arquitectónico

El patrón arquitectónico Modelo-Vista-Controlador (MVC[8]) (25), nótese en la figura 9, es utilizado para el desarrollo de la solución con el fin de separar en tres componentes distintos la interfaz de usuario, la lógica de negocio y los datos persistentes, potenciando la flexibilidad y la adaptabilidad a futuros cambios.

**El modelo:** es la representación de la información que maneja la aplicación. Son los datos puros que puestos en un contexto del sistema son mostrados al usuario por medio del controlador, proveen de información al usuario o a la aplicación misma.

**La vista:** constituye la representación del modelo en forma gráfica, disponible para la interacción con el usuario. En la aplicación la "vista" es la página HTML con contenido dinámico sobre el cual el usuario puede realizar operaciones.

**El controlador:** se encarga de responder a las solicitudes del usuario desde la interfaz, manejando los diferentes eventos a través de las funcionalidades necesarias y la información perteneciente al modelo.

**(MVC).Modelo Vista Controlador:** es un estilo de arquitectura de software que separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos.

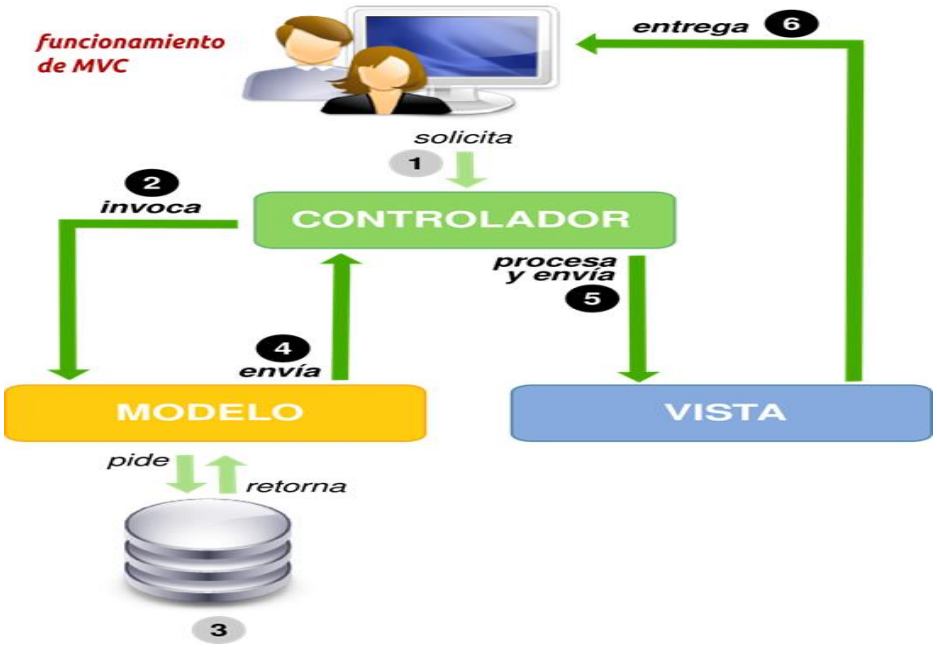



**Figura 9.** Representación del patrón MVC.

### 3.2.2 Patrones de diseño

Los patrones de diseño son la base para la búsqueda de soluciones a problemas comunes en el desarrollo de software y otros ámbitos referentes al diseño de interacción o interfaces. A continuación se realiza una breve descripción de los patrones utilizados durante la implementación y dónde se encuentran evidenciados en el sistema. (26)

**Patrones GoF:**

- ✓ **Instancia única (Singleton):** garantiza que se cree una sola instancia de una clase específica logrando de esta forma que en todos los lugares en las que se requiera se utilice el mismo objeto eliminando de esta forma altos consumos de memoria. Muchas de las clases de Joomla utilizan un modelo pseudo-singleton que permite crear instancias de objetos y de acceso.

- ✓ **El patrón de la fábrica (Factory):** se utiliza para construir y devolver objetos. El patrón de la fábrica se utiliza en los casos en donde diferentes clases, por lo general derivan de una clase abstracta, son instanciadas dependiendo de los parámetros. Joomla proporciona la clase estática JFactory, que implementa el patrón de la fábrica. Esta clase es importante porque nos permite acceder de forma sencilla a instancias de objetos globales. En los casos en donde JFactory y una clase provean un método para devolver una instancia de la clase, en general debes usar el método JFactory con preferencia. Si la clase proporciona un método getInstance () más amplio que JFactory, puede que quieras usar el método de clase para obtener una instancia adaptados específicamente a tus necesidades.

**Patrones GRAPS:**

- ✓ **Creador:** resuelve el problema de quien debe crear una instancia de alguna clase. Este se evidencia en la clase Comprobación.php y VerifyContent.php.
- ✓ **Experto**: indica cual es la clase que debe asumir una determinada responsabilidad (clase experta), teniendo en cuenta la información que aporta para cumplirla. Este se evidencia en la definición de las clases según las funcionalidades que realizan. Ejemplo: ConvertirPdfText.php y VerifyContent.php.
- ✓ **Controlador:** define quién debe responder a determinados eventos. Una clase controladora debe ser la encargada de manejar los eventos dentro de la funcionalidad, así la aplicación del patrón conlleva a separar la lógica de negocios de la capa de presentación, al aplicar estos principios, el controlador no realiza las actividades mencionadas sino que las delega en otras clases. La clase Comprobación.php es un ejemplo de la aplicación de este patrón.

Expuestos los patrones utilizados tanto los encargados de la asignación de responsabilidades (GRAPS), como el patrón arquitectónico (MVC), el cual divide en tres componentes distintos la lógica de negocio, la interfaz de usuario y los datos persistentes, se hace necesario el uso de un lenguaje que describa este último, ya que el mismo representa la información del sistema a través de una base de datos, para ello se emplea un modelo de datos.

### 3.3 Diseño de la base de datos

El modelo de datos es un lenguaje utilizado para la descripción de una base de datos. Por lo general, permite describir las estructuras de datos de la base (el tipo de los datos que incluye la base y la forma en que se relacionan), las restricciones de integridad (las condiciones que los datos deben cumplir para reflejar correctamente la realidad deseada) y las operaciones de manipulación de los datos (agregado, borrado, modificación y recuperación de los datos de la base) (27). El diseño de la base de datos es de gran importancia para el almacenamiento de los datos, y para permitir a los usuarios recuperarlos y actualizarlos en base a sus peticiones posteriores.

El modelo de datos describe la representación lógica y física de los datos persistentes en el sistema, es frecuentemente necesitado cuando se tiene un modelo de objetos y el mecanismo de almacenamiento se basa en una base de datos relacional.
El modelo de datos del portal tiene alta complejidad, debido a que está desarrollado en el CMS Joomla con un número de entidades manejadas por el sistema, por lo que no se contemplarán todas en el modelo.

A continuación solo se visualizarán aquellas entidades principales que conforman el sistema base para lograr un mayor entendimiento del mismo.

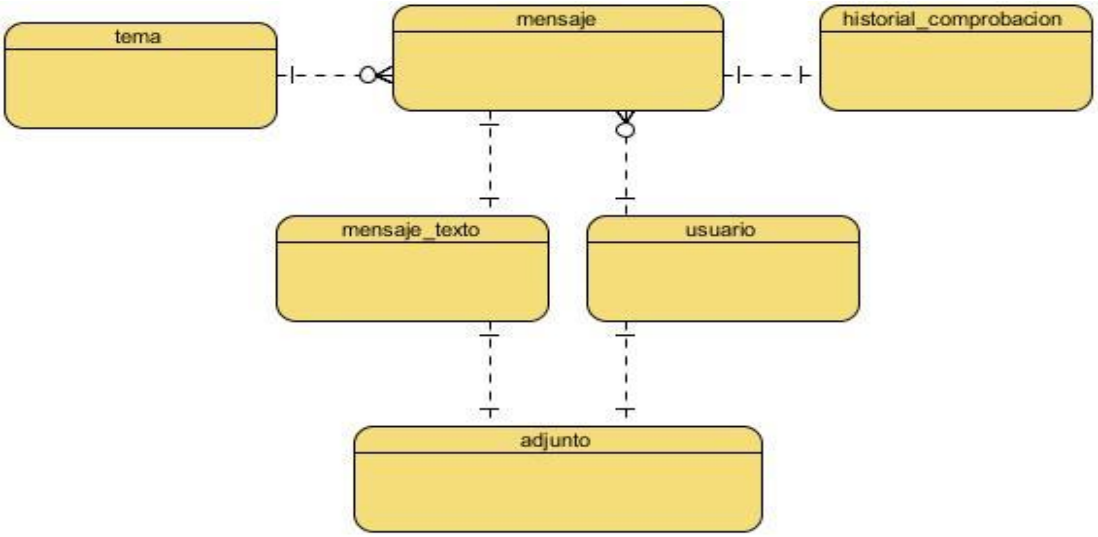



**Figura 10.** Diagrama Entidad-Relación de la base de datos.

### 3.4 Tarjetas CRC

Según la bibliografía consultada XP estimula el uso de tarjetas CRC como un mecanismo eficaz para pensar en el software en un contexto orientado a objetos, las mismas identifican y organizan las clases orientadas a objetos, siendo este el único producto de trabajo de diseño que se genera como parte del proceso XP.

Las tarjetas están compuestas por el nombre de la clase colocado como título, en la parte izquierda se colocan las responsabilidades (funcionalidades) y en la parte derecha las clases que se implican en cada responsabilidad.

- ✓ **Clase:** es cualquier persona, cosa, evento, concepto, pantalla o reporte.
- ✓ **Responsabilidades**: las responsabilidades de una clase son las entidades que conoce y las que realizan, sus atributos y métodos.
- ✓ **Colaboradores:** los colaboradores de una clase son las demás clases con las que trabaja en conjunto para llevar a cabo sus responsabilidades.

A continuación se muestra una selección de las tarjetas CRC confeccionadas por el equipo de desarrollo así como el formato general de dichas tarjetas. Las tarjetas seleccionadas son ProjectLegislativo y VerifyContent, ya que estas representan algunas de las clases más significativas que compone la solución.

**Tabla 9.** Formato general de la tarjeta CRC.

| Nombre de clase | |
|---|---|
| **Responsabilidades** | **Colaboradores** |

**Tabla 10.** Tarjeta CRC ProjectLegislativo.

| Tarjeta CRC | |
|---|---|
| **Clase:** ProjectLegislativo | |
| **Responsabilidades** | **Colaboradores** |
| ✓ __construct<br>✓ getId_proyecto<br>✓ getId_moderador<br>✓ getTitulo<br>✓ getTipo_proyecto<br>✓ getEmisor<br>✓ getDireccion<br>✓ getComentario<br>✓ setId_proyecto<br>✓ setId_moderador<br>✓ setTitulo<br>✓ setTipo_proyecto<br>✓ setEmisor<br>✓ setDireccion<br>✓ setComentario<br>✓ getAll<br>✓ getListAll<br>✓ selectDirection | ✓ Modelo |

**Tabla 11.** Tarjeta CRC VerifyContent.

| Tarjeta CRC | |
|---|---|
| **Clase :**VerifyContent | |
| **Responsabilidades** | **Colaboradores** |
| ✓ __construct<br>✓ convertPdf | ✓ ConvertirPdfText |
| ✓ posInitText<br>✓ posEndText<br>✓ isTextExact<br>✓ extractText<br>✓ getLengContent | |

| | |
|---|---|
| ✓ getCountSubSensible<br>✓ getPorCuantos<br>✓ deleteEndIniCaracter<br>✓ getPorTanto<br>✓ resolución<br>✓ decretoLey<br>✓ ley<br>✓ decreto<br>✓ clasificacionPorCuantos<br>✓ comprobacionFirma | |

### 3.5 Estándares de codificación

Los estándares de codificación son pautas de programación que no están enfocadas a la lógica del programa, sino a su estructura y apariencia física para facilitar la lectura, comprensión y mantenimiento del código. Los estándares permiten que el código que se desarrolló sea de una mejor calidad, que el mantenimiento de los programas se vuelva menos complejo, y provoca que baje la tasa de errores ingenuos. A continuación un resumen de los estándares de codificación aplicados en la implementación de la solución:

- ✓ **Indentación:** la indentación debe ser a cuatro espacios sin caracteres de tabulación. Esto es debido a que ciertos IDE's de desarrollo introducen caracteres de tabulación cuando indentan un texto automáticamente.
- ✓ **Estructuras de control:** las estructuras de control deben tener un espacio entre las palabras clave de la estructura, el signo de apertura de paréntesis para distinguir entre las llamadas de las funciones y el signo de llaves deben estar sobre la línea de la estructura.
- ✓ **Llamadas de funciones:** las funciones deben ser llamadas sin espacios entre el nombre de la función, el signo de paréntesis y el primer parámetro, espacios entre cada coma por parámetro y sin espacios entre el último paréntesis, el signo de paréntesis cerrado y el signo de punto y coma (;).
- ✓ **Estilos de comentarios:** el estilo de los comentarios debe ser como el estilo de comentarios para C (/* */ o //), no debe de utilizarse el estilo de comentarios de Perl (#).
- ✓ **Inclusión de archivos:** cuando se incluya un archivo de dependencia incondicionalmente se debe utilizar require_once y cuando sea condicionalmente, utilice include_once.
- ✓ **Bloques de código:** siempre se debe utilizar las etiquetas <?php?> para abrir un bloque de código. No utilizar el método de etiquetas cortas, porque esto depende de las directivas de configuración en el archivo PHP.INI y hace que el script no sea tan portable.

- ✓ **Nombres:** los nombres de las clases deben de iniciar con letra mayúscula. Los nombres de las variables y de las funciones pueden iniciar con letra minúscula, pero si estas tienen más de una palabra, cada nueva palabra debe iniciar con letra mayúscula (el nombre puede escribirse separado por signos de guión mayor). Si una función, en una clase, es privada; deberá comenzar con el signo de guión mayor para una fácil identificación. Las constantes deben de escribirse siempre en mayúsculas y tanto estas como las variables globales deben de tener como prefijo el nombre de la clase a la que pertenecen.
- ✓ **Constantes:** los nombres de las constantes pueden contener caracteres alfanuméricos, guiones bajos (_) y números. Todas las letras deben estar en mayúscula. Las constantes deben ser definidas como miembros de una clase usando el modificador "const". No se recomienda definir constantes con alcance global (utilizando la función "define").

### 3.6 Tareas de ingeniería

En la planificación de la iteración según la metodología XP se definen las actividades para las siguientes tres o cuatro semanas, aquí el cliente elige las HU de mayor valor para ser implementadas en la iteración planeada y a continuación los programadores dividen las mismas en tareas más pequeñas denominadas tareas de ingeniería, eligen las tareas que desean implementar, las analizan en mayor detalle y realizan una estimación de su tiempo de desarrollo. Finalmente, el cliente ordena en función de sus necesidades las HU estimadas, dejando para iteraciones posteriores aquellas que sobrepasan la capacidad productiva de la iteración.

En la tabla 12 se muestra el formato general de una tarea de ingeniería, mientras que las tablas 13 y 14 representan las tareas de ingeniería verificar relaciones entre normas y verificar relación autoridad-facultad respectivamente, por ser estas algunas de las funcionalidades críticas que fueron implementadas para dar cumplimiento a la propuesta de solución.

**Tabla 12.** Formato general de las tareas de ingeniería.

| Tarea | |
|---|---|
| **Número tarea**: número consecutivo a partir del 1. | **Número historia:** identifica la HU |
| **Nombre tarea**: | |
| **Tipo de tarea**: desarrollo/ corrección/ mejora/ otra | **Puntos estimados:** permite estimar la duración de la implementación, representando en días. |
| **Fecha inicio**: comienzo de la tarea | **Fecha fin**: fin de la tarea |
| **Programador responsable**: persona encargada de la realización de la tarea de ingeniería. | |
| **Descripción**: explica en qué consiste la tarea de ingeniería. | |

**Tabla 13.** Tarea de ingeniería Verificar relaciones entre normas.

| Tarea | |
|---|---|
| **Número tarea**: 1 | **Número historia**: 13 |
| **Nombre tarea**: Verificar relaciones entre normas | |
| **Tipo de tarea**: desarrollo | **Puntos estimados**: 3 |
| **Fecha inicio:** 26/2/2014 | **Fecha fin:** 28/2/2014 |
| **Programador responsable:** Gabriel Alejandro Gonzales y Alejandro Pedraza Fernández | |
| **Descripción:** el sistema muestra y clasifica los por cuantos, permite al especialista analizar y comentar sobre estos. El comentario debe ser clasificado en adición, eliminación y modificación. | |

**Tabla 14.** Tarea de ingeniería Verificar relación autoridad-facultad.

| Tarea | |
|---|---|
| **Número tarea**: 2 | **Número historia**: 13 |
| **Nombre tarea**: Verificar relación autoridad-facultad | |
| **Tipo de tarea**: desarrollo | **Puntos estimados**: 3 |
| **Fecha inicio:** 28/2/2014 | **Fecha fin:** 1/3/2014 |
| **Programador responsable:** Gabriel Alejandro Gonzales y Alejandro Pedraza Fernández | |
| **Descripción:** el sistema valida que el pie de firma del Proyecto legislativo se corresponda con el promulgador del mismo, muestra el por tanto y permite al especialista analizar y comentar sobre el contenido que se analiza. | |

### 3.7 Diagrama de despliegue

El diagrama de despliegue permite evaluar de forma visual cómo se encuentran relacionados físicamente los componentes de la aplicación. En este caso la aplicación se encuentra hospedada en un servidor web Apache y se comunica con un gestor de base de datos MySQL.

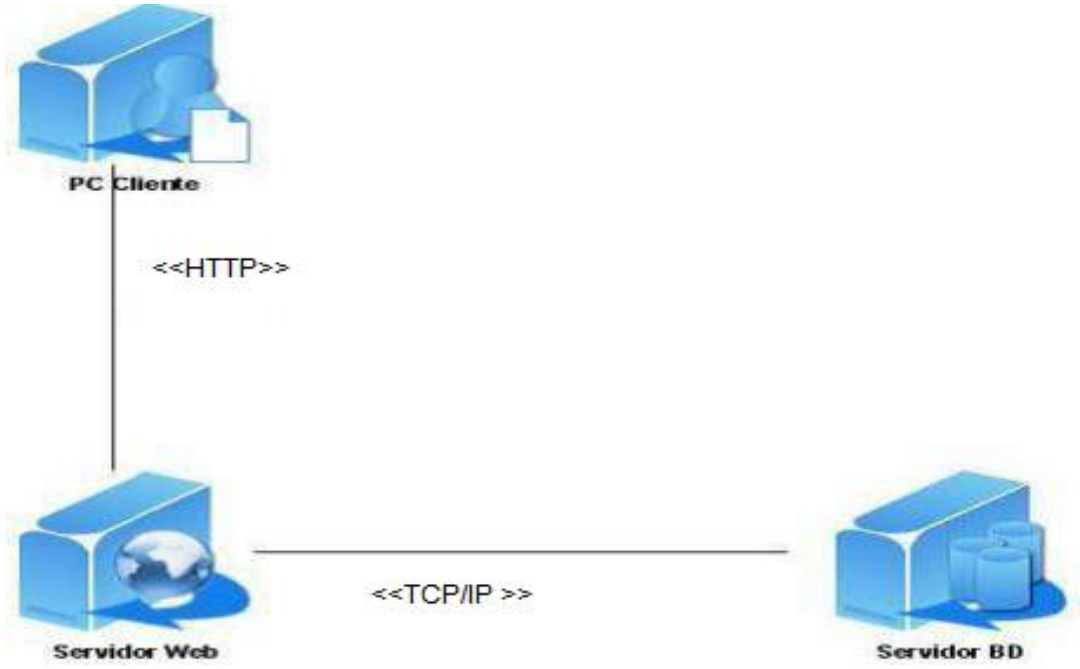



**Figura 11.** Diagrama de despliegue del sistema.

**Nodos:**

- ✓ **PC cliente:** representa una computadora desde la cual el usuario accede a la aplicación.
- ✓ **Servidor de aplicación Apache:** representa una estación donde se instala el servidor Apache sobre el cual se despliega la aplicación.
- ✓ **Servidor de base de datos MySQL:** representa el servidor donde se instala el Sistema Gestor de Base de Datos MySQL que responde a las peticiones hechas por la aplicación.

**Protocolos:**

- ✓ **TCP/IP (Protocolo de comunicación y transmisión/ Protocolo de Internet):** se utiliza en la comunicación entre el servidor y la base de datos para realizar operaciones sobre la información de las tablas.
- ✓ **HTTP (Protocolo de transferencia de hipertexto):** establece un esquema de comunicación cliente –servidor. El cliente es el navegador web que realiza las peticiones a las que el servidor se encarga de dar respuesta.

## 3.8 Interfaces de la aplicación

El portal web de la Facultad de Derecho de la UH, figura 12, se enfoca en la utilización de cada uno de los componentes desarrollados, los cuales permiten la comunicación entre los usuarios de la comunidad. Para ello se lleva a cabo la gestión de la información relacionada con los temas de carácter jurídico e institucional.

La aplicación está constituida por una serie de funcionalidades con el objetivo de facilitar la corrección a nivel de estructura y contenido de las nuevas normas, además de permitir el almacenamiento de estos proyectos, materiales de estudio referente a la facultad y todos aquellos informes generales que se generen a partir del proceso de consulta popular, la interfaz principal de la misma se evidencia en la figura 12.


**Figura 12.** Vista principal de la aplicación.

**Foro**

En la figura 13 se muestra el componente foro, el mismo tiene como funcionalidad que el usuario exponga sus dudas, criterios e ideas acerca de alguna temática abordada en el sitio, específicamente aquellos temas ligados al estudio y verificación de los proyectos legislativos. El especialista o moderador se encargará de visualizar los comentarios que más tarde estarán presentes en el Informe General que emite la aplicación.

Mediante la diversificación de criterios se aclaran dudas determinadas acerca del Proyecto legislativo en cuestión, permitiendo al usuario proponer cambios y emitir su posición ante el mismo.

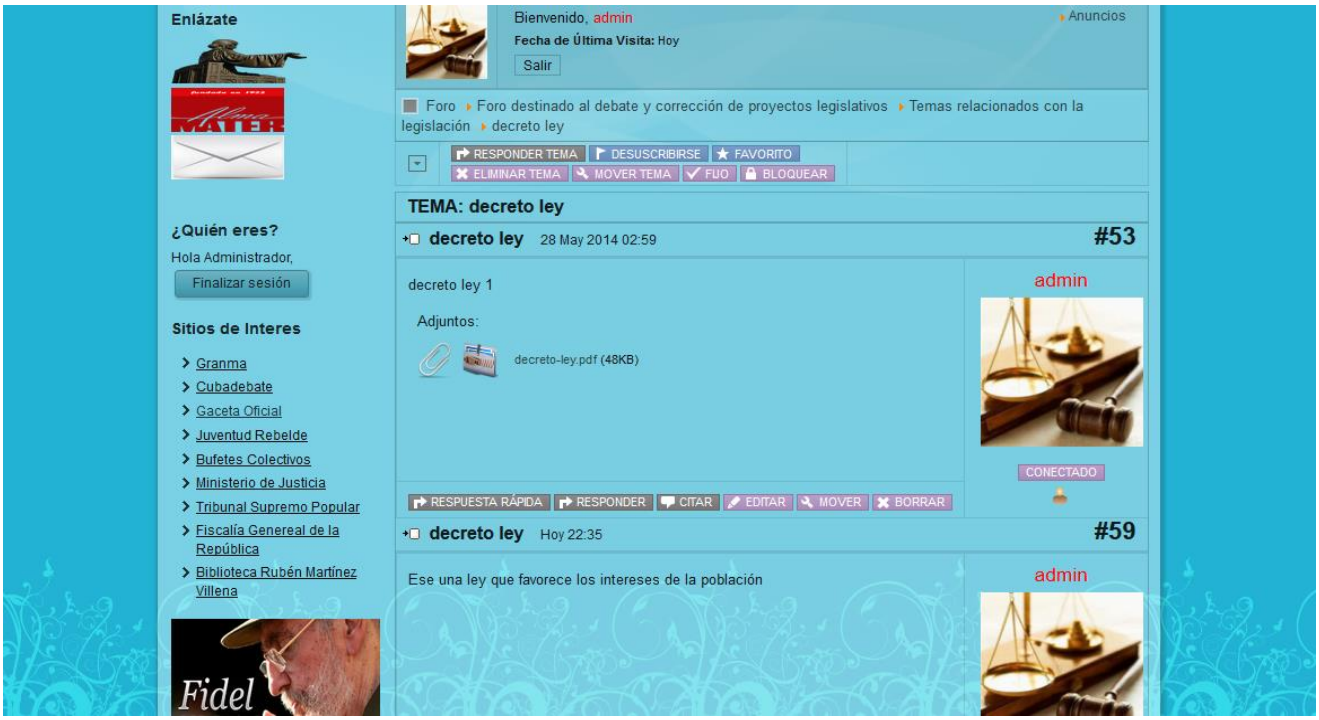



**Figura 13.** Vista principal del componente foro.

### Repositorio institucional

El componente repositorio institucional presentado en la figura 14, permite al usuario realizar descargas de diversos documentos previamente archivados por los administradores y especialistas de la comunidad.

**Figura 14.** Vista principal del repositorio institucional.

### Verificar contenido

El rol encargado de analizar el contenido de los tipos de normas y sus relaciones, encuentra en el componente verificar contenido expuesto en la figura 15, el apoyo a la hora de hallar ambigüedades y errores que no se tuvieron presente en el momento de redacción del texto normativo.

Verifica la existencia de puntos importantes en las normas, como la correcta relación entre el pie de firma con su correspondiente órgano emisor, y el anuncio de la fecha de entrada en vigor del Proyecto legislativo en análisis.

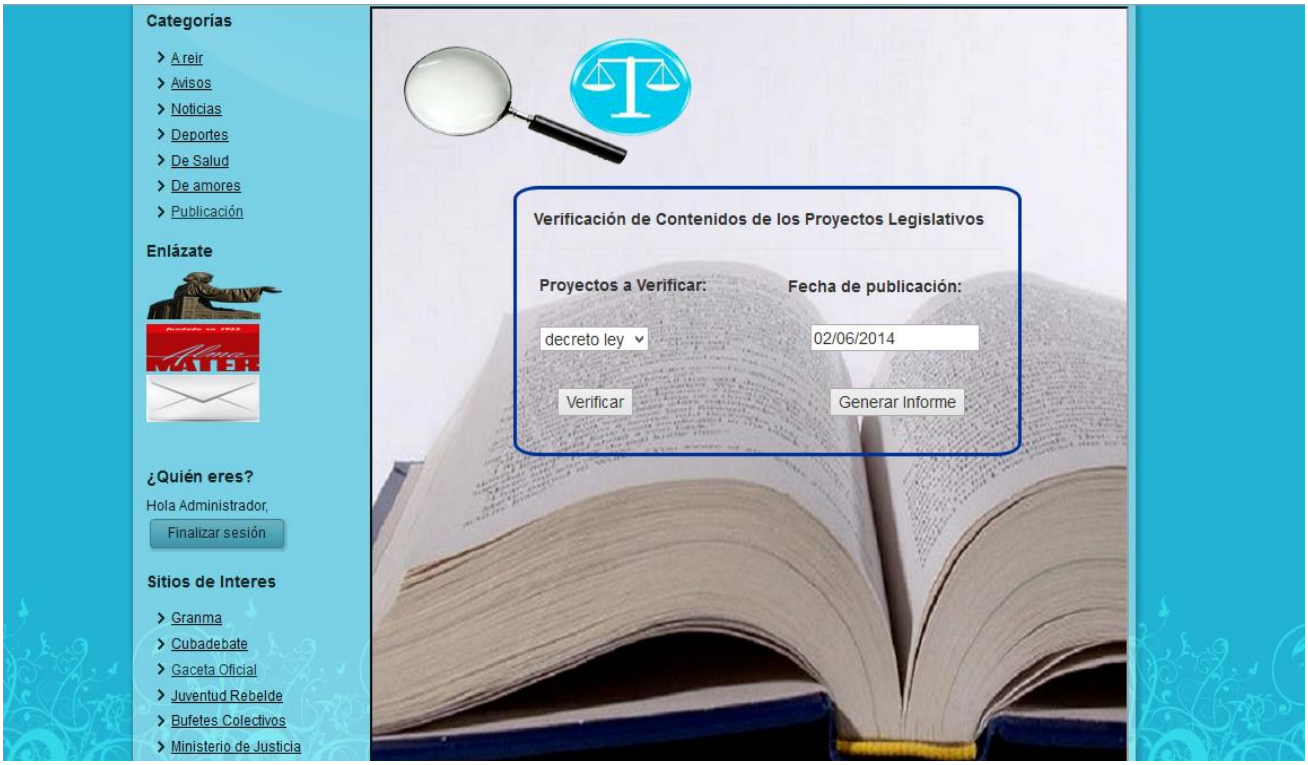



**Figura 15.** Vista principal de la verificación de contenido.

### Verificación de estructura normativa

Para analizar la estructura normativa de cada norma es desarrollado el componente mostrado en la figura 16, el mismo se basa en las comparaciones de cada etiqueta XML con las etiquetas del esquema (XSD[9]) que responde a la norma cubana de documentos legislativo. Permite cargar el documento en estudio y posteriormente mostrar aquellos errores existentes en el mismo.

**XSD:** es un formato para definir la estructura de un documento XML. XSD sustituye al anterior formato DTD, y añade funcionalidad para definir la estructura XML con más detalle.


**Figura 16.** Vista principal de la verificación de estructura normativa.

## 3.9 Pruebas

Las pruebas constituyen una etapa imprescindible durante el proceso de desarrollo del software. Su objetivo principal es asegurar que el software cumpla con las especificaciones requeridas y eliminar los posibles defectos que este pudiera tener. De forma más específica los propósitos de las pruebas son:

- ✓ Realizar la validación del software desarrollado, entendiendo como validación el proceso que determina si el software satisface los requisitos.
- ✓ Realizar la verificación del software desarrollado, entendiendo como verificación el proceso que determina si los productos de una fase satisfacen las condiciones de la misma.(28)

**Niveles de prueba:**

A la hora de evaluar dinámicamente un sistema se debe comenzar por los componentes más simples y pequeños e ir avanzando progresivamente hasta probar todo el software en su conjunto. Las pruebas se aplican en distintos niveles de trabajo, dentro de estos se distinguen:

- ✓ **Pruebas de unidad:** prueba individual a las unidades separadas de un sistema de software.

- ✓ **Pruebas de integración:** los componentes individuales son combinados con otros componentes para asegurar que la comunicación, enlaces y los datos compartidos ocurran apropiadamente.
- ✓ **Pruebas del sistema:** son usualmente conducidas para asegurar que todos los módulos trabajan como sistema sin error. Es similar a la prueba de integración pero con un alcance mucho más amplio.
- ✓ **Pruebas de aceptación:** son realizadas principalmente por los usuarios con el apoyo del equipo del proyecto. El propósito es confirmar que el sistema está terminado, que desarrolla puntualmente las necesidades de la organización y que es aceptado por los usuarios finales.

**Métodos de prueba:** un método de prueba es un enfoque sistemático, independiente del nivel en que se enmarque la prueba, que ayuda a encontrar buenos conjuntos de casos de prueba[10] para detectar diferentes tipos de errores. (29)

1. **Pruebas de caja blanca:** se comprueban los componentes internos. Comprueba los caminos lógicos del software proponiendo casos de prueba que ejerciten conjuntos específicos de condiciones. Se puede examinar el estado del programa en varios puntos para determinar si el estado real coinciden con el esperado o mencionado.

Este método de casos de prueba usa los detalles procedimentales del programa. Se busca obtener casos de prueba que:

- ✓ Garanticen que se ejecuta por lo menos una vez todos los caminos independientes de cada módulo.
- ✓ Verificar las decisiones lógicas (V/F).
- ✓ Ejecutar las estructuras internas de datos para asegurar su validez.

Dentro del método de caja blanca se incluyen técnicas de prueba tales como:

- ✓ **La prueba del camino básico:** permite al diseñador de casos de prueba obtener una medida de la complejidad lógica de un diseño procedimental y usar esa medida como guía para la definición de un conjunto básico de caminos de ejecución.

Los pasos que se siguen para aplicar esta técnica son:

- ✓ A partir del diseño o del código fuente, se dibuja el grafo de flujo asociado.
- ✓ Se calcula la complejidad ciclomática del grafo.
- ✓ Se determina un conjunto básico de caminos independientes.

**Casos de prueba:** especifican una forma de probar la solución, incluye: la entrada, las condiciones bajo las cuales ha de probarse y los resultados esperados.

- ✓ Se preparan los casos de prueba que obliguen a la ejecución de cada camino del conjunto básico.

- ✓ **La prueba de condición:** es un método de diseño de casos de prueba que ejercita las condiciones lógicas contenidas en el módulo de un programa.
- ✓ **La prueba de flujo de datos:** se selecciona caminos de prueba de un programa de acuerdo con la ubicación de las definiciones y los usos de las variables del programa.
- ✓ **La prueba de bucles:** es una técnica de prueba de caja blanca que se centra exclusivamente en la validez de las construcciones de bucles.

2. **Pruebas de caja negra:** a este tipo de prueba también se le conoce como prueba de caja opaca o inducida por los datos. Se centran en lo que se espera de un módulo, esta prueba se limita a brindar solo datos como entrada y estudiar la salida, sin preocuparse de lo que pueda estar haciendo el módulo internamente, es decir, solo trabaja sobre su interfaz externa.

Este tipo de pruebas permite encontrar:

- ✓ Funciones incorrectas o ausentes.
- ✓ Errores de interfaz.
- ✓ Errores en estructuras de datos o en accesos a las bases de datos externas.
- ✓ Errores de rendimiento.
- ✓ Errores de inicialización y terminación.

Dentro de la prueba de caja negra se incluyen varias técnicas de prueba tales como:

- ✓ **Partición de equivalencia:** divide el campo de entrada en clases de datos que tienden a ejercitar determinadas funciones del software.
- ✓ **Análisis de valores límites:** prueba la habilidad del programa para manejar datos que se encuentran en los límites aceptables.
- ✓ **Grafos de causa-efecto:** permite al encargado de la prueba validar complejos conjuntos de acciones y condiciones.

### 3.9.1 Pruebas realizadas a la solución

La metodología XP divide las pruebas en dos grupos: pruebas unitarias, desarrolladas por los programadores, encargadas de verificar el código de forma automática y las pruebas de aceptación, destinadas a evaluar si al final de una iteración se obtuvo la funcionalidad requerida, además de comprobar que dicha funcionalidad sea la esperada por el cliente.

### 3.9.1.1 Pruebas de unidad

Se le denomina prueba de unidad al proceso de separar y probar el correcto funcionamiento de las partes individuales de un programa para asegurar que cada uno de estos se ejecuta correctamente por separado.

Ventajas de usar este tipo de prueba:

- ✓ Los errores son más fáciles de localizar.
- ✓ Los errores están más acotados.
- ✓ Se fomenta el cambio.
- ✓ Se reducen los “efectos secundarios”.
- ✓ Se da más seguridad al programador.

A continuación se muestran las pruebas realizadas para este nivel mediante el método de caja blanca con la técnica de prueba de camino básico.

La figura muestra las sentencias de código del método fechaAplicacion (), la cual se encuentra enumerada y posteriormente el grafo de flujo correspondiente a este método.

```
public function fechaAplicacon($fechaPublic)                    //1
{
    $iniPos = $this->pos;                                       //1
    $dias = 3;                                                  //1
    if(preg_match("/vigor/",$this->contenido,$coincidencias,0,$iniPos))
    {                                                           //2
        $posIni = $this->postIniText($coincidencias[0],$iniPos,$this->contenido);
        $posFin = $posIni + 20;

        $text = $this->extractText($posIni,$posFin,$this->contenido);

        if(preg_match("/[0-9]+/",$text,$coincidencias,0,$posIni))
        {                                                       //3
            $dias = $coincidencias[0];
        }                                                       //4
    }                                                           //5

    $fechaAplicacion = $this->dameFecha($fechaPublic,$dias);    //6

    return $fechaAplicacion;                                    //7
}                                                               //8
```

**Figura 17.** Método que muestra la fecha de entrada en vigor del Proyecto legislativo.

**Notación del grafo de flujo**

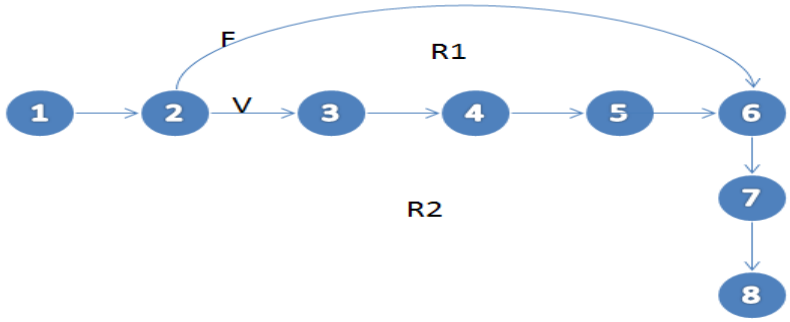



**Figura 18.** Grafo de flujo asociado al método.

Una vez construido el grafo de flujo asociado al procedimiento anterior se determina la complejidad ciclomática, la cual garantiza la menor cantidad de casos de pruebas a realizar, para que se ejecute al menos una vez cada sentencia de código. El cálculo de la complejidad ciclomática es necesario efectuarlo mediante tres vías o fórmulas, se debe utilizar el mismo grafo en cada caso:

**Fórmula 1**.V (G)= (A-N)+2 A: cantidad total de aristas del grafo.
Resultado. N: cantidad total de nodos del grafo.
V (G) = (8-8) + 2
V (G) =2
**Fórmula 2**.V (G)= P+1 P: cantidad total de nodos predicados.
Resultado. Un nodo es predicado cuando parten del mismo 2 o más aristas.
V (G) = 1+1
V (G) = 2
**Fórmula 3**.V (G)=R
R: cantidad total de regiones existentes en el grafo, se incluye el área exterior del grafo, como una región más.
**Resultado** V (G) = 2
A continuación se especifican los caminos básicos que puede tomar el algoritmo durante su ejecución. En estas representaciones se subrayan los elementos de cada camino que los hacen independientes a los demás.

**Tabla 15.** Caminos básicos del flujo.

| Número | Caminos básicos |
|---|---|
| 1 | 1-2-3-4-5-6-7-8 |
| 2 | 1-2-6-7-8 |

Luego de haber extraído los caminos básicos del flujo, se procede a ejecutar los casos de pruebas para este procedimiento. Para realizarlos es necesario cumplir con las siguientes exigencias:

- ✓ **Descripción:** se hace la entrada de datos necesaria, validando que ningún parámetro obligatorio pase nulo al procedimiento o no se entre algún dato erróneo.
- ✓ **Condición de ejecución:** se especifica cada parámetro para que cumpla una condición deseada para ver el funcionamiento del procedimiento.
- ✓ **Entrada:** se muestran los parámetros que entran al procedimiento.
- ✓ **Resultados esperados:** se expone el resultado que se espera que devuelva el procedimiento.
- ✓ **Resultados:** se muestra el resultado obtenido.

Seguidamente se presenta los casos de pruebas de caja blanca generados.

**Tabla 16.** Caso de prueba para el camino básico #1.

| **Camino básico#1: 1-2-3-4-5-6-7-8** | |
|---|---|
| Descripción | Se obtiene la fecha de entrada en vigor del Proyecto legislativo a partir de n días después de la fecha de publicación. |
| Condición de ejecución | Se debe tener la fecha de publicación ($fechaPublic) |
| Entrada | $iniPos = $this->pos;<br><br>$dias = 3; |
| Resultado esperado | Se obtiene la fecha. |
| Resultado | Satisfactorio. |

**Tabla 17.** Caso de prueba para el camino básico #2.

| **Camino básico#2: 1-2-6-7-8** | |
|---|---|
| Descripción | Se obtiene la fecha de entrada en vigor del Proyecto legislativo a partir de 3 días de ser publicado. |
| Condición de ejecución | Se debe tener la fecha de publicación ($fechaPublic) |
| Entrada | $iniPos = $this->pos;<br><br>$dias = 3; |
| Resultado esperado | Se obtiene la fecha. |
| Resultado obtenido | Satisfactorio. |

#### 3.9.1.1.1 Resultado de las pruebas unitarias realizadas

Para validar el código implementado, se generaron un total de 20 casos de pruebas. De las pruebas unitarias realizadas tres estuvieron relacionadas con la funcionalidad ValidarXML perteneciente al componente verificar estructura normativa, donde 18 de los casos de pruebas mencionados con anterioridad resultaron satisfactorios representando un 90%, evidenciando la lógica aplicada en el código probado, mientras que las 2 pruebas restantes resultaron fallidas representando un 10%. En una segunda iteración de un total de 20 pruebas se obtuvo como resultado 20 pruebas satisfactorias representando un 100%, quedando probado más del 80% del código de la aplicación. En la siguiente figura se muestra de forma gráfica resultado.


**Figura 19.** Representación de los casos de pruebas satisfactorios y fallidos.

### 3.9.1.2 Pruebas de aceptación

Las pruebas de aceptación se realizan a partir de las historias de usuarios, es por ello que en este tipo de prueba se utilizó el método de prueba de caja negra. Durante las iteraciones las historias de usuarios escogidas serán traducidas a prueba de aceptación. (30)

En ella se especifican, la perspectiva del cliente, y los escenarios para probar que la historia de usuario ha sido implementada correctamente. Una historia de usuario puede tener todas las pruebas de aceptación que desee para asegurar su funcionamiento. El objetivo específico de esta prueba es garantizar que los requerimientos han sido cumplidos y que el sistema ha sido aceptable. (7)

A continuación se muestra una selección de las pruebas de aceptación propuestas a realizarse por cada iteración.

**Casos de prueba de la iteración 1.**

**Tabla 18.** Caso de prueba_ autenticar usuario en el sistema.

| Caso de prueba de aceptación | |
|---|---|
| **Código:** HU 2_P1 | **Historia de usuario :**2 |
| **Nombre:** Probar_autenticar usuario en el sistema. | |
| **Descripción:** prueba para la funcionalidad autenticar usuario. | |
| **Condiciones de Ejecución:** el usuario debe estar previamente registrado.<br>El usuario y contraseña deben ser válidos. | |
| **Pasos de Ejecución:** se intenta autentificar un usuario en el sistema con los datos válidos. | |
| **Resultados Esperados:** el usuario se autentica correctamente en el sistema. | |

**Tabla 19.** Caso de prueba_ registrar usuario en el sistema.

| Caso de prueba de aceptación | |
|---|---|
| **Código:** HU 3_P1 | **Historia de usuario :**3 |
| **Nombre:** Probar_registrar usuario en el sistema. | |
| **Descripción:** prueba para la funcionalidad registrar usuario. | |
| **Condiciones de Ejecución:** el usuario debe llenar correctamente los campos requeridos. | |
| **Pasos de Ejecución:** se intenta registrar un usuario en el sistema con los datos válidos. | |
| **Resultados Esperados:** el usuario se registra correctamente en el sistema. | |

**Casos de prueba de la iteración 2.**

**Tabla 20.** Caso de prueba_ verificar contenido del Proyecto legislativo.

| Caso de prueba de aceptación | |
|---|---|
| **Código:** HU 13_P1 | **Historia de usuario :**13 |

| |
|---|
| **Nombre:** Probar_verificar contenido del Proyecto legislativo. |
| **Descripción:** prueba para la funcionalidad verificar contenido del nuevo Proyecto legislativo. |
| **Condiciones de Ejecución:** el usuario debe estar autenticado.<br><br>Se utilizará un usuario con datos válidos y con permisos para ejecutar esta funcionalidad. |
| **Pasos de Ejecución:** en el menú principal proceso de consulta se selecciona la opción verificar contenido, se selecciona el nuevo Proyecto legislativo a verificar.<br><br>El usuario entra al sistema la fecha de publicación en la Gaceta Oficial del nuevo Proyecto legislativo. |
| **Resultados Esperados:** el sistema permite al usuario emitir y clasificar un comentario sobre el contenido a analizar, las relaciones entre normas (análisis de los POR CUANTO), la relación autoridad facultad (análisis del POR TANTO).<br>El sistema emite un mensaje que muestre si el pie de firma es el correcto, también muestra la fecha de entrada en vigencia del Proyecto legislativo que se está analizando y verifica si dicho proyecto se publicará en la Gaceta Oficial.<br>El sistema muestra el contenido referente a las disposiciones finales del Proyecto legislativo y permite al usuario verificar si existe una correspondencia entre estas disposiciones y los POR CUANTOS del preámbulo. |

**Tabla 21.** Caso de prueba_ funcionalidad cargar documento XML.

| **Caso de prueba de aceptación** | |
|---|---|
| **Código:** HU 14 _P1 | **Historia de Usuario:** 14 |
| **Nombre:** Probar_funcionalidad cargar documento XML. | |
| **Descripción:** prueba para la funcionalidad de cargar documento XML en el sistema. | |
| **Condiciones de Ejecución:** el usuario debe estar autenticado en el sistema.<br><br>Se utilizará un usuario con datos válidos y con permisos para ejecutar esta funcionalidad. | |
| **Pasos de ejecución**: en el menú principal proceso de consulta se selecciona la opción verificar estructura normativa, el usuario intenta cargar el documento XML al sistema. | |

| **Resultados esperados:** el documento XML es cargado correctamente. |
|---|

**Tabla 22.** Caso de prueba_ funcionalidad cargar esquema XSD en el sistema.

| **Caso de prueba de aceptación** | |
|---|---|
| **Código:** HU14_P2 | **Historia de Usuario:** 14 |
| **Nombre:** Probar_funcionalidad cargar esquema XSD. | |
| **Descripción:** prueba para la funcionalidad de cargar esquema XSD en el sistema. | |
| **Condiciones de Ejecución:** el usuario debe estar autenticado en el sistema.<br><br>Se utilizará un usuario con datos válidos y con permisos para ejecutar esta funcionalidad. | |
| **Pasos de ejecución**: el usuario intenta cargar el esquema XSD al sistema. | |
| **Resultados esperados:** el esquema XSD es cargado correctamente. | |

**Tabla 23.** Caso de prueba_ validar el formato del documento XML con el esquema XSD definido.

| **Caso de prueba de aceptación** | |
|---|---|
| **Código:** HU14_P3 | **Historia de Usuario:** 14 |
| **Nombre:** Probar_validar el formato del documento XML con el esquema XSD definido. | |
| **Descripción:** prueba para la funcionalidad de validar el formato del documento XML con el esquema XSD definido. | |
| **Condiciones de Ejecución:** el usuario debe estar autenticado en el sistema.<br><br>Se utilizará un usuario con datos válidos y con permisos para ejecutar esta funcionalidad. | |
| **Pasos de ejecución**: se carga el documento XML y el esquema XSD al sistema.<br><br>Se realiza la validación del formato del documento XML con el esquema XSD definido. | |
| **Resultados esperados:** se ha validado correctamente el documento XML y se muestra un mensaje si la operación es exitosa. | |

#### 3.9.1.2.1 Resultado de las pruebas de aceptación realizadas

Las pruebas a las funcionalidades se realizaron en tres iteraciones, obteniéndose en la primera 6 no conformidades significativas y 4 no significativas. Para la segunda iteración 3 significativas y 3 no significativas, lográndose en la tercera iteración 0 no conformidades.

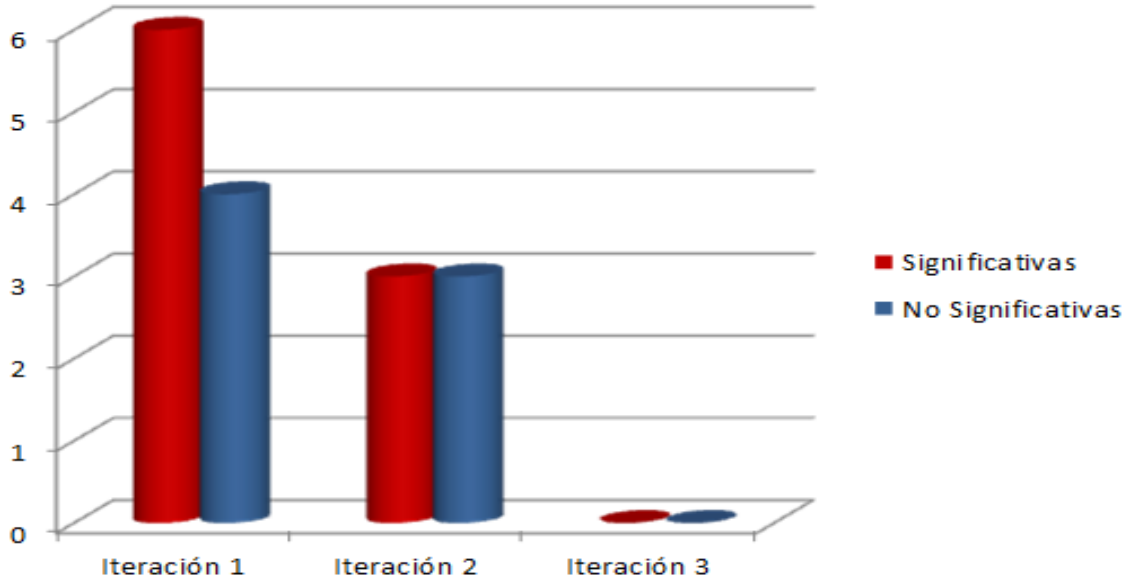



**Figura 20.** No conformidades significativas y no significativas detectadas por iteración.

### 3.9.1.3 Pruebas de usabilidad

Evaluar la usabilidad del sitio permite identificar los elementos y los niveles de compromiso con el cumplimiento de los requisitos de efectividad, eficiencia y satisfacción para un contexto de uso determinado. Las pruebas de usabilidad se realizaron mediante la lista de chequeo[11] relacionando el cumplimiento de aspectos incluidos en áreas o indicadores como se muestran en la figura siguiente. Para el cálculo del por ciento definido se responde una serie de preguntas en función de los aspectos analizados, y se le otorga una evaluación, el mismo se evalúa de 1 en caso de mal y le otorga el valor 0 en caso que el elemento revisado no presente errores, luego se calcula el porciento de preguntas evaluadas de bien en función del total y si el resultado es superior al 70 % se considera que la prueba arrojó resultados positivos.

#### 3.9.1.3.1 Resultado de las pruebas de usabilidad realizadas

Luego de realizar el procedimiento antes mencionado por cada indicador, se obtiene un intervalo de cumplimiento entre el 75% y 100% en cada uno de estos, evidenciándose el resultado esperado. En la figura 21 se muestra lo antes mencionado.

**Lista de chequeo:** lista diseñada por el Laboratorio Industrial de Pruebas de Software perteneciente a la empresa CALISOFT.

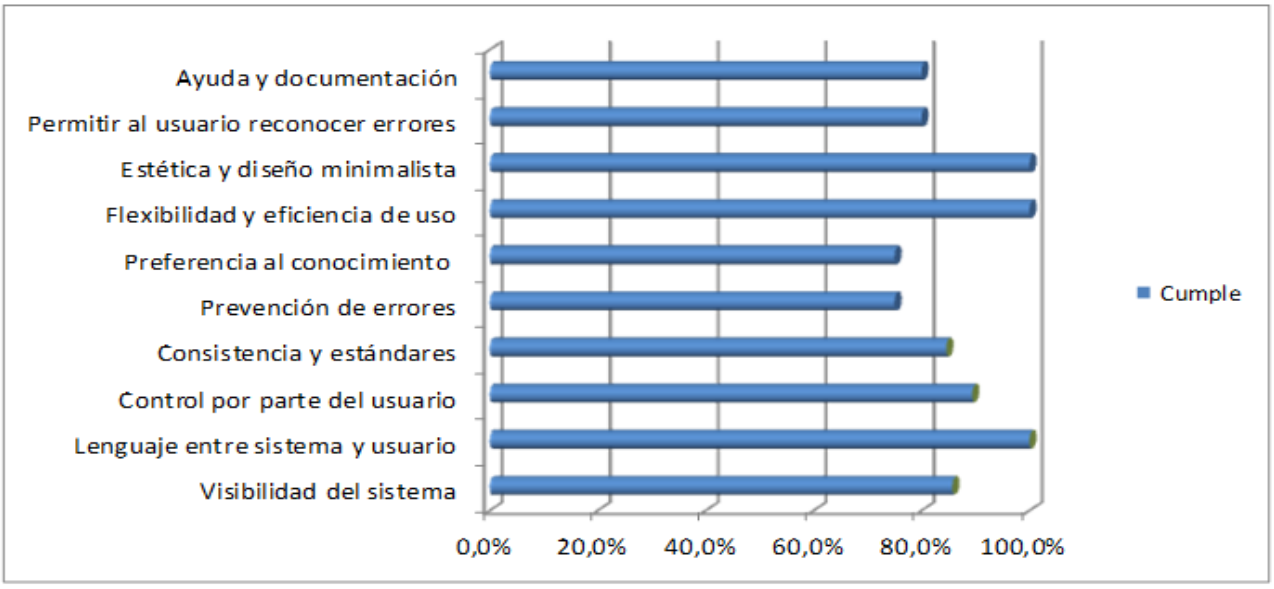



**Figura 21.** Resultados de la prueba de usabilidad.

## 3.1 Resultados de las pruebas realizadas al sistema

Es válido señalar que los resultados alcanzados han sido satisfactorios desde el punto de vista interno y funcional de la solución. Las no conformidades descubiertas fueron debidamente atendidas logrando un correcto comportamiento de la solución desarrollada ante diferentes situaciones. Luego de la tercera iteración la solución quedó libre de no conformidades.

En el Anexo 2 se puede observar el aval de aceptación emitida por la Junta Directiva Nacional y Sociedad Cubana de Derecho e Informática de la Unión de Juristas de Cuba.

## 3.2 Conclusiones del capítulo

- ✓ La utilización del patrón arquitectónico MVC, los patrones de diseño GRAPS y GOF, la representación de la información del sistema a través de una base de datos, así como la desagregación de las HU en tareas de ingenierías sentaron las bases de la arquitectura utilizada para la implementación de la propuesta de solución.

- ✓ La aplicación de las pruebas permitió brindar al cliente conformidad y seguridad ante las funcionalidades del sistema. A través del método de caja blanca y la técnica del camino básico se efectuaron las pruebas de unidad, validando de esta manera el correcto funcionamiento del código. Utilizando el método de caja negra se realizaron las pruebas de aceptación las cuales arrojaron no conformidades que fueron corregidas para garantizar la conformidad del cliente. Un resultado mayor que el 75% demostró la usabilidad del sistema después de realizarse las pruebas de usabilidad según lo planteado en la lista de chequeo diseñada por el Laboratorio Industrial de Pruebas de Software perteneciente a la empresa CALISOFT.

## CONCLUSIONES GENERALES

El desarrollo de la reciente investigación y los resultados generados por la misma, han permitido llegar a las siguientes conclusiones:

- ✓ Al desarrollar el servicio de asistencia a la consulta popular, e integrarlo al nuevo portal web de la Facultad de Derecho de la UH crea una mejora al proceso de consulta popular como ejercicio de participación ciudadana.
- ✓ El estudio de la teoría que sustenta la investigación, la selección de la metodología de desarrollo de software XP, así como las tecnologías, lenguajes y herramientas permitió sentar una base sólida para el desarrollo del sistema.
- ✓ El levantamiento de historias de usuario en conjunto con el cliente, la definición de una modelo conceptual a partir de la desagregación estructural/funcional de la propuesta de solución, la estimación del esfuerzo a necesitar para la implementación de cada HU, así como la definición de una arquitectura para el sistema, permitió el desarrollo evolutivo e incremental de la aplicación.
- ✓ El diseño desarrollado para la aplicación, permitió la implementación de funcionalidades que dieron solución al objetivo general de la investigación. Las pruebas realizadas al sistema demostraron efectividad del empleo de las herramientas, metodología seleccionada y garantizaron la conformidad del cliente.

## RECOMENDACIONES

Al concluir el presente trabajo de diploma, considerando cumplidos los objetivos trazados en el mismo, se recomienda:

- ✓ Realizar el despliegue de la aplicación en varias entidades para comprobar si la solución cumple desde el punto de vista tecnológico con las expectativas del cliente.
- ✓ Proponer al proyecto de rediseño del portal web para la Facultad de Derecho la utilización del protocolo de transferencia de hipertexto HTTPS.

## BIBLIOGRAFÍA REFERENCIADA

1. **Saumell Rodríguez, Mariana**.ELABORACIÓN Y PROCEDIMIE NTO DE UN PROYECTO LEGISLATIVO EN EL SENADO DE LA NACIÓN, nociones básicas, 2009. [*http://www.senado.gov.ar/bundles/senadomicrositios/pdf/ELABORACION%20Y%20PROCEDIMIENTO%20DE%20UN%20PROYECTO%20LEGISLATIVO.pdf]*
2. **Ecured**. Consulta Popular. [En línea] 2014. [Citado el: 20 de marzo del 2014.]. [http://www.ecured.cu/index.php/Consulta]
3. **Téllez Valdez, Julio.** Derecho informático. México, 2006.
4. **Ecured.** Portales. [En línea] 2014. [Citado el: 20 de marzo del 2014.] [http://www.ecured.cu/index.php/Portal_Web]
5. **García Gómez,Juan Carlos** . "Portales de internet: concepto, tipología básica y desarrollo". En: El profesional de la información, 2001, julio-agosto, v. 10, n. 7-8, pp. 4-13. *[http://eprints.rclis.org/14481/1/ELIS-Bahillo.pdf]*
6. **Lázara Ulloa Martínez, Deilys y Brito Dominico, Darianna.** Sistema de gestión de información de los procesos ambientales en el Centro de Inspección y Control Ambiental.
7. **BECK, K.** Planning eXtreme programming. Edtion ed. Boston: Addison-Wesley Professional, 2001.
8. **Ferré, Grau Xavier**. 2011. "Tutorial UML, Desarrollo Orientado a Objetos con UML." Clikear.com, 2011.
9. **Visual Paradigm** International Ltd. Visual-paradigm, 2013. [*http://www.visual-paradigm.com*.]. [*https://netbeans.org/community/releases/74/*]
10. **Wordpress.org.** [En línea] [Citado el: 21 de mayo del 2014.] [*https://es.wordpress.org/*]
11. **Drupal.** [En línea] [Citado el: 21 de mayo del 2014.] [*https://drupal.org*]
12. **Joomla.org.** [En línea] [Citado el: 21 de mayo del 2014.] [*https://joomla.org*]
13. **EcuRed.** Lenguaje de Programación. [En línea] 2014. [Citado el: 6 de junio del 2014.] [*www.ecured.cu/index.php/Lenguaje_de_Programación*].
14. **Eguiluz, Javier**. Introducción a JavaScript, 2014.[*librosweb.es/javascript/]*
15. **Astarita, Emilio** .MANUAL<xhtml/>. [En línea] [Citado el: 12 de junio del 2014.] [*https://manual-xhtml.blogspot.com/]*
16. **Pérez, Iván Nieto**. Diccionario de siglas CSS (Cascading Style Sheets), 2008. [*http://www.elcodigo.net/tutoriales/diccionario.html.*]
17. **CBox**. CodeBox. Glosario php, 2008. [*http://www.codebox.es/glosario.].*
18. [*https://netbeans.org/community/releases/74/*]
19. **Álvarez.** Sistemas Gestores de Base de Datos, 2007.
20. **Apache.** Documentación del Servidor HTTP Apache 2.0, 2013. [*http://httpd.apache.org/docs/2.0/es/.].*

21. **Ron E. Jeffries, Ann Anderson, Chet Hendrickson**. (octubre de 2000). Extreme Programming instalado. Addison-Wesley. ISBN 0201708426.
22. **Díez, Antonio**. IRqA y el desarrollo de proyectos: Experiencias Prácticas. I Jornadas de Ingeniería de Requisitos JIRA 2001. s.n. Sevilla, España, 2001.
23. **Caldero Solís, Manuel**. Una explicación de la programación extrema (XP). Una explicación de la programación extrema (XP). [Online] 2003. [*http://www.apolosoftware.com/.]*
24. **Kruchten, Philippe.** Architectural Blueprints. The "4+1" View Model of Software Architecture. 1998.
25. **2009.** debug_mode=on. [En línea] 2009. [Citado el: 10 de 03 de 2010.] [*http://es.debugmodeon.com/articulo/el-patron-mvc].*
26. **Prieto, Félix.** 2009. Patrones de diseño. 2009.
27. **Sommerville, Ian.** 2005. Ingeniería de Software. Séptima edición. Madrid: Pearson Educación S.A., 2005. 84-7829-074-5.) javaDefinicion.de,2013.[*http://definicion.de/modelo-de-datos/.]*
28. **PRESSMAN, R. S.** Software Engineerinng: A Practitioner's Approach,Seventh Edition Edtion ed. New York: McGraw-Hill Companies,Inc., 2010.
29. **Ortola Rendón, Ariadna y Perez Castellanos, Manuel.** Modelación y construcción de los componentes Gestor de actividades y Calendario para el Subsistema Planificación por Objetivos del Sistema Integral de Gestión de Entidades Cedrux. UCI. Habana, 2011.
30. **ALLENDE, R.** Desarrollo de Portales y Extranet con Plone. Edtion ed., 2006.

## BIBLIOGRAFÍA CONSULTADA

1. **Castro, Raúl.** Instrucción No. 1Del Presidente de Los Consejos de Estado y de Ministros para la Planificación de los Objetivos y Actividades en los Órganos, Organismos de La Administración Central del Estado, Entidades Nacionales y las Administraciones Locales del Poder Popular. Habana, 2012.
2. **Bernal Vidal, Néstor y Galán Ramírez, Yuliet.** Proceso de Planificación por Objetivos en las entidades de las FAR. UCI. Habana, 2008.
3. **Rendón Artola, Ariadna y Castellano Pérez, Manuel A.** Modelación y construcción de los componentes Gestor de actividades y Calendario para el Subsistema Planificación por Objetivos del Sistema Integral de Gestión de Entidades Cedrux.UCI. Habana, 2011.
4. **Féranndez González, Mairelys y Zorrilla Rivera, Osley.** Diseño e implementación del componente Ajuste al Costo del Subsistema Costos y Procesos del Sistema Integral de Gestión de Entidades CEDRUX.UCI.Habana,2010.
5. **Pressman, Roger.** Ingeniería de Software. Un enfoque práctico, 1998.
6. **Ramírez, Jaime.** Unidad de Programación. Métodos de Prueba del Software.
7. **OpenBravo.**OpenBravo,2013 [*http://www.openbravo.com/es.].*
8. **Chaviano Gómez, Enrique y Carrascoso Puebla, Yoan Arlet. 2009.** Propuesta de arquitectura orientada a servicios para el módulo de inventario del ERP cubano. UCI.Habana, 2009.
9. **González Doria, Heidi.** Métricas en el desarrollo del Software. Capitulo 4. Universidad de las Américas Puebla, **2010.**
10. **Machado Scull, Sandy y Galán Ramírez, Yuliet. 2009.** Glosario de términos del Subsistema Dirección por objetivos. UCI.Habana , 2009.
11. **Hernández Ponce, Lidiarys y Montes de Oca Montero, Arnoldo. 2009.** Análisis, diseño e implementación de los componentes para la Gestión de Factores que Influyen en el Plan y Configuración de Perfil para el Sub-sistema de Dirección por Objetivo. UCI.Habana , 2009.
12. **Salazar, Ing. Tte. Leonel.** Curso de Capacitación por Roles: Generalidades de los Patrones de Diseño. s.n. Habana, 2008.
13. **Saavedra Jorge.** El mundo informático, 2013.

[*http://jorgesaavedra.wordpress.com/2006/08/17/patrones-grasp-craig-larman/.].*

14. **Prieto, Félix. 2009.** Patrones de diseño.
15. **Díez, Antonio. 2001.** IRqA y el desarrollo de proyectos: Experiencias Prácticas. I Jornadas de Ingeniería de Requisitos JIRA 2001. s.n. Sevilla, España , 2001.
16. **Sommerville, Ian. 2005.** Ingeniería de Software. Septima edición. Madrid : Pearson Educación S.A., 2005. 84-7829-074-5.
17. **UCID. 2009.** Proceso de desarrollo y gestión de proyectos de software. UCID. Habana , 2009.

18. **Reyna** Rafael. Ingeniería De Software I. *Capitulo I HERRAMIENTAS CASE,2013* [*http://www.scribd.com/doc/3062020/Capitulo-I-HERRAMIENTAS-CASE.].*
19. **Allen Rob**. ZEND FRAMEWORK,2010 [*http://framework.zend.com/manual/en/.*].
20. **Guardado Iván.** Doctrine, 2008. [*http://www.doctrine-project.org.*]
21. **Gómez, Santiago Pavón.** Java Swing, 2011. *[http://jungla.dit.upm.es/~santiago/docencia/apuntes/Swing/index.htm.]*
22. **Ricardo Dario, Bernabeu.2010**. HEFESTO: Metodología para la Construcción de un Data Warehouse.Córdoba, Argentina , 2010.
23. **Jiménez C., W.** Planificación, 1982.
24. **Terry G. y Franklin S.** *Principios de Administración.* México, CECSA, 1987.
25. **Filgueiras Sainz de Rozas, Miriam L.** Planeación Estratégica y Dirección por Objetivos. s.n. Habana , 2010.
26. **Stoner, J.** *Administración.* México: Prentice - Hall Interamericana, 1996.
27. **Gómez Baryolo, Oiner, Morejón Borbón, Yoandry y García Tejo, Darien.** ARQUITECTURA TECNOLÓGICA PARA EL DESARROLLO DE SOFTWARE. UCI. Habana ,2010.
28. **Visconti, Marcelo y Astudillo, Hernán.** Fundamentos de Ingeniería de Software. Departamento de Informática. Universidad Técnica Federico Santa María. Valparaiso.España, 2010.
29. **Caldero Solís, Manuel**. Una explicación de la programación extrema (XP). Una explicación de la programación extrema (XP). [Online] 2003. [*http://www.apolosoftware.com/.]*
30. **Kruchten, Philippe.** Architectural Blueprints. The "4+1" View Model of Software Architecture. 1998.
31. **Borroto Cruz, Dr. Eugenio Radamés y Aneiros R, Ramón.** La Dirección por Objetivos. Escuela Nacional de Salud Pública. La Habana , 2003.
32. **Hamilton Reyna, Yaremis.** Plan de producto.Planificación por Objetivos v1.0. Centro de soluciones de gestión.Departamento de productos. Habana, 2010.
33. **Borroto Cruz, Eugenio Radamés, Castell-Florit serrate, Pastor y Díaz Rojas, Pedro A.** Planeación Estratégica y Dirección por Objetivos. Escuela Nacional de Salud Pública. Habana , 2010.
34. **ALLENDE, R.** Desarrollo de Portales y Extranet con Plone. Edtion ed., 2006.
35. **Rodríguez Pomeda, Jesús.** PLANIFICACIÓN Y CONTROL DE LA EMPRESA: LOS SISTEMAS DE APOYO A LA DIRECCIÓN. IADE-UAM. Madrid , 1996.
36. **Lázara Ulloa Martínez, Deilys y Brito Dominico, Darianna.** Sistema de gestión de información de los procesos ambientales en el Centro de Inspección y Control Ambiental.

# ANEXOS

**Anexo 1.** Análisis de los lineamientos resultantes por etapa.

| **Etapas** | **Primera etapa** | | **Segunda etapa** | | **Tercera etapa** | |
|---|---|---|---|---|---|---|
| **CONCEPTOS** | **Totales** | **%** | **Totales** | **%** | **Totales** | **%** |
| Lineamientos originales | 291 | 100 | 311 | 100 | 311 | 100 |
| Se mantuvieron | 94 | 32 | 165 | 53 | 225 | 72 |
| Modificados o integrados con otros | 197 | 68 | 146 | 47 | 86 | 28 |
| Modificados | 181 | 62 | 146 | 47 | 86 | 28 |
| Lineamientos integrados con otros | 16 | 6 | - | - | - | - |
| Se incorporaron nuevos | 36 | - | - | - | 2 | - |
| **Total de lineamientos resultantes por etapa** | **311** | | **311** | | **313** | |

**Anexo 2.** Aval de aceptación.

Unión Nacional de Juristas de Cuba
Calle 21 esq. a D. Vedado. Plaza de la Revolución.
Ciudad de la Habana. CP 10400. Cuba
Tel: (537) 8329680 //8326209// 8326513-14//8326616
x: (537)8333382 E. mail: unjc@unjc.co.cu //cuba.juristas@yahoo.

AVAL DE ACEPTACIÓN

La Sociedad Cubana de Derecho e Informática declara:

Ante nosotros se ha presentado el proyecto de investigación-desarrollo "Servicio de asistencia al proceso de consulta popular en la Facultad de Derecho de la Universidad de La Habana".

El producto constituye un valioso aporte a la materialización de la línea temática: Participación ciudadana y Toma de Decisiones en el proceso de elaboración jurídica todo lo cual forma parte de los objetivos del proceso de informatización del país y en su integración con similares esfuerzo a nivel regional.

En correspondencia la Junta Directiva Nacional expresa su satisfacción respecto al resultado investigativo que satisface las necesidades del cliente y reconoce el esfuerzo de los alumnos y tutores de la Tesis en cuestión.

Para que así conste, se expide el presente Aval de Aceptación. Dado en La Habana a los 27 días del mes de junio del 2014. "Año 56 de la Revolución".

Junta Directiva Nacional
Sociedad Cubana de Derecho e
Informática de la Unión de Juristas de Cuba

Junta Directiva Nacional
PRESIDENCIA