

**Universidad de las Ciencias Informáticas**

**Facultad 7**



**Trabajo de Diploma para optar por el título de Ingeniero en  
Ciencias Informáticas**

---

**Título: Generación dinámica de formularios basados en  
plantillas XML del sistema de Gestión de la Información del  
Expediente de Proyecto en su variante web**

*Autores: Yelena Martínez Matos*

*José Luis Prieto Labrador*

*Tutora: Ing. Jacqueline Marín Sánchez*

*Co\_Tutor: Ing. Arian Seguí García*

**La Habana, junio 2011**

**“Año 53 de la Revolución”**

### Datos de Contacto

***Tutora: Ing. Jacqueline Marín Sánchez.***

Graduado de Ingeniero en Ciencias Informáticas en la Universidad de las Ciencias Informáticas (UCI) en el curso 2006-2007. Cuenta con la categoría docente de Instructor. Ha impartido las asignaturas: ISW I y II, Proceso de Desarrollo de Software, Formación Pedagógica, Historia y Ética de la Informática, así como Pruebas y Evaluación de Software. Ha cumplido con la función de Asesora de la Calidad durante 4 cursos.

**Correo electrónico:** [jmarin@uci.cu](mailto:jmarin@uci.cu)

***Co\_Tutor: Ing. Arian Seguí García.***

Graduado de Ingeniero en Ciencias Informáticas en la Universidad de las Ciencias Informáticas (UCI) en el curso 2008-2009. Instructor recién graduado. Ha impartido la asignatura de Preparación para la Prueba de Nivel de Programación (PNP). Pertenece actualmente al Grupo de Calidad del CESIM, desempeñando las funciones de Jefe de pruebas.

**Correo electrónico:** [asegú@uci.cu](mailto:asegú@uci.cu)



*A mi padre que a pesar de no estar conmigo físicamente, siempre estuvo y estará en mi corazón apoyándome y dándome fuerzas para seguir adelante.*

*A mi mamá y a mi tía por estar presente en todos los momentos importantes de mi vida.*

*Yelena*

*A mis padres por ser ellos la fuente de inspiración, y mí que hacer de cada día, ser el ejemplo que me ha hecho lo que soy, a mis hermanos por todo su cariño y apoyo.*

*También a mi familia en sentido general y a todos mis amigos y amistades.*

*José Luis*



*A mi madre y mi tía que son lo más sagrado en mi vida, por estar a mi lado en todo momento, por tenerme confianza cuando he tenido que tomar mis propias decisiones.*

*A mi segunda familia Quintana, Irleis, Tato y Cristian por acogerme en su casa y aguantar mis malacrianzas.*

*A mi novio que aunque no esté presente ha sabido apoyarme en todo momento y soportarme en el transcurso de nuestra relación, a él que me dio fuerzas en mis peores momentos.*

*A Mercy que ha sido como mi abuela y me acogió en sus brazos desde que entré a la universidad y siempre me tiene presente en sus oraciones.*

*A mis amigos de la infancia Nieves, Yusmila, Lisandra y Diuver.*

*A mis mejores amistades en el transcurso de mis 6 años en la universidad Liliana, Cary, Lisdany y La Kuka (Yannia).*

*A la Mily, Dianela y el Lore por ser mis compañeros del proyecto.*

*A la gente de la Aldea por acogerme (Yerry, el Prieto, Alberto, Rey y Kuko), que con ellos disfruté mis mejores temporadas en la universidad.*

*A Yurien que más que mi oponente fue mi tutor y amigo a la vez, por aguantar mi carácter, ayudarme en todo lo que le pedía y por ser una persona especial.*

*A mi amigo Cutú y el flaco que siempre estaban ahí cuando me hacía falta cualquier cosa.*

*A todos los que de una forma u otra me apoyaron y me dieron su mano para que terminara mi carrera.*

*Al proyecto Calidad y en general al UCI por darme la oportunidad de graduarme en una Universidad de excelencia.*

*Yelena*



*Agradezco en primer lugar a mis padres, hermanos y los amigos que han sido como hermanos para mí, a ellos mis agradecimientos y las gracias por ser el apoyo incondicional siempre presente sin importar el momento y las circunstancias.*

*A las amistades que hice en la universidad gracias por su ayuda y compañía, Adel, Alexander, Braudis, Jorge Luis, Libisdey, Yanet, Plinio, Leonel, Yarisley, Daimy entre otros que han dedicado su tiempo a compartir momentos alegres.*

*A mi compañera de tesis Yelena que solo ella pudo soportar 10 meses llamándome la atención para obtener buenos resultados, a su dedicación y compromiso que hicieron posible el logro de nuestros objetivos.*

*A mis tíos que han sido como padres y madres para mí en especial a: Julito, Reyna ya mi tía Puchi la que llamamos así cariñosamente los cuales me han dado los consejos de seguir hacia delante sin importar las dificultades y los problemas.*

*Agradezco además a los tutores y al oponente por ser la ayuda en el momento necesario; al tribunal por ser las personas que hicieron posible a través de sus críticas constructivas obtener un trabajo de calidad.*

*José Luis*

### Resumen

La Universidad de las Ciencias Informáticas (UCI) ha puesto en práctica varias estrategias para mejorar y agilizar la gestión de la información que se realiza en los proyectos productivos. Se han implementado mecanismos para desarrollar dicho proceso, es por ello que se confeccionó un Expediente de Proyecto (EP), con el objetivo de documentar el producto y su elaboración. Para mejorar la gestión del EP se realizó una herramienta para la Gestión de la Información del Expediente de Proyecto (GIEP 1.0). Herramienta que fue confeccionada en el Grupo Calidad del Centro de Informática Médica (CESIM); al ser una aplicación de escritorio carga los formularios de manera estática, no permite que la documentación generada por ella sea flexible a cambios en cuanto a su estructura.

El objetivo de este trabajo de diploma es desarrollar un módulo de generación dinámica de formularios basados en plantillas XML en su variante web, para garantizar que cualquier cambio realizado en los documentos del EP sea aceptado y logre guardar toda la información sin temor a pérdidas. Para lograrlo, se hizo uso de las tecnologías .NET, esto facilita el trabajo con la documentación de los proyectos; además garantiza un fácil acceso a la información y la organización de los documentos del EP; también permite un mejor manejo de la información que contienen los EP, trabaja con versiones actuales de dicho expediente y crea formularios de forma dinámica a partir de cualquier plantilla XML del EP.

**Palabras Claves:** Gestión de la Información, Expediente de Proyecto.



## Índice

<b>Datos de Contacto</b>	<b>1</b>
<b>Resumen</b>	<b>1</b>
<b>Introducción</b>	<b>1</b>
<b>Capítulo 1: Fundamentación Teórica</b>	<b>5</b>
<b>1.1 Estado del Arte</b>	<b>5</b>
1.1.1 Ámbito internacional	6
1.1.2 Ámbito a nivel UCI	9
1.1.3 Análisis crítico de las soluciones existentes	10
<b>1.2 Herramientas y Tecnologías</b>	<b>10</b>
1.2.1 Lenguajes de programación	10
1.2.1.2 ¿Por qué usar C#?	12
1.2.2 Lenguajes de Programación Web	12
1.2.2.2 ¿Por qué usar ASP.NET?	14
1.2.3 Framework	15
1.2.4 Entorno de desarrollo integrado (siglas en inglés IDE)	15
1.2.5 Metodologías de Desarrollo de Software	16
1.2.5.1 El Proceso Unificado de Desarrollo de Software (RUP)	17
1.2.5.2 Programación Extrema (siglas en inglés XP)	20
1.2.5.3 ¿Por qué usar RUP?	21
1.2.6 Lenguaje de modelado	22
1.2.7 Herramientas CASE	23
1.2.7.3 ¿Por qué usar Enterprise Architect 7.0 (EA)?	25
<b>Capítulo 2: Características del Sistema</b>	<b>26</b>
<b>2.1 Objeto de Automatización</b>	<b>26</b>
<b>2.2 Propuesta del sistema</b>	<b>26</b>
<b>2.3 Descripción del Modelo del Dominio</b>	<b>26</b>
2.3.1 Glosario de Términos del Dominio	27
<b>2.4 Especificación de Requisitos</b>	<b>28</b>
2.4.1 Requisitos Funcionales	28
2.4.2 Requisitos No Funcionales	29
<b>2.5 Definición de los Casos de Uso (CU)</b>	<b>30</b>
<b>2.6 Diagrama de Casos de Uso del Sistema (CUS)</b>	<b>30</b>
<b>2.7 Definición de los Actores</b>	<b>31</b>
<b>2.8 Descripción de Casos de Uso</b>	<b>31</b>
<b>Capítulo 3: Diseño del Sistema</b>	<b>34</b>
<b>3.1 Flujo de trabajo Análisis y Diseño</b>	<b>34</b>
<b>3.2 Modelo de diseño</b>	<b>34</b>

3.2.1 Descripción de la Arquitectura .....	35
3.2.2 Fundamentación del uso de patrones .....	36
3.2.3 Diagrama de clases .....	37
3.2.4 Diagramas de Secuencia del diseño .....	39
<b>3.3 Descripción de las clases del diseño .....</b>	<b>39</b>
<b>Capítulo 4: Implementación y Pruebas .....</b>	<b>42</b>
<b>4.1 Modelo de implementación .....</b>	<b>42</b>
4.1.1 Diagrama de Componentes .....	42
4.1.2 Diagrama de Despliegue .....	44
4.1.3 Implementación de las principales funcionalidades .....	45
<b>4.2 Pruebas .....</b>	<b>49</b>
4.2.1 Prueba de Caja Negra .....	49
4.2.2 Pruebas de los componentes .....	49
<b>Conclusiones .....</b>	<b>52</b>
<b>Recomendaciones .....</b>	<b>53</b>
<b>Referencias Bibliográficas .....</b>	<b>54</b>
<b>Bibliografía .....</b>	<b>58</b>
<b>Glosario de Términos .....</b>	<b>62</b>



## Introducción

En la actualidad, el desarrollo de las Tecnologías de la Información y las Comunicaciones (TIC) ha cobrado auge. Las empresas informáticas de hoy día se trazan la meta de brindar una respuesta rápida, eficaz y con calidad a los clientes que cada vez son más exigentes, no únicamente en cuanto al precio, sino también en la confiabilidad y seguridad que deben poseer los productos de software. Para lograr con éxito la satisfacción de sus expectativas, estas empresas deben procesar una enorme cantidad de información que permitirá tomar decisiones cada vez más precisas y con mayor rapidez.

La utilización de las TIC para el manejo y almacenamiento masivo de documentos, actualmente representa una necesidad vital para cualquier entidad, beneficia a un número cada vez mayor de usuarios, que van desde las grandes empresas hasta el ciudadano común.

Cuba es un país que se ha dado a la tarea de informatizar la sociedad, abarcando los sectores de la educación, la salud y algunas de las empresas estatales. El desarrollo de software en el país se está convirtiendo en una fuente de desarrollo económico-social.

La Universidad de las Ciencias Informáticas (UCI), es una de las instituciones que se dedica al desarrollo de software de alta calidad. Esta emplea un modelo de formación vinculado a la producción y a la investigación, donde debe tenerse en cuenta todo procedimiento y plantilla que se defina. Todo esto ha impuesto un reto en cuanto a la organización de la producción, con la máxima de que la cantidad no puede afectar la calidad.

Con el objetivo de garantizar el manejo adecuado de la información surge el término Gestión Documental, el cual es un conjunto de normas, técnicas y prácticas, usadas para administrar el flujo de documentos en una organización, permitir la recuperación de información, determinar el tiempo que los documentos deben guardarse, eliminar los que ya no tienen utilidad y asegurar la conservación indefinida de los documentos más valiosos, aplicando principios de racionalización y economía.

Se puede afirmar entonces que “la gestión documental abarca el ciclo de vida completo de los documentos, es decir, el tratamiento secuencial y coherente que se da a los documentos desde que se producen o reciben en las distintas unidades hasta el momento en que son eliminados o conservados, en función de su valor testimonial o histórico como fuente para el conocimiento de la trayectoria de la Universidad.” [1]

Con el objetivo de organizar y guardar la información que se genera en el proceso de desarrollo de software, surge el Expediente de Proyecto (EP); este esquema de expediente y grupo de plantillas tienen como objetivo influir en la estandarización de la documentación y la creación de una cultura de calidad en la organización. Para asegurar la gestión documental de los EP existen diversas herramientas, pero estas trabajan con los documentos ya creados y no con la información de los proyectos en sí.

Para una mejor organización en la producción, la universidad está dirigida por centros de desarrollo, uno de ellos es el Centro de Informática Médica (CESIM), en el cual se desarrolló una herramienta de escritorio para la Gestión de la Información del Expediente de Proyecto (GIEP). Esta aplicación en su versión 1.0 permite el llenado fácil de la información del EP, además de generar documentos finales en formato PDF y garantizar el control de versiones de los mismos.

GIEP 1.0 soluciona la existencia de información redundante, pero los formularios que se cargan son estáticos y no permiten modificaciones, trayendo consigo que la aplicación no permita cambiar la estructura de los documentos del EP, debido a que habría que modificar las interfaces de la aplicación; al no hacer uso de una base de datos centralizada, la información no puede ser accedida desde otras estaciones de trabajo y la aplicación solo funciona con la versión 2.0 del EP.

Además no posee un sistema para la generación dinámica de formularios basados en plantillas XML (estos definen la estructura de los documentos del EP). Se necesita que el sistema posea un mecanismo de razonamiento para que en dependencia de la información contenida en estas plantillas XML la aplicación genere dinámicamente un formulario para una plantilla XML determinada.

La situación problemática antes planteada da origen al siguiente **problema a resolver**: ¿Cómo permitir la generación dinámica de formularios a partir de una plantilla XML del sistema de Gestión de la Información del Expediente de Proyecto en su variante web?

Se define como **objeto de estudio**: el proceso de generación dinámica de formularios, siendo el **campo de acción**: el proceso de generación dinámica de formularios a partir de una plantilla XML del sistema GIEP en su variante web.

Para dar solución al problema se define como **objetivo general**: Desarrollar el módulo para la generación dinámica de formularios basados en plantillas XML del sistema GIEP en su variante web.

## Tareas de la investigación:

- Analizar diferentes sistemas que generen formularios dinámicamente.
- Evaluar las tecnologías y librerías existentes que puedan ser usadas en el trabajo.
- Seleccionar Metodología, lenguaje de programación, plataforma y Entorno Integrado de Desarrollo (IDE) para elaborar la herramienta.
- Realizar la especificación de los requisitos de software.
- Construir los artefactos requeridos en la ingeniería de software del sistema comprendidos en las fases de análisis y diseño.
- Analizar los aspectos comunes entre los módulos que conformarán el sistema para su posterior integración.
- Definir la prioridad de los componentes a implementar en el sistema.
- Elaborar la documentación correspondiente a los Flujos de Trabajo propuestos por la Metodología seleccionada.
- Implementar las funcionalidades requeridas en la aplicación.
- Desarrollar pruebas de funcionalidad al sistema obtenido.

La aplicación contará con los siguientes **Beneficios**:

### Permite:

- Un mejor manejo de la información que contienen los EP.
- Trabajar con las versiones actuales del EP.
- Crear formularios de forma dinámica a partir de cualquier plantilla XML del EP.

El presente documento está estructurado en cuatro capítulos:

**Capítulo 1:** Fundamentación Teórica: En este capítulo se brinda una panorámica sobre la situación problemática y se hace un análisis de las soluciones ya existentes en la nación así como internacionalmente, además de explicar el ambiente de desarrollo empleado para la realización del nuevo sistema informático.

**Capítulo 2:** Características del Sistema: En este capítulo se describen los principales conceptos asociados a los requisitos funcionales y no funcionales, además se presenta una vista gráfica del negocio a través del modelo del dominio.

**Capítulo 3:** Diseño del Sistema: Este capítulo muestra la definición arquitectónica y diseño del sistema. El mismo presenta una vista gráfica del Modelo de Diseño, estructurada por diagramas de clases y de interacción, así como descripción de cada una de las clases.

**Capítulo 4:** Implementación y Pruebas: En este capítulo se plasman los resultados de la implementación, representándose en diagramas de componentes para cada casos de uso, la vista global del sistema, descripción de la estructura interna de la aplicación, así como el códigos de los métodos más relevantes. Se describen además, los resultados arrojados por la realización de pruebas funcionales al sistema.

### Capítulo 1: Fundamentación Teórica

En el presente capítulo se expone el estado del arte donde se analizan diferentes sistemas informáticos existentes en Cuba y en el mundo, similares al que se desea implementar, a partir de la realización de este trabajo, se describen las herramientas y metodologías a utilizar para el desarrollo del mismo.

#### 1.1 Estado del Arte

En la actualidad, coexisten en el mundo los más diversos sistemas de gestión documental: desde el simple registro manual de la correspondencia que entra y sale, hasta los más sofisticados sistemas informáticos que manejan no sólo la documentación administrativa.

Ya sea en papel o en formato electrónico, sino que además controlan los flujos de trabajo del proceso de tramitación de los expedientes, capturan información desde bases de datos de producción, contabilidad y otros, enlazan con el contenido de archivos, bibliotecas, centros de documentación y permiten realizar búsquedas sofisticadas y recuperar información de cualquier lugar. [2]

En la universidad se trabaja diariamente para lograr la estandarización de la documentación que se genera en el proceso de desarrollo de software en los proyectos, con el objetivo de optimizar este proceso en la UCI se diseña el Expediente de Proyecto (EP), el cual ha sido implantado en cada uno de los proyectos productivos vigentes de la universidad.

El EP está compuesto por plantillas; dicho expediente tiene como objetivo la organización de los documentos permitiendo una mayor agilidad y calidad en los productos ofrecidos por la universidad.

La confección del EP es una tarea trabajosa por la cantidad de información que se registra y el tiempo que requiere su elaboración. A partir de estos problemas y de algunas experiencias de los proyectos con el uso del EP, se determina analizar la manera en que se podría agilizar y mejorar este proceso. Para ello se elabora un método estructurado en esquemas XML, que es un lenguaje de esquema utilizado para describir la estructura y las restricciones de los contenidos de los documentos XML de una forma muy precisa, más allá de las normas sintácticas impuestas por el propio lenguaje XML. [3]

# Capítulo 1: Fundamentación Teórica

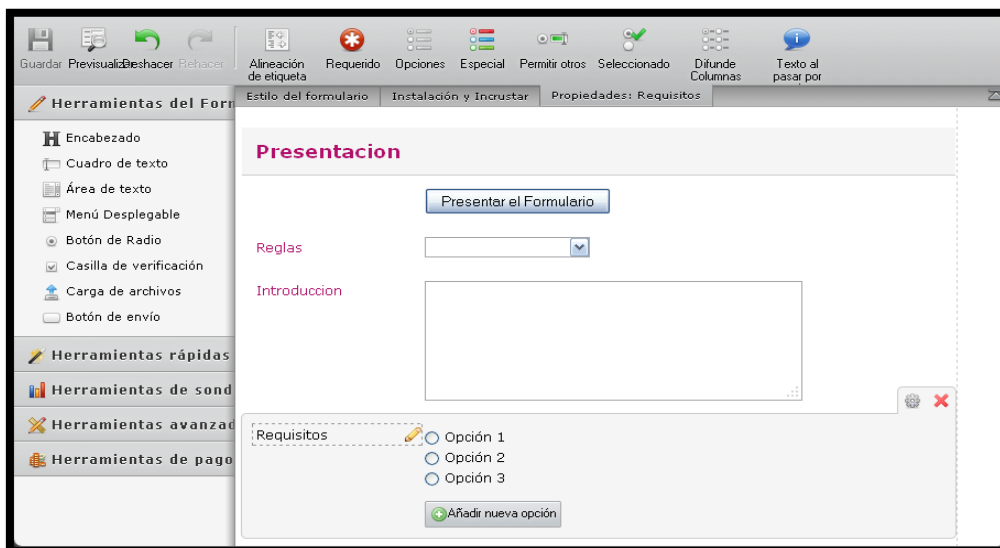
Hoy día se pueden encontrar muchas soluciones de generación de formularios, a continuación se brindan características referentes a la generación y creación de formularios de algunas de estas soluciones existentes.

## 1.1.1 Ámbito internacional

**Jotform:** es una herramienta gratis, que permite crear formularios en línea de una forma bastante sencilla, pero con una eficiencia destacable teniendo en cuenta de que se trata de un servicio gratis. Los formularios que se pueden crear son bastante completos ya que proporciona herramientas básicas y avanzadas dependiendo del tipo de formulario que se desee crear.

Jotform ofrece una interfaz amigable para el usuario, su entorno de trabajo se encuentra en idioma español, se tiene acceso a información de ayuda, guía para usuarios e incluso un foro de discusión. [4] Sin embargo para tener acceso y trabajar con esta herramienta se necesita Internet, no lee ficheros XML, no guarda información y no genera dinámicamente formularios.

A continuación se muestra en la Figura 1 como queda un formulario haciendo uso de la herramienta Jotform.



**Figura 1: Formulario con la herramienta Jotform**

**Microsoft InfoPath:** es una herramienta que permite la creación de formularios dinámicos basados en XML, permite crear etiquetas personalizadas que ofrecen flexibilidad para organizar y presentar información, garantiza que sus usuarios rellenen un formulario trabajando sin conexión, puede ser de gran ayuda para reducir la

## Capítulo 1: Fundamentación Teórica

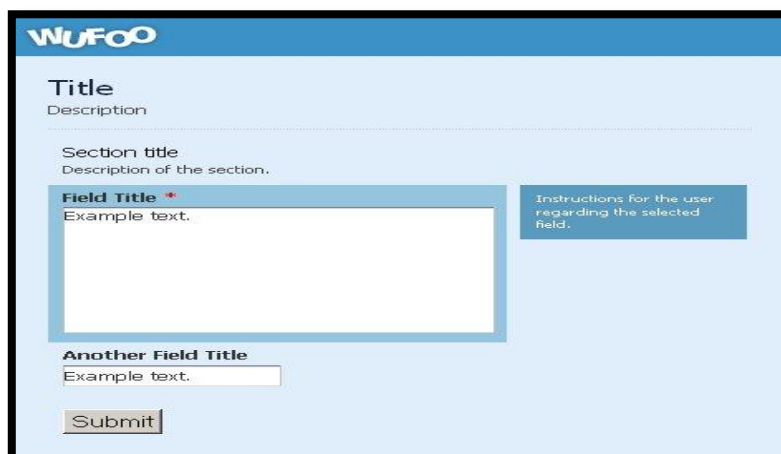
---

información repetida e ineficiente mientras se mejora la colaboración y la toma de decisiones y utiliza esquemas XML existentes como base de un formulario. [5]

Sin embargo la instalación de este producto requiere clave de licencia y medios digitales para su instalación y además no genera dinámicamente formularios basados en una plantilla XML determinada, sino que genera formularios dinámicos.

**Wufoo:** es una aplicación de Internet que ofrece muchos tipos de formularios, como también diferentes diseños de formularios que están organizados por categorías, para descargar de forma totalmente gratuita. Al diseñar un formulario con Wufoo, se crea automáticamente una base de datos necesaria para hacer la recolección y la comprensión de sus datos.

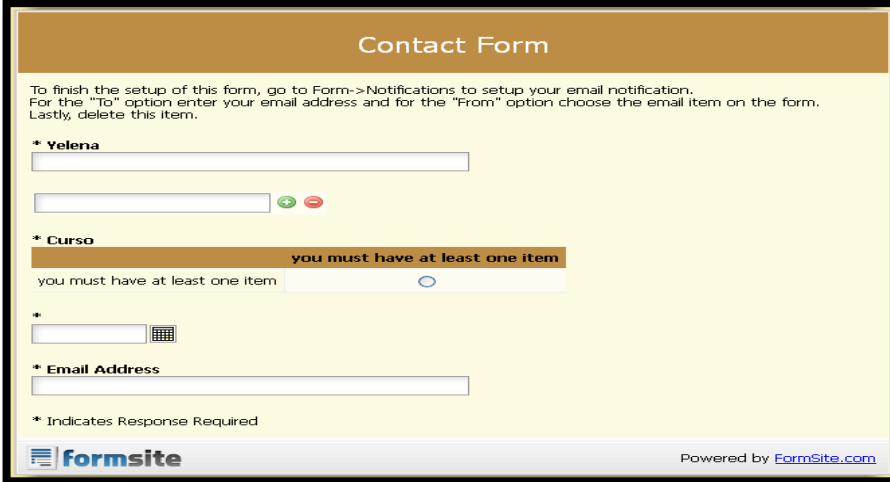
Para trabajar con esta aplicación es necesario conexión a Internet y tener una cuenta en Wufoo, no trabaja con plantillas XML y no posee un sistema para la generación dinámica de formularios. [6]

The image shows a screenshot of a form created using the Wufoo tool. The form has a light blue background and a dark blue header with the 'WUFOO' logo. The form structure includes a 'Title' field with a 'Description' label below it. Below the title is a 'Section title' field with a 'Description of the section.' label. The main body of the form contains a large text input field with a 'Field Title \*' label and 'Example text.' placeholder. To the right of this field is a blue box containing the text 'Instructions for the user regarding the selected field.'. Below the large field is another smaller text input field with the label 'Another Field Title' and 'Example text.' placeholder. At the bottom left of the form is a 'Submit' button.

*Figura 4: Formulario creado por la herramienta Wufoo*

**Formsite:** es una herramienta para generar páginas de formularios en Internet y encuestas. Básicamente permite empezar desde cero o usar algunos de los modelos ya predefinidos y con solo arrastrar los componentes de los diferentes menús que aparecen en la página online de la herramienta el usuario diseña el formulario que desee. Sin embargo no cuenta con una interfaz en el idioma español, para trabajar con dicha herramienta se necesita conexión a Internet, no genera de forma dinámica los formularios y no guarda información.

A continuación en la Figura 3 se muestra como queda un formulario haciendo uso de la herramienta Formsite. [7]



The image shows a screenshot of a web form titled "Contact Form". At the top, there is a brown header with the title. Below the header, there is a paragraph of instructions: "To finish the setup of this form, go to Form->Notifications to setup your email notification. For the 'To' option enter your email address and for the 'From' option choose the email item on the form. Lastly, delete this item." The form contains several fields: a text field labeled "\* Yelena", a text field with a green plus icon and a red minus icon, a dropdown menu labeled "\* Curso" with a warning message "you must have at least one item" and a radio button, a text field with a calendar icon, and a text field labeled "\* Email Address". At the bottom left is the "formsite" logo and at the bottom right is the text "Powered by FormSite.com".

*Figura 2: Formulario creado por la herramienta Formsite*

**Microsoft Access:** es una herramienta que permite crear formularios que contengan un formato complicado, imágenes, casillas de verificación, listas desplegadas o áreas de texto que tengan tipos de datos, almacena la información contenida en el formulario, esto lo realiza mediante una base de datos; crea una aplicación Web sencilla para mostrar o editar los datos de una base de datos ubicada en la intranet.

Sin embargo se utiliza principalmente para el almacenamiento, la recuperación y la elaboración de informes de los datos, realiza la generación de formularios a través de las tablas creadas por el usuario y no de una plantilla XML determinada y además se necesita la licencia del producto. [8]

A continuación se muestra en la Figura 4 como queda un formulario creado por la herramienta Access.



Id	Organización	Nombre	Apellido	Dirección de corr
1	Organización A	Anna	Bedecs	
2	Organización B	Antonio	Gratacos Solso	
3	Organización C	Thomas	Axen	
4	Organización D	Christina	Lee	
5	Organización E	Martin	O'Donnell	
6	Organización F	Francisco	Pérez-Olaeta	
7	Organización G	Ming-Yang	Xie	
8	Organización H	Elizabeth	Andersen	

Figura 4: Formulario creado por la herramienta Microsoft Access

## 1.1.2 Ámbito a nivel UCI

En la Universidad de las Ciencias Informáticas se cuenta con un sistema informático que permite manejar la información de los documentos del EP:

Una herramienta de escritorio para la **Gestión de la Información en el Expediente de Proyecto** (GIEP 1.0)

Esta herramienta es de fácil uso y posee un tiempo de respuesta de las acciones acorde con los requisitos de hardware especificados. La estructuración de la aplicación en pequeños módulos, permite que se puedan realizar cambios en el código fuente sin afectar el resto de los componentes o en caso de afectación, esta sería mínima; facilitando la inclusión de mejoras a la aplicación para futuras iteraciones.

Sin embargo no define las funcionalidades que permitan la integración con otras herramientas como Redmine y el Alfresco, además la retroalimentación de información contenida en estas y no permite que varios usuarios actualicen el Expediente de Proyecto en tiempo real, es decir que los cambios realizados por un usuario son guardados en la computadora donde el mismo esté trabajando, por lo que los cambios no pueden ser visibles por otros usuarios que se encuentren trabajando en ese expediente. Además no permite la generación dinámica de formularios.

## **Generador Automático de Plantillas en la Web**

Durante el curso 2009 – 2010 en la Facultad 1 de la UCI se desarrolló un Generador automático de plantillas en la web, que genera plantillas a través de la web, logrando que los usuarios creen los formularios deseados, para poder recoger y consultar la información necesaria de una actividad. El sistema automatiza la creación de plantillas de datos a través de la web para que cualquier usuario pueda crear sus formularios para la recogida de los datos que considere necesarios, brinda la posibilidad de modificarlos y eliminarlos, si se desea, también automatiza la disponibilidad de cada plantilla creada para cualquier usuario que se interese en llenar sus datos, así como el almacenamiento de los datos de los interesados. [9]

### **1.1.3 Análisis crítico de las soluciones existentes**

Una vez analizadas las herramientas antes descritas, se llega a la conclusión que las mismas no son factibles para la realización de este trabajo, debido a que no permiten la generación dinámica de formularios, no hacen uso de plantillas XML, no permiten realizar modificaciones en la estructura de los documentos creados y no responden a las necesidades existentes en los proyectos productivos de la universidad para gestionar y trabajar con la información de los EP.

Debido a ello se realizó una herramienta que es capaz de solucionar dichos problemas, siendo necesario para esto un estudio de las tecnologías y herramientas para la elaboración de la misma.

## **1.2 Herramientas y Tecnologías**

### **1.2.1 Lenguajes de programación**

En el mundo del desarrollo del software muchos programadores usan lenguajes de programación de alto nivel y orientados a objetos; la decisión del uso de uno u otro para el desarrollo de un software determinado está en las librerías que estos utilizan, en las cuales radica la verdadera riqueza del lenguaje. La elección final de uno u otro lenguaje también dependerá del posible conocimiento que se tenga de la sintaxis del mismo y de su adaptación al medio para el que se quiere programar, ya que es diferente programar para red local, para Windows, Mac o Linux. A continuación se presentarán algunas características, ventajas y desventajas de los principales

lenguajes de programación orientados a objetos que más se usan en la actualidad. [10]

### 1.2.1.1 Lenguaje C++

C++ es un lenguaje imperativo orientado a objetos, es un súper conjunto de C, que nació para añadirle cualidades y características de las que carecía. El resultado es que como su ancestro, sigue muy ligado al hardware subyacente, manteniendo una considerable potencia para programación a bajo nivel, pero se le han añadido elementos que le permiten también un estilo de programación con alto nivel de abstracción.

C++ no es un lenguaje orientado a objetos puro (en el sentido en que puede serlo Java por ejemplo), se trata simplemente del sucesor de un lenguaje de programación hecho por programadores (de alto nivel) para programadores, al cual le han ido añadiendo todos los elementos que la práctica aconsejaba como necesarios. [11]

### 1.2.1.2 El lenguaje C#

Actualmente se está utilizando con gran efectividad el nuevo lenguaje C# desarrollado por la empresa Microsoft Corporation, como una recopilación de lo mejor de C++ y Java. Es un lenguaje orientado a objetos, tiene una sintaxis muy parecida a Java y posee la potencia de C++. Tiene algunas ventajas sobre los restantes lenguajes de alto nivel orientados a objetos. El estándar del lenguaje C# por excelencia está comprendido en la especificación ECMA-334 (Estándar internacional que especifica la representación y semántica de programas escritos en C# así como la sintaxis y restricciones de este lenguaje) de la ECMA International (Organización internacional que se basa en membresías de estándares para la comunicación y la información). [12]

#### 1.2.1.2.1 Comparación de C# con C++

A continuación se expone una comparación del C# frente a otros lenguajes, viendo sus principales características y ventajas frente al resto y así algunas desventajas que presentan otros lenguajes con respecto a él que pudiesen atrasar el desarrollo de la aplicación que se desea.

## Ventajas frente a C/C++

- ✓ Compila a código intermedio independiente del lenguaje en que haya sido escrita la aplicación e independiente de la máquina donde vaya a ejecutarse.
- ✓ Realiza la recolección automática de basura.
- ✓ Elimina el uso de punteros, en C# no son necesarios aunque permite utilizarlos.
- ✓ Posee capacidades de reflexión.
- ✓ No hay que preocuparse por archivos de cabecera ".h".
- ✓ Es flexible en cuanto al orden de definición de las clases y las funciones.
- ✓ No hay necesidad de declarar funciones y clases antes de invocarlas.
- ✓ No existen las dependencias circulares. Soporta definición de clases dentro de otras.
- ✓ No existen funciones ni variables globales, todo pertenece a una clase. [12]

### 1.2.1.2 ¿Por qué usar C#?

Finalmente se ha comprobado que el lenguaje C++ es muy potente y estable pero complicado para algunos propósitos. Java es muy similar al C# y su sintaxis es amigable pero eventualmente es menos eficiente en la ejecución de programas de escritorio frente a los desarrollados en C#. Además, este último aporta características novedosas que simplifican grandemente las labores de implementación.

Se enriquece con la potencia de C++ y la sencillez y modernidad de Java pero mejorando lo que debe ser mejorado. Ha alcanzado gran auge y se utiliza para desarrollar gran cantidad de aplicaciones desde mediano a gran tamaño, obviando esfuerzo de programadores en actividades de rutina. Está debidamente estandarizado. Se usa en múltiples grupos de proyecto en la Facultad 7 de la UCI. Es gratis tanto el C# como la plataforma que lo soporta. Resulta, por tanto, el adecuado para las aspiraciones y necesidades de este trabajo. [12]

### 1.2.2 Lenguajes de Programación Web

Son los lenguajes más utilizados para el desarrollo de los servicios, de los motores de búsqueda, de las páginas dinámicas para la interfaz y para la recuperación de los materiales de la colección digital. A continuación se da una breve reseña de los lenguajes de programación web que más se utilizan en la actualidad.

### 1.2.2.1 ASP

ASP es la tecnología pionera en las aplicaciones Web que se ejecutan en el servidor. Desarrollada por Microsoft y optimizada para su ejecución en servidores Windows con tecnología NT bajo IIS, aunque también hay opciones para Windows 98 con el Personal Web Server y para Linux con Chilisoft.

Al ser una tecnología propietaria, no tiene la enorme cantidad de módulos extra que sí tiene PHP, aunque abriendo objetos COM trabaja fácilmente con archivos dll.

Usa Visual Basic Script, lo que para los desarrolladores Visual Basic era una ventaja pues no tenían que aprender otro lenguaje, para otros es una desventaja, pues consideran que no es más que un parche Web de un lenguaje ya existente, y que no fue creado expresamente para los servidores Web, como sí pasa con PHP. También soporta el lenguaje JavaScript (JavaScript de Microsoft). Se comunica con las bases de datos vía Conectividad abierta de Base de Datos (siglas en inglés ODBC), y la comunicación con SQL Server es óptima. [13]

### 1.2.2.2 ASP.NET

ASP.NET es un modelo de desarrollo Web que incluye los servicios necesarios para construir aplicaciones Web con la menor cantidad de código. Es parte de la plataforma .NET, luego las aplicaciones que se implementen sobre ASP.NET pueden estar codificadas en cualquier lenguaje compatible con el Common Language Runtime (CLR) de .NET (Visual Basic, C#, JScript .NET y J#).

Una de las principales ventajas de ASP.NET es la gran cantidad de lenguajes que soporta: VB.NET (que deriva del Visual Basic); C# (una versión mejorada de C++) y JScript.NET (que deriva de JavaScript). ASP.NET constituye un entorno abierto en el que se puede combinar código HTML, scripts y componentes ActiveX del servidor para crear soluciones dinámicas y de calidad para la Web. El Proyecto Mono lo ha llevado a Linux, Solaris, Mac y Unix. Las páginas que utilizan esta tecnología tienen la extensión “.asp”. [14]

### 1.2.2.3 XML

Es un Lenguaje de Etiquetado Extensible muy simple, pero estricto que juega un papel fundamental en el intercambio de una gran variedad de datos. Es un lenguaje muy similar a HTML pero su función principal es describir datos y no mostrarlos como es el

caso de HTML. XML es un formato que permite la lectura de datos a través de diferentes aplicaciones y permite estructurar, almacenar e intercambiar información. [15]

### 1.2.2.4 JavaScript

JavaScript es un lenguaje de programación que se utiliza principalmente para crear páginas web dinámicas. Una página web dinámica es aquella que incorpora efectos como texto que aparece y desaparece, animaciones, acciones que se activan al pulsar botones y ventanas con mensajes de aviso al usuario. Técnicamente, es un lenguaje de programación interpretado, por lo que no es necesario compilar los programas para ejecutarlos. Los programas escritos con JavaScript se pueden probar directamente en cualquier navegador sin necesidad de procesos intermedios. Es un lenguaje de programación bastante sencillo y pensado para hacer las cosas con rapidez, a veces con ligereza. [16]

### 1.2.2.2 ¿Por qué usar ASP.NET?

ASP.NET presenta varias ventajas sobre los lenguajes PHP, ASP y JavaScript (JSP) las cuales se enumeran a continuación:

1. Caché: se puede almacenar en la caché del servidor tanto páginas enteras, como controles personalizados o simples variables. En páginas críticas con mucha carga de base de datos es muy útil almacenar datos de la base de datos en la caché, reduciendo enormemente el consumo de recursos.
2. Carpetas especializadas, como por ejemplo `app_code` que compila automáticamente las clases que se alojan en él, o la carpeta `app_theme` que alojan ficheros que marcan los temas de estilos de la Web.
3. Los archivos de configuración `Web.config` y `Machine.config` permiten realizar operación de configuración en ficheros que hasta ahora había que realizar en el servidor.
4. La extraordinaria compatibilidad con esquemas XML y los servicios Web.
5. La multitud de controles Web que permiten mucha funcionalidad con poco código. Desde enlace con las bases de datos o enseñar fácilmente todos los datos, hasta simples etiquetas, hiperenlaces o generadores de imágenes. [17]

## 1.2.3 Framework

Un framework es un término muy utilizado últimamente en el campo de la informática, se utiliza para referirse a un conjunto de bibliotecas, que se utilizan para implementar la estructura de un modelo para una aplicación. Esto se realiza con el objetivo de promover la reutilización de código, posibilitando que no sea necesario perder tiempo en reinventar la rueda. Existen diferentes tipos de framework, para diferentes propósitos, algunos orientados al desarrollo de aplicaciones web, o para un determinado sistema operativo o lenguaje. En un sentido muy amplio el framework que se utiliza determina la arquitectura del software. [18]

### 1.2.3.1 Framework .NET 3.5

Microsoft .NET Framework 3.5 contiene muchas y nuevas características que se agregan de forma incremental a .NET Framework 2.0 y 3.0 e incluye .NET Framework 2.0 Service Pack 1 y .NET Framework 3.0 Service Pack 1.

.NET Compact Framework versión 3.5 amplía la compatibilidad con aplicaciones móviles distribuidas al incorporar la tecnología Windows Communication Foundation (WCF). También agrega nuevas características de lenguaje como LINQ, incluye nuevas Application Programming Interface (API) basadas en los comentarios de la comunidad y mejora la depuración con herramientas y características de diagnóstico actualizadas.

.NET Framework 3.5 incorpora características mejoradas en áreas concretas de ASP.NET y Visual Web Developer. El avance más significativo es la mejora de la compatibilidad con el desarrollo de sitios web habilitados para AJAX. ASP.NET agrega compatibilidad con el desarrollo de AJAX centrado en el servidor mediante un conjunto de nuevos controles de servidor y nuevas API. Puede habilitar una página ASP.NET 2.0 existente en AJAX agregando un control ScriptManager y un control UpdatePanel, de modo que la página pueda actualizarse sin que sea necesario realizar una actualización de la página completa. [19]

### 1.2.4 Entorno de desarrollo integrado (siglas en inglés IDE)

En el mercado existen aplicaciones informáticas, llamadas Entornos Integrados de Desarrollo, que incluyen a todos los programas necesarios para realizar todas las fases de puesta a punto de un programa.

Además, un IDE suele proporcionar otras herramientas de software muy útiles para los programadores, tales como: depuradores de código, ayuda en línea de uso del lenguaje, etc. Todo ello, con el fin de ayudar y facilitar el trabajo al programador.

### 1.2.4.1 Visual Studio 2008

Visual Studio 2008 permite incorporar características del nuevo Windows Presentation Foundation sin dificultad tanto en los formularios de Windows existentes como en los nuevos. Ahora es posible actualizar el estilo visual de las aplicaciones al de Windows Vista debido a las mejoras en Microsoft Foundation Class Library (MFC) y Visual C++. Visual Studio 2008 permite mejorar la interoperabilidad entre código nativo y código manejado por .NET. Esta integración más profunda simplificará el trabajo de diseño y codificación. [20]

Visual Studio 2008 ahora permite la creación de soluciones multiplataforma adaptadas para funcionar con las diferentes versiones de .Net Framework: 2.0. (Incluido con Visual Studio 2005), 3.0 (incluido en Windows Vista) y 3.5 (incluido con Visual Studio 2008). [21]

Visual Studio 2008 brinda ventajas al desarrollador en 3 pilares fundamentales:

- ✓ Mejor productividad del Desarrollador.
- ✓ Administración del ciclo de vida de las aplicaciones.
- ✓ Desarrollo sobre últimas tecnologías.

### 1.2.4.2 MonoDevelop 2.2

Es un IDE diseñado para que los desarrolladores de .NET migren sus aplicaciones creadas en Visual Studio a Linux de forma rápida y que puedan mantener un único código base para ambas plataformas.

### 1.2.5 Metodologías de Desarrollo de Software

El término de metodología se define como un conjunto de métodos eficientes, orientados a conseguir un objetivo propuesto. Son un conjunto de procesos que organizados dan una secuencia de pasos a seguir para obtener los hitos propuestos y finalmente el producto final. Como no existe una metodología de software universal, esta debe guiar a la organización en el desarrollo de sus objetivos. Por ello, la organización, su estructura, canales de información, recursos humanos y físicos,



deben ser los adecuados y garantizar el correcto funcionamiento de los procesos y métodos definidos. [22]

### 1.2.5.1 El Proceso Unificado de Desarrollo de Software (RUP)

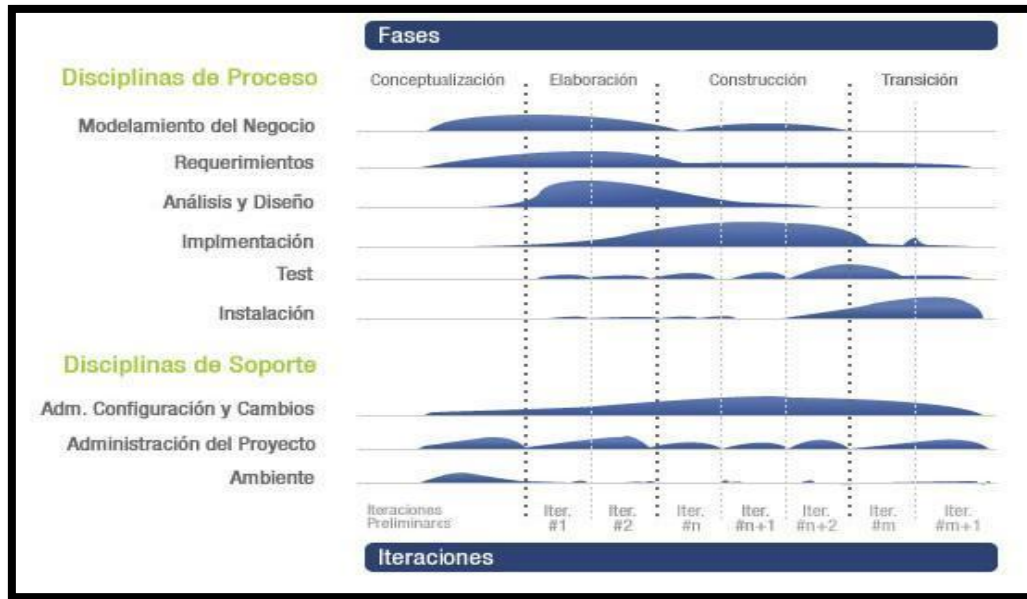
Un proceso define quién está haciendo qué, cuándo y cómo para alcanzar un determinado objetivo. Un proceso de desarrollo de software es un conjunto de actividades que guían los esfuerzos de las personas implicadas en el proyecto, a modo de plantilla que explica los pasos necesarios para terminar el proyecto. Este conjunto de actividades, tiene la misión de transformar los requerimientos del usuario en un producto software. Como RUP es un proceso, en su modelación define como sus principales elementos:

- ✓ *Trabajadores (Quién)*: Define el comportamiento y responsabilidades de un individuo, grupo de individuos, sistema automatizado o máquina, los cuales se encargan de realizar las actividades y son los responsables de una serie de artefactos.
- ✓ *Actividades (Cómo)*: Es una tarea que tiene un propósito claro, es realizada por un trabajador y manipula elementos.
- ✓ *Artefactos (Qué)*: Productos tangibles del proyecto que son producidos, modificados y usados por las actividades.
- ✓ *Flujo de actividades (Cuándo)*: Secuencia de actividades realizadas por trabajadores y que produce un resultado de valor observable.

RUP divide su ciclo de vida en cuatro fases dentro de las cuales se llevan a cabo un grupo de iteraciones.

- ✓ *Conceptualización (Concepción o Inicio)*: Se describe el negocio y se delimita el proyecto describiendo sus alcances con la identificación de los casos de uso del sistema.
- ✓ *Elaboración*: Se define la arquitectura del sistema y se obtiene una aplicación ejecutable que responde a los casos de uso que la comprometen.
- ✓ *Construcción*: Se obtiene un producto listo para su utilización que está documentado y tiene un manual de usuario.
- ✓ *Transición*: El release ya está listo para su instalación en las condiciones reales. Puede implicar reparación de errores.

- Flujos de Trabajo.
  - ✓ *Modelamiento del negocio*: Describe los procesos de negocio, identificando quiénes participan y las actividades que requieren automatización.
  - ✓ *Requerimientos*: Define qué es lo que el sistema debe hacer, para lo cual se identifican las funcionalidades requeridas y las restricciones que se imponen.
  - ✓ *Análisis y diseño*: Describe cómo el sistema será realizado a partir de las funcionalidades previstas y las restricciones impuestas, por lo que indica con precisión lo que se debe programar.
  - ✓ *Implementación*: Define cómo se organizan las clases y objetos en componentes, cuáles nodos se utilizarán y la ubicación en ellos de los componentes y la estructura de capas de la aplicación.
  - ✓ *Prueba (Testeo)*: Busca identificar los defectos y corregirlos antes de la instalación final del sistema, se verifica la integración apropiada de los componentes y que se satisfacen los requerimientos de los clientes.
  - ✓ *Instalación (Despliegue)*: Produce release del producto y realiza actividades (empaquete, instalación, asistencia a usuarios, y otras) para entregar el software a los usuarios finales.
  - ✓ *Administración del proyecto*: Involucra actividades con las que se busca producir un producto que satisfaga las necesidades de los clientes.
  - ✓ *Administración de configuración y cambios*: Describe cómo controlar los elementos producidos por todos los integrantes del equipo de proyecto.
  - ✓ *Ambiente*: Contiene actividades que describen los procesos y herramientas que soportarán el equipo de trabajo del proyecto; así como el procedimiento para implementar el proceso en una organización.



*Ilustración 1: Fases e Iteraciones de la Metodología RUP*

- Características de RUP.
  - ✓ Creado por Jacobson, Rumbaugh y Booch.
  - ✓ Unifica los mejores elementos de metodologías anteriores.
  - ✓ Preparado para desarrollar grandes y complejos proyectos.
  - ✓ Orientado a objetos.
  - ✓ Utiliza UML como lenguaje de representación visual.

El ciclo de vida de RUP se caracteriza por ser: dirigido por casos de uso, centrado en la arquitectura y además por ser iterativo e incremental.

Los casos de uso reflejan lo que los usuarios futuros necesitan y desean, lo cual se capta cuando se modela el negocio y se representa a través de los requerimientos. A partir de aquí los casos de uso guían el proceso de desarrollo, obteniéndose un producto final que cuenta con la calidad requerida por el cliente.

La arquitectura muestra la visión común del sistema completo, que describe los elementos del modelo que son importantes para la construcción del software, creándose de esta manera los cimientos del sistema que son necesarios como base para comprenderlo, desarrollarlo y producirlo. RUP propone que se desarrolle el software mediante iteraciones, comenzando por los casos de uso relevantes desde el punto de vista para la arquitectura y además que cada fase se desarrolle en iteraciones, donde en cada iteración se involucran actividades de todos los flujos de trabajo, aunque se desarrollan fundamentalmente algunos flujos más que otros. [23]

## 1.2.5.2 Programación Extrema (siglas en inglés XP)

Es una metodología ágil centrada en potenciar las relaciones interpersonales como clave para el éxito en desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores, y propiciando un buen clima de trabajo. XP se basa en realimentación continua entre el cliente y el equipo de desarrollo, comunicación fluida entre todos los participantes, simplicidad en las soluciones implementadas y coraje para enfrentar los cambios. XP se define como especialmente adecuada para proyectos con requisitos imprecisos y muy cambiantes, y donde existe un alto riesgo técnico. Los principios y prácticas son de sentido común pero llevadas al extremo, de ahí proviene su nombre.

- Fases de XP.

✓ *Fase I: Exploración.*

En esta fase, los clientes plantean a grandes rasgos las historias de usuario 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. Se prueba la tecnología y se exploran las posibilidades de la arquitectura del sistema construyendo un prototipo. La fase de exploración toma de pocas semanas a pocos meses, dependiendo del tamaño y familiaridad que tengan los programadores con la tecnología.

✓ *Fase II: 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. Una entrega debería obtenerse en no más de tres meses. Esta fase dura unos pocos días.

✓ *Fase III: 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. Esto se logra escogiendo las historias que fueren la creación de esta arquitectura, sin embargo, esto no siempre es posible ya

que es el cliente quien decide qué historias se implementarán en cada iteración (para maximizar el valor de negocio). Al final de la última iteración el sistema estará listo para entrar en producción.

✓ *Fase IV: Producción.*

La fase de producción requiere de pruebas adicionales y revisiones de rendimiento antes de que el sistema sea trasladado al entorno del cliente. Al mismo tiempo, se deben tomar decisiones sobre la inclusión de nuevas características a la versión actual, debido a cambios durante esta fase. Es posible que se rebaje el tiempo que toma cada iteración, de tres a una semana. Las ideas que han sido propuestas y las sugerencias son documentadas para su posterior implementación (por ejemplo, durante la fase de mantenimiento).

✓ *Fase V: Mantenimiento.*

Mientras la primera versión se encuentra en producción, el proyecto XP debe mantener el sistema en funcionamiento al mismo tiempo que desarrolla nuevas iteraciones. Para realizar esto se requiere de tareas de soporte para el cliente. De esta forma, la velocidad de desarrollo puede bajar después de la puesta del sistema en producción. La fase de mantenimiento puede requerir nuevo personal dentro del equipo y cambios en su estructura.

✓ *Fase VI: Muerte del Proyecto.*

Es cuando el cliente no tiene más historias para ser incluidas en el sistema. Esto requiere que se satisfagan las necesidades del cliente en otros aspectos como rendimiento y confiabilidad del sistema. Se genera la documentación final del sistema y no se realizan más cambios en la arquitectura. La muerte del proyecto también ocurre cuando el sistema no genera los beneficios esperados por el cliente o cuando no hay presupuesto para mantenerlo. [24]

### 1.2.5.3 ¿Por qué usar RUP?

RUP es la más utilizada para el diseño, implementación y documentación de sistemas informáticos. Es altamente configurable, ya que permite construir solamente los artefactos que se necesiten para el desarrollo de un producto. Además de traer consigo un grupo de beneficios como la estandarización, facilita el entendimiento por parte de los usuarios del software, que no necesariamente deben tener conocimientos

previos sobre el tema y facilita además, el desarrollo del producto a distancia de los clientes.

## 1.2.6 Lenguaje de modelado

Un lenguaje de modelado de objetos se puede definir como el conjunto estandarizado de símbolos y de modos de disponerlos para modelar (parte de) un diseño de software orientado a objetos.

### 1.2.6.1 Lenguaje Unificado de Modelado 2.0 (UML)

Lenguaje Unificado de Modelado (UML), es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad. Es un lenguaje gráfico para visualizar, especificar, construir y documentar los artefactos que se crean durante el proceso de un desarrollo de software.

Se utiliza para definir un sistema de software, para detallar los artefactos en el sistema y para documentar y construir. Se puede aplicar en una gran variedad de formas para dar soporte a una metodología de desarrollo de software (como el Proceso Unificado Racional), pero no especifica en sí mismo qué metodología o proceso usar.

Intenta solucionar el problema de propiedad de código que se da con los desarrolladores, al implementar un lenguaje de modelado, y se crea una documentación también común, que cualquier desarrollador con conocimientos de UML será capaz de entender, independientemente del lenguaje utilizado para el desarrollo.

Permite la modificación de todos sus miembros mediante estereotipos y restricciones. Su utilización es independiente del lenguaje de programación y de las características de los proyectos, ya que ha sido diseñado para modelar cualquier tipo de proyectos, tanto informáticos como de arquitectura, o de cualquier otra rama. Está consolidado como el lenguaje estándar en el análisis y diseño de sistemas de cómputo.

- Objetivos de UML.
- ✓ *Visualizar*: UML permite expresar de una forma grafica un sistema de forma que otro lo puede entender.
- ✓ *Especificar*: UML permite especificar cuáles son las características de un sistema antes de su construcción.

- ✓ *Construir*: A partir de los modelos especificados se pueden construir los sistemas diseñados.
- ✓ *Documentar*: los propios elementos gráficos sirven como documentación del sistema desarrollado que pueden servir para su futura revisión.

Un modelo UML está compuesto por tres clases de bloques de construcción:

- ✓ *Elementos*: los elementos son abstracciones de cosas reales o ficticias (objetos, acciones, entre otros).
  - ✓ *Relaciones*: relacionan los elementos entre sí.
  - ✓ *Diagramas*: son colecciones de elementos con sus relaciones.
- Características de UML.
    - ✓ Es un lenguaje de representación visual.
    - ✓ Permite combinar diversos elementos gráficos y crear diagramas.
    - ✓ Se usa solo para modelar sistemas que usan tecnología orientada a objetos.
- [25]

### 1.2.7 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. [26]

#### 1.2.7.1 Enterprise Architect 7.0 (EA)

Enterprise Architect es una herramienta CASE para el diseño y construcción de sistemas de software. EA soporta la especificación de UML 2.0, que describe un lenguaje visual por el cual se pueden definir mapas o modelos de un proyecto. EA es una herramienta progresiva que cubre todos los aspectos del ciclo de desarrollo, proporcionando una trazabilidad completa desde la fase inicial del diseño a través del

despliegue y mantenimiento. También provee soporte para pruebas, mantenimiento y control de cambio. Algunas de las características claves de Enterprise Architect son:

- ✓ Crear elementos del modelo UML para un amplio alcance de objetivos.
- ✓ Ubicar esos elementos en diagramas y paquetes.
- ✓ Crear conectores entre elementos.
- ✓ Documentar los elementos que ha creado.
- ✓ Generar código para el software que está construyendo.

Al utilizar EA, se puede realizar ingeniería directa y generar código C++, C#, Delphi, Java, Python, PHP, VB.NET y clases de Visual Basic, sincronizar códigos y elementos del modelo, diseñar y generar elementos de base de datos. La documentación de alta calidad puede ser rápidamente exportada desde sus modelos en industria estándar formato RTF e importar a Word para una personalización y presentación final. Enterprise Architect sustenta todos los diagramas y modelos UML.

Puede modelar procesos de negocio, sitios web, interfaces de usuario, redes, configuraciones de hardware, mensajes y más. Estimar el tamaño de su proyecto en esfuerzo de trabajo en horas. Capturar y trazar requisitos, recursos, planes de prueba, solicitudes de cambio y defectos. Desde los conceptos iniciales hasta el mantenimiento y soporte, Enterprise Architect tiene las características que precisa para diseñar y administrar su desarrollo e implementación. [27]

### 1.2.7.2 Visual Paradigm

Es una herramienta UML profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. El software de modelado UML ayuda a una más rápida construcción de aplicaciones de calidad, mejores y a un menor coste. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación.

- Características.
  - ✓ Producto de calidad.
  - ✓ Soporta aplicaciones web.
  - ✓ Las imágenes y reportes generados, no son de muy buena calidad.
  - ✓ Varios idiomas.
  - ✓ Generación de código para Java y exportación como HTML.



- ✓ Fácil de instalar y actualizar.
- ✓ Compatibilidad entre ediciones.
- ✓ En adición al soporte de Modelado UML esta herramienta provee el modelado de procesos de negocios, además de un generador de mapeo de objetos-relacionales para los lenguajes de programación Java .NET y PHP.
  
- Visual Paradigm ofrece:
  - ✓ Entorno de creación de diagramas para UML 2.1.
  - ✓ Diseño centrado en casos de uso y enfocado al negocio que generan un software de mayor calidad.
  - ✓ Uso de un lenguaje estándar común a todo el equipo de desarrollo que facilita la comunicación.
  - ✓ Capacidades de ingeniería directa (versión profesional) e inversa.
  - ✓ Modelo y código que permanece sincronizado en todo el ciclo de desarrollo.
  - ✓ Disponibilidad de múltiples versiones.
  - ✓ Disponibilidad de integrarse en los principales IDEs.
  - ✓ Disponibilidad en múltiples plataformas. [28]

### 1.2.7.3 ¿Por qué usar Enterprise Architect 7.0 (EA)?

Enterprise Architect es una herramienta muy poderosa porque es rápida, rica en funcionalidad, multiusuario, que conduce el éxito de cualquier proyecto de software, además presenta acompañamiento en todo el proceso de desarrollo, modelado de XML y otros; presenta características excelentes como realizar la trazabilidad y soporta diferentes lenguajes como Java, C#, C++ y VB.NET.

Como resultado del análisis realizado durante el presente capítulo, se identificaron las principales características de las herramientas y sistemas analizados, así como elementos significativos de la generación de formularios a tener en cuenta en el desarrollo del sistema propuesto. Para el mismo se seleccionaron las herramientas, tecnologías, lenguaje de programación y metodología más adecuadas.

### Capítulo 2: Características del Sistema

En este capítulo se realiza la descripción del sistema a desarrollar, para lograr una mayor claridad y comprensión por parte de los desarrolladores. Los puntos que se abordan son los siguientes: descripción del modelo de dominio, especificación de los requisitos funcionales y no funcionales, así como la descripción textual de ellos.

#### 2.1 Objeto de Automatización

Resulta de vital importancia que GIEP garantice que los cambios ocurridos en las plantillas XML no afecten el buen funcionamiento de la aplicación por lo que se desea obtener el siguiente proceso: El llenado de la información de los EP haciendo uso de formularios generados dinámicamente.

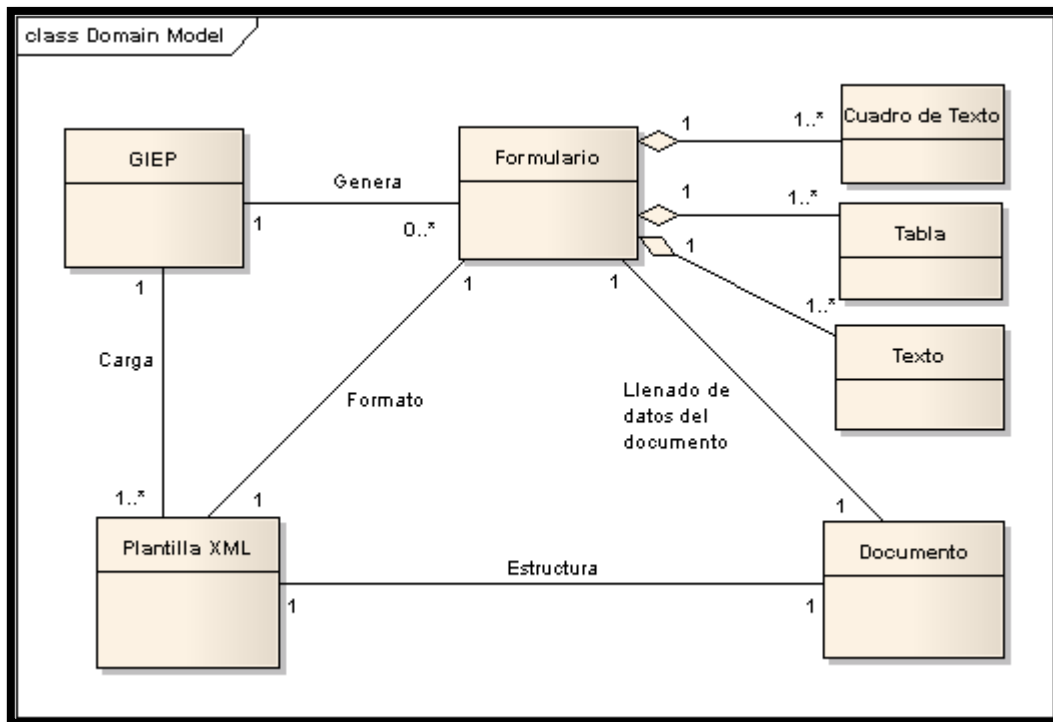
#### 2.2 Propuesta del sistema

Se propone la implementación de una herramienta que genere dinámicamente formularios basados en plantillas XML, para facilitar el llenado de información del Expediente de Proyecto permitiendo que varios usuarios actualicen o modifiquen la información de dicho EP, es decir agilizar el trabajo con los documentos del EP, por la cantidad de información que se registra en cada expediente, la herramienta cuenta con el objetivo de mantener los cambios realizados al mismo.

#### 2.3 Descripción del Modelo del Dominio

El modelo de dominio es una representación visual estática del entorno real objeto del proyecto. Es un diagrama con los objetos que existen (reales) relacionados con el proyecto que se va a desarrollar y las relaciones que hay entre ellos. Este modelo ayuda a comprender los conceptos que utilizan los usuarios, es decir, los conceptos con los que trabajan y con los que deberá trabajar la aplicación.

El modelo de dominio se describe mediante diagramas de UML, especialmente mediante diagramas de clases. En la ilustración 2 se representa el modelo de dominio realizado.



*Ilustración 2: Modelo de Dominio*

### 2.3.1 Glosario de Términos del Dominio

**GIEP:** es la herramienta de escritorio encargada de realizar una serie de funcionalidades para facilitar el trabajo con la documentación del EP.

**Documento:** es una plantilla del EP en la cual se recogen los datos.

**Plantilla XML:** es un archivo o fichero utilizado por la herramienta GIEP para guardar, cargar y mostrar información necesaria.

**Formulario:** representación gráfica de una plantilla siendo el formulario utilizado por el usuario para el llenado de información.

**Cuadro de texto:** son contenedores de texto que conforman el formulario.

**Texto:** elemento que conforma el formulario.

**Tabla:** elemento que permite organizar la información en columnas y filas que conforman el formulario.

### 2.4 Especificación de Requisitos

Como parte del modelado se definieron los requisitos, los cuales son una “condición o capacidad que necesita un usuario para resolver un problema o lograr un objetivo.” [29]

#### 2.4.1 Requisitos Funcionales

Los requisitos funcionales “son capacidades o condiciones que el sistema debe cumplir, se mantienen invariables sin importar con que propiedades o cualidades se relacionen por lo que no alteran la funcionalidad del producto”. [29]

A partir del modelo de dominio representado anteriormente, se alcanzó la visión necesaria del objeto a automatizar y quedaron determinadas las funcionalidades. Con ello, se pretende determinar de manera clara y concisa lo que debe hacer el sistema. A continuación se muestran los requisitos que debe cumplir el sistema:

**RF1 Cargar Expediente de Proyecto (EP):** El sistema debe permitir que se pueda cargar el EP creado por el usuario en el *Módulo Configuración*.

**RF2 Seleccionar plantilla:** El sistema debe permitir que el usuario seleccione la plantilla (asociada al documento) deseada.

**RF3 Mostrar formulario a partir de la plantilla XML:** El sistema debe generar un formulario asociado a la plantilla que el usuario seleccionó.

**RF4 Modificar información del Documento:** El sistema debe permitir que el usuario pueda modificar la información del documento.

**RF4.1 Guardar información del Documento:** El sistema debe permitir guardar toda la información generada por el usuario.

**RF5 Gestionar imágenes:** El sistema debe permitir que se pueda realizar las diferentes funcionalidades sobre una imagen.

**RF5.1 Cargar imágenes**

**RF5.2 Salvar imágenes**

**RF6 Control de versiones:** El sistema debe permitir al usuario guardar versiones del documento (plantilla) seleccionada.

### RF6.1: Guardar Versión

**RF7 Imprimir Documento:** El sistema debe permitir al usuario imprimir el documento seleccionado en el formato deseado en el *Módulo Imprimir Documento*.

### 2.4.2 Requisitos No Funcionales

Los requisitos no funcionales “son las propiedades o cualidades que el sistema debe tener”. [29] Los requisitos no funcionales son aquellos requisitos que hacen que el sistema sea usable, rápido, confiable y agradable para los usuarios.

Con el propósito de responder a las necesidades de los proyectos productivos de la UCI se definió un conjunto de propiedades y cualidades que debe cumplir el sistema, las cuales se describen a continuación.

#### **Apariencia o interfaz externa:**

La interfaz del sistema debe ser sencilla y amigable; de fácil comprensión para el usuario.

#### **Usabilidad:**

El sistema podrá ser usado por personas que tengan conocimientos básicos sobre el manejo de sistemas web.

#### **Portabilidad:**

El sistema será compatible con la plataforma del sistema operativo Linux.

#### **Software:**

Navegador Web: Mozilla FireFox v3.6 y v4.0 son las versiones definidas en el documento de arquitectura de desarrollo del CESIM; además de ser el navegador que soporta el software que se está desarrollando.

Sistema Operativo: Windows XP Service Pack3 es el sistema operativo utilizado en el CESIM para el desarrollo de aplicaciones.

Servidor de Aplicaciones: Internet Information Services.

### Hardware:

✚ PC\_Cliente:

- Tarjeta de red.
- Al menos 512 MB de memoria RAM o superior.
- 100 MB de disco duro.
- Procesador 2.0 MHz como mínimo.

✚ Servidor de Aplicación requiere:

- Tarjeta de red.
- Al menos 1GB de RAM.
- 1GB de espacio libre en el disco para la instalación de la aplicación.
- Procesador de 2.0 MHz como mínimo.

### 2.5 Definición de los Casos de Uso (CU)

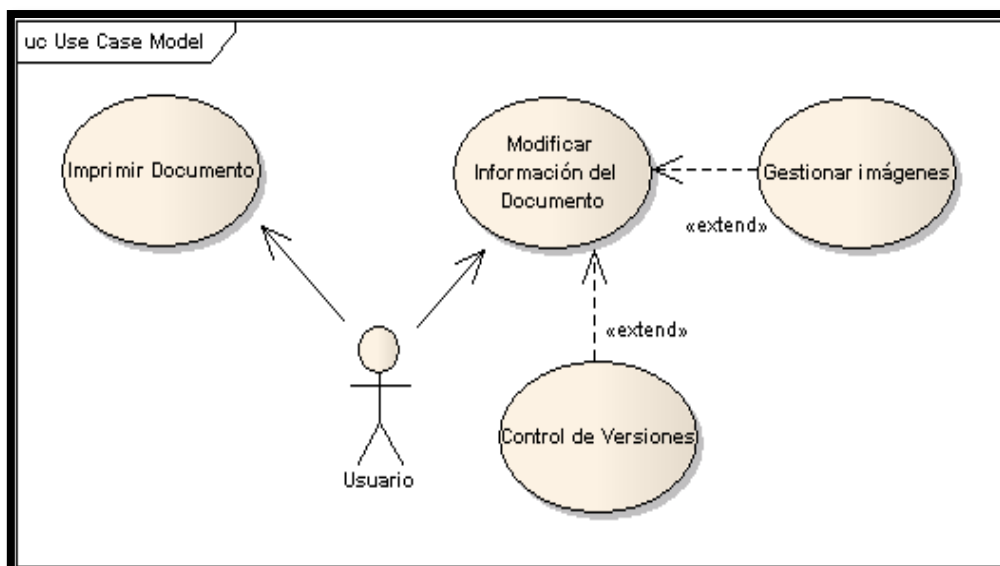
**CUS 1** Modificar información del Documento

**CUS 2** Gestionar imágenes

**CUS 2** Control de Versiones

**CUS 4** Imprimir Documento (Módulo Editor de Plantilla)

### 2.6 Diagrama de Casos de Uso del Sistema (CUS)



*Ilustración 3: Diagrama de Casos de Uso del Sistema*

### 2.7 Definición de los Actores

Un actor del sistema no es parte de la aplicación en desarrollo, es un agente externo que intercambia información con el mismo en pos de obtener un resultado esperado. Los actores definidos para el sistema en cuestión son:

Actores	Justificación
Usuario	Es el encargado de seleccionar la plantilla y realizar el llenado de información del documento.

Tabla 1: Descripción del Actor del Sistema

### 2.8 Descripción de Casos de Uso

#### Caso de Uso Modificar información del Documento

Caso de uso	
<b>CUS 1</b>	Modificar información del Documento
<b>Propósito</b>	Permitir la gestión de la información.
<b>Actores:</b> Usuario	
<b>Resumen:</b> El caso de uso se inicia cuando el sistema muestra el EP con sus correspondientes plantillas, el usuario da clic sobre la plantilla XML, el sistema busca la plantilla seleccionada; este visualiza el formulario y el usuario decide realizar el llenado de información en el formulario; el sistema brinda la posibilidad de guardar la información.	
<b>Referencias</b>	RF1, RF2, RF3, RF4, RF4.1
<b>Flujo Normal de Eventos</b>	
<b>Acción del actor</b>	<b>Respuesta del sistema</b>
2. Selecciona la plantilla XML.  5. Realiza el llenado de información en el formulario.  6. Selecciona la opción “ <b>Guardar</b> ” la	1. Carga y muestra el EP con sus plantillas.  3. Carga la plantilla XML (asociada al documento seleccionado).

## Capítulo 2: Características del Sistema

información.	4. Genera y muestra el formulario asociado a la plantilla seleccionada.  7. El sistema guarda la información.
<b>Flujo Alterno</b>	
1. Selecciona la opción “ <b>Cerrar</b> ”.	2. Cierra el formulario mostrando la interfaz principal del módulo.
<b>El Caso de Uso termina</b>	

*Tabla 2: Descripción del CU Modificar información del Formulario*

### Caso de uso Gestionar imágenes

Caso de uso	
<b>CUS 2</b>	Gestionar imágenes
<b>Propósito</b>	Permitir la gestión de las imágenes.
<b>Actores:</b> Usuario	
<b>Resumen:</b> El caso de uso se inicia cuando el usuario decide gestionar imagen permitiendo cargar y guardar la imagen.	
<b>Referencias</b>	RF5, RF5.1, RF5.2
Flujo Normal de Eventos	
Acción del actor	Respuesta del sistema
1. Da clic sobre el componente donde va la imagen.  3. Selecciona la imagen a añadir al formulario.  4. Selecciona la opción <b>Cargar</b> .  6. Selecciona la opción <b>Salvar</b> .	2. Muestra una nueva ventana mostrando un componente para realizar los criterios de búsquedas.  5. Carga la imagen seleccionada y la muestra.  5. Cierra la ventana y muestra la imagen en el formulario.



## Capítulo 2: Características del Sistema

<b>El Caso de Uso termina</b>	

*Tabla 3: Descripción del CU Gestionar imágenes*

Caso de uso	
<b>CUS 3</b>	Control de versiones
<b>Propósito</b>	Permitir la gestión de versiones.
<b>Actores:</b> Usuario	
<b>Resumen:</b> El caso de uso se inicia cuando el usuario selecciona la opción <i>control de versiones</i> ; el sistema le brinda la posibilidad de escoger la versión que desea.	
<b>Referencias</b>	RF6, RF6.1
Flujo Normal de Eventos	
Acción del actor	Respuesta del sistema
1. Selecciona la plantilla XML.  3. Da clic en el botón Control de Versiones.	2. Muestra la plantilla con un selector donde se muestran las versiones de ese documento, si el usuario desea trabajar con una de las versiones de ese documento selecciona la versión.  4. Guarda el documento con una Nueva Versión y actualiza el selector de versiones de ese documento.
<b>El Caso de Uso termina</b>	

*Tabla 4: Descripción del CU Control de versiones*

En este capítulo se identificaron los requisitos a cumplir por el sistema, tanto funcional como no funcional. Se realizó la definición de los Casos de Usos del Sistema, donde se encuentran las principales funcionalidades a implementar, además se definió un modelo del dominio ya que no se pudieron identificar los procesos de negocio.

### Capítulo 3: Diseño del Sistema

En el presente capítulo se abordarán los aspectos relacionados con el flujo de trabajo de análisis y diseño; incluyendo, los diagramas de clases del diseño, diagramas de interacción y la descripción de las clases de diseño.

#### 3.1 Flujo de trabajo Análisis y Diseño

El objetivo principal de esta disciplina es transformar los requerimientos a una especificación que describa cómo implementar el sistema. El análisis fundamentalmente consiste en obtener una visión que se preocupa de ver qué hace el software a desarrollar, por tal motivo este se interesa en los requerimientos funcionales. Por otro lado, el diseño es un refinamiento que toma en cuenta los requerimientos no funcionales, por lo cual se centra en cómo el sistema cumple sus objetivos.

Los objetivos específicos del análisis y diseño son:

1. Transformar los requerimientos al diseño del futuro sistema.
2. Desarrollar una arquitectura para el sistema.
3. Adaptar el diseño para que sea consistente con el entorno de implementación.

Al principio de la fase de elaboración hay que definir una arquitectura candidata: crear un esquema inicial de la arquitectura del sistema, identificar clases de análisis y actualizar las realizaciones de los casos de uso con las interacciones de las clases de análisis. Durante la fase de elaboración se va refinando esta arquitectura hasta llegar a su forma definitiva. En cada interacción hay que analizar el comportamiento para diseñar componentes. [30]

#### 3.2 Modelo de diseño

El diseño indica cómo lograr los requisitos relevados en el análisis, usando la metodología definida anteriormente, se puede describir, cómo el sistema llevará a cabo cada una de las funciones definidas, esta es una de las etapas más largas en el desarrollo de un sistema. La especificación del diseño aborda diferentes aspectos del

modelo de diseño y se completa a medida que el diseñador refina su propia representación del software.

### 3.2.1 Descripción de la Arquitectura

La programación por capas es un estilo de programación en el que el objetivo primordial es la separación de la lógica de negocios de la lógica de diseño; un ejemplo básico de esto consiste en separar la capa de datos de la capa de presentación al usuario. La ventaja principal de este estilo es que el desarrollo se puede llevar a cabo en varios niveles y, en caso de que sobrevenga algún cambio, sólo se ataca al nivel requerido sin tener que revisar entre código mezclado.

**Capa de Presentación:** es la encargada de interactuar con el usuario, solicitándole y/o mostrándole información de ser necesaria. Esta capa se comunica únicamente con la capa de negocio.

**Capa de Negocio:** es donde residen los programas que se ejecutan, se reciben las peticiones del usuario y se envían las respuestas tras el proceso. Se denomina capa de negocio (e incluso de lógica del negocio) porque es aquí donde se establecen todas las reglas que deben cumplirse. Esta capa se comunica con la capa de presentación, para recibir las solicitudes y presentar los resultados, y con la capa de datos, para solicitar al gestor de base de datos almacenar o recuperar datos de él. También se consideran aquí los programas de aplicación.

**Capa de Datos:** es donde residen los datos y es la encargada de acceder a los mismos. Está formada por uno o más gestores de bases de datos que realizan todo el almacenamiento de datos, reciben solicitudes de almacenamiento o recuperación de información desde la capa de negocio. [31]

A continuación en la ilustración 4 se muestra la arquitectura empleada en el sistema.

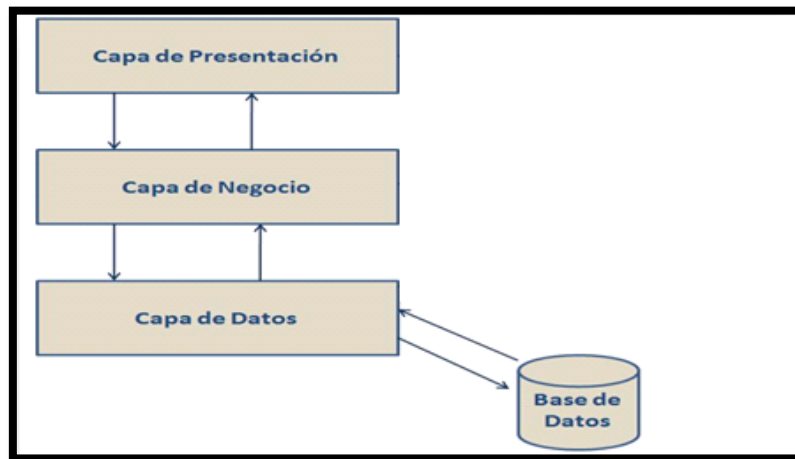


Ilustración 4: Figura de la Arquitectura en Capas

### 3.2.2 Fundamentación del uso de patrones

Los patrones de diseño de software constituyen un conjunto de principios generales y expresiones que ayudan a desarrollar software. Los patrones GRASP<sup>1</sup> describen los principios fundamentales de diseño de objetos para la asignación de responsabilidades. Dentro de este grupo se identifican 4 patrones fundamentales experto, creador, alta cohesión y el controlador.

En los diagramas de clases elaborados se aplican dichos patrones, se utilizan a fin de distribuir responsabilidades en las mismas, y establecer sus relaciones, tratando de que no estén muy sobrecargadas de funcionalidades ni exista mucha dependencia entre ellas.

El patrón **Experto** es el principio básico de asignación de responsabilidades. Indica que la responsabilidad de la creación de un objeto debe recaer sobre la clase que conoce toda la información necesaria para crearlo. Su problema radica en el principio general para asignar responsabilidades a los objetos, teniendo como solución asignar una responsabilidad al experto en información.

En este trabajo el patrón *Experto* es bastante utilizado ya que la herramienta GIEP contiene la clase GeneradorDinámico que es la encargada de crear todos los componentes a mostrar en la interfaz.

<sup>1</sup> Patrones Generales de Software para Asignación de Responsabilidades (General Responsibility Assignment Software Patterns)

El patrón **Creador** ayuda a identificar quién debe ser el responsable de la creación (o instanciación) de nuevos objetos o clases.

En este trabajo se muestra el uso del patrón *Creador* mediante las clases Document, Chapter, StartPage donde la clase Document es la encargada de contener objetos de las clases Chapter y StartPage estas a su vez contienen objetos de las clases Imagen y IElement las cuales son las encargadas de definir y ejecutar acciones.

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

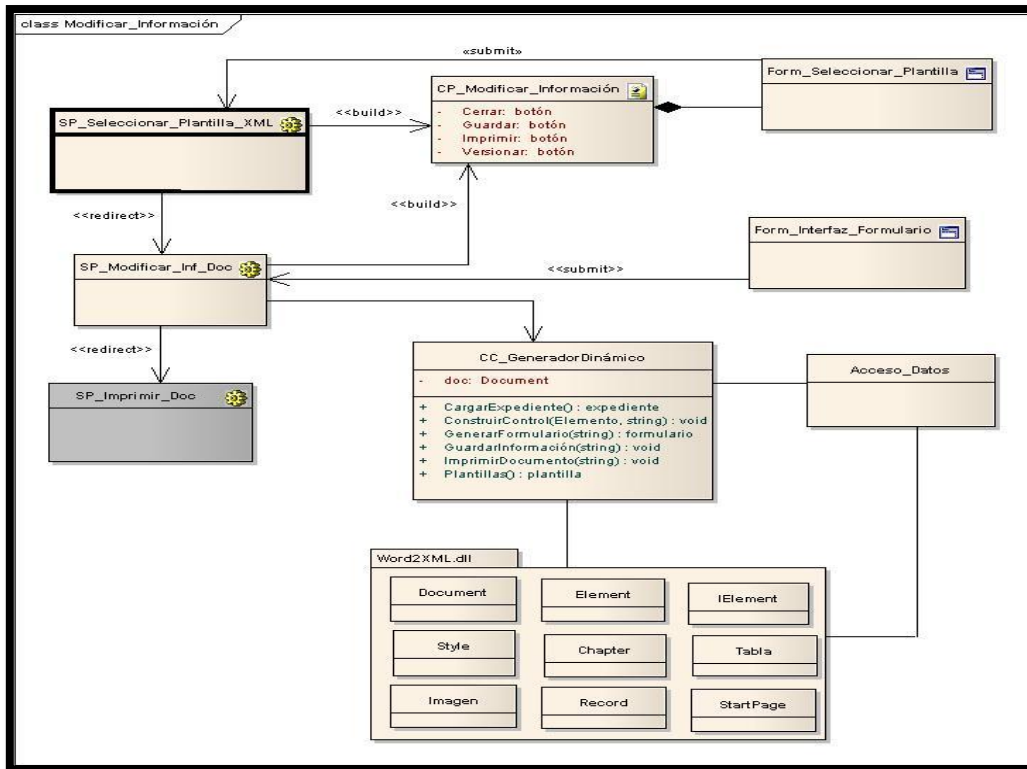
En este trabajo se muestra el uso del patrón *Controlador* el cual se evidencia por la clase GeneradorDinámico la cual es encargada de manejar las peticiones del sistema y de todas las acciones a realizar.

El patrón **Alta cohesión** La cohesión es una medida de cuán relacionadas y enfocadas están las responsabilidades de una clase. Una alta cohesión caracteriza a las clases con responsabilidades estrechamente relacionadas que no realicen un trabajo enorme. Una baja cohesión hace muchas cosas no afines o realiza trabajo excesivo. [32]

En GIEP las responsabilidades de las clases se asignaron de manera que haya *Alta Cohesión*, ejemplo de ello son las clases Element y la clase Imagen.

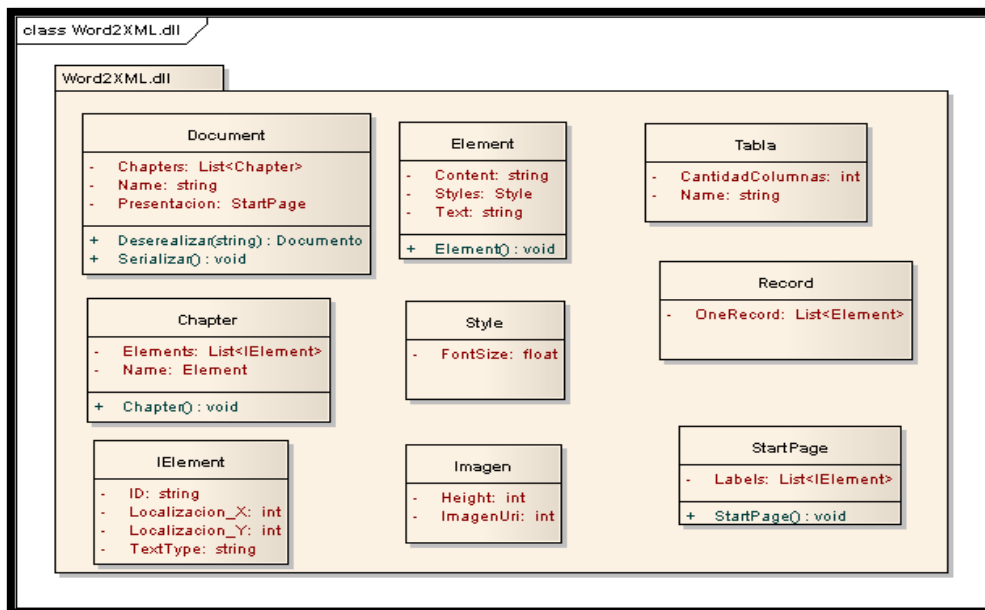
### 3.2.3 Diagrama de clases

En el diagrama de clases de diseño se muestran los atributos y métodos de cada clase y se representa de una forma sencilla la colaboración y las responsabilidades de las distintas clases que forman el sistema. El diagrama de clases de diseño representa la parte estática del sistema. Contiene las clases del diseño y sus relaciones. A continuación se presentarán los diagramas de clases del diseño de los principales casos de uso. [30]



**Ilustración 6: Diagrama de clase del diseño CU Modificar\_Información\_Documento**

A continuación se muestran en la Ilustración 7 detalladamente, las clases que están contenidas dentro de la librería Word2XML.dll, las cuales son utilizadas en el diseño de todos los Casos de Uso.



**Ilustración 7: Librería de clases utilizada en la clase de diseño**

## 3.2.4 Diagramas de Secuencia del diseño

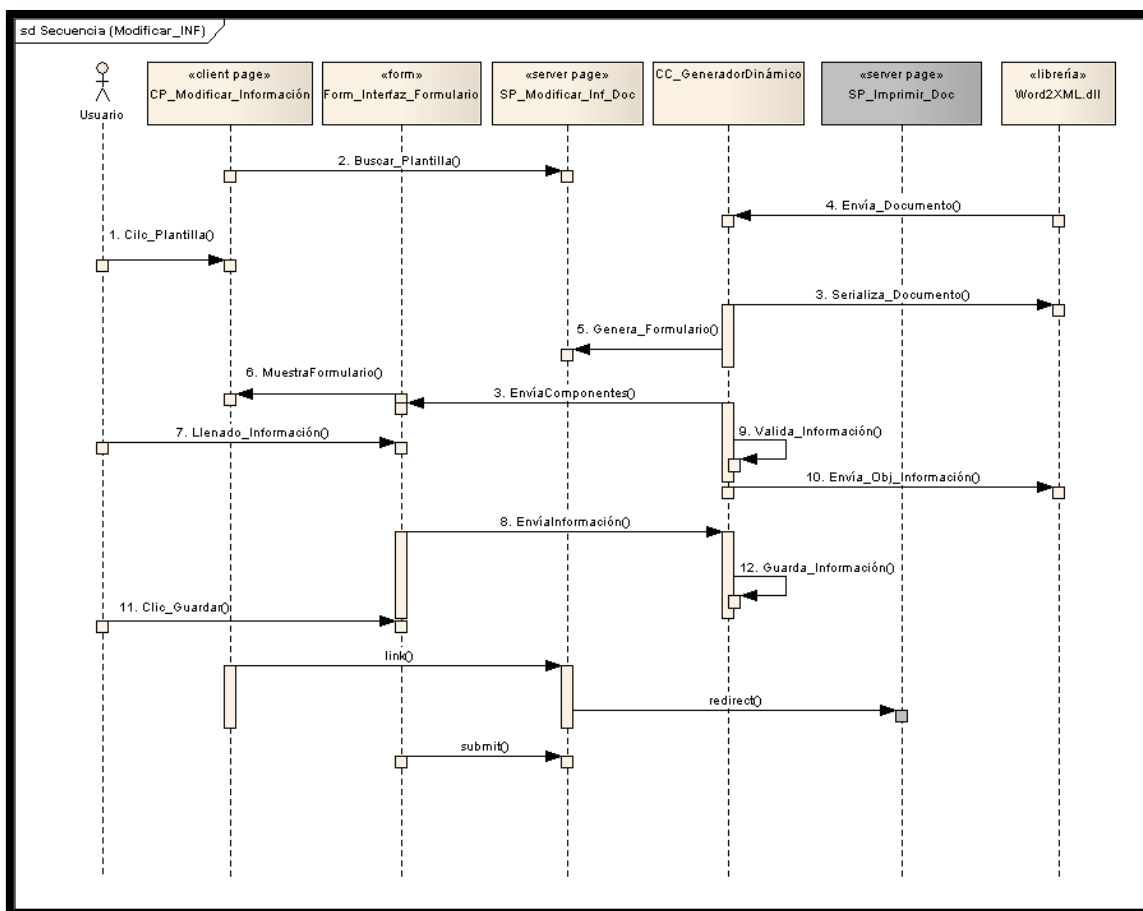


Ilustración 8: Diagrama de Secuencia del CU Modificar\_Información\_Documento

## 3.3 Descripción de las clases del diseño

<b>Nombre: CC_GeneradorDinámico</b>	
<b>Tipo de clase: Controladora</b>	
<b>Atributo</b>	<b>Tipo</b>
doc	Documento
<b>Para cada Responsabilidad</b>	
Nombre:	Plantillas()
Descripción:	Carga las plantillas correspondientes al expediente de proyecto.

## Capítulo 3: Diseño del Sistema

Nombre:	CargarExpediente()
Descripción:	Muestra la estructura del Expediente de Proyecto.
Nombre:	ConstruirControl(string obj, Elemento)
Descripción:	Se encarga de crear un componente de tipo Control a partir del objeto pasado por parámetros con atributos dados por el elemento.
Nombre:	GenerarFormulario(string obj)
Descripción:	Se encarga de representarle al usuario un formulario del documento seleccionado por el mismo.
Nombre:	GuardarInformación(string obj)
Descripción:	Se encarga de guardar toda la información del documento que el usuario lleno en el formulario.
Nombre:	ImprimirDocumento(string obj)
Descripción:	Se encarga de llamar al Módulo Imprimir Documento pasándole el objeto documento.

<b>NOMBRE:</b>	CI_InterfazPrincipal
<b>Tipo de Clase:</b>	Interfaz
<b>Atributo</b>	<b>Tipo</b>
Guardar	Botón
Imprimir	Botón
Cerrar	Botón



## Capítulo 3: Diseño del Sistema

Versionar	Botón
<b>Descripción:</b>	Esta clase muestra una interfaz para guardar la información contenida en el formulario, imprimir el documento y cancelar la operación realizada. Además de permitir guardar versiones del documento.

<b>NOMBRE:</b>	CE_Document
<b>Tipo de Clase:</b>	Entidad
<b>Atributo</b>	<b>Tipo</b>
Chapters	List<Chapter>
Name	string
Presentación	StartPage
<b>Para cada responsabilidad</b>	
Nombre:	Serializar()
Descripción:	Se encarga de convertir el documento en un XML.
Nombre:	Deserializar(string obj)
Descripción:	Se encarga de convertir una plantilla XML a documento.

En este capítulo quedaron expuestos una serie de elementos y diagramas que muestran como está construido el sistema en términos de clases del diseño. Se arribó a la conclusión de que haciendo un diseño detallado de la estructura de la futura aplicación, dará la posibilidad de comprender mejor la lógica del sistema en general y optimizará el tiempo de implementación.

### Capítulo 4: Implementación y Pruebas

En este capítulo quedan plasmados todos los detalles referentes a la implementación del sistema, mostrando de esta forma los diagramas de componentes, dando una vista detallada de cada uno de los paquetes en que se ha dividido y los múltiples elementos físicos y archivos que conforman la aplicación, con el objetivo de lograr una mayor claridad y comprensión del modelo, seguidamente el diagrama de despliegue que muestra la distribución del sistema en los diferentes elementos de hardware que le darán soporte.

#### 4.1 Modelo de implementación

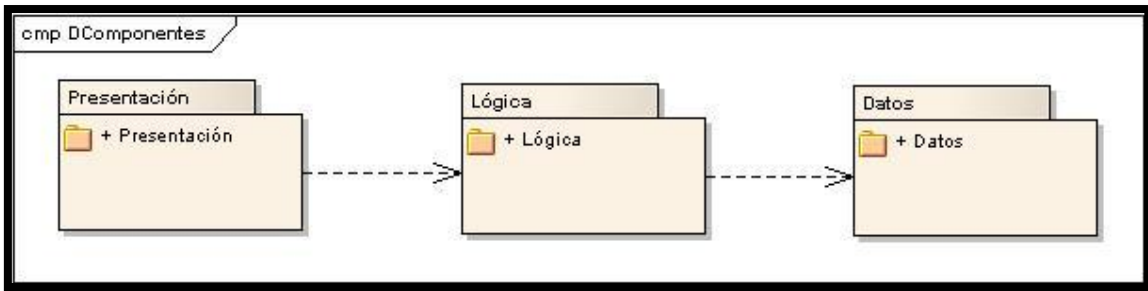
El modelo de implementación describe como los elementos del modelo de diseño y las clases, se implementan en términos de componentes, ficheros de código fuente, ejecutables, entre otros. El diagrama de componentes conforman lo que se conoce como un modelo de implementación, al describir los componentes y construir su organización y dependencia entre los nodos físicos en que funcionará la aplicación. [33]

##### 4.1.1 Diagrama de Componentes

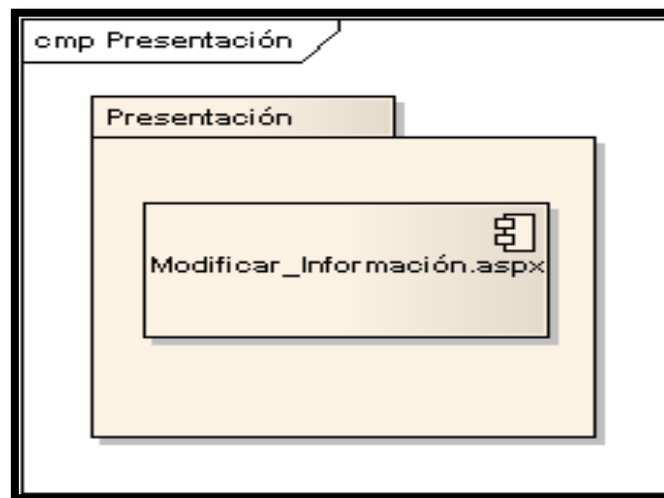
Un diagrama de componentes muestra las organizaciones y dependencias lógicas entre componentes software, sean éstos componentes de código fuente, binarios o ejecutables. Normalmente contienen componentes, interfaces y relaciones entre ellos y como todos los diagramas, también puede contener paquetes utilizados para agrupar elementos del modelo.

Los diagramas de componentes muestran los componentes software que constituyen una parte reusable, sus interfaces, y sus interrelaciones, en muchos aspectos se puede considerar que un diagrama de componentes es un diagrama de clases a gran escala. Cada componente en el diagrama debe ser documentado con un diagrama de componentes más detallado, un diagrama de clases, o un diagrama de casos de uso. [34]

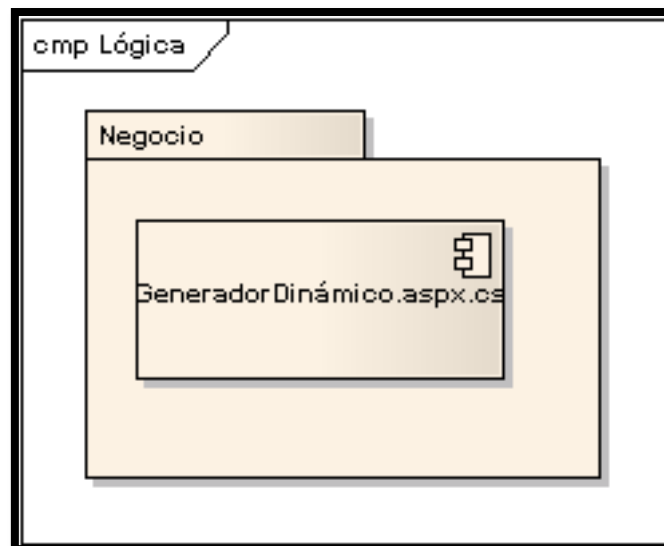
A continuación se muestra en la ilustración 9 una vista general del diagrama de componentes y el las ilustraciones 10, 11 y 12 una vista más detallada de los paquetes, así como los componentes que contienen cada paquete.



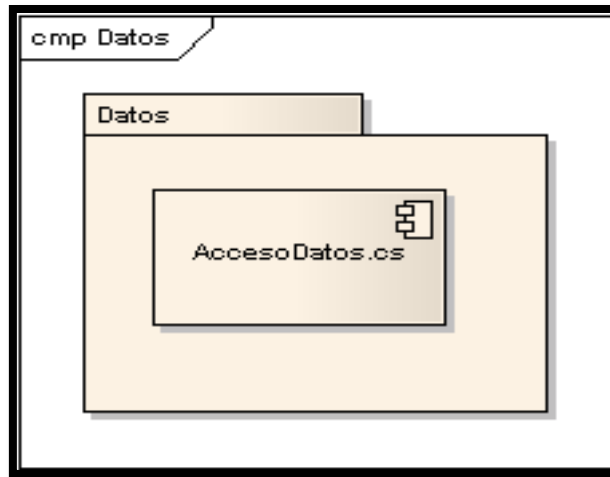
*Ilustración 9: Vista General del Diagrama de Componentes*



*Ilustración 10: Vista Detallada del Paquete Presentación*



*Ilustración 11: Vista Detallada del Paquete Lógica del Negocio*



*Ilustración 12: Vista Detallada del Paquete Datos*

### 4.1.2 Diagrama de Despliegue

El diagrama de despliegue es utilizado para modelar el hardware<sup>2</sup> empleado en las implementaciones de sistemas y las relaciones entre sus componentes. El modelo de despliegue es un modelo de objetos que describe la distribución física del sistema en términos de cómo se distribuye la funcionalidad entre los nodos de cómputo. [35]

El diagrama de despliegue correspondiente al sistema desarrollado, como se muestra en la ilustración 13, consta con una computadora cliente, en la cual los usuarios pueden visualizar e interactuar con la aplicación que se encuentra en el servidor de aplicación. Este servidor es el encargado de responder las peticiones de las computadoras clientes y a su vez está conectado al servidor de base de datos, el cual es el encargado de administrar los datos necesarios para el funcionamiento correcto de la aplicación **alasGIEP**.

<sup>2</sup> Hardware: corresponde a todas las partes tangibles de una computadora.

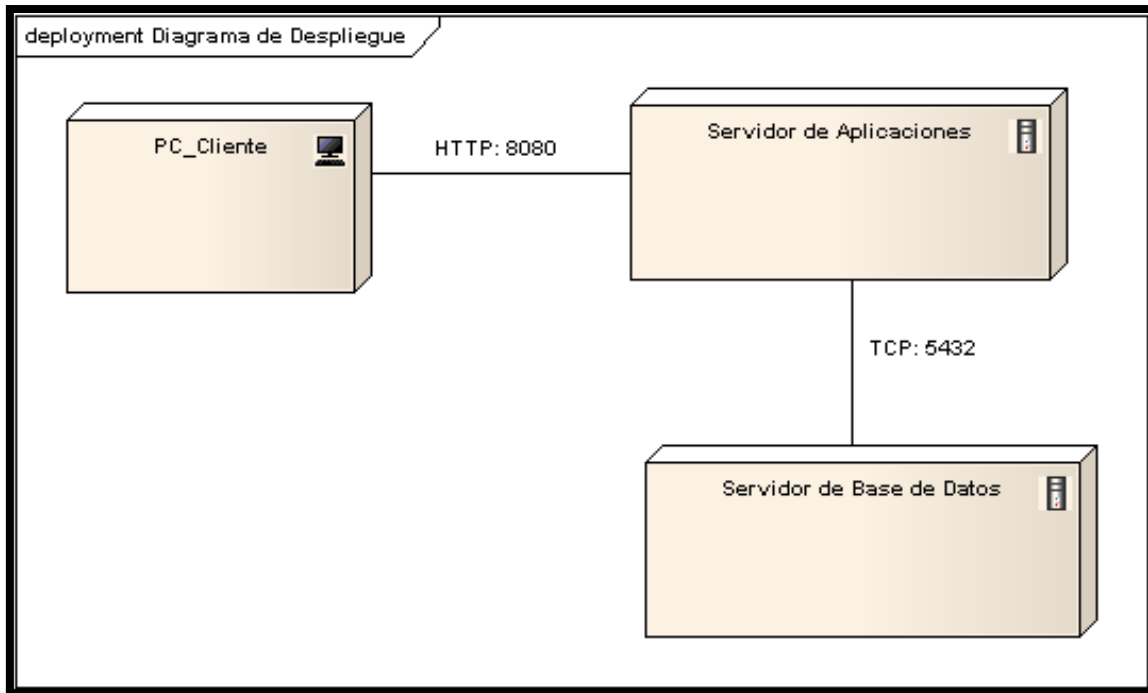


Ilustración 13: Diagrama de Despliegue

### 4.1.3 Implementación de las principales funcionalidades

#### ✓ Cargar Expediente de Proyecto y Cargar XML

Para mostrar el Expediente de Proyecto con sus respectivas plantillas se realizó la implementación de dos métodos importantes CargarExpediente () el cual a través de un componente de la web llamado TreeView es el encargado de cargar la estructura del expediente generado por el usuario en el módulo Configuración (Trabajo de Diploma Alexander y Daymi); además de hacer uso de la recursividad para la creación de carpetas con sus respectivas plantillas XML con el método CargarXML () que crea el conjunto de documentos que conforman el EP.

```
protected void tv_Init(object sender, EventArgs e)
{
    TreeNode raiz = new TreeNode("Documentos");
    raiz.ImageUrl = "img/carpeta.png";
    raiz.Expanded = false;
    CargarExpediente(raiz, Request.PhysicalApplicationPath + "/documentos");
    tv.Nodes.Add(raiz);
}
```

```
public void CargarExpediente(TreeNode raiz, string path)
{
    char a=' ';
    foreach (string childVdir in Directory.GetDirectories(path))
    {
        TreeNode hijo = new TreeNode(Path.GetFileName(childVdir));
        hijo.ImageUrl = "img/carpeta.png";
        hijo.Expanded = false;
        raiz.ChildNodes.Add(hijo);
        CargarXML(hijo, path + "/" + hijo.Text);
        CargarExpediente(hijo, path + "/" + hijo.Text);
    }
}
public void CargarXML(TreeNode raiz, string path)
{
    foreach (string xml in Directory.GetFiles(path))
    {
        string file = Path.GetFileNameWithoutExtension(Path.GetFileName(xml));

        TreeNode doc = new TreeNode(file);
        doc.ImageUrl = "img/xml.png";
        doc.Expanded = false;
        raiz.ChildNodes.Add(doc);
    }
}
```

*Ilustración 15: Fragmento de código Cargar Expediente y Cargar XML*

### ✓ Generar Formulario

Para la generación dinámica de formularios, se utilizó la clase *GeneradorDinámico.aspx.cs* la cual es la encargada de crear físicamente el formulario utilizando un objeto de la clase Document la cual permite serializar y deserializar la plantilla XML para su posterior trabajo para la generación dinámica; esta funcionalidad crea la interfaz amigable para el usuario, así el mismo realiza el llenado de información más rápidamente.

```
public void GenerarFormulario(string path)
{
    doc = Deserializar(path);
    NombreDoc = doc.Name.Text;
    var nuevo = new AccordionPane { };
    nuevo.ID = "nuevo";
    if (doc.Presentation.Labels.Count != 0)
    {
        var ap1 = new AccordionPane { ID = doc.Name.Text };
        ap1.HeaderContainer.Controls.Add(new Label { Text = "Presentacion" });
        for (int i = 0; i < doc.Presentation.Labels.Count; i++)
        {
            if (doc.Presentation.Labels[i] is Element)
            {
                ap1.ContentContainer.Controls.Add(ConstruirElemento(doc.Presentation.Labels[i]));
            }
            if (doc.Presentation.Labels[i] is Imagen)
            {
                ap1.ContentContainer.Controls.Add(ConstruirImagen(doc.Presentation.Labels[i]));
            }
            if (doc.Presentation.Labels[i] is Word2Xml.Library.Entities.Table)
            {
                ap1.ContentContainer.Controls.Add(ConstruirTabla(doc.Presentation.Labels[i]));
            }
        }
        MyAccordion.Panes.Add(ap1);
    }
}
```

*Ilustración 14: Fragmento del código que Genera Formulario*

```
if (doc.Chapters.First() != null)
{
    foreach (Chapter cap in doc.Chapters)
    {
        var ap = new AccordionPane { ID = cap.Name.Text };
        ap.HeaderContainer.Controls.Add(new Label { Text = cap.Name.Text });
        if (cap.Elements.First() != null)
        {
            foreach (IElement e in cap.Elements)
            {
                if (e is Element)
                {
                    ap.ContentContainer.Controls.Add(ConstruirElemento(e));
                }
                if (e is Word2Xml.Library.Entities.Table)
                {
                    ap.ContentContainer.Controls.Add(ConstruirTabla(e));
                }
                if (e is Imagen)
                {
                    ap.ContentContainer.Controls.Add(ConstruirImagen(e));
                }
            }
        }
        MyAccordion.Panes.Add(ap);
    }
}
acordion = MyAccordion;
```

*Ilustración 15: Fragmento del código que Genera Formulario*

### ✓ Guardar Información

La información de todas las plantillas que conforman el Expediente de Proyecto UCI (EP) se guardará en documentos XML con una jerarquía organizada; esto lo logra capturando los ID de cada componente que se utiliza en la creación del formulario y se compara con el Request de la página web; que no es más que al refrescar la página esta envía los datos introducidos por el usuario en cada componente. A continuación se presentan fragmentos del código guardar información del formulario.

```
protected void Guardar(object sender, EventArgs e)
{
    if (IsValid)
    {
        string path = Request.PhysicalApplicationPath + ddl_tipo_doc.SelectedValue + "\\\" + tbName.Text + ".xml";
        if (!string.IsNullOrEmpty(path))
        {
            if (File.Exists(path))
            {
                lblError.Text = "El documento \"'\" + tbName.Text + "\"' ya existe.";
                ModalPopupExtender.Show();
            }
            else
            {
                string var = string.Empty;
                CargarHiperfunciones(Direccion);
                List<string> listaID = new List<string>();
                List<System.Web.UI.WebControls.Table> Listtablas = new List<System.Web.UI.WebControls.Table>();
                List<Control> controlesentablas = new List<Control>();
            }
        }
    }
}
```

```
for (int i = 0; i < MyAccordion.Panes.Count; i++)
{
    for (int j = 0; j < MyAccordion.Panes[i].ContentContainer.Controls.Count; j++)
    {
        if (MyAccordion.Panes[i].ContentContainer.Controls[j] is System.Web.UI.WebControls.Table)
        {
            Listtablas.Add(MyAccordion.Panes[i].ContentContainer.Controls[j] as System.Web.UI.WebControls.Table);
        }
        else
        {
            var id = MyAccordion.Panes[i].ContentContainer.Controls[j].ID;
            listaID.Add(var);
        }
    }
}
for (int t = 0; t < Listtablas.Count; t++)
{
    for (int m = 0; m < Listtablas[t].Rows.Count; m++)
    {
        for (int n = 0; n < Listtablas[t].Rows[m].Cells.Count; n++)
        {
            for (int c = 0; c < Listtablas[t].Rows[m].Cells[n].Controls.Count; c++)
            {
                controlesentablas.Add(Listtablas[t].Rows[m].Cells[n].Controls[c]);
            }
        }
    }
}
```

```
for (int g = 0; g < controlesentablas.Count; g++)
{
    listaID.Add(controlesentablas[g].ID);
}
for (int i = 0; i < doc.Presentation.Labels.Count; i++)
{
    if (doc.Presentation.Labels[i] is Element)
    {
        for (int j = 0; j < listaID.Count; j++)
        {
            if (listaID[j] == (doc.Presentation.Labels[i] as Element).ID)
            {
                try
                {
                    var component = (from comp in Request.Form.AllKeys
                                     where comp.EndsWith("$" + listaID[j])
                                     select comp).Single();
                    (doc.Presentation.Labels[i] as Element).Content = Request[component].ToString();
                }
                catch { }
            }
        }
    }
}
```



### 4.2 Pruebas

La prueba es el proceso de ejecución de un programa con la intención de descubrir un error. Las mismas son un elemento crítico para la garantía de calidad del software y representan una visión final de las especificaciones, del diseño y de la codificación.

Las pruebas comienzan desde la fase de inicio del software hasta la fase de construcción, siendo esta última fase donde tiene mayor volumen el flujo de trabajo de prueba. Uno de los tipos de prueba que existen son las pruebas de caja negra. [36]

#### 4.2.1 Prueba de Caja Negra

Las pruebas de caja negra, también denominada Prueba de Comportamiento, se centran en los requisitos funcionales del software. La prueba de caja negra intenta encontrar errores de las siguientes categorías:

1. Funciones incorrectas o ausentes.
2. Errores de interfaz.
3. Errores en estructuras de datos o en accesos a bases de datos externas.
4. Errores de rendimiento.
5. Errores de inicialización y de terminación.

Estas pruebas se refieren a las pruebas que se llevan a cabo sobre la interfaz del software. Los casos de prueba pretenden demostrar que las funciones del software son operativas, que la entrada se acepta de forma adecuada y que se produce un resultado correcto. Una prueba de caja negra examina algunos aspectos del modelo fundamental del sistema sin tener mucho en cuenta la estructura lógica interna del software.

#### 4.2.2 Pruebas de los componentes

Se entiende por caso de prueba, según la Ingeniería del software, al conjunto de condiciones o variables bajo las cuáles el analista determinará si el requisito de una aplicación es parcial o completamente satisfactorio. A continuación se muestran los casos de prueba realizados a los casos de uso Seleccionar Plantilla XML, Modificar

## Capítulo 4: Implementación y Pruebas

información del Documento, Gestionar imágenes y Control de versiones se muestran en las tablas 6, 7, 8.

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad.	Resultado	Flujo Central
SC1: Gestionar imágenes.	EC1.1: Gestionar imagen.	Se permite gestionar imágenes en el formulario.	Satisfactorio	<ul style="list-style-type: none"> <li>- Se selecciona el componente donde va la imagen.</li> <li>- El sistema muestra los criterios de búsquedas.</li> <li>- El usuario selecciona la imagen añadir al formulario</li> </ul>

*Tabla 6: DCP del CU Gestionar imágenes*

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad.	Resultado	Flujo Central
1: Modificar información del documento.	EC1.1: Llenar información.	Se permite realizar el llenado de información en el formulario.	Satisfactorio	<ul style="list-style-type: none"> <li>- El sistema muestra el EP con sus respectivas plantillas.</li> <li>- El usuario selecciona la plantilla XML que desea.</li> <li>- El sistema busca la plantilla que el usuario seleccionó.</li> <li>- El sistema genera el</li> </ul>

## Capítulo 4: Implementación y Pruebas

	EC 1.2: Selecciona la opción <b>Cerrar</b> .	Permite cerrar el formulario.		<ul style="list-style-type: none"> <li>- El usuario selecciona la opción <b>Cerrar</b>.</li> <li>- El sistema cierra el formulario mostrando la interfaz principal del módulo.</li> </ul>
--	---	-------------------------------	--	---

*Tabla 7: DCP del CU Modificar Información del Documento*

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad.	Resultado	Flujo Central
SC1: Control de versiones.	EC1.1: Guardar nuevas versiones.	Se permite guardar la nueva versión del documento seleccionado.	Satisfactorio	<ul style="list-style-type: none"> <li>- El usuario selecciona la plantilla</li> <li>- El sistema muestra la plantilla con sus respectivas versiones en un selector.</li> <li>- El usuario da clic en la opción <b>Control de versiones</b>.</li> <li>- El sistema guarda el</li> </ul>

*Tabla 8: DCP del CU Control de versiones*

En este capítulo queda plasmado como está construido el sistema a través del diagrama de componentes de la aplicación en general. También fue mostrado el diagrama de despliegue, el cual ilustra los nodos que serán usados para la distribución física de la aplicación. Como parte del flujo de prueba se realizaron las pruebas correspondientes a través de 3 casos de pruebas, los cuales arrojaron en la primera iteración realizada 14 no conformidades de las cuales 2 eran significativas y las restantes no significativas. En una última iteración el sistema cumple con las funcionalidades previstas.

### Conclusiones

Las aplicaciones informáticas que existen dedicadas a la generación de formularios, son herramientas potentes, pero estas no trabajan con los expedientes de proyecto, ni con ficheros XML para la generación de los formularios. Además no realizan la generación de forma dinámica; por lo que estas aplicaciones no son adaptables a las exigencias del proyecto Calidad.

Se realizó un análisis de las tendencias actuales de las tecnologías, y se seleccionaron las más adecuadas para la implementación del sistema. Como lenguaje de programación C# y ASP.net, la metodología utilizada para guiar el proceso de desarrollo es RUP, se emplea como lenguaje de modelado UML, como herramienta CASE para visualizar los modelos, Enterprise Architect; y para realizar el diseño de la aplicación, se utilizó la Arquitectura 3 Capas.

Como resultado final, se obtuvo una herramienta que posibilita la generación dinámica de formularios basados en plantillas XML, la cual agiliza el proceso de llenado de información de los expedientes de proyecto.

### Recomendaciones

Los autores recomiendan:

- Trabajar en mejoras en cuanto al diseño del sistema, teniendo en cuenta la opinión de los usuarios.
- Desarrollar aplicativos con la herramienta propuesta, ya que existe la documentación necesaria para el mismo.
- Instalar al navegador con el que trabaja la aplicación, un plugins para la corrección ortográfica.

### Referencias Bibliográficas

- [1]. Introducción [En línea] Consultado Septiembre 2010, Universidad de Málaga. Archivo General.2006
- [2]. *Monográfico Calidad del Software / Software de calidad. Número 137, s.l.: NOVATICA, Enero-Febrero 2004*
- [3]. Estado del Arte [En línea] Consultado Febrero 2011, [http://kuainasi.ciens.ucv.ve/ideas07/documentos/articulos\\_ideas/Articulo35.pdf](http://kuainasi.ciens.ucv.ve/ideas07/documentos/articulos_ideas/Articulo35.pdf).
- [4]. Jotform [En línea] 2010 <http://netic360.blogspot.com/2010/01/jotform-crear-formularios-online.html>
- [5]. Microsoft InfoPath [En línea] 2010 <http://www.microsoft.com/latam/office/infopath/prodinfo/using.mspix>
- [6]. Wufoo [En línea] Diciembre 2010 <http://www.incubaweb.com/wufoo-herramienta-para-crear-formularios-online/>
- [7]. Formsite [En línea] Diciembre 2010 <http://cosassencillas.wordpress.com/2007/04/17/formsite-sistema-para-generar-formularios-web/> <http://www.formsite.com>
- [8]. Microsoft Access [En línea] 2010 <http://www.aulafacil.com/Access2/CursoAccess/Lecc-22-Acc.htm>
- [9]. Generador Automático de Plantillas en la Web [En línea] 2010 Trabajo de Diploma Generador automático de plantillas en la Web.UCI, 2010.
- [10]. Reyes Prieto Dariel Fernando y Seguí García Arian Lenguajes de Programación [En línea] Consultado Diciembre 2010 Trabajo de Diploma Herramienta para la gestión de la información del Expediente de Proyecto; Junio 2009
- [11]. Lenguaje C# [En línea] Diciembre 2010 Microsoft Corporation. MSDN Microsoft DeveloperNetwork. Visual Studio Developer Center. [En línea] Microsoft Corporation. [Citado en:] <http://msdn.microsoft.com/en-gb/vstudio/bb265237.aspx>

- [12]. Ventajas frente a C/C# [En línea] Consultado Septiembre 2010 [Citado en:] Archer, Tom. A FONDO C#. Redmond: McGraw-Hill/INTERAMERICANA DE ESPAÑA, S.A.U, 2001.ISBN: 84-481-3246-7
- [13]. ASP [En línea] Noviembre 2010 <http://www.subgurim.net/Articulos/asp-net-general>
- [14]. ASP .NET [En línea] Consultado Noviembre 2010 ASP.NET Master Pages Overview (Microsoft Developer Network)
- [15]. XML [En línea] Noviembre 2010 [Citado en:] <http://www.desarrolloweb.com/articulos/>, Eguiluz Pérez, 25-03-2009
- [16]. JavaScript [En línea] 2010 <http://www.subgurim.net/Articulos/asp-net-general>
- [17]. González Polanco Liset y Pérez Betancourt Yadian Guillermo ¿Por qué usar ASP.NET? [En línea] Consultado Diciembre 2010. Análisis y Diseño del subsistema Préstamos del proyecto Modernización del Sistema Bancario Cubano. Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas, Mayo 2009
- [18]. Framework [En línea] Noviembre 2010 <http://www.pedrov.info/>, [Citado en:] <http://msdn.microsoft.com/es-es/library/zw4w595w.aspx>
- [19]. Framework .NET 3.5 [En línea] 2010 [Citado en:] [http://www.carlospes.com/curso\\_de\\_ingenieria\\_del\\_software/04\\_06\\_entornos\\_integrados\\_de\\_desarrollo.php](http://www.carlospes.com/curso_de_ingenieria_del_software/04_06_entornos_integrados_de_desarrollo.php)
- [20]. Visual Studio 2008 [En línea] 2010 <http://www.msdn2.microsoft.com>
- [21]. Visual Studio 2008 [En línea] 2010 <http://msmvps.com/blogs/ffagas/archive/2007/08/02/overview-de-visual-studio-2008.aspx>
- [22]. Metodologías de Desarrollo de Software [Consultado en:] Jacobson Ivar, Booch Grady, Rumbaugh James. (2000).
- [23]. Proceso Unificado de Desarrollo de Software (RUP) [Citado en:] D. d. IS, "Introducción a la ingeniería de software," 2007

- [24]. Maza Capote Idalmys y Fabars Hardy Yusdelis. Programación Extrema (XP) [Citado en:] Trabajo de Diploma Sistema para la gestión de la información de los discos contenedores de imágenes médicas (alasSMIS), Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas, Junio de 2009
- [25]. Lenguaje Unificado de Modelado 2.0 [En línea] 2010 Larman, Craig. (2004). "UML y Patrones. Introducción al análisis y diseño orientado a objetos". Félix Varela.
- [26]. Herramientas CASE [Citado en:] Material Básico de Bibliografía Básica de Ingeniería de Software
- [27]. Enterprise Architect [En línea] 2010 [Citado en:]  
[Http://www.thedigitalmap.com/EasyDEM/download/aide/html/documentos/robotikerWebServices.pdf](http://www.thedigitalmap.com/EasyDEM/download/aide/html/documentos/robotikerWebServices.pdf)
- [28]. Visual Paradigm [Disponible en]:  
[http://www.softpile.com/Development/Editors\\_and\\_IDEs/Review\\_19078\\_index.html](http://www.softpile.com/Development/Editors_and_IDEs/Review_19078_index.html)
- [29]. Especificación de requisitos [En línea] Consultado Diciembre 2010 Jacobson, Booch et al. 2004
- [30]. González Lisdany y Cruz Yordan Flujos de trabajo Análisis y Diseño [En línea] Consultado Febrero 2011. Plataforma para la Gestión de la Calidad en el Centro de Informática Médica (CESIM). . Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas, Junio 2010
- [31]. Descripción de la Arquitectura 3 Capas [En línea] Abril 2011  
<http://kernelerror.net/programacion/php/arquitectura-3-capas/>
- [32]. Valencia Yudisleydis y Peña Amarilis. Fundamentación del uso de patrones [En línea] Consultado Marzo 2011. Herramienta para automatizar el proceso de evaluación del programa de mejoras en los proyectos productivos de la UCI, basados en el modelo CMMI. Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas, Junio 2010
- [33]. Pérez López Yunelis y San Juan Rosabal Antonio. Modelo de implementación [En línea] Consultado Marzo 2011. Sistema para la Gestión y Análisis de Información Estadística en la salud pública cubana: Subsistema Editor de Plantillas. Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas, Junio 2009



## Referencias Bibliográficas

---

[34]. Valdés Rodríguez Jorge Fernando. Diagrama de Componentes. Desarrollo del Módulo de Gestión de Recursos Humanos del Sistema Integral de Gestión Administrativa de la Facultad 7. Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas, Julio 2010

[35]. Diagrama de Despliegue [En línea] Consultado Marzo 2011 Entorno virtual de aprendizaje. Ingeniería de software 2. Clase práctica para estudiantes. UCI:s.n., 2008-2009.

[36]. Pérez Vázquez, Maridalia y Rodríguez Hidalgo, Guillermo William. Pruebas [En línea] Consultado Abril 2011. Desarrollo de componentes de Interfaz de usuarios para la representación de información mediante gráficas en el polo Petrosoft Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas, Junio 2010

### Bibliografía

- [1]. Introducción [En línea] Consultado Septiembre 2010, Universidad de Málaga. Archivo General.2006
- [2]. *Monográfico Calidad del Software / Software de calidad. Número 137, s.l.: NOVATICA, Enero-Febrero 2004*
- [3]. [http://kuainasi.ciens.ucv.ve/ideas07/documentos/articulos\\_ideas/Articulo35.pdf](http://kuainasi.ciens.ucv.ve/ideas07/documentos/articulos_ideas/Articulo35.pdf).
- [4]. <http://netic360.blogspot.com/2010/01/jotform-crear-formularios-online.html>
- [5].<http://www.microsoft.com/latam/office/infopath/prodinfo/using.msp>x
- [6]. Wufoo [En línea] <http://www.incubaweb.com/wufoo-herramienta-para-crear-formularios-online/>
- [7]. Formsite [En línea] <http://cosassencillas.wordpress.com/2007/04/17/formsite-sistema-para-generar-formularios-web/> <http://www.formsite.com>
- [8]. Microsoft Access [En línea] <http://www.aulafacil.com/Access2/CursoAccess/Lecc-22-Acc.htm>
- [9]. Generador Automático de Plantillas en la Web [En línea] 2010 Trabajo de Diploma Generador automático de plantillas en la Web.UCI, 2010.
- [10]. Reyes Prieto Dariel Fernando y Seguí García Arian Lenguajes de Programación. Trabajo de Diploma Herramienta para la gestión de la información del Expediente de Proyecto; Junio 2009
- [11]. Lenguaje C# [En línea] Microsoft Corporation. MSDN Microsoft DeveloperNetwork. Visual Studio Developer Center. [Citado en:] <http://msdn.microsoft.com/en-gb/vstudio/bb265237.aspx>
- [12]. Ventajas frente a C/C# [En línea] Archer, Tom. A FONDO C#. Redmond: McGraw-Hill/INTERAMERICANA DE ESPAÑA, S.A.U, 2001.ISBN: 84-481-3246-7
- [13]. ASP [En línea] <http://www.subgurim.net/Articulos/asp-net-general>
- [14]. ASP .NET [En línea] ASP.NET Master Pages Overview (Microsoft Developer Network)

- [15]. XML [En línea] <http://www.desarrolloweb.com/articulos/>, Eguiluz Pérez, 25-03-2009
- [16]. JavaScript [En línea] <http://www.subgurim.net/Articulos/asp-net-general>
- [17]. González Polanco Liset y Pérez Betancourt Yadian Guillermo ¿Por qué usar ASP.NET? Análisis y Diseño del subsistema Préstamos del proyecto Modernización del Sistema Bancario Cubano. Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas, Mayo 2009
- [18]. Framework [En línea] <http://www.pedrov.info/>, <http://msdn.microsoft.com/es-es/library/zw4w595w.aspx>
- [19]. Framework .NET 3.5 [En línea]  
[http://www.carlospes.com/curso\\_de\\_ingenieria\\_del\\_software/04\\_06\\_entornos\\_integrados\\_de\\_desarrollo.php](http://www.carlospes.com/curso_de_ingenieria_del_software/04_06_entornos_integrados_de_desarrollo.php)
- [20]. Visual Studio 2008 [En línea] <http://www.msdn2.microsoft.com>
- [21]. Visual Studio 2008 [En línea]  
<http://msmvps.com/blogs/ffagas/archive/2007/08/02/overview-de-visual-studio-2008.aspx>
- [22]. Metodologías de Desarrollo de Software [Consultado en:] Jacobson Ivar, Booch Grady, Rumbaugh James. (2000).
- [23]. Proceso Unificado de Desarrollo de Software (RUP) [Citado en:] D. d. IS, "Introducción a la ingeniería de software," 2007
- [24]. Maza Capote Idalmys y Fabars Hardy Yusdelis. Programación Extrema (XP) Trabajo de Diploma Sistema para la gestión de la información de los discos contenedores de imágenes médicas (alasSMIS), Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas, Junio de 2009
- [25]. Lenguaje Unificado de Modelado 2.0 [En línea] Larman, Craig. (2004). "UML y Patrones. Introducción al análisis y diseño orientado a objetos". Félix Varela.
- [26]. Herramientas CASE. Material Básico de Bibliografía Básica de Ingeniería de Software

- [27]. Enterprise Architect [En línea]  
<http://www.thedigitalmap.com/EasyDEM/download/aide/html/documentos/robotikerWebServices.pdf>
- [28]. Visual Paradigm [Disponible en]:  
[http://www.softpile.com/Development/Editors\\_and\\_IDEs/Review\\_19078\\_index.html](http://www.softpile.com/Development/Editors_and_IDEs/Review_19078_index.html)
- [29]. Especificación de requisitos [En línea] Jacobson, Booch et al. 2004
- [30]. González Lisdany y Cruz Yordan Flujos de trabajo Análisis y Diseño. Plataforma para la Gestión de la Calidad en el Centro de Informática Médica (CESIM). . Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas, Junio 2010
- [31]. Descripción de la Arquitectura 3 Capas [En línea]  
<http://kernelerror.net/programacion/php/arquitectura-3-capas/>
- [32]. Valencia Yudisleydis y Peña Amarilis. Fundamentación del uso de patrones [En línea] Consultado Marzo 2011. Herramienta para automatizar el proceso de evaluación del programa de mejoras en los proyectos productivos de la UCI, basados en el modelo CMMI. Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas, Junio 2010
- [33]. Pérez López Yunelis y San Juan Rosabal Antonio. Modelo de implementación. Sistema para la Gestión y Análisis de Información Estadística en la salud pública cubana: Subsistema Editor de Plantillas. Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas, Junio 2009
- [34]. Valdés Rodríguez Jorge Fernando. Diagrama de Componentes [En línea] Consultado Mayo 2011. Desarrollo del Módulo de Gestión de Recursos Humanos del Sistema Integral de Gestión Administrativa de la Facultad 7. Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas, Julio 2010
- [35]. Diagrama de Despliegue. Entorno virtual de aprendizaje. Ingeniería de software 2. Clase práctica para estudiantes. UCI:s.n., 2008-2009.
- [36]. Pérez Vázquez, Maridalia y Rodríguez Hidalgo, Guillermo William. Pruebas. Desarrollo de componentes de Interfaz de usuarios para la representación de información mediante gráficas en el polo Petrosoft Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas, Junio 2010

[37]. Lopez Quesada, Juan Antonio. Ingeniería de software. Pruebas del software  
[http://www.google.com/url?sa=t&source=web&cd=6&ved=0CD0QFjAF&url=http%3A%2F%2Fdis.um.es%2F~lopezquesada%2Fdocumentos%2FTema\\_6.ppt&rct=j&q=pruebas%20de%20particion%20de%20equivalencia%20&ei=hQkBTro-JcLX0QHA3sTSBg&usg=AFQjCNFGwArbfJsKXxXt-v6Naydl4TvAvg&cad=rja](http://www.google.com/url?sa=t&source=web&cd=6&ved=0CD0QFjAF&url=http%3A%2F%2Fdis.um.es%2F~lopezquesada%2Fdocumentos%2FTema_6.ppt&rct=j&q=pruebas%20de%20particion%20de%20equivalencia%20&ei=hQkBTro-JcLX0QHA3sTSBg&usg=AFQjCNFGwArbfJsKXxXt-v6Naydl4TvAvg&cad=rja).

[38]. La prueba del software  
[http://eisc.univalle.edu.co/materias/Material\\_Desarrollo\\_Software/Pruebas.pdf](http://eisc.univalle.edu.co/materias/Material_Desarrollo_Software/Pruebas.pdf).

### Glosario de Términos

**Extensible Markup Language (XML):** Un lenguaje de marcado extensible que puede usarse para almacenar datos en un formato estructurado, basado en texto y definido por el usuario.

**Framework:** Es una estructura de soporte definida en la cual otro proyecto puede ser organizado y trabajado. Un framework puede incluir soporte de programas, bibliotecas y un lenguaje interpretado entre otros software para ayudar a desarrollar y unir los diferentes componentes de un proyecto.

**Herramienta Case:** Aplicación informática destinada a aumentar la productividad en el desarrollo de software, reduciendo el coste de las mismas en tiempo y dinero, se utiliza para el modelado del sistema.

**IDE:** Entorno Integrado de Desarrollo, es un programa compuesto por un conjunto de herramientas para un programador. Puede ser exclusivo de un lenguaje o puede ser utilizado en más de uno.