Universidad de las Ciencias Informáticas

Facultad 7

Trabajo de Diploma para optar por el Título de

Ingeniero en Ciencias Informáticas

Herramienta para el diseño de plantillas del sistema de gestión del expediente de proyecto en su variante Web

Autores: Yanicet García Acosta
Reinaldo Castillo González

Tutores: Ing. Jacqueline Marín Sánchez
Ing. Arian Segui Garcia

La Habana, junio del 2011

"Año 53 de la Revolución"

DATOS DEL CONTACTO

Tutores:

Tutor: Ing. Jacqueline Marín Sánchez

Cotutor: Ing. Arian Seguí García

Tutor principal:

Ing. Jacqueline Marín Sánchez

Graduada en Ingeniería en Ciencias Informáticas en el año 2007 en la UCI. Categoría Docente: Instructor. En cursos anteriores impartió la asignatura de Ingeniería de Software, en el presente curso impartió la asignatura de Historia de la Informática y asignaturas del perfil de Calidad. Es asesora de Calidad del CESIM.

Empresa: UCI Dirección: Carretera a San Antonio Km. 2 1/2 Reparto Torrens, Facultad No.7, La Habana.

e-mail: jmarin@uci.cu.

Co tutor:

Ing. Arian Seguí García

Profesor graduado de Ingeniero en Ciencias Informáticas en el año 2009 en la UCI. Categoría docente: Instructor Recién Graduado. Ha impartido las asignaturas de P5 y Pruebas y Evaluación de Software.

Empresa: UCI Dirección: Carretera a San Antonio Km. 2 1/2 Reparto Torrens, Facultad No.7, La Habana.

E-mail: asegui@uci.cu.

DEDICATORIA

De Yanicet:
A mis padres, Martha Acosta y Alcides García, por dedicar toda su vida al cuidado y bienestar de mis hermanos y mío.
A mis queridos hermanos, Yasel y José Miguel que representan para mi más de la mitad de mi vida.
Muy en especial a mi sobrinito Carlitos, por el amor tan inmenso que despierta en mi hacia él.
A todos mis amigos y amigas de la Universidad.

De Reinaldo:
Les dedico absolutamente esta tesis a mis padres Lidia Lina González y Pedro Castillo, gracias a ellos se hizo realidad este sueño.
A mi hermano Reinier Castillo, gracias a él yo siempre tuve fuerzas para demostrar que si podía llegar.
A todas mis amistades, a todo el que de una forma u otra ha conocido un pedacito de Rey
Agradezco a la Revolución, a la vida por darme la oportunidad de estudiar en esta maravillosa escuela, en la cual he dejado parte de mi juventud.
Al Comandante en Jefe "Fidel Castro Ruz", al Guerrillero Heroico "Ernesto Ché Guevara".

AGRADECIMIENTOS

De Yanicet:
Muy en especial a mis padres, Martha Acosta y Alcides García, por su amor incondicional, dedicación, apoyo y entrega para conmigo, por sus consejos, por confiar en mí y preocuparse por mis estudios desde siempre.
A mis queridos hermanos, Yasel y José Miguel quienes han estado presente siempre brindándome su ayuda y cariño.
Muy en especial también a Ariel, por su innegable apoyo durante la realización de este trabajo y por su ayuda incondicional durante todo este tiempo.
A mis amigos por estar en los momentos buenos y malos: Aylen León, Taisbel Guzmán, Dianela Castillo, Lisandra Estupiñan, Karel Gómez, Damián Almeida, Alexey Pérez, Lisandra García, Viviana, Eloy García, Luna, Tony, Luis, Marlon, a los chicos de Valle Grande en especial al flaco y a Felito, en fin a todos mis amigos de la universidad, que de una forma u otra han colaborado con mi formación profesional.
A Reinaldo Castillo, mi compañero de tesis, por su empeño y dedicación.
A Yurien Ricardo, que más que un profesor, ha sido un amigo incondicional y un ejemplo a seguir en mi carrera.
Y mis tutores Jacqueline y Arian, por su ayuda y guía en la realización de este trabajo y a todo el proyecto de Calidad.

De Reinaldo:
Gracias, infinitamente gracias "Mima" (la pura) por hacer de ese niño intranquilo y malcriado, este hombre que hoy soy, quiero que sepas, que en mi vida lo más especial, lo más grande eres tú, no tendría palabras para expresar todo lo que siento por ti, gracias por todo tu cariño, por todo tu amor, por toda tu entrega, gracias por hacer que este sueño se haya convertido en realidad. Te AMO con la vida, nunca lo dudes.

Gracias, ("Papi") mi puro por ese ejemplo que me has dado siempre, gracias por estar ahí cuando necesité un consejo, gracias por hacer posible que hoy sea este hombre que ves, por inculcarme tantos valores humanos , por hacerme ver que en la vida, siempre podemos seguir hacia delante , infinitamente gracias, te quiero un montón.
Bro, mi "Teti", mi hermano del alma gracias por ser ese segundo padre que siempre has sido, gracias por dedicar todo tu esfuerzo en mi educación, en mi bienestar, gracias por ser mi amigo, "mi mejor amigo", mi compañero, mi ejemplo a seguir, gracias por estar ahí cuando más lo necesité, gracias a Dios por darme la oportunidad de tener un hermano como tú, infinitamente gracias por estar siempre a mi lado.

Agradecer a mi familia , por ser la familia más linda y hermosa del mundo , mi abuelita querida "Mami" eres mi mayor tesoro, a mis primas Carla y Diana , más que mis primas son mis hermanas , a mi tío Enrique, el cual es como un padre para mí, yo soy el reflejo más pequeño de él, a mi abuelo Pedro por ser ese fabuloso abuelo que siempre ha sido ; a mis tíos Jorgito(el gordo) , Luisito, Panchito ,a mis tías Delia , Elisa , Lyli, a mi abuelo Pablo por sus resabios , a mi abuela Nena , a todo mis primitos.

Agradecer a todos mis amigos y compañeros de estudios, gracias a ellos por estar ahí cuando más me hacían falta, el Yherry mi hermanito, Alfredito, Omarito, Pochet, Yurien, Lore a la gente de la Aldea (Alberto, Kuko, Pire, el Prieto, Susy). En fin a todos los que de una forma u otra han estado a mi lado en estos cinco años de carrera.

Agradecer a todas esas personas que me han abierto su corazón, que han convivido conmigo en las buenas y en las malas, con las cuales viví momentos felices.

Agradecer al fútbol, mi deporte preferido, gracias por desestresarme cuando más lo necesité, gracias a él conocí mucha gente buena en esta vida.

Agradecer a mi compañera de tesis Yanicet por su entrega para con el desarrollo de este trabajo y por aguantarme tanto.

Agradecer a mis tutores Segui, Jacqueline por su empeño en el desarrollo de este trabajo.

A todos los que de una forma u otra colaboraron en el desarrollo de este trabajo, el Msc. Karel Osorio, a la 1er Tte. Ing. Yaima Álvarez, a mi profesores, todos mis profesores desde la primaria hasta ahora, todos han sido muy importantes en mi formación profesional.

Agradezco a la Revolución, por darme la posibilidad de vivir en este país libre y soberano, agradezco al indiscutible Comandante en jefe "Fidel Castro Ruz" y al Guerrillero heroico Ernesto "Ché" Guevara.

RESUMEN

La Universidad de las Ciencias Informáticas ha desarrollado mecanismos para mejorar el proceso de gestión de la información, para ello se confeccionó un Expediente de Proyecto (EP). En el grupo de calidad de la facultad 7 se desarrolló un sistema para gestionar la Información del Expediente de Proyecto (GIEP). A pesar de estos beneficios, GIEP no es flexible a los cambios que puedan sufrir las plantillas del EP, está basada en solo una versión del mismo; además carece de un editor de plantillas, es por ello que las mismas son estáticas y no pueden ser modificadas.

El presente trabajo de diploma tiene como objetivo desarrollar una aplicación que permita diseñar las plantillas del sistema de gestión del expediente de proyecto en su variante Web. Para el desarrollo del mismo se utilizó como metodología de desarrollo RUP, como lenguaje de modelado UML en su versión 2.1, como lenguaje de programación C#, como herramienta CASE Enterprise Architect en su versión 7.5 y como Entorno Integrado de Desarrollo (IDE) Visual Studio 2008.

Como beneficios el sistema mejorará el proceso de conformación de las plantillas del Expediente de Proyecto, proporcionará un ambiente visual amigable para el diseño de las plantillas, así como la flexibilidad a los cambios que puedan tener las plantillas del EP, además de posibilitar la modificación del diseño de las plantillas del EP.

Palabras claves: Gestión de la información, Expediente de proyecto (EP), Sistema de gestión del expediente de proyecto (GIEP), Esquemas XML.

TABLA DE CONTENIDOS

INTRODUCCIÓN

La informática ha aportado cambios significativos en la vida de la sociedad contemporánea. Su impacto a nivel global se ve reflejado en todas las esferas y actividades sociales y cada día juega un papel fundamental en la vida cotidiana del hombre.

Casi la totalidad de los trabajos, trámites, procesos que se realizan en la actualidad están vinculados a la informática, que se convierte en una nueva fuente económica para muchos países. Esto ha propiciado que los avances tecnológicos centren sus objetivos en mejorar la calidad y ampliar sus aplicaciones, principalmente en la producción de software. (1)

La interpretación de la información es actualmente una de las actividades de la informática más importante de la sociedad moderna; como consecuencia existe un alto porcentaje del trabajo cotidiano que se dedica al procesamiento y comunicación de la información. Los grandes beneficios de la informática en el campo de los sistemas de información, han hecho que su uso se masifique por todas las latitudes y que sea mayor el número de usuarios que prefieren utilizar las computadoras y las tecnologías para automatizar los procesos que involucren documentos electrónicos en su trabajo diario. (2)

El crecimiento de la documentación provoca grandes problemas, resultando necesario una planificación de su producción y un control de su gestión. En cualquier organización, una adecuada gestión es esencial para la consecución de un buen funcionamiento. La planificación de dicha gestión se convertirá en una fase básica para alcanzar una gestión correcta y conveniente; es por ello que se hace necesaria la implantación de sistemas que permitan controlar la gestión de los documentos que se generan en la entidad. Estos sistemas favorecerán establecer la organización y tratamientos adecuados a la documentación y por consiguiente una eficiente y rápida recuperación de la información.

Cuba pretende entrar al mercado mundial de desarrollo de software, es por ello que la informática se está convirtiendo en una fuente de desarrollo económico-social y para esto se viene realizando un trabajo significativo en materia de la informatización de la sociedad. La formación y capacitación de nuevos profesionales en las diferentes ramas de la informática constituyen un eslabón fundamental en el desarrollo

de la industria del software a nivel nacional, pero para ello tiene que lograr una mayor calidad en sus productos, ya que el exceso de información constituye uno de los principales problemas. Es frecuente encontrar un gran número de información redundante e innecesaria mezclada con la que es realmente importante.

La Universidad de las Ciencias Informáticas (UCI), creada en el año 2002 en el marco de la Batalla de Ideas, con el objetivo de ayudar e impulsar el desarrollo de software en el país, se ha convertido en un centro por excelencia de referencia en la industria de software cubano. Con su poco tiempo de creación ya se han desarrollado una serie de normativas rectoras del proceso de desarrollo de software, todos con buenos resultados, por lo que constituyen un reto en cuanto a organización y calidad en el proceso de producción. Con más calidad en los productos, se logrará una mayor flexibilidad, corrección, seguridad e integridad en el software.

La esfera productiva de la Universidad está estructurada por centros de desarrollo, estos a su vez se dividen en departamentos que agrupan proyectos afines a una temática. En sus inicios, la información que generaban los proyectos productivos, como parte del proceso de desarrollo de software, presentaban diferentes deficiencias, era desorganizada, descentralizada y escasa, lo cual no garantizaba la total calidad del proceso y el producto. Por este motivo la dirección de calidad de la UCI propone el uso de un Expediente de Proyecto (EP) para estandarizar y agrupar la documentación.

La confección del EP resultaba trabajosa por la cantidad de información que exigía y el tiempo que requería su elaboración, además se registraba información redundante e innecesaria. Como consecuencia se afectaba la calidad y entrega en tiempo de la misma.

Con el objetivo de solucionar estos problemas, en el grupo de calidad de la facultad 7 se desarrolló un sistema para Gestionar la Información del Expediente de Proyecto (GIEP), el cual elimina el exceso de información, ahorra tiempo en la confección de los documentos y brinda fácil acceso a la información.

La herramienta permite:

- Eliminar el exceso de información recurrente.
- Ahorrar tiempo en la confección de los documentos.
- Capturar y gestionar imágenes.
- Rapidez en el acceso a la información del EP.
- Facilitar la revisión de los documentos para el equipo de desarrollo y el equipo de calidad.

A pesar de estos beneficios, GIEP no es flexible a los cambios que puedan sufrir las plantillas del EP, está basada en solo una versión del mismo; trayendo como consecuencia que al variar la versión del EP se podrán incrementar el número de plantillas, así como algunas ya no serán necesarias; por lo que no se puede controlar el cambio. Además carece de un editor de plantillas, es por ello que las mismas son estáticas y no pueden ser modificadas.

En la actualidad se está desarrollando una plataforma Web como variante de GIEP que permitirá:

- Centralizar el trabajo en el EP.
- Proporcionar un control de versiones robusto.
- Flexibilidad a los cambios que pueda sufrir dicho EP.
- La personalización de la estructura del EP y de las reglas de acceso a los documentos.
- Editar vistas de impresión.

Para garantizar un mejor funcionamiento del sistema se necesita que la creación de las plantillas sea ágil, sencilla y flexible al cambio. Por lo expuesto anteriormente se define como problema a resolver: ¿Cómo facilitar el diseño de las plantillas del Expediente Proyecto para el uso de la herramienta GIEP en su variante Web?; teniéndose como **objeto de estudio:** el diseño de plantillas de documentos, enmarcando el **campo de acción:** en el diseño de las plantillas del Expediente del Proyecto en la UCI. Para darle solución a la problemática planteada se define como **objetivo general**: desarrollar una herramienta para el diseño de las plantillas del sistema de gestión del expediente de proyecto en su variante Web.

Con vista a darle cumplimiento al objetivo planteado se han definido las siguientes tareas investigativas:

- Analizar los diferentes sistemas que existen para el diseño de plantillas de documentos.
- Evaluar las tecnologías y librerías existentes que puedan ser usadas para elaborar la herramienta en el trabajo.
- Seleccionar Metodología, lenguaje de programación, plataforma y Entorno Integrado de Desarrollo (IDE) para elaborar la herramienta dentro de los establecidos por el CESIM (Centro de Informática Médica).
- Elaborar la documentación correspondiente a los Flujos de Trabajo propuestos por la Metodología seleccionada.
- Implementar la herramienta que de solución al problema planteado.
- Desarrollar pruebas funcionales al sistema obtenido.

Como resultado de la investigación se espera obtener una herramienta capaz de:

- Proporcionar un ambiente visual amigable para el diseño de las plantillas.
- Agilizar y mejorar el proceso de conformación de las plantillas del Expediente de Proyecto.
- Proporcionar flexibilidad a los cambios que puedan tener las plantillas del EP.
- Modificar el diseño de una plantilla del EP.

El presente trabajo de diploma está estructurado de la siguiente forma:

Capítulo 1: Este capítulo describe las condiciones actuales del diseño de plantillas de documentos, describiendo las herramientas existentes, tanto a nivel mundial como nacional. Se realiza una comparación entre tecnologías, con vista a la selección de las más apropiadas para el desarrollo de la solución propuesta.

Capítulo 2: En este capítulo se especifican los requisitos funcionales y no funcionales del sistema. Se desarrolla el Modelo de caso de uso del sistema.

Capítulo 3: En este capítulo queda plasmado el Modelo del diseño de la aplicación.

Capítulo 4: En este capítulo se describe la arquitectura correspondiente a la implementación perteneciente al sistema, se muestra el Modelo de implementación del sistema. Quedan plasmados los resultados de las pruebas realizadas.

CAPÍTULO 1. **FUNDAMENTACIÓN TEÓRICA**

En la actualidad no existe una homogeneidad en la documentación que se construye por parte de los proyectos, lo que dificulta la retroalimentación que pudiera obtenerse de la misma. Para poder lograr mejores productos en el proceso de desarrollo de software se hace necesario que la gestión de documentos generados se realice con mayor eficiencia y calidad. Es indispensable registrar y documentar todo lo que sucede en cada fase del proceso, de ello dependerá el éxito de cada proyecto. De esta forma se facilitará al cliente y al propio equipo de desarrollo conocer la situación en cada momento del avance obtenido. Además se podrán gestionar transformaciones en el producto de una manera eficiente y rápida sin la necesidad de afectar otras fases del desarrollo de software, aunque exista una relación entre cada una de ellas.

Al inicio de la creación de la UCI, como parte de los mecanismos que permitan mejorar la calidad de los procesos de desarrollo de software, estandarizar la documentación de todos los proyectos que se realizan, facilitar la formación y adiestramiento de los equipos de proyecto en el uso de modelos y estándares propios; se diseña el Expediente de Proyecto (EP). Este ya ha sido implantado en cada uno de los proyectos vigentes en la universidad, el mismo se confecciona para registrar de forma documental el proceso de desarrollo de software dentro de los proyectos productivos. En él se gestiona toda la información referente al proceso de elaboración de un software, así como la información referente al propio proyecto. El EP está compuesto por plantillas que no es más que una forma de dispositivo que proporciona una separación entre la forma o estructura y el contenido. Es un instrumento que permite guiar, portar o construir un diseño o esquema predefinido.

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: 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)

El desarrollo de un software requiere de una amplia documentación y la conformación de numerosas plantillas. En la universidad la creación de estas plantillas se realiza manualmente, lo cual provoca que las mismas sean vulnerables al cambio, volviéndose inestable la integridad de las plantillas. Esto resulta engorroso en el proceso de confección de las mismas. Es por eso que surge la necesidad del desarrollo de una herramienta que diseñe plantillas del expediente de proyecto, lo cual favorecerá a mejorar el proceso de conformación de las plantillas del EP.

1.1. Algunas herramientas existentes

Para el desarrollo del sistema se realiza un estudio de algunas de las herramientas existentes, para analizar su funcionamiento con respecto a la problemática planteada y comprobar si son factibles para el presente trabajo de diploma.

1.1.1. Ámbito Internacional

Microsoft InfoPath: La principal característica de InfoPath es la capacidad de poder crear y ver documentos XML con soporte para XML Schema. Es una aplicación usada para desarrollar formularios de entrada de datos basados en XMLEl InfoPath 2010, es un programa de agrupación de información que utiliza formularios electrónicos implementados mediante correo. Combina el ambiente del sistema de Office Microsoft con un conjunto de herramientas que le permiten a las organizaciones armar formularios de solución que ayuden a validar la información, integrarla a los sistemas de información y guiar a los usuarios completando el formulario. Puede ser de gran ayuda para reducir la información repetida e ineficiente mientras se mejora la colaboración y la toma de decisiones. La instalación de este producto requiere clave de licencia y medios digitales. No brinda ayuda al trabajo de diploma porque no posee un editor de plantillas. (4)

Typefi: Integrado con el software Adobe Indesign Typefi NLM es un escritorio de usuario; producto que entrega de forma automatizada el diseño altamente flexible, la composición y la paginación de los documentos de cualquier longitud a un costo asequible. Permite a los editores utilizar el contenido XML. Tiene las funciones y herramientas necesarias para producir revistas y artículos con Adobe Indesign. Incluye

la capacidad de generar tablas, el tamaño y lugar de las cifras y la inserción de las líneas. También tiene la característica de forma automática de alinear las columnas de la tabla. El precio de Typefi NLM, incluye el costo de creación de la plantilla de Indesign. (5)

AltovaXMLSpy: Algunas de las características más destacables de **AltovaXMLSpy** son:

- Validación del código XML.
- Visualización de los esquemas en XML.

AltovaXMLSpy es un editor de XML completo para modelar, editar, transformar y depurar las tecnologías XML. Proporciona todas las funciones que necesita para desarrollar códigos XML más complejos y aplicaciones Web. El mismo es compatible con Windows XP, Windows 7 y Windows server 2003-2008. Para utilizar el software se debe acceder a la tienda Online para adquirir una licencia. Altova no proporciona una interfaz amigable al usuario para el diseño de plantillas. (6)

1.1.2. Ámbito Nacional

En el país existen diversos software que emplean esquemas XML para sus fines particulares, pero no se encontró un software que resuelva la problemática de diseñar plantillas del expediente de proyecto.

Generación de una biblioteca de Objetos de aprendizaje a partir de contenidos preexistentes: La propuesta se basa en la creación de procesos que permitan la conversión de contenidos y materiales de la plataforma origen en objetos de aprendizaje reusables en otras plataformas, para lo cual se apoya en los estándares. Las especificaciones del estándar están definidas mediante plantillas con una estructura XML que contienen la información del modelo, así como de la estructura y el contenido de los objetos. Estrategia de interoperabilidad para la transferencia de datos entre sistemas ERP en Cuba: es una estrategia de

interoperabilidad[1] para sistemas ERP[2]que generalice y rija metodológicamente el proceso de interoperabilidad en Cuba y permita solucionar los problemas asociados al intercambio de información. XML se utiliza como lenguaje intermedio para intercambiar información (obtenerla o servirla). Una vez que se dispone de la información, se puede realizar una aplicación para gestionar y mantener dichos datos, como puede ser una aplicación Web interna. (7)

1.1.3. Análisis Crítico de las herramientas

Una vez realizado el análisis de las herramientas más acordes a las necesidades planteadas, se puede concluir que las mismas no responden a las necesidades requeridas porque:

- Muchas de ellas son propietarias.
- Los costes de soporte técnico y mantenimiento son muy elevados.
- No presentan editores de plantillas.
- No poseen una interfaz amigable al usuario para el diseño de las mismas.

1.2. Tecnologías

Para el desarrollo del sistema se realiza un análisis de las herramientas y tecnologías existentes, para posteriormente seleccionar las más factibles de acuerdo a la compatibilidad de las mismas con las características del sistema.

UML (Lenguaje Unificado de Modelado)

[1]Interoperabilidad: La disponibilidad de mecanismos que permitan intercambiar procesos y/o datos entre sistemas heterogéneos.

[2]ERP: Sistemas de Planificación de Recursos Empresariales.

El Lenguaje Unificado de Modelado (UML) es un lenguaje de modelado visual que se usa para especificar, visualizar, construir y documentar artefactos de un sistema de software. Captura decisiones y conocimiento sobre los sistemas que se deben construir. Se usa para entender, diseñar, hojear, configurar, mantener, y controlar la información sobre tales sistemas. Está pensado para usarse con todos los métodos de desarrollo, etapas del ciclo de vida, dominios de aplicación y medios. El lenguaje de modelado pretende unificar la experiencia pasada sobre técnicas de modelado e incorporar las mejores prácticas actuales en un acercamiento estándar.

UML incluye conceptos semánticos, notación, y principios generales. Tiene partes estáticas, dinámicas, de entorno y organizativas. Está pensado para ser utilizado en herramientas interactivas de modelado visual que tengan generadores de código así como generadores de informes. La especificación de UML no define un proceso estándar pero está pensado para ser útil en un proceso de desarrollo iterativo. Pretende dar apoyo a la mayoría de los procesos de desarrollo orientados a objetos.

Ventajas

- Es un lenguaje conocido.
- Estándar.
- Fácil de aprender.

Desventajas

- No ha sido diseñado para modelar procesos de negocios.
- Implica un enfoque orientado a objetos.
- UML no tiene todavía una semántica formal. (8)

1.2.1. BPMN

BPMN[3] es una notación gráfica para el modelado conceptual de procesos de negocio. Proporciona la capacidad de entender y definir procesos de negocio, tanto internos como externos, a través de un diagrama de procesos de negocio. Es un estándar de notificación que provee una representación gráfica (mediante diagramas) para expresar procesos de una empresa. Se diseñó con el objetivo de facilitar la comprensión por parte de todos los implicados (expertos TIC, analistas de negocio, directivos, etc.) que participan en el proceso. (8)

BPMN toma un perfil orientado a procesos en el modelado de sistemas. Este se está convirtiendo rápidamente en el lenguaje de elección para tratar la complejidad de los entornos SOA (arquitectura orientada a servicios). BPMN se creó con notaciones especiales para representar eventos que se ejecutan al recibir un mensaje y los flujos de datos (mensajes) entre las organizaciones y aplicaciones. Esta notación facilita la orquestación de servicios y permite la implementación de una arquitectura empresarial flexible, interoperable y ágil. BPMN crea un puente estándar para unir la brecha existente entre el diseño de proceso de negocio, la comunicación de requerimientos de negocio y la implementación de procesos. (9)

Ventajas

- Considera un único diagrama para la representación de los procesos (BPD, diagrama de procesos del negocio).
- Pensado para ser asignado con naturalidad a lenguajes de ejecución.

[3]BPMN: Notación de modelado de un proceso de negocio

- Fácil de entender. (10)

Desventajas

- Existe un grado importante de ambigüedad en el proceso de negocio modelado.
- No se utilizan todas las potencialidades del modelamiento necesarias para disminuir el grado de ambigüedad.
- El cliente no garantiza el cumplimiento de los estándares mínimos de modelado de proceso. (11)

1.2.2. Selección del lenguaje de modelado a utilizar

Para el desarrollo del sistema se usará UML v 2.0, pues permite modelar el sistema de manera clara y entendible para el equipo de desarrollo. Además permite modelar todo el ciclo de desarrollo de software, los cuales ahorrarán tiempo de trabajo a los programadores.

1.2.3. Herramientas CASES

1.2.3.1 Visual Paradigm

Visual Paradigm para UML 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 ayuda a una construcción más rápida 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, así como una serie de tutoriales, demostraciones interactivas y proyectos. (12)Es importante señalar que la herramienta de modelado Visual Paradigm no es gratuita, pero la compañía Visual Paradigm UML Comunity, tiene disponible distintas versiones y facilita licencias especiales para fines académicos, sin interés de lucro. (13)

1.2.3.2 Rational Rose

Es una de las más poderosas herramientas CASE de modelado visual para el análisis y diseño de sistemas basados a objetos. Se utiliza para modelar un sistema antes de proceder a construirlo. Cubre todo el ciclo de vida de un proyecto.

Rational Rose tiene las características siguientes:

- Característica de control por separado de componentes, modelo que permite una administración más granular y el uso de modelos.
- Soporte de ingeniería Forward y/o reversa para algunos de los conceptos más comunes de Java 1.5.
- Capacidad de análisis de calidad de código.
- Publicación Web y generación de informes para optimizar la comunicación dentro del equipo.

Desventajas:

- Necesidad de alta capacidad de procesamiento.
- Existen varios lenguajes de programación que no soporta o que sólo lo hace a medias. (14)

1.2.3.3 Enterprise Architec 7.5

Enterprise Architect (EA) es una herramienta CASE (Computer Aided Software Engineering) 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.

- Realizar ingeniería inversa del código existente en varios lenguajes.

La documentación de alta calidad puede ser rápidamente exportada desde sus modelos en industria estándar a 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, etc. Posibilita 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 iníciales hasta el mantenimiento y soporte, Enterprise Architect tiene las características que precisa para diseñar y administrar su desarrollo e implementación. *(15)*

1.2.4. Selección de la herramienta CASE a utilizar

Enterprise Architect 7.5 se empleará porque es una herramienta 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 característica excelentes como realizar la trazabilidad y soporta diferentes lenguajes como Java, C#, C++ y VB.NET.

1.3. Metodologías de desarrollo de software

1.3.1. XP [4]

La Programación Extrema (XP), mejor conocida por su nombre en inglés Extreme Programming , es una de las llamadas Metodologías Ágiles de desarrollo de software más exitosa de los tiempos recientes. Esta se basa en la simplicidad, la comunicación y la reutilización del código desarrollado. Es utilizada en proyectos con equipos de desarrollo pequeños y con plazos de entrega corto, de manera que se pueda llevar a cabo

[4]XP: Programación Extrema

una programación rápida o extrema. Una particularidad que tiene, es que el usuario final debe ser miembro del equipo.

La programación extrema (XP) se basa en varios valores fundamentales:

- La comunicación: requiere de una amplia comunicación entre clientes y desarrolladores.
- La simplicidad: al desarrollar y codificar los módulos del sistema. Es mejor hacer hoy algo simple, que hacerlo complicado y probablemente no usarlo mañana.
- La retroalimentación: concreta y frecuente del equipo de desarrollo, el cliente y los usuarios finales. XP ofrece una mayor oportunidad de dirigir el esfuerzo eficientemente. (16)

Cuando usar XP:

- Cuando los clientes no tienen ideas claras de los requerimientos y los van cambiando.
- Para proyectos de riesgo: fecha fija de entrega, algo nunca hecho por el grupo, algo nunca hecho por la comunidad de desarrolladores.
- No es apto para proyectos con mucho personal (entre 2 y 10 programadores).
- Integra gerentes y clientes a la formulación de preguntas, la negociación de cronogramas y alcance, la creación de las pruebas.
- Automatiza las pruebas; es posible en casi todos los dominios. Es lícito repensar el diseño para facilitar el ensayo.
- El objetivo es entregar el software tal cual se necesita y en el momento en que se necesita. Incidentalmente, los proyectos XP muestran mayor productividad. (17)

1.3.2. SCRUM

El Scrum es un modelo de referencia, que define un conjunto de prácticas y roles, que pueden tomarse como punto de partida para definir el proceso de desarrollo que se ejecutará durante un proyecto. El resultado final en esta metodología se consigue de forma iterativa e incremental. Al comienzo de cada

iteración (sprint) se determina que partes se van a construir, tomando como criterios la prioridad para el negocio, y la cantidad de trabajo que se podrá abordar durante la iteración.

Está especialmente indicada para proyectos con un rápido cambio de requisitos. SCRUM entiende el desarrollo de software como algo muy complejo, sujeto a multitud de variaciones e incertidumbres, sobre todo al comienzo. También trata de que todos los involucrados en el proyecto conozcan perfectamente en qué punto del desarrollo se encuentra el proyecto y qué falta por hacer. Es ideal para proyectos con un constante cambio de requerimientos. (18)

Trabajar bajo la metodología SCRUM ofrece las siguientes ventajas:

Ventajas:

- Ofrece la posibilidad de ajustar la funcionalidad en base a la necesidad de negocio del cliente.
- Brinda la visualización del proyecto día a día.

Desventajas:

- No genera toda la evidencia o documentación de otras metodologías.
- No es apto para todos los proyectos.
- Es necesario complementarlo con otros procesos (XP).

1.3.3. RUP

Las siglas RUP[5] en inglés significa Rational Unified Process (Proceso Unificado Racional) es un producto del proceso de ingeniería de software que proporciona un enfoque disciplinado para asignar tareas y

[5]RUP: Proceso unificado de desarrollo.

responsabilidades dentro de una organización del desarrollo. Su meta es asegurar la producción del software de alta calidad que resuelve las necesidades de los usuarios dentro de un presupuesto y tiempo establecido.

Presenta como características esenciales:

En RUP las actividades de desarrollo están dirigidas por los casos de usos, esto se refiere a la utilización de los Casos de Usos para el desenvolvimiento y desarrollo de las disciplinas con los artefactos, roles y actividades necesarias.

RUP es un proceso iterativo e incremental que plantea la implementación del proyecto a realizar en iteraciones; con lo cual se pueden definir objetivos por cumplir en cada iteración y así poder ir completando todo el proyecto iteración por iteración. Como parte del enfoque iterativo se encuentra la flexibilidad para acomodarse a nuevos requisitos o a cambios tácticos en los objetivos del negocio.

El desarrollo bajo el Proceso Unificado está centrado en la arquitectura. Este proceso se centra en establecer una arquitectura de un sistema y una arquitectura ejecutable construida como un prototipo evolutivo. Este diseño arquitectónico sirve como una sólida base sobre la cual se puede planificar y manejar el desarrollo de software basado en componentes.

RUP divide el proceso en cuatro fases:

- *Inicio*: en esta primera fase se define el ámbito y objetivos del proyecto, así como las funcionalidades y capacidades del producto.
- *Elaboración*: en esta fase se define una arquitectura básica y se planifica el proyecto considerando recursos disponibles.
- *Construcción*: esta fase proporciona un producto construido junto con la documentación basada en la arquitectura de la línea base.
- *Transición*: se libera el producto y se entrega al usuario para un uso real.

En RUP se han agrupado las actividades en grupos lógicos definiéndose 9 flujos de trabajo principales. Los 6 primeros son conocidos como flujos de ingeniería y los tres últimos como de apoyo, los cuales son Modelamiento del Negocio, Requerimientos, Análisis y Diseño, Implementación, Prueba o Testeo, Instalación, Administración de Configuración y Cambios y Ambiente.

Una particularidad de esta metodología es que, en cada ciclo de iteración, se hace exigente el uso de artefactos, siendo por este motivo, una de las metodologías más importantes para alcanzar un grado de certificación en el desarrollo del software. (19)

1.3.4. Selección de la metodología a utilizar

Se realizó un estudio de las metodologías más usadas para el desarrollo del software y de acuerdo con las características y complejidad de este sistema, se decide trabajar con RUP. Esto le permite a los desarrolladores entender con profundidad las funcionalidades que deben implementar. RUP genera una documentación detallada de todo el proceso de software desde sus inicios, a diferencia de metodologías ágiles como XP y SCRUM.

RUP resulta más adaptable para los proyectos a largo plazo que necesiten un desarrollo iterativo capaz de mantener un buen control sobre los cambios, además detecta la mayor cantidad de riesgos en las primeras fases y se basa en las mejores prácticas de la Ingeniería de Software.

1.4. Lenguajes de Programación

1.4.1. C#

C# es un lenguaje de programación orientado a objetos desarrollado y estandarizado por Microsoft como parte de su plataforma .NET. Aunque es posible escribir código para la plataforma .NET en muchos otros lenguajes, C# es el único que ha sido diseñado específicamente para ser utilizado en ella, por lo que programarla usando C# es mucho más sencillo e intuitivo que hacerlo con cualquiera de los otros lenguajes ya que C# carece de elementos heredados innecesarios en .NET. Por esta razón, se suele decir que C# es el lenguaje nativo de .NET. (20)

1.4.2. C++

Fue diseñado para ser compatible con el anterior C (de manera que todos los programas escritos en C pudieran ser también programas C++ y pudieran ser compilados con un compilador de C++). La principal aportación a C++ fue la compatibilidad para el nuevo concepto de objeto. C++ incorporó compatibilidad para clases (que son "plantillas" de objetos) y permitió que toda una generación de programadores de C pensaran en términos de objetos y su comportamiento. C++ puede dar como resultado aplicaciones muy potentes, pero debe asegurarse de que el código funciona bien. Un error en la escritura del programa puede hacer que toda la aplicación falle o se comporte de forma inesperada. Como el objetivo al diseñar C++ era retener la compatibilidad con el anterior C, C++ fue incapaz de escapar de la naturaleza de bajo nivel de C.

Desventajas.

- Puede ser difícil de manejar.
- Lenguaje de muy "bajo nivel"
- Exigen que mucho código para funcionar correctamente.
- Tiene que escribir su propio código para manejar aspectos como la gestión de memoria y el control de errores. (21)

1.4.3. Java

Este es un lenguaje desarrollado por la compañía Sun Microsystem en los años noventa. Está inspirado en C++ y se proyectó con la finalidad de obtener un producto de pequeñas dimensiones, simple y portátil sobre diferentes plataformas y sistemas operativos ya sea a nivel de código fuente como a nivel de código binario. Su sintaxis ha sido trabajada mejorando la de C++ logrando mayor sencillez y legibilidad. Presenta mayor robustez al simplificar la gestión de memoria y eliminar las complejidades del manejo explícito de punteros. Presenta capacidades avanzadas de ejecución multi-hilo y proporciona mecanismos de carga dinámica de clases en tiempo de ejecución. Se puede compilar y ejecutar en cualquier plataforma de sistema operativo por ejemplo en Windows, Solaris o Linux gracias a su máquina virtual. (22)

1.4.4. Selección del lenguaje de programación a utilizar

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 se enriquece con la potencia de C++ y la sencillez y modernidad de Java. Ha alcanzado gran auge y se utiliza para desarrollar gran cantidad de aplicaciones desde mediano a gran tamaño. Está debidamente estandarizado. Se usa en múltiples grupos de proyecto en la Facultad 7 de la UCI. Resulta, por tanto, el adecuado para el desarrollo de la aplicación.

1.5. Entorno Integrado de Desarrollo (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.

Así, por ejemplo, en el caso del lenguaje C se necesita: un editor, un preprocesador, un compilador y un enlazador.

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. (23)

1.5.1. Visual Studio 2008

Visual Studio 2008 permite incorporar características del nuevo WPF[6]sin dificultad tanto en los formularios de Windows existentes como en los nuevos. Visual Studio 2008 permite mejorar la interoperabilidad entre

[6] WPF: Windows Presentation Fundation. Tecnología de Microsoft, presentada como la parte de Windows Vista.

código nativo y código manejado por .NET. Esta integración más profunda simplificará el trabajo de diseño y codificación.

Visual Studio 2008 provee una serie de herramientas para desarrollo, así como características de depuración, funcionalidad en base de datos y características innovadoras para la creación de aplicaciones en una variedad de plataformas. Incluye realces como un diseñador visual para desarrollo rápido con el .NET Framework 3.5, esto ayuda mucho a los que desarrollan en Web por que se incluyen las características de Microsoft Expression Web. Visual Studio 2008 provee a desarrolladores con todas las herramientas y el framework para poder crear aplicaciones Web con el soporte de AJAX.[7]

Visual Studio 2008 ofrece a desarrolladores nuevas herramientas para la fácil creación de aplicaciones conectadas en las ultimas plataformas, incluyendo Web, Windows Vista, Office 2007, SQL Server 2008 y Windows Server 2008. Para la Web, se tiene ASP.NET, AJAX y otras tecnologías como Silverlight, WPF, etc. que dará la posibilidad de crear aplicaciones con rica interfaz de usuario, para poder dar una experiencia de usuario sin precedentes.

Visual Studio 2008 provee un nuevo lenguaje de consultas integrado para el manejo de la información, llamado Microsoft Language Integrated Query (LINQ), que es lo que hará la vida más fácil para programadores individuales para poder construir soluciones que analicen y actúen sobre la información. (24)

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) (25)

1.5.2. NetBeans 3.5

[7] AJAX: Asynchronous Java Script And XML. Técnica de desarrollo Web

NetBeans es un entorno de desarrollo, hecho principalmente para el lenguaje de programación Java. La Plataforma NetBeans es una base modular y extensible usada como una estructura de integración para crear aplicaciones de escritorio grandes. Empresas independientes asociadas, especializadas en desarrollo de software, proporcionan extensiones adicionales que se integran fácilmente en la plataforma y que pueden también utilizarse para desarrollar sus propias herramientas y soluciones. Es una herramienta para programadores pensada para escribir, compilar, depurar y ejecutar programas. NetBeans IDE es un producto libre y gratuito sin restricciones de uso. (26)

1.5.3. Selección del IDE a utilizar

Se usa Visual Studio 2008 porque permite a los desarrolladores crear rápidamente aplicaciones conectadas con sistemas ya existentes en los proyectos sin importar la plataforma que sobre la cual estén creadas, brindándole al mismo tiempo a estas aplicaciones de alta calidad al momento de crear la capa de presentación mejorando la experiencia del usuario. Desde el punto de vista del negocio se cuenta con aplicaciones amigables para el usuario al final de forma rápida y al estar conectadas a toda la organización permitirá tomar decisiones económicas en corto tiempo.

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.

1.6. Plataforma de desarrollo

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 usan para implementar la estructura de un modelo de 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.

.NET Framework 3.5 amplía la compatibilidad con aplicaciones móviles distribuidas al incorporar la tecnología WCF que es el modelo de programación unificado de Microsoft para generar las aplicaciones orientadas a servicios. También agrega nuevas características de lenguaje como LINQ, incluye nueva interfaz de programación de aplicaciones(API) basadas en los comentarios de la comunidad y mejora la depuración con herramientas y características de diagnóstico actualizadas. Language-Integrated Query (LINQ) agrega funciones de consulta de uso general a .NET Framework 3.5que se aplican a diferentes orígenes de información, como bases de datos relacionales, datos XML y objetos en memoria.

.NET Compact Framework 3.5 admite el generador de perfiles de CLR[8], que sólo estaba disponible con la versión completa de .NET Framework. El generador de perfiles de CLR permite ver el montón administrado de un proceso e investigar el comportamiento del recolector de elementos no utilizados. El generador de perfiles de CLR y su documentación asociada están incluidos en las herramientas avanzadas de .NET Framework 3.5. Admite además la herramienta de configuración, que proporciona información sobre la versión del motor en tiempo de ejecución y funciones administrativas.

.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 Script Manager y un control Update Panel, de modo que la página pueda actualizarse sin que sea necesario realizar una actualización de la página completa. (27)

En este capítulo se realizó un estudio acerca de las herramientas y tecnologías existentes actualmente, realizándose comparaciones entre ellas y según las características presentadas por las mismas y la compatibilidad con el sistema, se escogió como metodología de desarrollo RUP, como lenguaje de

8 CLR: componente de máquina virtual de la plataforma .Net de Microsoft

modelado se utilizará UML v 2.0. Para lograr un mejor entendimiento en el modelo del mismo se utilizará Enterprise Architect 7.5, así como para su desarrollo se escogió Visual Studio 2008 acompañado del Framework.Net 3.5.

CAPÍTULO 2. **CARACTERÍSTICAS DEL SISTEMA**

En este capítulo se realiza un análisis a partir de los resultados arrojados de la investigación previa, donde es necesaria la identificación de los conceptos asociados al Modelo de dominio, para definir los requisitos funcionales y no funcionales. Además de un prototipo de interfaz externa. Se definirá el Modelo del diseño. Y finalmente quedarán definidos los procesos que se automatizarán en la aplicación.

2.1. Objeto de automatización

Debido a los cambios que se exponen los documentos del EP, es de gran importancia que el diseño de las plantillas utilizadas por GIEP, se genere de forma dinámica; es por ello que se necesita la automatización del proceso de diseño y generación de las plantillas del EP.

2.2. Propuesta del sistema

Se propone el desarrollo de un sistema para el diseño de plantillas del Expediente de Proyecto. La herramienta contará con un panel de componentes, y un panel de propiedades de dichos componentes. Además brindará la posibilidad de diseñar cualquier plantilla del EP, la cual estará conformada por la elección de los componentes necesarios para la misma y posteriormente generará el XML de dicha plantilla.

2.3. Modelo del dominio

Modelo de dominio o Modelo conceptual, es una representación visual de los conceptos u objetos del mundo real significativos para un problema o área de interés. Define un modelo de clases común para todos los implicados en el desarrollo, representadas en objetos del dominio, además sirve como interlocutor entre clientes y desarrolladores. El propósito fundamental de este modelo es generar una terminología común y sentar las bases del entendimiento del desarrollo y no para definir el sistema completo. Cualquiera que sea la solución de casos de uso que se haya elegido, los conceptos e ideas propias del dominio del problema son las mismas. Un mismo Modelo de dominio contempla cualquiera de las soluciones analizadas. Además es global, es decir, se realiza para todos los casos de uso y no para uno en particular. (28)

Al mismo tiempo se logran identificar los siguientes conceptos:

GIEP: herramienta para la gestión de la información del expediente de proyecto**.**

Plantilla XML: es un archivo o fichero utilizado por la herramienta GIEP para la elaboración de un estándar, que carga y guarda información.

Documento: es el artefacto que se genera en el proceso de desarrollo de software que se recoge en el Expediente de Proyecto (EP).

Partiendo de la descripción anterior, el Modelo de dominio se estructuró de la siguiente forma:

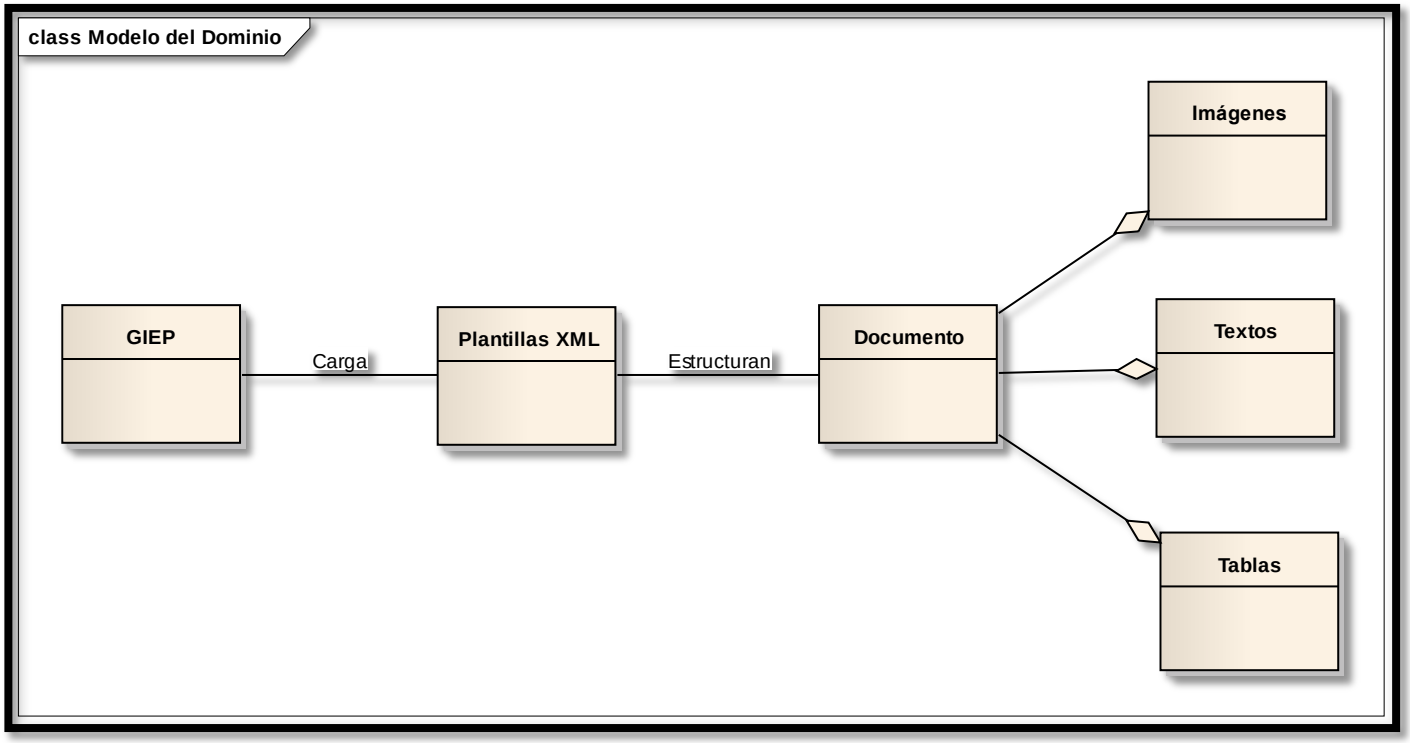


Figura # 1: Modelo del dominio

2.4. Especificación de requisitos de software

El objetivo fundamental del Flujo de trabajo es guiar el desarrollo al sistema correcto, lo que se consigue mediante una descripción de los requisitos de la aplicación, es decir, condiciones o capacidades que el sistema debe cumplir. Los requisitos deben ser especificados por escrito, descritos como una característica del sistema a construir y posibles de probar o verificar. (29)

2.4.1. Requisitos funcionales

Son capacidades o condiciones que el sistema debe cumplir sin alterar la funcionalidad del producto, esto quiere decir que se mantienen invariables sin importar con qué propiedades o cualidades se relacionan. Su principal tarea consiste en la generación de especificaciones correctas que describan con claridad, sin ambigüedades, en forma consistente y compacta, el comportamiento del sistema. (29)

Una vez conocido los conceptos que rodean al objeto de estudio, se analiza ¿Qué debe hacer el sistema para que se cumplan los objetivos planteados al inicio del trabajo de diploma?, enumerándose a través de requisitos funcionales las instrucciones que el sistema debe ser capaz de efectuar. Dentro de ellos se incluyen las acciones que podrán ser ejecutadas por el usuario y las acciones ocultas que debe realizar el sistema.

A continuación se describen los requisitos funcionales que debe cumplir el sistema.

RF1 Diseñar plantilla expediente proyecto.

- **RF1.1** Insertar componentes.
- **RF1.2** Mover/Cortar componentes.
- **RF1.3** Copiar componentes.
- **RF1.4** Eliminar componentes.

RF2 Gestionar documento

- **RF2.1** Crear documento.
- **RF2.2** Modificar documento.
- **RF2.3** Eliminar plantilla.

RF3 Gestionar propiedades de componentes.

- **RF3.1** Modificar propiedades de componentes.
- **RF3.2** Visualizar propiedades de componentes.

RF4 Generar plantilla XML.

- **RF4.1** Salvar XML.
- **RF4.2** Cargar XML.

2.4.2. Requisitos no funcionales

Los requisitos no funcionales son características requeridas del sistema, del proceso de desarrollo, del servicio prestado o de cualquier otro aspecto del desarrollo, que señala una restricción del mismo; que se aplican de manera general como un todo, más que a rasgos particulares del mismo. (30)

Estos requerimientos son importantes para que clientes y usuarios puedan valorar las características no funcionales del producto, pues si se conoce que el mismo cumple con todas las funcionalidades requeridas, las propiedades no funcionales pueden marcar la diferencia entre un producto bien aceptado y uno con poca aceptación.

2.5.2.1 Usabilidad

Estos requerimientos describen los niveles apropiados de usabilidad, dados los usuarios finales del producto, para ello deben revisarse las especificaciones de los perfiles de usuarios y definirlas clasificaciones de sus niveles de experiencia.

RNF1 Las interfaces de la aplicación deberán ser intuitivas para el usuario, típico de herramientas de escritorio.

RNF2 Contará con una *Ayuda* que permitirá el aprendizaje sobre el uso de la aplicación para un mejor entendimiento de la misma.

2.5.2.2 Apariencia o Interfaz Externa

Requerimiento que describe la apariencia del producto. Es válido destacar que no se trata del diseño de la interfaz en detalle, sino que especifican cómo se pretende que sea la interfaz externa del producto.

RNF3 Una herramienta sencilla, fácil de usar, presenta las funcionalidades explícitas, apoyándose en una barra de herramientas con íconos intuitivos y elementos organizados a ambos lados (panel de componentes y panel de propiedades).

2.5.2.3 Hardware

Estos requerimientos especifican las características lógicas para cada interfaz entre el producto y los componentes de hardware del sistema.

RNF4 Requerimientos mínimos del hardware.

- Procesador 600 MHz o superior.
- 128 MB de memoria RAM.
- RNF Monitor VGA o superior.
- Ratón.

2.5. Definición del modelo de casos de uso del sistema

El Modelo de casos de uso permite que los desarrolladores de software y los clientes lleguen a un acuerdo sobre los requerimientos, es decir, sobre las condiciones y posibilidades que debe cumplir el sistema. El

mismo contiene actores, casos de uso y sus relaciones, además constituye una entrada para el análisis, el diseño y las pruebas. (29)

Actores del sistema

Los actores representan terceros fuera del sistema que colaboran con el sistema. Al identificar los actores del sistema se identifica el entorno externo del sistema.

Definición de los actores del sistema.

Actor del Sistema	Descripción
Usuario	Es el encargado del diseño de la plantilla del expediente de proyecto.

2.6. Definición de los casos de uso del sistema (CUS).

Los casos de uso son el componente clave del modelado. Su propósito es ilustrar cómo un sistema permite a un actor cumplir una meta, ilustrando todos los posibles caminos apropiados que ellos pueden tomar para cumplirla, así como las situaciones que podrían hacerlo fallar.

Casos de uso del Sistema.

CUS1 Gestionar documento.

CUS2 Gestionar propiedades de componentes.

CUS3 Diseñar plantillas del expediente de proyecto.

CUS4 Generar plantillas XML.

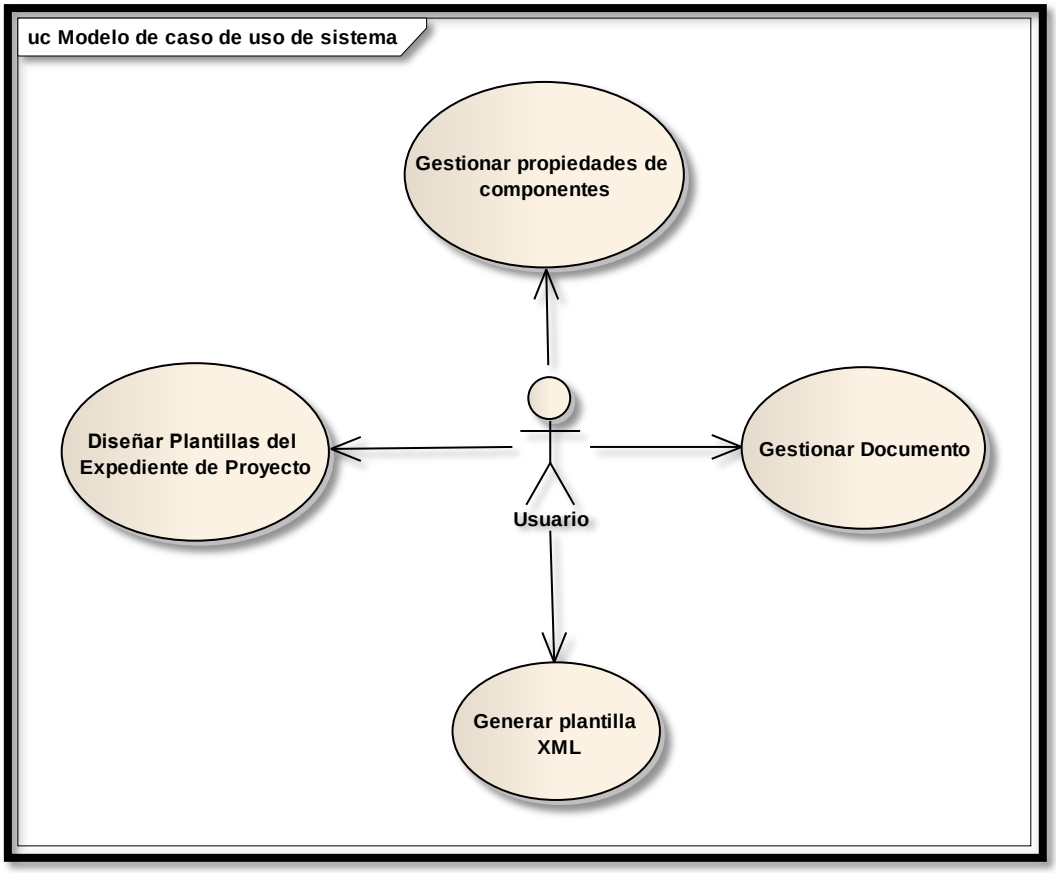


Figura # 2: Modelo de caso de uso del Sistema

2.6.1. Especificación de los casos de uso

Un caso de uso consta principalmente de una especificación textual (llamada Especificación del caso de uso), que contiene una descripción del flujo de eventos, describiendo la interacción entre actores y el sistema. (29)

Caso de Uso Diseñar plantillas de expediente de proyecto

Caso de Uso 1:	Diseñar plantillas de expediente de proyecto.
Actores:	Usuario

Propósito:	Realizar el diseño de las plantillas del expediente de proyecto satisfactoriamente.
Referencias:	RF1.1, RF1.2, RF1.3, RF1.4.
Descripción:	El usuario podrá realizar el diseño de cualquier plantilla del expediente de proyecto, con todos los respectivos componentes que la conforman, así como realizar cualquier modificación con el mismo.

Caso de Uso Gestionar documento.

Caso de Uso 2:	Gestionar documento.
Actores:	Usuario
Propósito:	Permite al usuario gestionar el documento (crear documento, modificarlo y eliminarlo).
Referencias:	RF2.1,RF2.2, RF2.3, RF2.4
Descripción:	El usuario tendrá la posibilidad de crear un documento, el cual estará conformado por todos los componentes que el usuario determine. Además de permitir crear uno nuevo y modificarlo si se requiere, permitiendo también la eliminación del mismo.

Caso de Uso Gestionar propiedades de componentes.

Caso de Uso 3:	Gestionar propiedades de componentes.
Actores:	Usuario
Propósito:	Modificar y visualizar las propiedades de todos los componentes.
Referencias:	RF 3, RF3.1, RF3.2.
Descripción:	El actor tendrá la posibilidad de ver y posteriormente modificar las propiedades de los componentes que formen parte de la

	plantilla.

Caso de Uso Generar plantilla XML.

Caso de Uso 4:	Generar plantilla XML.
Actores:	Usuario
Propósito:	Generar la plantilla XML del documento que se diseñe.
Referencias:	RF4.1, RF4.2
Descripción:	El actor podrá obtener la plantilla XML de la plantilla diseñada así como el proceso inverso, o sea, cargar la plantilla XML.

2.7.2 Descripción expandida de Casos de uso

Los casos de uso expandidos son muy útiles para alcanzar un conocimiento más profundo de los procesos y de los requerimientos. Constituyen un documento narrativo que describe la secuencia de eventos de un actor que utiliza el sistema para completar un proceso.

Caso de Uso 1:	Diseñar plantillas del expediente de proyecto. Ver Anexo 3
Actores:	Usuario
Propósito:	Permitir al usuario diseñar la plantilla del expediente de proyecto.
Resumen:	El actor podrá diseñar la plantilla con todos los componentes que esta requiera.
Referencias:	RF1.1, RF1.2, RF1.3, RF1.4.
Precondiciones.	
Acción del Actor.	**Respuesta del Sistema.**

1. El usuario inserta el componente que desee.	2. El sistema lo muestra en el lugar donde el usuario escogió.
3. El usuario selecciona la opción: a) Cortar/Mover componente  Ver sección **Cortar/Mover.** b) Copiar componente  Ver sección **Copiar.** c) Eliminar componente  Ver sección **Eliminar.**	

Sección: Cortar/Mover componente	
Acción del Actor.	**Respuesta del Sistema.**
	4. El sistema elimina el componente marcado de esa posición.
Sección: Copiar componente.	
Acción del Actor.	**Respuesta del Sistema.**
	4. El sistema realiza una copia del componente escogido por el usuario, dejando el marcado en el mismo lugar.
Sección: Eliminar componente.	
Acción del Actor.	**Respuesta del Sistema.**
	4. El sistema muestra un mensaje de confirmación¨ Está seguro de eliminar el componente¨.Sí\No

5. El usuario presiona el botón Si.	6. El sistema elimina el componente.

Caso de Uso 2:	Gestionar documento Ver Anexo 4
Actores:	Usuario
Propósito:	Permite al usuario gestionar la plantilla (crear documento, modificarlo y eliminarlo).
Resumen:	El actor podrá diseñar el documento con todos los componentes que esta requiera.
Referencias:	RF2.1, RF2.2, RF2.3, RF2.4.
Precondiciones.	
Acción del Actor.	Respuesta del Sistema.
1. El usuario puede seleccionar la opción: a) Nuevo documento  Ver sección **Nuevo documento.** b) Eliminar documento  Ver sección **Eliminar documento.** c) Salvar documento  Ver sección **Salvar documento.**	

Sección: Nuevo documento.	
Acción del Actor.	**Respuesta del Sistema.**
1. El usuario agrega	2. El sistema muestra el documento recientemente

un nuevo documento a diseñar.	agregado.
Sección: Eliminar documento.	
Acción del Actor.	**Respuesta del Sistema.**
	7. El sistema muestra una ventana de confirmación "Está seguro de eliminar el documento" Si/No.
8. El usuario presiona el botón Si.	9. El sistema elimina el documento.
Sección: Salvar documento.	
Acción del Actor.	**Respuesta del Sistema.**
	1. El sistema muestra una ventana para introducir el nombre y dirección donde guardar la plantilla (fichero con extensión: XML.).
2. El usuario introduce los datos y se guarda toda la información en la plantilla (fichero con extensión: XML).	

Caso de Uso 3:	Gestionar propiedades de componentes. Ver Anexo5
Actores:	Usuario
Propósito:	Permite al usuario ver las propiedades del componente y modificarlas.

Resumen:	El actor puede ver las propiedades de cada componente y posteriormente realizar cambios en las mismas.
Referencias:	RF 3, RF3.1, RF3.2.
Precondiciones.	
Acción del Actor.	**Respuesta del Sistema.**
	1. El sistema muestra el panel de propiedades del componente que está seleccionado.
2. El usuario modifica cualquier propiedad ya sea: el nombre, color del componente, ubicación del mismo, además del tipo de letra en caso de los textos, y adicionar columnas en caso de las tablas.	3. El sistema muestra el componente con las nuevas propiedades.

Caso de Uso 4:	Generar plantilla XML. Ver Anexo 6
Actores:	Usuario

Propósito:	Generar la plantilla XML de la plantilla que se diseñe y realizar el proceso inverso, es decir, cargar la plantilla XML.
Resumen:	El actor puede obtener la plantilla XML del diseño realizado y además puede cargarlo.
Referencias:	RF4.1, RF4.2.
Precondiciones.	
Acción del Actor.	**Respuesta del Sistema.**
1. El actor selecciona la opción **:** a) Salvar documento  Ver sección **Salvar Documento.** b) Cargar documento  Ver sección **Cargar Documento.**	
Sección: Salvar documento.	
Acción del Actor.	**Respuesta del Sistema.**
	1. El sistema muestra una ventana para introducir el nombre y dirección donde guardar la plantilla (fichero con extensión: XML.).
2. El usuario introduce los datos y se guarda toda la información en una plantilla (fichero con extensión: XML).	
Sección: Cargar Documento.	
Acción del Actor.	**Respuesta del Sistema.**

	1. El sistema muestra una ventana para buscar la plantilla (fichero con extensión: XML) donde esté guardada.
2. El usuario escoge la plantilla y selecciona el botón de Cargar.	3. El sistema muestra el diseño de la plantilla cargada por el usuario.

En este capítulo se definieron las características de la aplicación a construir, las cuales se especifican en términos de requisitos funcionales y no funcionales, siendo modelados gráficamente mediante un Diagrama de casos de uso del sistema. Además se identificó el actor que interactuará con las funcionalidades previstas para cada uno de estos casos de usos, los cuales fueron descritos textualmente. Finalmente, se concluye que, con el desarrollo de este capítulo quedaron sentadas las bases que darán paso al diseño y construcción de la herramienta para el diseño de plantillas del sistema de gestión del expediente de proyecto en su variante Web.

CAPÍTULO 3. **DISEÑO DEL SISTEMA**

En el presente capítulo se desarrolla el flujo de análisis y diseño, es uno de los flujos de trabajo de RUP, el cual contribuye a la definición de una arquitectura estable y sólida, que crea un plano del Modelo de implementación. El propósito del mismo es definir la estructura y elementos del diseño, describir los casos de uso en términos de clases de diseño y sus objetos, representándose gráficamente en Diagramas de clases, justificar los patrones empleados, así como estructurar el Modelo de datos.

3.1. Arquitectura del sistema

La arquitectura de software se centra tanto en los elementos estructurales significativos del sistema como en subsistemas, clases, componentes, nodos y las colaboraciones que tienen lugar entre estos elementos a través de las interfaces. La arquitectura abarca elementos importantes cómo:

- La organización del sistema.
- Los elementos que componen el sistema y sus interfaces, unido a su comportamiento.
- La composición de los elementos estructurales y del comportamiento en subsistemas progresivamente más grandes.
- Estilos de arquitectura que guían esta organización: elementos y sus interfaces, sus colaboraciones y su composición. (29)

3.1.1. Arquitectura orientada a objetos

Debido a que la implementación de la herramienta es orientada a objetos, se comporta como el estilo arquitectónico: Arquitectura orientada a objeto, el cual refleja la estructura del lenguaje de programación. Permite al diseñador del software construir una estructura de programa relativamente fácil de modificar y ajustar a escala.

Se basa en la bien conocida abstracción de procedimientos/funciones/métodos. Además presenta características como:

- Facilita la modularidad y reusabilidad
- Las clases encapsulan datos y métodos.
- La comunicación entre componentes se realiza mediante la invocación de servicios ofertados por ellos.

3.2. Modelo de diseño

El Modelo de diseño es un modelo de objetos que describe la realización de los casos de uso y al mismo tiempo constituye una abstracción del Modelo de implementación y del código fuente. Constituye una entrada esencial a las actividades de implementación y prueba. El mismo constituye un vehículo de análisis durante la fase de elaboración, pero se refina con un diseño detallado durante la fase de construcción. (31)

3.2.1. Fundamentación del uso de patrones

Los patrones de diseño son soluciones a problemas comunes de diseño en un determinado contexto, y que están presentes en el desarrollo de software. Permiten formalizar un vocabulario común entre los diseñadores del sistema y estandarizar el modo en que se realiza el diseño. Su utilización permite entender de manera fácil, el modo de mantener y ampliar el sistema.

Contribuye a la reutilización y al diseño de componentes de software, a la organización del código, la flexibilidad y extensibilidad, y la facilidad de realizar cambios en el sistema. Los GRASP son patrones generales de software para asignación de responsabilidades, es el acrónimo de "General Responsibility Assignment Software Patterns”. Aunque se considera que más que patrones propiamente dichos, son una serie de "buenas prácticas" de aplicación recomendable en el diseño de software. (31)

Experto: La responsabilidad de realizar una labor es de la clase que tiene o puede tener los datos involucrados (atributos). Una clase, contiene toda la información necesaria para realizar la labor que tiene encomendada.

Problema: ¿Cuál es el principio general para asignar responsabilidades a los objetos?

Solución: Asignar una responsabilidad al experto en información.

En GIEP_Designer se evidencia en la clase Principal, como se muestra a continuación en la figura 3.

```
public partial class Principal : QRibbonForm
{
    controladora obj_control;
    QTabControl  tabc;
    QTabPage tabp_clone = new QTabPage();
    Plantilla_Doc plan;

    public Principal()...

    private void Diseñador_Load(object sender, EventArgs e)...

    public void creartapControl()...

    public QTabPage creartabpage()...

    public void LlenarToolbook()...

    private void Nueva_hoja_Click(object sender, EventArgs e)...

    private void eliminarDocumentoToolStripMenuItem_Click(object sender, EventArgs e)...

    private void toolStripButton1_Click(object sender, EventArgs e)...
```

Figura # 3: Patrón experto

Creador: El patrón creador ayuda a identificar quién debe ser el responsable de la creación (o instanciación) de nuevos objetos o clases. La nueva instancia deberá ser creada por la clase que tiene la información necesaria para realizar la creación del objeto, o usa directamente las instancias creadas del objeto, o almacena o maneja varias instancias de la clase. En la herramienta a implementar se evidencia el uso de este patrón en el siguiente ejemplo: donde en la clase Plantilla_Doc, se realizan instancias de otras clases, contribuyendo al uso de los métodos y atributos de dichas clases.

Figura # 4: Patrón creador

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

Este patrón sugiere que la lógica de negocios debe estar separada de la capa de presentación, esto para aumentar la reutilización de código y a la vez tener un mayor control.

Se recomienda dividir los eventos del sistema en el mayor número de controladores para poder aumentar la cohesión y disminuir el acoplamiento. Como se muestra en el siguiente fragmento de código perteneciente a la implementación de la herramienta GIEP_Designer.

```
public partial class Principal : QRibbonForm
{
    controladora obj_control;
    QTabControl  tabc;
    QTabPage tabp_clone = new QTabPage();
    Plantilla_Doc plan;

    public Principal()...

    private void Diseñador_Load(object sender, EventArgs e)...

    // Crear tabcontrol.
    public void creartapControl()...

    // Crear tabpage.
    public QTabPage creartabpage()...

    // Crear componentes.
    public void LlenarToolbook()...

    Adicionar , Eliminar (Documentos).

    Adicionar , Copiar, Eliminar, Pegar(Capítulo).

    Copiar, Eliminar, Cortar , Pegar(Componentes).

    Métodos ,eventos.

    Leer , Salvar XML.
```

Figura # 5: Patrón controlador

Alta cohesión: Se dice que la información que almacena una clase debe de ser coherente y está en la mayor medida de lo posible relacionada con la clase. En la implementación de la herramienta en la clase Páginas se refleja el uso de este patrón, como se muestra a continuación.

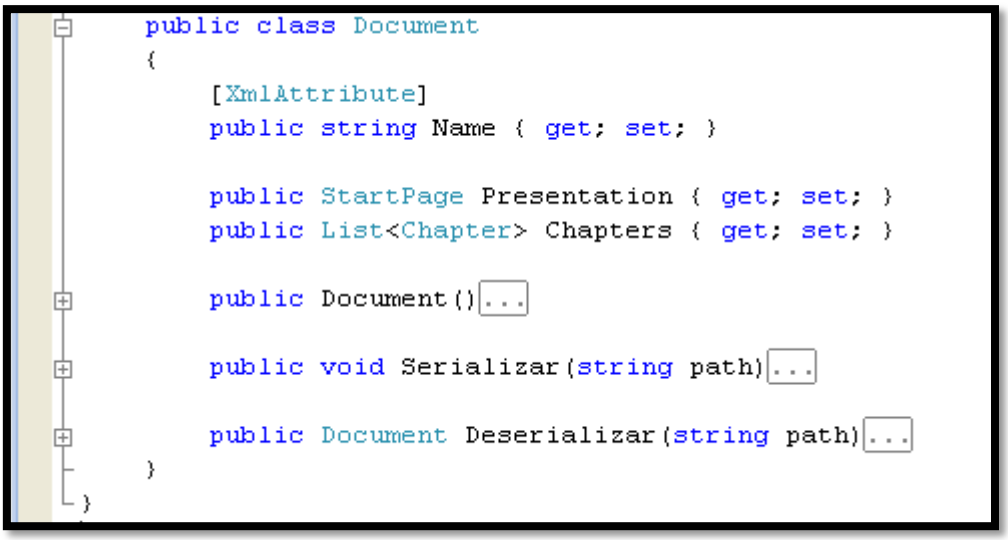


Figura # 6: Patrón alta cohesión

Bajo acoplamiento: Es la idea de tener las clases lo menos ligadas entre sí que se pueda. De tal forma que en caso de producirse una modificación en alguna de ellas, se tenga la mínima repercusión posible en el resto de las clases, potenciando la reutilización, y disminuyendo la dependencia entre las clases, como es el caso de la clase Documento que se muestra a continuación.

```
public class Document
{
    [XmlAttribute]
    public string Name { get; set; }

    public StartPage Presentation { get; set; }
    public List<Chapter> Chapters { get; set; }

    public Document()...

    public void Serializar(string path)...

    public Document Deserializar(string path)...
}
}
```

Figura # 7: Patrón bajo acoplamiento

3.3. Diagrama de clase del diseño

Representa las clases que serán utilizadas dentro del sistema y las relaciones que existen entre ellas. Sirve para visualizar las relaciones entre las clases que involucran el sistema, logrando una mejor muestra de la aplicación para la implementación. A continuación se muestra el diagrama de clase de diseño de los casos de uso Diseñar plantilla del EP y Generar plantilla XML respectivamente, el resto se podrán visualizar en el Anexo 1.

Modelo de clases del diseño – CU Diseñar plantilla del expediente de proyecto

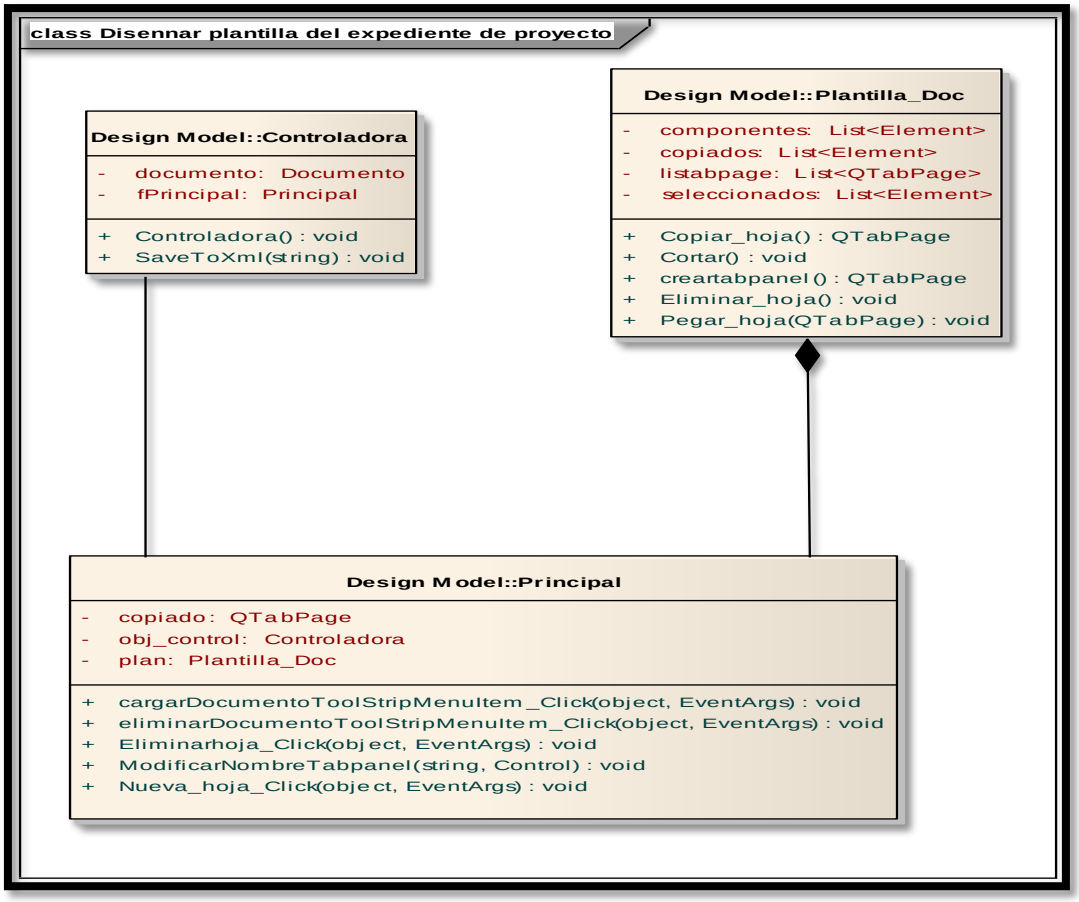


Figura # 8: Modelo de clases del diseño CU Diseñar plantilla del expediente de proyecto

Modelo de clases del diseño – CU Generar plantilla XML

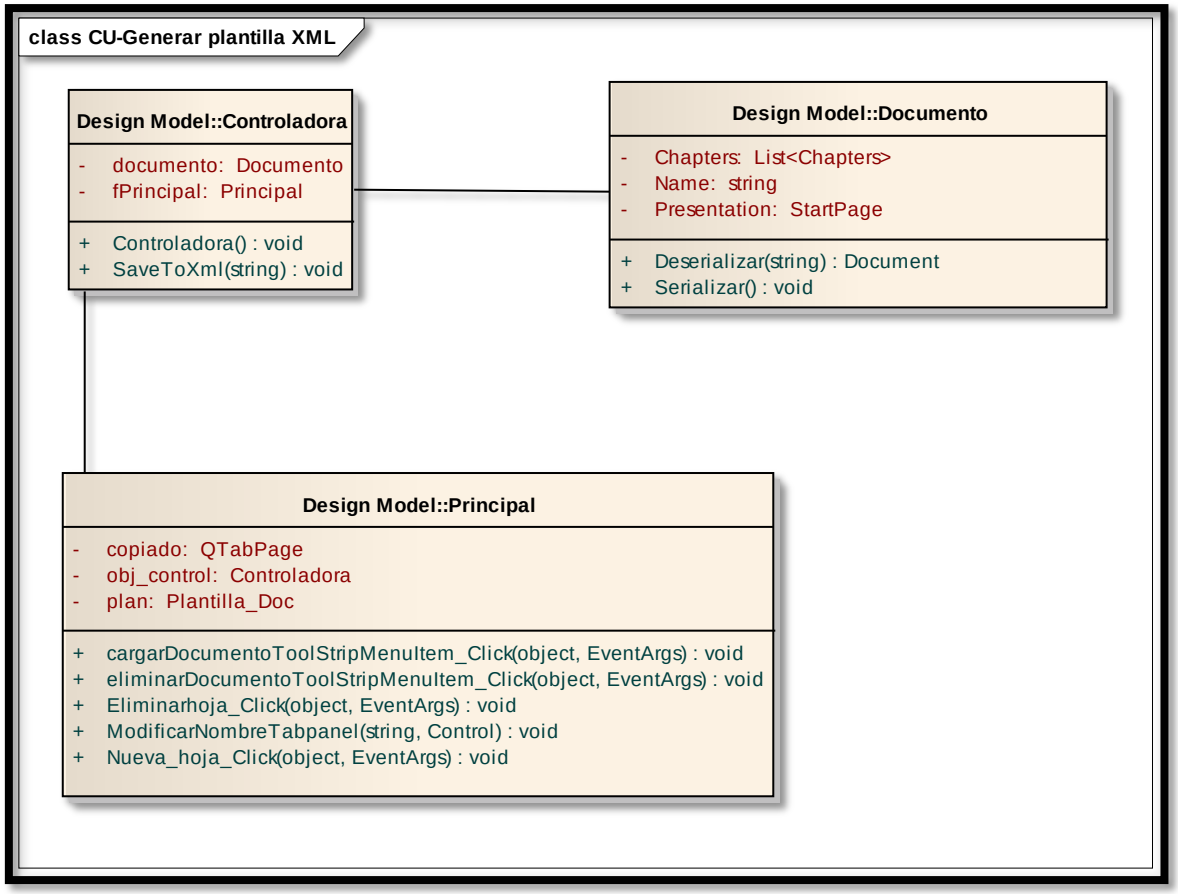


Figura # 9: Modelo de clases del diseño CU Generar plantilla XML

3.4. Descripción de las clases del diseño

A continuación se describen las clases del diseño representadas en los diagramas mostrados anteriormente.

Nombre: Controladora
Tipo de clase: Controladora

Atributo	**Tipo**
Documento	Documento
fPrincipal	Principal
Para cada responsabilidad:	
Nombre:	Controladora()
Descripción:	Es el constructor de la clase.
Nombre:	SaveToXML(string path)
Descripción:	Salva el XML del documento diseñado

Nombre:Principal	
Tipo de clase: Interfaz	
Atributo	**Tipo**
Copiado	QTabPage
Ob_control	Controladora
Plan	Plantilla_Doc
Para cada responsabilidad:	
Nombre:	Eliminar _Hoja(objectsender, EventArgs e)
Descripción:	Elimina la hoja que desee.
Nombre:	Modificar_Nombre (string nombre, Control TabPage)
Descripción:	Modifica el nombre que seleccione.
Nombre:	NuevaHoja (objectsender, EventArgs e)
Descripción:	Añade una hoja al documento.

Nombre:Plantilla_Doc	
Tipo de clase: Entidad	
Atributo	**Tipo**

Copiados	List<Element>
Seleccionados	List<Element>
Componentes	List<Element>
Listabpage	List<QTabPage>
Para cada responsabilidad:	
Nombre:	Copiar _Hoja()
Descripción:	Hace una copia de la hoja..
Nombre:	Cortar()
Descripción:	Mueve el componente seleccionado de lugar.
Nombre:	Pegar_Hoja(QTabPage copiado)
Descripción:	Hace una copia de la hoja con todos los componentes que la conforman..
Nombre:	Eliminar_Hoja()
Descripción:	Elimina la hoja seleccionada.

Nombre: Element	
Tipo de clase: Entidad	
Atributo	**Tipo**
Content	String
ID	String
Text	String
TexType	String
Para cada responsabilidad:	
Nombre:	Clone()
Descripción:	
Nombre:	CompareTo(Elementother)
Descripción:	Compara las posiciones de los componentes.

Nombre: Imagen	
Tipo de clase: Entidad	
Atributo:	**Tipo:**
Height	Float
ID	String
ImagenUri	String
Wight	Float
Para cada responsabilidad:	
Nombre:	Imagen()
Descripción:	Es el constructor de la clase

Nombre: Tabla	
Tipo de clase: Entidad	
Atributo:	**Tipo:**
Headers	Header
ID	String
Name	String
Records	List<Record>
Para cada responsabilidad:	
Nombre:	Tabla()
Descripción:	Es el constructor de la clase

Nombre: Chapter	
Tipo de clase: Entidad	
Atributo:	**Tipo:**
Element	List<IElement>

Name	Element
Page	Int
Wight	Float
Para cada responsabilidad:	
Nombre:	Chapter()
Descripción:	Es el constructor de la clase

Nombre: Documento	
Tipo de clase: Entidad	
Atributo:	**Tipo:**
Chapters	List<Chapters>
Name	String
Presentation	StartPage
Para cada responsabilidad:	
Nombre:	Serializar(Document doc, string path)
Descripción:	Convierte los datos del documento en un XML
Nombre:	Deserializar(string path)
Descripción:	Convierte el XML en un documento.

3.5. Diagrama de secuencia

Los diagramas de secuencia muestran los objetos como líneas de vida a lo largo de la página, y con sus interacciones en el tiempo representadas como mensajes dibujados como flechas desde la línea de vida origen hasta la línea de vida destino (32). Además realizan gráficamente las interacciones del actor y de las operaciones a que dan origen. A continuación se citarán los diagramas de secuencia de los casos de uso: Diseñar plantilla del expediente de proyecto y Generar plantilla XML, para ver el resto puede ir a Anexo 2

Diagrama de secuencia: CU–Diseñar plantilla del expediente de proyecto.

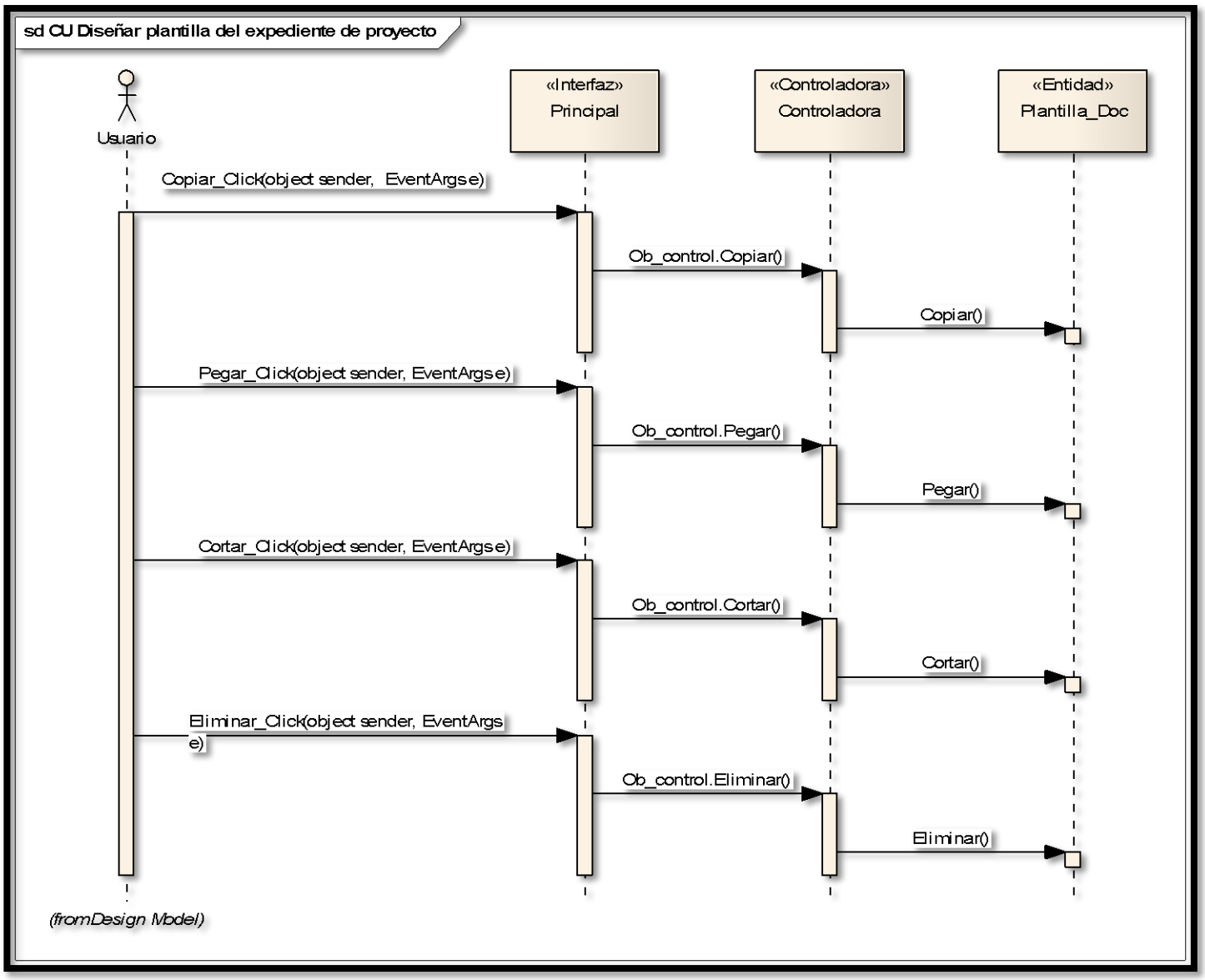


Figura # 10: Diagrama de secuencia CU Diseñar plantilla del expediente de proyecto

Diagrama de secuencia: CU–Generar plantilla XML

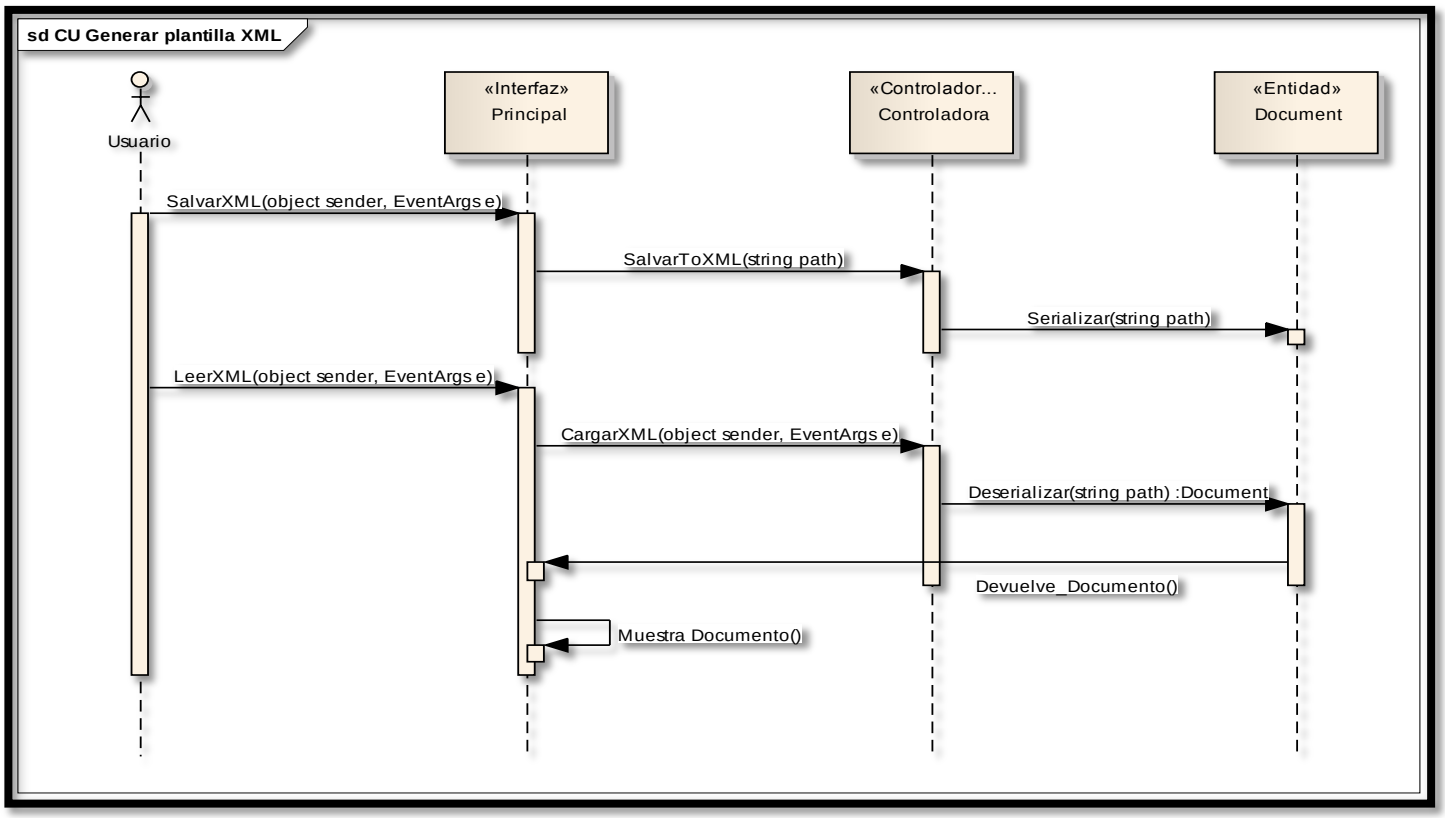


Figura # 11: Diagrama de secuencia CU Generar plantilla XML

En el capítulo se detallaron características del diseño de la aplicación, donde se definen las clases y modelos que las conforman. Se especificaron los principios y patrones de diseño del sistema y se realizó una descripción de las clases del diseño de la aplicación. Además se detallaron los diagramas de secuencia de cada uno de los casos de uso.

CAPÍTULO 4. **IMPLEMENTACION Y PRUEBA**

El propósito fundamental de este capítulo es definir cómo desarrollar la arquitectura definida durante el diseño. Se implementan las clases y subsistemas definidos en el capítulo anterior en términos de componentes. Se realizarán los diagramas de componentes y de despliegue, y finalmente se desarrollarán pruebas de caja negra a la herramienta para garantizar un buen funcionamiento de la misma.

4.1. Modelo de componentes

Un componente representa una parte física del sistema, los mismos pueden ser una librería, un ejecutable, una tabla, etc. El diagrama de componentes permite conocer a los desarrolladores la estructura física que tiene el sistema y como se relacionan sus partes.

A continuación se muestra cómo quedará la estructura de la implementación de la herramienta para el diseño de plantillas del sistema de gestión del expediente de proyecto en su variante Web.

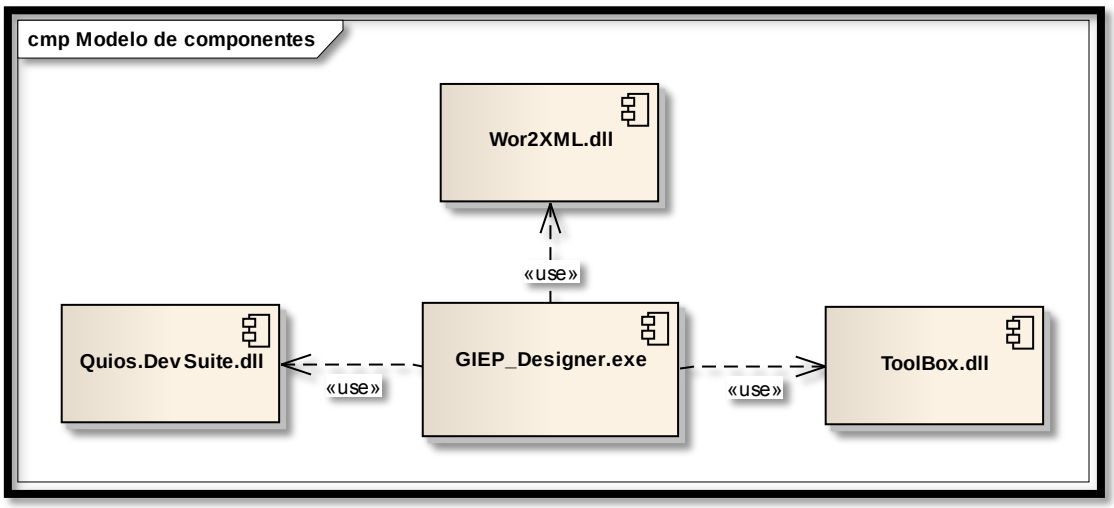


Figura # 12: Modelo de componentes

4.2. Modelo de despliegue

El diagrama de despliegue define la arquitectura física del sistema por medio de nodos interconectados. Se modela la topología del hardware sobre el que se ejecuta el sistema. Para la puesta en práctica de la herramienta, solo se necesita una computadora cliente donde correrá la aplicación en cada una.

4.3. Pruebas

La prueba es el proceso de ejecución de un programa con la intención de descubrir un error. Se puede ir realizando pruebas 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.

Existen casos de pruebas para los diferentes tipos de pruebas: caja negra y caja blanca. En el proceso de pruebas en cuestión, se hace uso de los casos de prueba para la realización de las pruebas de caja negra. La partición equivalente es un método de prueba de caja negra que divide el campo de entrada de un programa en clases de datos de los que se pueden derivar casos de prueba. Una clase de equivalencia representa un conjunto de estados válidos o no válidos para condiciones de entrada. Típicamente, una condición de entrada es un valor numérico específico, un rango de valores, un conjunto de valores relacionados o una condición lógica. (33)

En este caso se muestran los diseños de casos de prueba de los casos de usos más significativos.

El siguiente es un ejemplo donde se detalla el caso de uso **Diseñar plantillas del expediente de proyecto.**

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad
SC1: Insertar componente	EC1.1: Insertar componente	En este escenario el usuario arrastra el componente hacia la región de diseño.
	EC1.2: Insertar componente fuera de la región editable.	En este escenario el usuario arrastra el componente hacia una región no editable.

	EC1.3: Insertar componente y presionar la tecla Esc (escape).	En este escenario el usuario arrastra el componente y se arrepiente de la acción presionando la tecla de escape (Esc).
SC2: Mover componente	EC2.1: Mover el componente	En este escenario el usuario tiene la posibilidad de cortar el componente y pegarlo donde desee.
SC3: Copiar componente	EC3.1: Copiar componente	El usuario en este escenario realiza una copia del componente hacia donde lo necesite.
SC4: Eliminar componente	EC4.1: Eliminar componente	El usuario elimina el componente que desee.

Matriz de datos del caso de uso **Diseñar plantillas del expediente de proyecto**, de la SC1 Insertar componente.

ID del escenario	Escenario	Respuesta del sistema	Resultado de la prueba	Flujo central
EC:1.1	Insertar componente	El sistema muestra el componente en la posición que el usuario escogió.	Satisfactorio	1. El usuario selecciona el componente y lo arrastra hasta la región editable 2. El sistema muestra el componente en las plantillas.
EC:1.2	Insertar componente	El sistema muestra un	Satisfactorio	1. El usuario selecciona el

	fuera de la región editable.	mensaje de error.		componente y lo arrastra fuera de la región de edición. 2. El sistema muestra un mensaje de error.
EC:1.3	Insertar componente y presionar la tecla Esc (escape).	El sistema no permite insertar el componente.	Satisfactorio	1. El usuario selecciona el componente y lo arrastra, pero se arrepiente de la acción y presiona la tecla Esc (escape). 2. El sistema no permite insertar componentes.

Matriz de datos del caso de uso **Diseñar plantillas del expediente de proyecto**, de la SC2 Mover componente

ID del escenario	Escenario	Respuesta del sistema	Resultado de la prueba	Flujo central
EC:2.1	Mover componente	El sistema elimina el componente.	Satisfactorio	1. El usuario selecciona el componente y da clic en la opción

				de cortar. 2. El sistema elimina el componente de esa posición.

Ejemplo de la matriz de datos del caso de uso **Diseñar plantillas del expediente de proyecto**, de la SC3 Copiar componente.

ID del escenario	Escenario	Respuesta del sistema	Resultado de la prueba	Flujo central
EC:3.1	Copiar componente	El sistema hace una copia del componente seleccionado.	Satisfactorio	1. El usuario selecciona el componente y da clic en la opción copiar. 2. El usuario da clic en la posición exacta donde desea hacer la copia del componente. 3. El sistema muestra el componente en la posición que el usuario

				seleccionó.

Matriz de datos del caso de uso **Diseñar plantillas del expediente de proyecto**, de la SC4 Eliminar componente.

ID del escenario	Escenario	Respuesta del sistema	Resultado de la prueba	Flujo central
EC:4.1	Eliminar componente	El sistema elimina el componente.	Satisfactorio	1. El usuario selecciona el componente y da clic en la opción de eliminar. 2. El sistema elimina el componente marcado.

Detalles del caso de uso **Generar plantilla XML.**

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad
SC1: Salvar XML.	EC1.1: Salvar XML.	El usuario luego de realizar el diseño de la plantilla, puede salvar el XML de dicho diseño.
SC2: Cargar XML.	EC2.1: Cargar XML.	El usuario puede cargar el XML, y continuar el diseño de la plantilla.

Matriz de datos del caso de uso **Generar plantilla XML**, de la SC1 Salvar XML.

ID del escenario	Escenario	Respuesta del sistema	Resultado de la prueba	Flujo central
EC:1.1	Salvar XML	Salvar XML	Satisfactorio	3. El usuario selecciona la opción “Salvar documento”. 4. El sistema muestra una ventana para introducir el nombre y dirección donde guardar la plantilla (fichero con extensión: XML.) 5. El usuario introduce los datos y se guarda toda la información en la plantilla (fichero con extensión: XML)

Matriz de datos del caso de uso **Generar plantilla XML**, de la SC2 Cargar XML.

ID del escenario	Escenario	Respuesta del sistema	Resultado de la prueba	Flujo central
EC:2.1	Cargar XML	Cagar XML	Satisfactorio	1. El usuario selecciona la opción “Cargar documento”. 2. El sistema muestra una ventana para buscar la plantilla (fichero con extensión: XML.) 3. El usuario selecciona la plantilla (fichero con extensión: XML) y lo carga. 4. El sistema muestra toda la información correspondiente a la plantilla (fichero con

				extensión: XML) cargada.

En este capítulo, se obtuvieron los diagramas de componentes, quedando conformado el modelo de implementación del sistema. Además se desarrollaron los diseños de casos de pruebas del sistema, obteniendo resultados satisfactorios del mismo.

CONCLUSIONES

Se le ha dado cumplimiento a los objetivos planteados obteniéndose los resultados que a continuación se mencionan:

Se realizó un estudio de las herramientas existentes para el diseño de plantillas de documentos, obteniéndose como resultado que las mismas no son viables para el desarrollo de la aplicación porque su funcionamiento no responde a las necesidades requeridas.

Se realizó un estudio concreto de las tendencias y tecnologías actuales, permitiendo seleccionar las herramientas adecuadas para el desarrollo de la solución propuesta.

Se conformaron los requisitos que debe cumplir el sistema, facilitando la identificación de las principales necesidades y los elementos que garantizarán un correcto desempeño del sistema implementado.

Se le dio cumplimento al objetivo de la investigación con la implementación de la herramienta GIEP Designer capaz de diseñar las plantillas del sistema de gestión del expediente de proyecto en su variante Web.

RECOMENDACIONES

Para darle continuidad a este trabajo y contribuir con los servicios que brinda la herramienta GIEP Designer, los autores recomiendan:

- Implementar una versión para sistemas operativos libres, aprovechando las potencialidades de la plataforma MONO.
- Agregar en forma de pluggins una funcionalidad para que exporte las plantillas a una base de datos determinada, para facilitar la integración con GIEP en su variante Web.
- Crear una variante Web para el diseñador de plantillas de GIEP.

REFERENCIAS BIBLIOGRÁFICAS

1. peremarques.pangea. [En línea] http://peremarques.pangea.org/si.htm.

2. vecam.org. [En línea] http://vecam.org/spip.php?page=article&id_article=697.

3. Mora, Beatriz, Ruiz, Francisco y García, Félix y Piattini, Mario. [En línea] http://kuainasi.ciens.ucv.ve/ideas07/documentos/articulos_ideas/Articulo35.pdf.

4. Donatec. [En línea] //www.donatec.cl/node/295.

5. [En línea] http://www.x-window.co.uk/images/downloads/!Typefi_NLM_A4Web.pdf.

6. altova. [En línea] http://www.altova.com/de/xml-editor.

7. joven .cu. [En línea] http://gutl.jovenclub.cu/aplicaciones/fet-herramienta-para-generar-horarios-escolares.

8. Introducción al análisis y diseño orientado a objetos2004.

9. BOC Business Objectives Consulting Ibérica, S.L.U.

10. [En línea] http://cibsi05.inf.utfsm.cl/presentaciones/sesion11/Hacia_una_definicion_de_procesos_de_negocios_seguros.pdf.

11. INTALIO (BPM + BPMN + BPEL + OPEN SOURCE)2010.

12. [En línea] http://www.freedownload/manager.org/es/downloads/Paradigma_Visual_para_UML_%28M%C3%8D%29_14720_p/.

13. CEDS e Learning.

14. rational.com. [En línea] http://www.rational.com.ar/herramientas/roseenterprise.html.

15. [En línea] Http://www.thedigitalmap.com/EasyDEM/download/aide/html/documentos/robotikerWebServices.pdf.

16. kmels.net. [En línea] http://kmels.net/files/2009/uvg/cc2003/Resources/Contenidos/XP/xp.pdf.

17. Extreme Programming . [En línea]

18. [En línea] http://arturoweb.wordpress.com/2008/02/14/scrum-metodologia-agil-de-desarrollo.

19. Aplicación de la Metodología RUP para el Desarrollo Rápido de Aplicaciones Basado en el Estándar J2EE.

20. [En línea] http://decsai.ugr.es/~cb/CSharp/languaje/intro.xml.

21. A fondo C++.

22. Java sin errores2001.

23. [En línea] http://www.carlospes.com/curso_de_ingenieria_del_software/04_06_entornos_integrados_de_desarrollo.php .

24. [En línea] http://mredison.wordpress.com/2007/12/02/caractersticas-de-visual-studio-2008/.

25. [En línea] http://www.msdn2.microsoft.com.

26. www.netbeans.org. [En línea] http://www.netbeans.org/index_es.html.

27. [En línea] http://msdn.microsoft.com/es-es/library/bb397835%28v=vs.90%29.aspx.

28. Rational Enterprise Edition, Ayuda extendida2003.

29. Jacobson, Ivar, Booch, Grady y Rumbaugh, James.El Proceso Unificado de Desarrollo de Software. . La Habana : Felix Varela, 2004.

30. [En línea] ttp://www.procuraduria.gov.co/descargas/contratacion_2006/licitaciones/licitacion112006bid_Anexo4_4Requerimientosfuncionalesynofuncionalesparte_2.pdf.

31. [En línea] http://merinde.rinde.gob.ve.

32. [En línea] http://es.scribd.com/doc/15493687/DIAGRAMAS-DE-SECUENCIA.

33. Pressman., Roger S. Ingeniería del software, un enfoque práctico. 2002.

BIBLIOGRAFÍA

A fondo C++.

Aplicación de la Metodología RUP para el Desarrollo Rápido de Aplicaciones Basado en el Estándar J2EE.

BOC Business Objectives Consulting Ibérica, S.L.U.

CEDS e Learning.

Extreme Programming .

http://cibsi05.inf.utfsm.cl/presentaciones/sesion11/Hacia_una_definicion_de_procesos_de_negocios_seguros.pdf.

http://www.msdn2.microsoft.com.

INTALIO (BPM + BPMN + BPEL + OPEN SOURCE)2010.

Introducción al análisis y diseño orientado a objetos2004.

Jacobson, Ivar, Booch, Grady y Rumbaugh, James.El Proceso Unificado de Desarrollo de Software. . La Habana : Felix Varela, 2004.

Java sin errores2001.

kmels.net. [En línea] http://kmels.net/files/2009/uvg/cc2003/Resources/Contenidos/XP/xp.pdf.

Mora, Beatriz, Ruiz, Francisco y García, Félix y Piattini, Mario. kuainasi.ciens.ucv.ve/ideas07/documentos/articulos_ideas/Articulo35.pdf.

Pressman., Roger S. Ingeniería del software, un enfoque práctico. 2002.

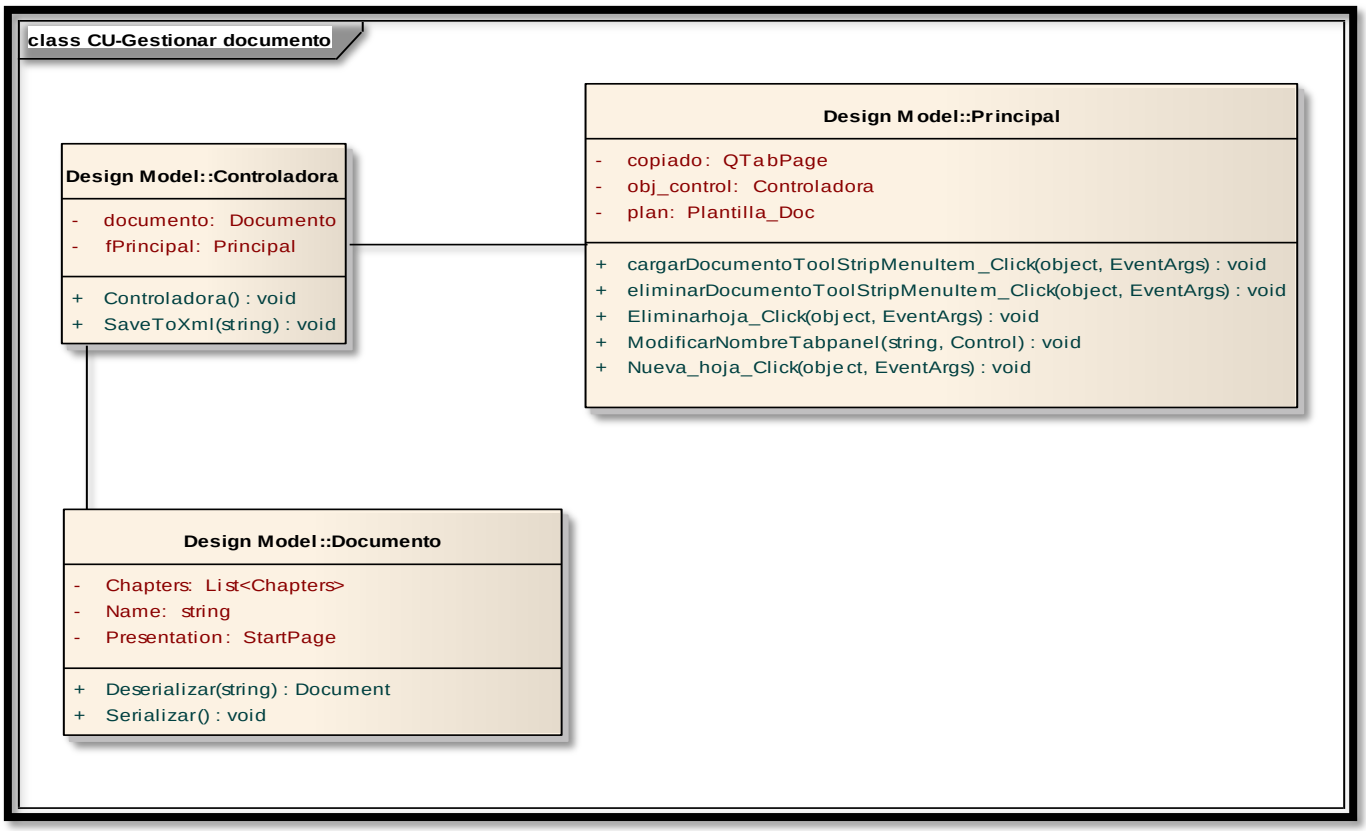
Rational Enterprise Edition, Ayuda extendida2003.

rational.com. [En línea] http://www.rational.com.ar/herramientas/roseenterprise.html.

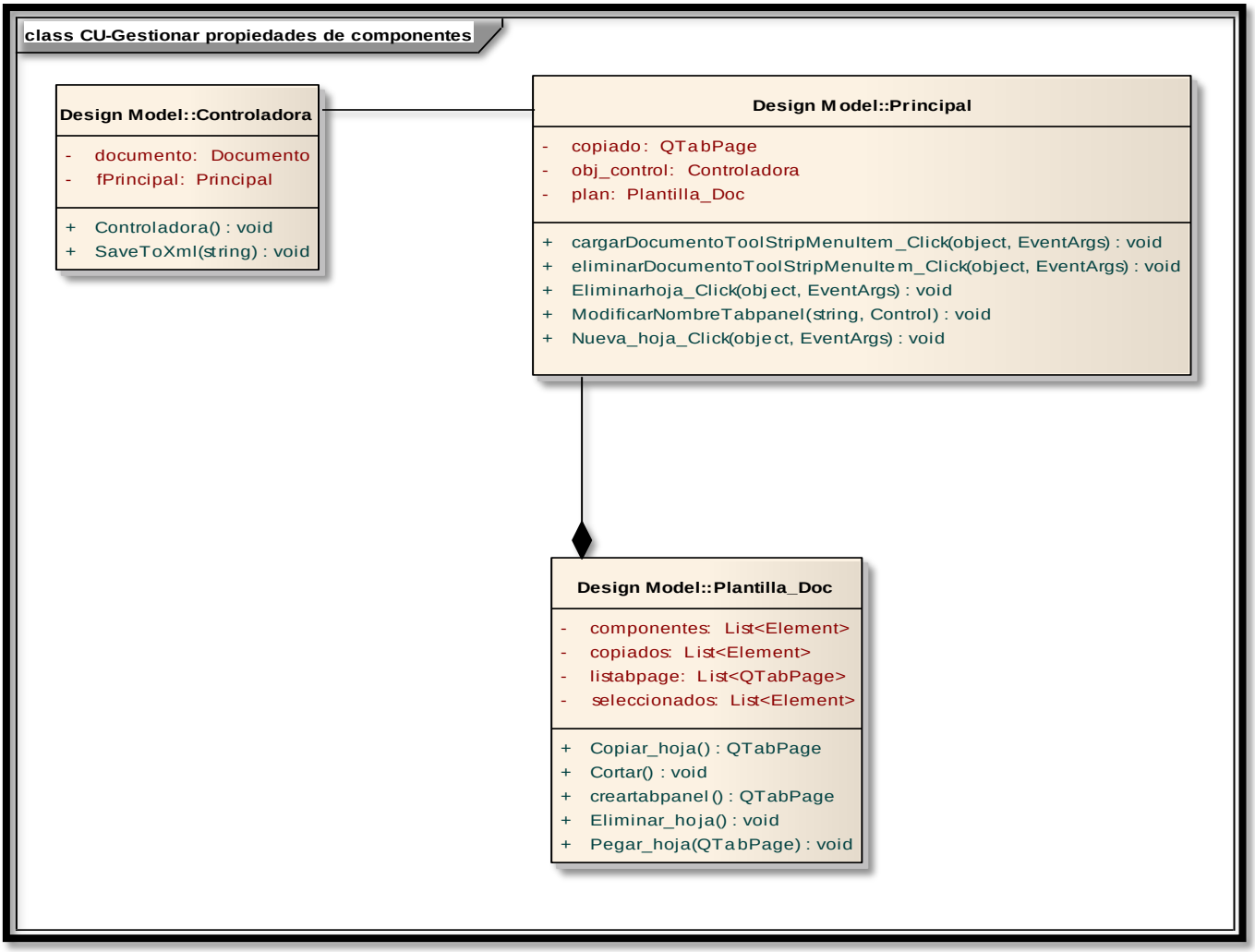
ANEXOS

Anexo 1

Modelo de clases del diseño – CU Gestionar documento



Modelo de clases del diseño – CU Gestionar propiedades de componentes



Anexo 2

Diagrama de secuencia: CU–Gestionar documento

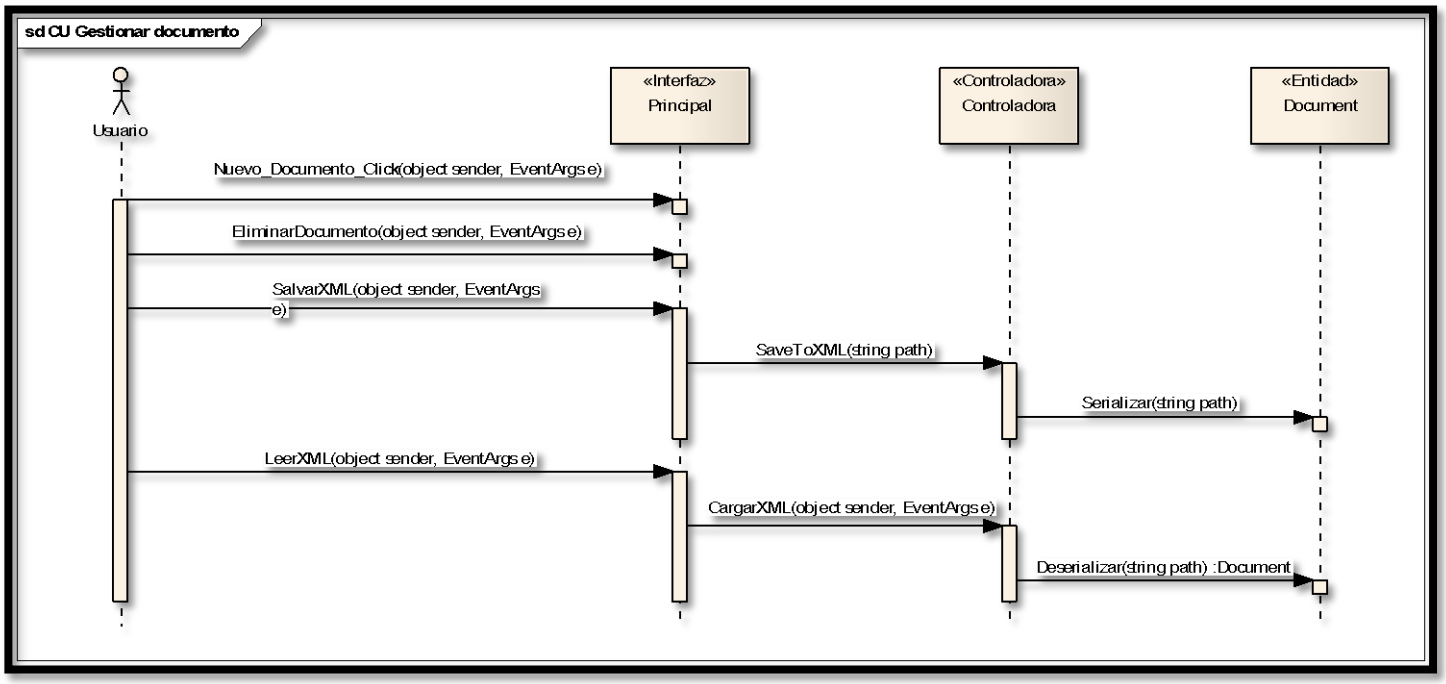
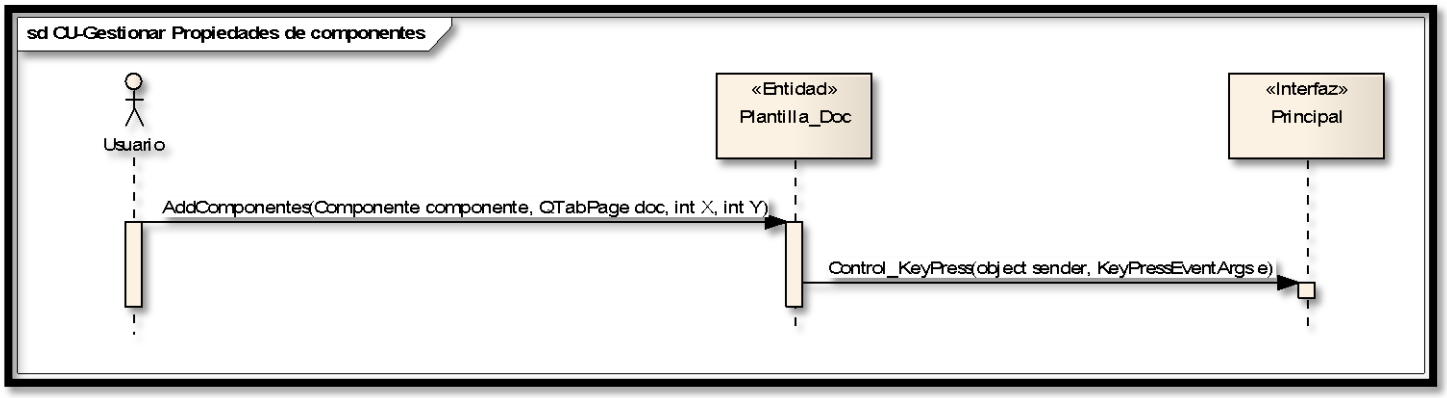
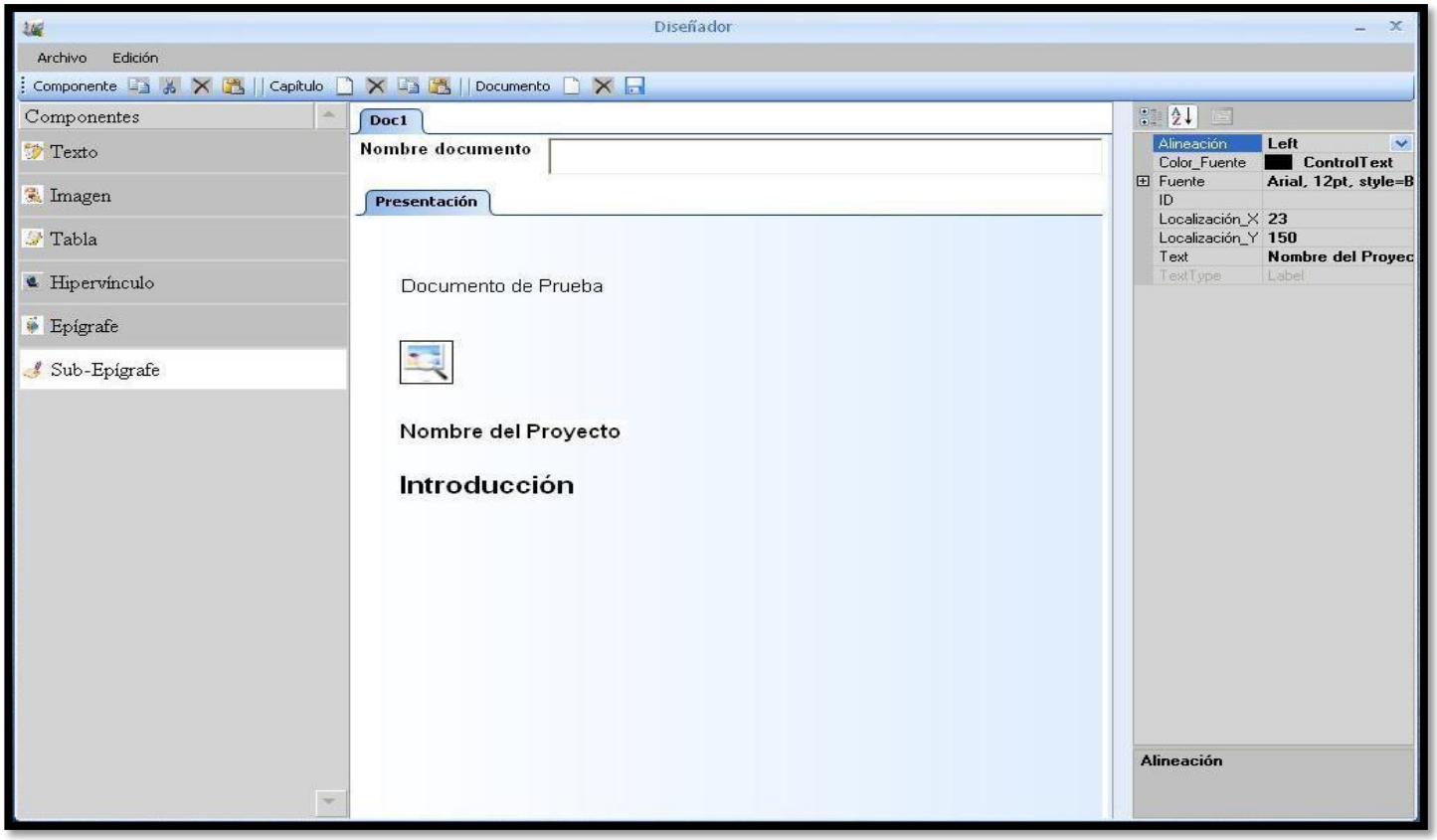


Diagrama de secuencia: CU–Gestionar propiedades de componentes



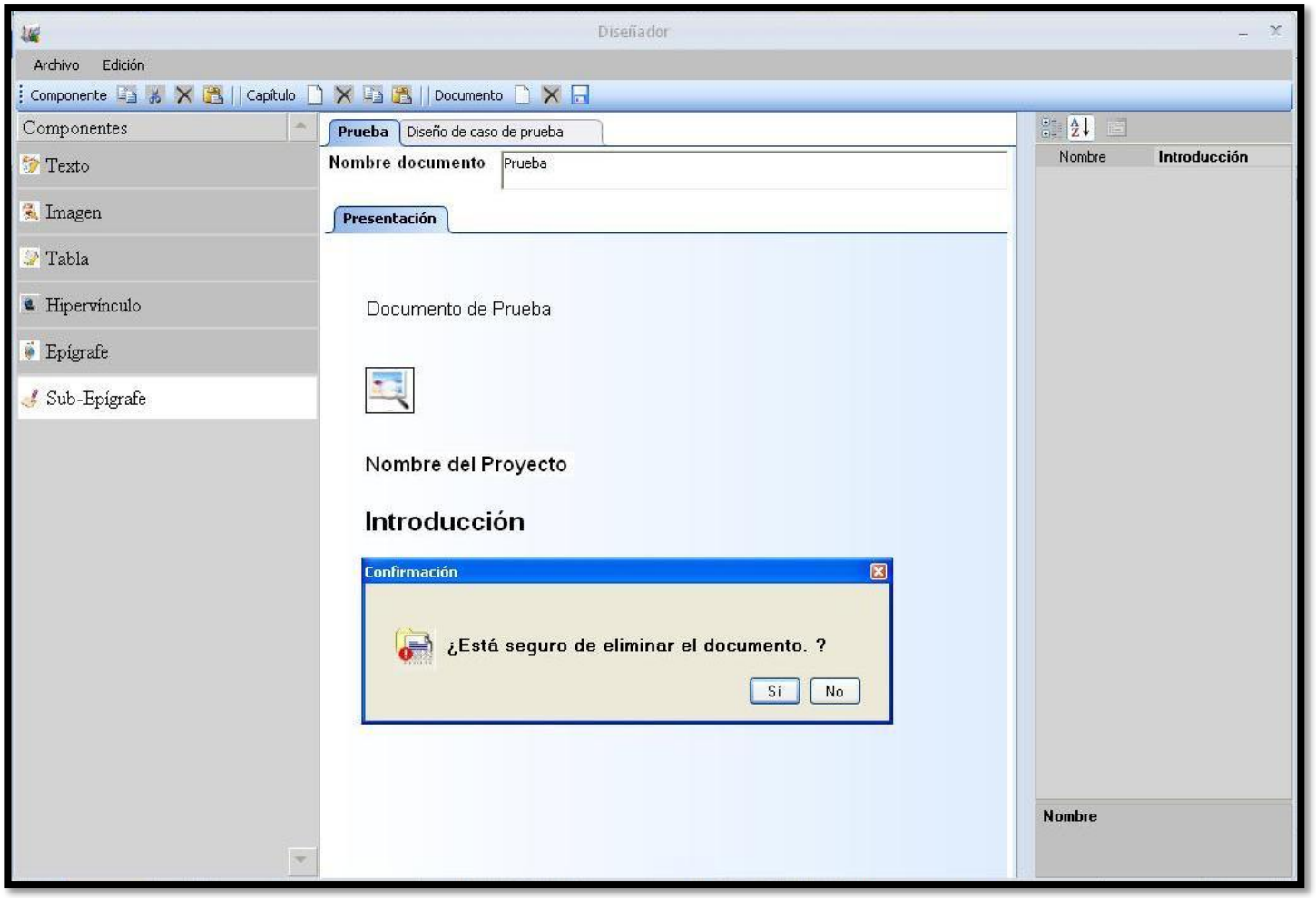
Anexo 3

Imagen correspondiente al caso de uso Diseñar plantilla del expediente de proyecto.



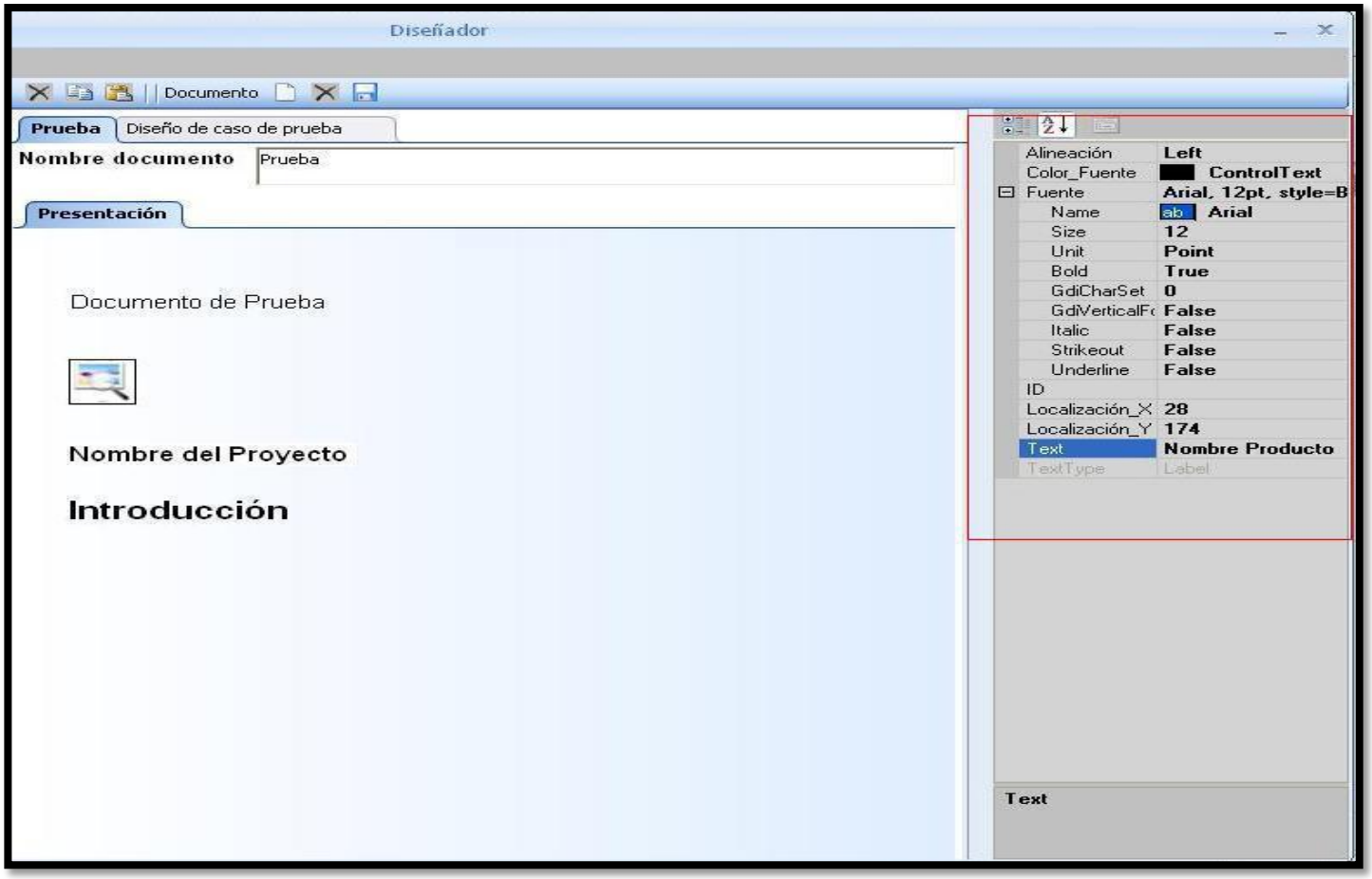
Anexo 4

Imagen correspondiente al caso de uso Gestionar documento.



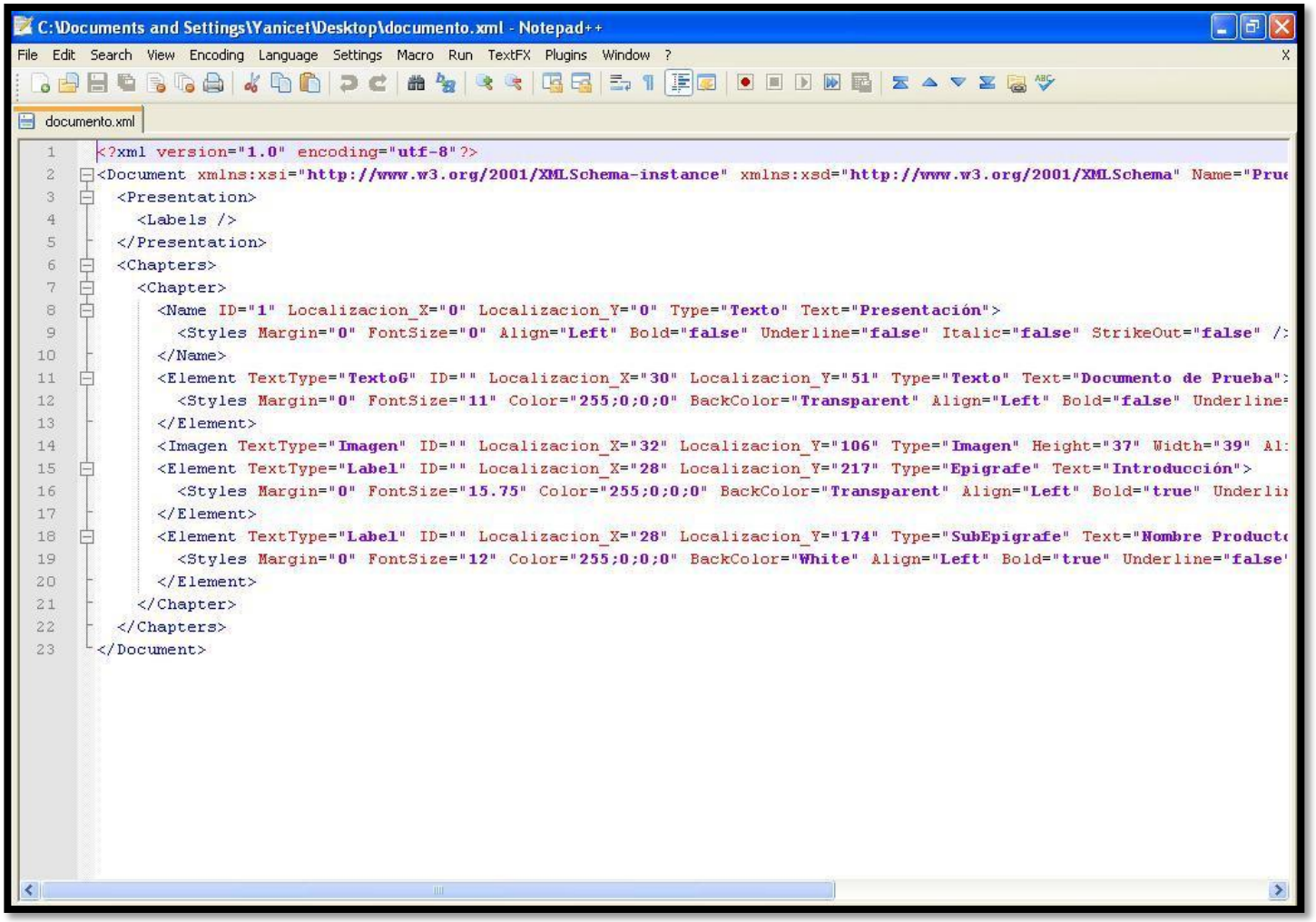
Anexo 5

Imagen correspondiente al caso de uso Gestionar propiedades de componentes.



Anexo 6

Imagen correspondiente al caso de uso Generar plantilla XML


C:\Documents and Settings\Yanicet\Desktop\documento.xml - Notepad++
File Edit Search View Encoding Language Settings Macro Run TextFX Plugins Window ?
documento.xml
<?xml version="1.0" encoding="utf-8"?>
<Document xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" Name="Prue
<Presentation>
<Labels />
</Presentation>
<Chapters>
<Chapter>
<Name ID="1" Localizacion_X="0" Localizacion_Y="0" Type="Texto" Text="Presentación">
<Styles Margin="0" FontSize="0" Align="Left" Bold="false" Underline="false" Italic="false" StrikeOut="false" />
</Name>
<Element TextType="TextoG" ID="" Localizacion_X="30" Localizacion_Y="51" Type="Texto" Text="Documento de Prueba"
<Styles Margin="0" FontSize="11" Color="255;0;0;0" BackColor="Transparent" Align="Left" Bold="false" Underline
</Element>
<Imagen TextType="Imagen" ID="" Localizacion_X="32" Localizacion_Y="106" Type="Imagen" Height="37" Width="39" Al
<Element TextType="Label" ID="" Localizacion_X="28" Localizacion_Y="217" Type="Epigrafe" Text="Introducción">
<Styles Margin="0" FontSize="15.75" Color="255;0;0;0" BackColor="Transparent" Align="Left" Bold="true" Underli
</Element>
<Element TextType="Label" ID="" Localizacion_X="28" Localizacion_Y="174" Type="SubEpigrafe" Text="Nombre Producto
<Styles Margin="0" FontSize="12" Color="255;0;0;0" BackColor="White" Align="Left" Bold="true" Underline="false"
</Element>
</Chapter>
</Chapters>
</Document>

GLOSARIO DE TÉRMINOS

1- **.NET:** Es una plataforma de Microsoft que provee un extenso conjunto de soluciones predefinidas para necesidades generales de la programación de aplicaciones, y administra la ejecución de los programas escritos específicamente con la plataforma.

2- **API:** Application Programming Interface es el conjunto de funciones y procedimientos (o métodos, en la programación orientada a objetos) que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción. Son usados generalmente en las bibliotecas.

3- **ASP.NET:** Es un framework para aplicaciones Web desarrollado y comercializado por Microsoft. Es usado por programadores para construir sitios Web dinámicos, aplicaciones Web y servicios Web XML.

4- **Clases:** Conjunto de objetos que comparten atributos, operaciones, relaciones y semántica, las mismas representan los conceptos fundamentales del sistema.

5- **Diagrama de clases:** Representación de los conceptos de importancia en el área de la aplicación, así como de las relaciones entre estos.

6- **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.

7- **Language Integrated Query o LINQ:** Es una tecnología integrada en .NET que proporciona la capacidad para consultar o manipular diversas fuentes de datos, independientes del proveedor utilizando de forma nativa la sintaxis de cualquier lenguaje de programación soportado por .NET.

8- **Librería:** Conjuntos de código ya realizado que se puede reutilizar en los programas y que ahorran mucho esfuerzo en la programación.

9- **RTF:** Es uno de los formatos nativos del Microsoft Word. Es un formato de texto compatible, en el sentido que puede ser migrado desde y hacia cualquier versión de Word.

10- **Script Manager:** El control es fundamental para la funcionalidad de AJAX en ASP.NET. El control administra todos los recursos AJAX de ASP.NET en una página. Esto incluye la descarga de scripts de Microsoft AJAX Library en el explorador y la coordinación de las actualizaciones parciales de página que se habilitan mediante el uso de controles Update Panel.

11- **Update Panel:** Estos controles son una parte fundamental de la funcionalidad AJAX en ASP.NET. Se usan con el control Script Manager para habilitar la representación de una página parcial. Esta representación reduce la necesidad de devoluciones de datos sincrónicas y actualizaciones de página completas cuando sólo se debe actualizar parte de la página.