UNIVERSIDAD NACIONAL DE LOJA

ÁREA DE ENERGÍA, INDUSTRIAS Y RECURSOS NATURALES NO RENOVABLES

CARRERA DE INGENIERÍA EN SISTEMAS

"GENERADOR DE CONTENIDOS ACADÉMICOS HIPERMEDIA BASADOS EN XML".

TESIS PREVIA A LA OBTENCIÓN DEL TITULO DE INGENIERO EN SISTEMAS

AUTORAS: **Verónica Maldonado Córdova.**

Jessenia Ramón Cabrera.

DIRECTOR: Ing. Milton Labanda Jaramillo.

LOJA-ECUADOR

2006

Ing. Milton Labanda Jaramillo.

DOCENTE DE LA CARRERA DE INGENIERÍA EN SISTEMAS DE LA UNIVERSIDAD NACIONAL DE LOJA.

C E R T I F I C O:

Que el presente trabajo de tesis denominado **"GENERADOR DE CONTENIDOS ACADÉMICOS HIPERMEDIA BASADOS EN XML**", de la autoría de las señoritas **Verónica Maldonado Córdova, y Jessenia Sidney Ramón**, ha sido revisado en su totalidad, por lo que autorizo su presentación y sustentación.

……

Ing. Milton Labanda Jaramillo

DIRECTOR DE TESIS

A U T O R I A

Los conceptos, análisis, diseño, conclusiones y recomendaciones del presente trabajo investigativo, corresponde exclusivamente a sus autoras.

Las Autoras.

Verónica J. Maldonado Córdova. Jessenia Sidney Ramón Cabrera.

A G R A D E C I M I E N T O

Queremos dejar constancia, haciendo extensivo el agradecimiento muy sincero a la Universidad Nacional de Loja, a la carrera de Ingeniería en Sistemas del Área de la Energía, las Industrias y los Recursos Naturales no Renovables, que nos ayudó a nuestra formación integral.

Al Ing. Milton Labanda Jarramillo, por su tiempo brindado en el asesoramiento del presente trabajo investigativo.

A nuestros amigos y a todas las personas que de alguna u otra manera nos ayudaron y contribuyeron para la culminación de este trabajo.

Las Autoras.

D E D I C A T O R I A

Dedico este trabajo a mis padres por su esfuerzo y dedicación; a mis hermanos por su cariño y soporte; y a todas aquellas personas que me apoyaron para hacer posible el desarrollo de este proceso investigativo.

Verónica

Dedico el presente trabajo a Dios; a mis padres por su amor, esfuerzo y comprensión; a mis hermanos por su apoyo y cariño; a mis tíos, a mis abuelitos y a todos mis amigos que me apoyaron en el transcurso de este trabajo.

Jessenia

ÍNDICE DE CONTENIDOS

ÍNDICE DE FIGURAS

Modelo de dominio

Casos de Uso

Casos de Uso Mantener Proyecto

Casos de Uso Mantener Tutor

Casos de Uso Mantener Página

Casos de Uso Agregar Recursos Página

Casos de Uso Abrir Proyecto

Diagrama de paquetes

Diagramas de clase

Diagrama de Componentes

Generación de Documentos

I N T R O D U C C I Ó N

A lo largo de la historia del hombre se pueden identificar grandes cambios que han marcado el desarrollo tecnológico y científico: grandes inventos y descubrimientos que han traído consigo el crecimiento del ser humano. Estos inventos han sido un gran soporte en la concepción de sistemas de información en el ámbito informático, ya que esto obedece a la transformación de la información en conocimiento, apostando por el desarrollo personal y el institucional, en un nuevo marco de trabajo: la empresa inteligente y, más específicamente, en un sistema de información de carácter interno.

La informática educativa se ha desarrollado con fuerza tras la comercialización de los ordenadores personales y ha permitido la incorporación de nuevos métodos de representación del conocimiento, puesto que los sistemas hipermedia están fuertemente ligados a un marco de trabajo conceptual que no limita aplicación en la construcción del conocimiento completando la necesidad de entornos de aprendizaje.

La Hipermedia ha tenido gran aceptación en lo referente a la realización de desarrollos informáticos orientados al aprendizaje, debido a que proporciona muchas de las características necesarias en la enseñanza como son: la interactividad, el uso de grandes bases de la información, la información multimedia, y la representación del conocimiento de forma similar a la forma de procesamiento de la información del alumno.

El desarrollo de los sistemas de información a través de la Web, ha suscitado un renovado interés por el desarrollo de la presente investigación, la misma que consta de los siguientes capítulos.

En el capítulo uno, se proporcionan conceptos básicos de hipermedia, su importancia así como las ventajas en la construcción de sistemas de hipermediales.

A demás se realiza un estudio de las tecnologías orientadas a objetos haciendo mayor énfasis en lenguaje Java que será utilizado posteriormente en la etapa de desarrollo del sistema.

Así también se ha llevado ha cabo un estudio del Lenguaje UML, el cual se compone de diferentes diagramas, los mismos que permiten avanzar en las diferentes etapas de desarrollo del proyecto.

Por último se brinda una perspectiva general del lenguaje XML como novedoso estándar para descripción e intercambio de datos en la generación de documentos.

En el capítulo dos, se describe el análisis realizado al sistema para poder alcanzar la implementación del mismo, además se detalla el modelo del sistema apoyado en los requerimientos y alcances del sistema determinados en la etapa de análisis.

En el capítulo tres, se da ha conocer el esquema XML así como las hojas de estilo XSLT que permitirá crear salidas en formatos HTML, PDF y WML.

Finalmente se presentan las conclusiones y recomendaciones de la investigación.

M E T O D O L O G Í A

Para el desarrollo del sistema se utilizó la metodología ICONIX por ser un proceso flexible y abierto, además nos permitió el uso de UML (Lenguaje Unificado de Modelado que unifica tres metodologías OMT, Bootch y OOSE) para el seguimiento de requisitos, proporcionado un resultado concreto y específico en el desarrollo real del proyecto.

Durante la etapa de análisis de requerimientos se realizó una revisión bibliográfica y una evaluación de la Hipermedia aplicada a la educación. De acuerdo con su aplicación y potencialidad se describe el sistema, se establecen metas, requerimientos y atributos del mismo, así mismo se desarrollo el modelo conceptual.

En todo el modelado del sistema se empleó UML para estructurar y organizar la información realizando los diagramas de clases, secuencia y de colaboración, integrando los requisitos funcionales del sistema.

A partir de toda la información recopilada en dichos diagramas se procedió a la generación del código fuente y la creación de las interfaces gráficas de usuario en base al diseño. Aplicando la programación orientada a objetos que brindan grandes beneficios, debido a la mantenibilidad, modificabilidad, reusabilidad y fiabilidad que son aplicables a todas las fases del ciclo de vida del proyecto.

Finalmente se realizó las pruebas necesarias al sistema para la depuración y control de errores, comprobando resultados y valorando el sistema hasta su disponibilidad para los usuarios.

EXPOSICIÓN Y DISCUSIÓN DE RESULTADOS

CAPITULO I
MARCO REFERENCIAL

1.1 Hipermedia

1.1.1 Conceptos generales

Hipertexto se refiere a un documento hipermedia donde todos los elementos de información contienen únicamente texto.

Multimedia hace referencia a sistemas que contienen y presentan texto, imágenes, sonido, video, etc. pero sin enlaces entre estos elementos de información.

Hipermedia “es la combinación de dos tecnologías HIPERtexto y multiMEDIA, hace referencia a una tecnología de construcción de (hiper)documentos que permite a los lectores encontrar fácilmente la información que realmente necesitan, de la manera que ellos decidan, a través de enlaces establecidos por el autor entre los diferentes elementos de información multimedia (texto, imagen, hipervínculos, etc.) que conforman el documento.” [1]

Hiperdocumento se compone, en principio, de dos tipos de elementos: por un lado estarían los nodos, o "contenedores" de la información multimedia; y por otro los enlaces, que interconectan los nodos permitiendo otras alternativas de navegación por la información diferente del clásico acceso secuencial "desde la primera a la última línea". Existe, no obstante, un tercer componente fundamental denominado ancla. Se trata de un mecanismo para señalar puntos incluidos en el interior de los nodos que sirven de origen o de destino a un determinado enlace entre nodos.

Sistemas Hipermediales, son los entornos que ofrecen a los usuarios todos los mecanismos para la creación, manipulación y consulta de hiperdocumentos. Así, respecto a su interface con los usuarios, proporcionan browsers o visualizadores para la navegación y, opcionalmente, visiones de mapas generales del hiperespacio del documento para que los usuarios conozcan su situación en cada momento. También incluyen herramientas de autor para el mantenimiento de los contenidos de los nodos y de la estructura de enlaces.

---

[1] http://www.cc.uah.es/hilera/docs/1998/a_cdm2/a_cdm2.htm

Ventajas:

- Hipermedia ofrece un medio adecuado para representar información poco o nada estructurada, que no puede ajustarse a los rígidos esquemas de las bases de datos tradicionales.
- Sus ergonómicos interfaces de usuario, muy intuitivos pues imitan el funcionamiento de la memoria humana.
- Los usuarios pueden hacer crecer el hiperdocumento o anotarlo sin modificarlo.
- Facilita la división en módulos y la consistencia de la información.
- Constituyen un marco idóneo para la autoría en colaboración.
- Facilitan diferentes modos de acceso a la información de manera que el usuario pueda elegir en cada momento el que más se ajuste a sus necesidades.

1.2 Tecnología Orientada a Objetos

La programación orientada a objetos es una metodología de diseño de software y un paradigma de programación que define los programas en términos de clases de objetos. Expresa un programa como un conjunto de objetos que se comunican entre ellos para realizar tareas. Esto difiere de los lenguajes tradicionales, en los que los datos y los procedimientos están separados y sin relación. Estos métodos están

pensados para hacer los programas y módulos más fáciles de escribir, mantener y reutilizar.

1.2.1 Conceptos básicos

Objeto: Es una instancia de una clase. Es una variable concreta de una clase, con su propia copia de las variables miembro.[2]

Clase: Define las variables y los métodos comunes a los objetos de ese tipo, pero luego, cada objeto tendrá sus propios valores y compartirán las mismas funciones.

Primero deberemos crear una clase antes de poder crear objetos o ejemplares de esa clase.

Mensajes: Son simples llamadas a las procesos del objeto con el se quiere comunicar para decirle que haga cualquier cosa.

Herencia: Significa que se puede crear una clase a través de una clase existente, y esta clase tendrá todas las variables y los métodos de su 'superclase', y además se le podrán añadir otras variables y métodos propios.

1.2.2 Características:

- **Abstracción**: La abstracción consiste en captar las características esenciales de un objeto, así como su comportamiento. Cada objeto en el sistema sirve como un modelo de un "agente" abstracto que puede realizar trabajo, informar y cambiar

[2] http://java.ciberaula.com/articulo/tecnologia_orientada_objetos.html

su estado, y “comunicarse” con otros objetos en el sistema sin revelar como se implementan estas características.

- **Encapsulación:** También llamada “ocultación de la información”, esto asegura que los objetos no puedan cambiar el estado interno de otros objetos de manera inesperada, solamente los propios métodos internos del objeto pueden acceder a su estado.

- **Polimorfismo:** “Indica la posibilidad de definir varias operaciones con el mismo nombre, diferenciándolas únicamente en los parámetros de entrada. Dependiendo del objeto que se introduzca como parámetro de entrada, se elegirá automáticamente cual de las operaciones se va a realizar. “[3]

- **Herencia:** Organiza y facilita el polimorfismo y la encapsulación permitiendo a los objetos ser definidos y creados como tipos especializados de objetos preexistentes. Estos pueden compartir (y extender) su comportamiento sin tener que reimplementar su comportamiento.

1.2.3 Ventajas de la programación y el modelado orientado a objetos

- Un problema macro se subdivide en unidades más pequeñas llamadas procesos.

- Es más cercano a la realidad.

- Es más fácil de mantener, reutilizar y hacer crecer el sistema.

- Evita redundancia en los procesos.

---

[3] http://www.ilustrados.com/publicaciones/EpyuVuuAlEAFjVSJEI.php

- Evita ambigüedades en su uso.
- Facilita la integridad de módulos.

1.2.4 Lenguajes de programación

- **C++**

Es un lenguaje más extenso en proyectos orientados a objetos, así como, en proyectos no orientados a objetos en absoluto.

- **Smalltalk**

Smalltalk fue el primer sistema puro de objetos. Todo en Smalltalk está definido en principio como objeto, incluyendo las propias clases.

- **Java**

El lenguaje de programación Java fue inventado por Sun Microsystems, empresa líder en servidores para Internet. Sun decidió entrar al mercado de los electrodomésticos y desarrollar pequeños programas para los mismos.

James Gosling miembro de este equipo con más experiencia en lenguajes de programación había notado la necesidad de desarrollar un lenguaje de programación llamado Oak que cubriera las necesidades que la empresa requería, para remediar las deficiencias que había notado en lenguajes como C y C++, tales como fiabilidad de código, robustez, y facilidad de desarrollo.

James Gosling, desarrollo un lenguaje derivado de C++ que intentaba eliminar las deficiencias del mismo. Llamó a este lenguaje Oak. Cuando Sun abandonó el proyecto se encontró con este lenguaje y decide lanzarlo oficialmente al mercado 1995.

Bill Joy, uno de los desarrolladores principales de Unís de Berkeley y cofundador de Sun, vio en Oak el lenguaje de programación adecuado para conquistar el creciente campo del Internet. Después de algunos ajustes en el diseño y un cambio de nombre, se presenta públicamente en agosto de 1995 el lenguaje de programación Java.

Características de java:

- **Orientado a Objetos.**

Java soporta las características esenciales del paradigma de la programación a objetos: encapsulación, herencia y polimorfismo. Java hace uso de la definición de entidades formadas por métodos y variables que reciben el nombre de clases, la instancia de alguna clase cualquiera en Java recibe el nombre de objeto.

- **Independencia de plataforma.**

Java ofrece la posibilidad de que los archivos que son generados para una aplicación sean independientes de plataformas, es decir, que se compilen una vez y se ejecuten en cualquier plataforma. Esto es posible gracias a que las aplicaciones hechas en Java generan archivos conocidos como bytecode.

- **Distribuido.**

Java se ha construido con extensas capacidades de interconexión TCP/IP. Existen librerías de rutinas para acceder a interactuar con protocolos como http y ftp.

- **Simple**

Java ofrece toda la funcionalidad de un lenguaje potente, pero sin las características menos usadas y más confusas de éstos.

No es necesario preocuparse de liberar memoria, al reciclador se encarga de ello y como es un thread de baja prioridad, cuando entra en acción, permite liberar bloques de memoria muy grandes, lo que reduce la fragmentación de la memoria.

- **Seguro**

Las aplicaciones de Java resultan extremadamente seguras, ya que no acceden a zonas delicadas de memoria o de sistema, con lo cual evitan la interacción de ciertos virus. Java no posee una semántica específica para modificar la pila de programa, la memoria libre o utilizar objetos y métodos de un programa sin privilegios de kernel del sistema operativo.

- **Portable**

Más allá de la portabilidad básica por ser de arquitectura independiente, Java implementa otros estándares de portabilidad para facilitar el desarrollo. Java construye sus interfaces de usuario a través de un sistema abstracto de ventanas de forma que las ventanas pueden ser implementadas en entornos Unix, Pc o Mac.

- **Multithreaded**

Java permite muchas actividades simultáneas en un programa. El beneficio de ser Multithreaded consiste en un mejor rendimiento interactivo y mejor comportamiento en tiempo real. Aunque el comportamiento en tiempo real esta limitado a las capacidades del sistema operativo subyacente (Unix, Windows, etc.), aún supera a los entornos de flujo único de programa (single-threaded) tanto en facilidad de desarrollo como en rendimiento.

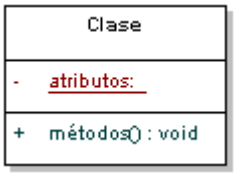
1.3 UML

El Lenguaje de Modelamiento Unificado es un lenguaje grafico para visualizar, especificar y documentar cada una de las partes que comprende el desarrollo de software. UML entrega una forma de modelar cosas conceptuales como lo son procesos de negocio y funciones de sistema, además de cosas concretas como lo son escribir clases en un lenguaje determinado, esquemas de bases de datos y componentes de software reusables.

1.3.1 Modelo de Clases

Clase: Es la unidad básica que encapsula toda la información de un objeto.

En UML, una clase es representada por un rectángulo que posee tres divisiones:



En donde:

- **Superior:** Contiene el nombre de la Clase.
- **Intermedio**: Contiene los atributos que caracterizan a la Clase.
- **Inferior:** Contiene los métodos u operaciones, los cuales son la forma como interactúa el objeto con su entorno.

Atributos y Métodos

Atributos: Los atributos o características de una Clase pueden ser de tres tipos, los que definen el grado de comunicación y visibilidad de ellos con el entorno.

Estos son:

- **public** (+,): Indica que el atributo será visible tanto dentro como fuera de la clase, es decir, es accesible desde todos lados.
- **private** (-,): Indica que el atributo sólo será accesible desde dentro de la clase.
- **protected** (#,): Indica que el atributo no será accesible desde fuera de la clase, pero si podrá ser accesado por métodos de la clase además de las subclases que se deriven.

Métodos: Los métodos u operaciones de una clase son la forma en como interactúa con su entorno, éstos pueden tener las características:

- **public** (+,): Indica que el método será visible tanto dentro como fuera de la clase, es decir, es accesible desde todos lados.

- **private** (-,): Indica que el método sólo será accesible desde dentro de la clase.

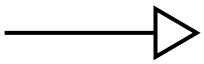
- **protected** (#,): Indica que el método no será accesible desde fuera de la clase, pero si podrá ser accesado por métodos de la clase además de métodos de las subclases que se deriven.

Relaciones entre clases:

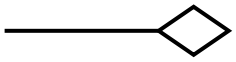
En UML, la cardinalidad de las relaciones indica el grado y nivel de dependencia, se anotan en cada extremo de la relación y éstas pueden ser:

- **uno o muchos:** 1 ..* (1 ..n)
- **0 o muchos:** 0..* (0..n)
- **número fijo:** m (m denota el número).

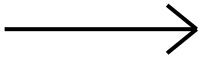
- **Herencia :** Indica que una subclase hereda los métodos y atributos especificados por una Superclase (public y protected), por ende la Subclase además de poseer sus propios métodos y atributos, poseerá las características y atributos visibles de la Superclase.



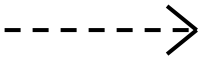
- **Agregación**: Es un tipo especial de asociación que representa una relación estructural entre un todo y sus partes.



- **Asociación**: Permite asociar objetos que colaboran entre sí. Cabe destacar que no es una relación fuerte, es decir, el tiempo de vida de un objeto no depende del otro.



- **Dependencia o Instanciación (uso):** "Es una relación semántica entre dos elementos en la cual un cambio en un elemento puede afectar a la semántica del otro elemento. Se representa como una línea discontinua dirigida."[4]



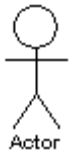
1.3.2 Diagrama de casos de uso.

Se emplean para visualizar el comportamiento del sistema, una parte de él o de una sola clase. De forma que se pueda conocer como responde esa parte del sistema. El diagrama de uso es muy útil para definir como debería ser el comportamiento de una parte del sistema, ya que solo especifica como deben comportarse y no como están implementadas las partes que define. Un caso de uso especifica un requerimiento funcional, es decir indica esta parte debe hacer esto cuando pase esto.

En el diagrama nos encontramos con diferentes figuras que pueden mantener diversas relaciones entre ellas:

[4] http://www.libros-electronicaos.net

- **Actor:** Un actor representa quien o que inicia una acción dentro del sistema, en otras palabras, es simplemente un rol que es llevado acabo por una persona o cosa. Un Actor en un diagrama Caso de uso es representado por una figura en forma de persona.



- **Uso-Caso:** El caso de uso en sí es representado por un ovalo que describe la funcionalidad que se requiere por el sistema.

- **Comunicación**: Este elemento representa la relación que existe entre un Caso de uso y un Actor, dicho elemento es representado simplemente por una línea recta que se extiende de la figura del actor hacia él ovalo del caso de uso.

- **Limite de Sistema (System Boundry)**: Empleado para delimitar los límites del sistema, y representado por un rectángulo con color de fondo distintivo.

- **Generalización:** Una generalización indica que un caso de uso es un caso especial de otro caso, en otros términos, representa una relación padre-hijo, donde el hijo puede ser suplido directamente por el padre en cualquier momento. Este elemento es representado por una línea con flecha que se extiende del caso de uso hijo hacia el caso de uso padre.

- **Inclusión:** Una inclusión es utilizada para indicar que un caso de uso depende de otro caso, dicho de otra manera, significa que la funcionalidad de determinado caso se requiere para realizar las tareas de otro. Este elemento es representado por una línea punteada con flecha y comentario <<include>> que se extiende del caso de uso base hacia el uso caso de inclusión.

- **Extensión:** Una extensión representa una variación de un caso de uso a otro, aunque similar a una generalización, una extensión representa una dependencia especifica, mientras una generalización no implica que los usos-casos dependen uno del otro. Este elemento es representado por una línea punteada con flecha y comentario <<extend>> que origina del caso de uso base hacia el uso caso de extensión.

1.3.3 Diagrama de objetos.

Muestra un conjunto de objetos y sus relaciones, son como fotos instantáneas de los diagramas de clases y cubren la vista de diseño estática o la vista de procesos estática desde la perspectiva de casos reales o prototípicos.

1.3.4 Diagrama de componentes.

Se utilizan para modelar la vista estática de un sistema. Muestra la organización y las dependencias entre un conjunto de componentes. Se relacionan con los diagramas de clases ya que un componente suele tener una o más clases, interfaces o colaboraciones.

En el se sitúan las librerías, tablas archivos, ejecutables y documentos que formen parte del sistema.

1.3.5 Diagramas de despliegue.

En el diagrama de despliegue se indica la situación física de los componentes lógicos desarrollados. Es decir se sitúa el software en el hardware que lo contiene. Cada Hardware se representa como un nodo.

1.3.6 Diagrama Secuencia.

El diagrama de secuencia forma parte del modelado dinámico del sistema. Se modelan las llamadas entre clases desde un punto concreto del sistema. Es útil para observar la vida de los objetos en sistema, identificar llamadas a realizar o posibles errores del modelado estático, que imposibiliten el flujo de información o de llamadas entre los componentes del sistema.

En el diagrama de secuencia se muestra el orden de las llamadas en el sistema. Se utiliza un diagrama para cada llamada a representar. Es imposible representar en un solo diagrama de secuencia todas las secuencias posibles del sistema, por ello se escoge un punto de partida. El diagrama se forma con los objetos que forman parte de la secuencia, estos se sitúan en la parte superior de la pantalla, normalmente en la izquierda se sitúa al que inicia la acción. De estos objetos sale una línea que indica su vida en el sistema.

1.3.7 Diagrama Colaboración.

El diagrama de colaboración destaca la organización estructural de los objetos que envían y reciben mensajes. Estos permiten describir una operación específica incluyendo sus argumentos y variables locales creadas durante su ejecución. Se muestran los objetos y mensajes que son necesarios para cumplir con un requerimiento o propósito, o con un conjunto de ellos.

1.4 XML

XML o Extensible Markup Lenguage se ha presentado como sucesor de HTML como lenguaje para presentación de contenidos en Internet. Pero XML es mucho más: es un metalenguaje que sirve para describir nuevos lenguajes cada cual adaptado a un grupo de contenidos especial. XML tiene múltiples aplicaciones; desde la simple publicación de contenidos, pasando por publicación de contenidos complejos que se adaptan al cliente en el que se va a presentar, hasta un nuevo paradigma de programación, denominada programación orientada a documentos, en el cual el elemento fundamental no es el programa, sino el documento. XML contiene alrededor un grupo de tecnologías: hojas de estilo (CSS) que especifican como se tiene que presentar la información a un cliente determinado, hojas de estilo transformadoras (XSLT), así como entornos completos de codificación que permiten aplicar diferentes transformaciones a un documento XML hasta que se presente al usuario final.

Características XML:

- **XML sirve para estructurar datos:** XML es un conjunto de reglas para diseñar formatos de texto que permitan estructurar los datos. XML facilita a la computadora la tarea de generar datos, leerlos, y asegurar que su estructura no es ambigua.

XML evita las fallas comunes en diseño de lenguajes: es extensible, independiente de la plataforma, soporta internacionalización y localización.

- **XML se parece un poco al HTML:** “Al igual que HTML, XML usa *etiquetas* palabras encerradas por '<' y '>' y *atributos* de la forma nombre="valor". Mientras HTML especifica lo que cada etiqueta y atributo significan, y a menudo cómo aparecerá en un navegador el texto que hay entre ellas, XML usa las etiquetas sólo para delimitar las piezas de datos, y deja la interpretación de los datos completamente a la aplicación que los lee. “[5]

- **XML es texto, pero no está pensado para ser leído:** Los programas que producen planillas de cálculo, libretas de direcciones y otros datos estructurados, a menudo guardan esos datos en disco, usando un formato binario o de texto. Los formatos de texto también permiten a los desarrolladores corregir más fácilmente sus aplicaciones. Igual que los de HTML, los archivos de XML son archivos de texto que la gente no necesita, pero puede leer, si surge la necesidad. Las reglas de XML son estrictas, y en esto se parece menos al HTML. Una etiqueta olvidada o un atributo sin comillas inutilizan un archivo XML.

- **XML lleva HTML a XHTML:** Hay una importante aplicación de XML que consiste en un formato de documento: XHTML de la W3C, el sucesor de HTML. XHTML tiene muchos de los mismos elementos de HTML. La sintaxis ha sido ligeramente cambiada para conformarse a las reglas de XML. Un documento "basado en XML" hereda la sintaxis de XML.

---

[5] http://www.w3.org/XML/1999/XML-in-10-points.html

- **XML es modular:** XML le permite definir un formato de documento combinando y rehusando otros formatos. *XML Schema* está diseñado para reflejar este soporte de la modularidad al nivel de definir la estructura de documentos XML, por medio de la facilidad para combinar dos esquemas para producir un tercero que cubre una estructura de documento combinada.

- **XML es la base de RDF y de la Web Semántica:** El Resource Description Framework (RDF) de la W3C es un formato de texto XML que soporta aplicaciones de descripción de recursos y metadatos, tales como listas de temas musicales, colecciones de fotos, y bibliografías. RDF integra las aplicaciones y los agentes en una Web Semántica.

- **XML es gratuito,** independiente de la plataforma y bien soportado: Al elegir XML como la base de un proyecto, se gana acceso a una comunidad grande y creciente de herramientas e ingenieros experimentados en la tecnología.

1.4.1 Ventajas del XML

Algunas de las ventajas del XML son:

- Los autores y proveedores pueden diseñar sus propios tipos de documentos usando XML, en vez de limitarse a HTML. Los tipos de documentos pueden ser explícitamente hechos a la medida de una audiencia.

- La información contenida puede ser más 'rica' y fácil de usar, porque las habilidades hipertextuales de XML son mayores que las de HTML.

- XML puede dar más y mejores facilidades para la representación en los visualizadores.

- Elimina muchas de las complejidades de SGML (Standart Generalized Markup Language), en favor de la flexibilidad del modelo, con lo que la escritura de programas para manejar XML será más sencilla que haciendo el mismo trabajo en SGML.

- La información será más accesible y reutilizable, porque la flexibilidad de las etiquetas de XML pueden utilizarse sin tener que amoldarse a reglas específicas de un fabricante, como es el caso de HTML.

- Los ficheros XML válidos son válidos también en SGML, luego pueden utilizarse también fuera de la Web, en un entorno SGML.

- XML es gratuito, independiente de la plataforma y bien soportado.

CAPITULO II

ANÁLISIS, DETERMINACION DE REQUERIMIENTOS

Y ALCANCES DEL SISTEMA

2.1 Análisis del sistema

El adelanto incontenible de las ciencias, el desarrollo vertiginoso de la técnica, la complejidad de la sociedad contemporánea, la facilidad de las relaciones internacionales, la desaparición de fronteras económicas, el nivel cultural universal, las expectativas de este siglo, manifestaciones todas estas que configuran un proceso globalizado que le ha exigido a la Universidad Nacional de Loja convertirse en el referente científico, técnico, cultural y político que forme profesionales comprometidos con el desarrollo productivo e integral de la sociedad.

En la educación el cambio y la innovación no se reducen a prácticas didácticas o al simple uso de ciertos recursos tecnológicos, metodologías y actividades utilizadas en el proceso de enseñanza aprendizaje, sino en la estructuración y en la organización de la información para la consecución de un aprendizaje significativo.

Aquí radica el propósito para la realización de un estudio que permita considerar al docente la aplicación de tecnologías hipermedia que le ayude a enriquecer el proceso educativo y valorar el nivel de aprendizaje individualizado de los estudiantes mediante la utilización de documentos hipermedia.

Para la realización de este trabajo se ha contado con la colaboración de 25 docentes y 40 estudiantes de las diferentes áreas de la Universidad Nacional de Loja, los mismos que fueron entrevistados para obtener información acerca de la utilización y actitudes ante los recursos hipermedia.

Tomando en cuenta que el docente es un factor determinante a la hora de cualquier innovación didáctica o tecnológica se realizó un estudio sobre las posibilidades de utilización y aplicación de los recursos multimedia e hipermedia por parte de los docentes de las distintas áreas de la U.N.L para lograr:

1. Fomentar la participación.

En general, se puede decir que hay diferencias entre los docentes de las distintas áreas de U.N.L, sobre la importancia que conceden a la utilización didáctica de los recursos hipermedia para ayudar a fomentar la participación del estudiante universitario.

Un 80% de los docentes consideran que el uso de las enciclopedias, diccionarios, periódicos electrónicos, documentos, CD-ROM, etc., fomentan la participación del estudiante en el aula, frente a un 20% de docentes que dan una baja valoración de los recursos hipermedia para fomentar la participación en asignaturas como; matemáticas, física, química y cálculo.

2. Formar criterio propio.

Las repuestas de los docentes en las distintas áreas de la U.N.L, acerca del uso de los recursos hipermedia para ayudar a formar criterio propio en el estudiante universitario fueron las siguientes:

Destacan con un 85% de los docentes que consideran que los materiales hipermedia ayudan a los estudiantes mucho a reflexionar y formar criterio propio, frente a un 15% de docentes que manifiestan que el uso de las aplicaciones hipermedia en la enseñanza no ayudan a desempeñar la función analizada.

3. Desarrollo de la creatividad y exploración.

Las manifestaciones de los docentes con respecto a sí los documentos hipermedia pueden incidir en el desarrollo de la creatividad y exploración en el estudiante universitario, proporcionaron los siguientes resultados:

Un 93% de los docentes consideran que los materiales hipermedia si desarrollan la creatividad y la exploración, porque desarrolla los sentidos, fomenta la iniciativa y evoluciona la imaginación, frente a un 7% de docentes que sostienen que los materiales hipermedia no contribuyen al desarrollo de la creatividad.

4. Establecer conclusiones.

A la pregunta de que si el uso de documentos hipermedia pueden ayudar a los estudiantes a establecer conclusiones, las respuestas de los docentes de las distintas áreas de la U.N.L, reflejaron los porcentajes siguientes:

Un 90% consideran que los recursos multimedia e hipermedia sirven mucho para establecer conclusiones especialmente en las asignaturas de matemáticas, física e inglés y un 10% de docentes dicen que los recursos multimedia e hipermedia ayudan poco a establecer conclusiones en las asignaturas de lenguaje y literatura.

5. Motivar y mantener el interés.

La importancia de despertar el interés y motivar al alumno durante en el proceso de enseñanza nos llevó a formular la pregunta a los docentes de la U.N.L. ¿los recursos multimedia e hipermedia motivan y mantienen el interés en el estudiante?

Un 93% de docentes valoran los recursos hipermedia para motivar y mantener la atención de los estudiantes universitarios en las asignaturas de inglés, comunicación, computación, lenguaje y literatura frente a un 7% de docentes que da bajas valoraciones en las asignaturas de matemáticas, física y química.

6. Mejorar el proceso de aprendizaje.

A la interrogante sobre el uso de nuevas tecnologías implementadas en la educación para mejorar el proceso de aprendizaje, los docentes manifestaron lo siguiente:

Un 89% de docentes expresan que las nuevas tecnologías son un aporte valioso en la educación debido a que ofrecen una aproximación informática a entornos de enseñanza, frente a un 11% de docentes que consideran que la hipermedia no mejora el proceso de aprendizaje en las asignaturas de matemáticas, física y química.

7. Actualizar los conocimientos.

A los docentes de las distintas áreas de la U.N.L se les indago, acerca de que si los documentos hipermedia pueden colaborar para que los estudiantes actualicen sus conocimientos, los resultados obtenidos fueron:

Un 100% de docentes opinan que gracias a la diversidad de información que contienen los documentos hipermedia los estudiantes pueden actualizar sus conocimientos.

Para conocer la actitud de los estudiantes de las distintas áreas de la U.N.L con respecto a la utilización de los recursos hipermedia en el proceso de aprendizaje. Se realizó el siguiente estudio.

1. Instrucción personalizada.

La opinión de los estudiantes acerca de que puedan tener una instrucción personalizada mediante el uso de los documentos hipermedia, fue:

Un 75% de los estudiantes universitarios consideran que son capaces de aprender más cuando puede controlar el curso de su aprendizaje y 25% afirman que necesitan la supervisión de los docentes para el análisis de los contenidos.

2. Flexibilidad tanto de tiempo como de espacio.

Las respuestas sobre si los documentos hipermedia proporcionan al estudiante una flexibilidad tanto de tiempo como de espacio, fueron las siguientes.

Un 87% de los estudiantes sostienen que los documentos hipermedia les permite organizar su estudio de la forma más conveniente según su manera de aprender, mientras un 13% considera que no debido a que no cuentan con el tiempo necesario para aprender con los documentos.

3. Mejorar la retención.

Se formuló la pregunta a los estudiantes de la U.N.L ¿consideran que los recursos multimedia e hipermedia mejoran la retención? y se obtuvo los siguientes resultados.

Un 90% de los estudiantes manifiestan que los elementos como animación, interacción, sonido, imágenes y texto, favorecen a la obtención de mejores resultados y una mayor retención de lo aprendido al participar en el proceso de aprendizaje un mayor número de capacidades sensoriales y un 10% aseguran que los recursos hipermedia no mejoran la retención.

4. Motivar y mantener el interés.

Los resultados obtenidos sobre si el uso de los recursos multimedia e hipermedia motivan y mantienen interés el estudiante, se presentan a continuación:

Un 88% de estudiantes consideran que los recursos hipermedia si motivan y mantienen su atención, frente a un 12% de estudiantes que opinan lo contrario.

5. Potencializar habilidades

Con respecto a sí los documentos hipermedia le permiten al estudiante potencializar sus habilidades, las respuestas fueron:

Un 92% de los estudiantes sostienen que los documentos hipermedia si les ha permitido mejorar sus habilidades porque permite el desarrollo de su creatividad, mientras un 8% piensa que no los documentos hipermedia no ayudan a desarrollar sus habilidades.

6. Formación a la carta

A la pregunta de ¿los documentos hipermedia que contienen variada información permiten una formación a la carta?, los estudiantes respondieron lo siguiente.

Un 86% de los estudiantes mantienen que la versatilidad que ofrecen los documentos hipermedia facilitan una formación a la carta, mientras un 14% considera que la información no se adapta a su ritmo de aprendizaje.

7. Nuevas Tecnologías

La evaluación de los estudiantes sobre si la utilización de las nuevas tecnologías produciría una mejora en el ámbito educativo, reveló los siguientes resultados.

Un 98% de los estudiantes manifiestan que la integración de nuevas tecnologías les ayudará a una actualización ágil y rápida de los contenidos mejorando así su aprendizaje y un 2% sostiene que no ayudaría mucho porque no ajusta a todas asignaturas.

De todos los datos obtenidos se concluye que los docentes y estudiantes de la Universidad Nacional de Loja consideran a los recursos hipermedia como una contribución valiosa a la innovación educativa.

Para lograr mejoras académicas en la universidad se hace necesario aplicar nuevos métodos de comunicación de información. Siendo la computación una de las áreas más importantes, que impulsa notablemente el desarrollo de todas las ciencias tecnológicas. El docente puede beneficiarse del potencial que tiene el computador y las tecnologías de multimedia para enriquecer el proceso educativo. Basándose en esto se crea este sistema que pretende facilitar un medio donde los docentes de la universidad puedan generar material complementario que incluya y expanda los contenidos de la asignatura con una presentación y secuenciación dinámica. Además el docente podrá proveer una cierta variedad de técnicas atractivas, para mantener la atención del estudiante y facilitar la transmisión del conocimiento. Impulsando e incentivando al estudiante a que busque su propia libertad a través del conocimiento científico.

La formación y la experiencia del docente en el uso de las tecnologías con fines pedagógicos resultan una variable clave en el éxito del docente. No se pretende modificar la forma de impartir la asignatura por parte del docente sino se ofrece la posibilidad de una aproximación informática desde una perspectiva técnica en la creación de documentos hipermedia que les permita a los estudiantes aprender de una manera más entretenida, rápida, flexible, etc.

2.2 Descripción del sistema.

El proyecto consiste en una aplicación hecha en Java que está formada por; un administrador de proyectos y un editor de texto. Su principal función es la generación de documentos académicos. Para la elaboración de documentos XML el docente podrá elegir un menú de opciones para ingresar texto, insertar imagen, insertar tablas

y crear vínculos. Sin que el usuario tenga conocimiento previo en hipermedia. Cada documento XML creado en el sistema será validado por un XML-Schema que define los elementos que puede contener el documento, además se manipulará hojas de estilo XSLT y código java para convertir el documento en otros formatos de salida HTML y PDF.

Toda la información que contenga el proyecto se guardará en documentos XML. XML es un lenguaje que permite definir etiquetas semánticas que organizan un documento en diferentes partes. Los archivos generados podrán ser guardados en cualquier dispositivo de almacenamiento como disco duro, CD-ROM, CD-Writer y disco de $3\frac{1}{2}$

.

2.2.1 Metas del sistema

Nuestra meta principal es crear un generador de contenidos académicos hipermedia basados en XML y XSLT, para lograr el cumplimiento de la misma se debe realizar:

- Generar un esquema XML para el control de documentos académicos.
- Definir esquemas XSLT para la transformación de los documentos XML en páginas HTML y documentos PDF.
- Crear una herramienta que permita la creación de páginas Web tomando como fuente documentos XML y XSLT.
- Establecer un mecanismo que permita la incorporación de texto, imágenes e hipervínculos en los documentos académicos hipermedia.

- Proveer a los docentes una herramienta sobre la cual tengan un extenso control para la generación de todos los aspectos de los contenidos por medio de documentos XSLT.

2.2.2 Requerimientos Funcionales

CÓDIGO	REQUERIMIENTO
R01	Crear nuevo proyecto
R02	Abrir proyectos existentes
R03	Guardar proyectos
R04	Ingresar las propiedades del proyecto
R05	Crear documentos XML en un proyecto
R06	Eliminar documentos XML en un proyecto.
R07	Actualizar las modificaciones que el usuario defina en el proyecto.

R08	Manejo de documentos a través de un editor de texto.
R09	Ingresar elementos hipermedia (texto, imagen) considerando el esquema XML propuesto.
R10	Eliminar elementos hipermedia de los documentos XML.
R11	Actualizar las modificaciones realizadas en documento XML.
R12	Utilizar esquemas de salida XSLT.
R13	Generar páginas HTML a partir de documentos XML y archivos XSLT definidos.
R14	Generar documentos PDF a partir de documentos XML y archivos XSLT definidos.
R15	Almacenamiento físico de las páginas HTML y documentos PDF.

2.2.3 Atributos del sistema.

CÓDIGO	ATRIBUTO	DETALLES Y RESTRICCIONES DE FRONTERA
A01	Plataforma.	Windows, Linux.
A02	Monousuario.	Manejo de una sola persona.
A03	Manipulación	Fácil navegación a través de teclado y Mouse.

A04	Ayuda	Tutor para el manejo de la herramienta.
A05	Documentos	Se podrá crear varios documentos a la vez.
A06	La interfaz de usuario será amigable.	El sistema presentará ventanas y cuadros de dialogo de fácil navegación.
A07	Tiempo de respuesta.	Las acciones del sistema serán inmediatas.

2.2.4 Glosario de términos.

TERMINO	**DESCRIPCIÓN**
ComponentesArbol	Permite manejar y almacenar los tutores que van ha ser administrados por el proyecto.
Docente (Actor)	Es el usuario que da inicio a las acciones del sistema.
DocumentoHTML	Es el código que expresa el significado de los datos que sé autodefinen, exclusivamente en el editor.
EquipoEditor	Maneja estándares de edición como cortar, copiar, pegar e insertar caracteres.

Pagina	Conjunto de normas para describir un documento hipermedia.
Proyecto	Contiene todos los elementos que se administran en el sistema para la generación de componentes.
Tutor	Es aquel que contiene todas las páginas generadas.
VistaPagina	Es el editor que permite al docente ingresar todos los datos (texto, imágenes, vínculos, tablas, etc.) que van ha contener el documento XML.

2.2.5 Modelo de dominio del sistema

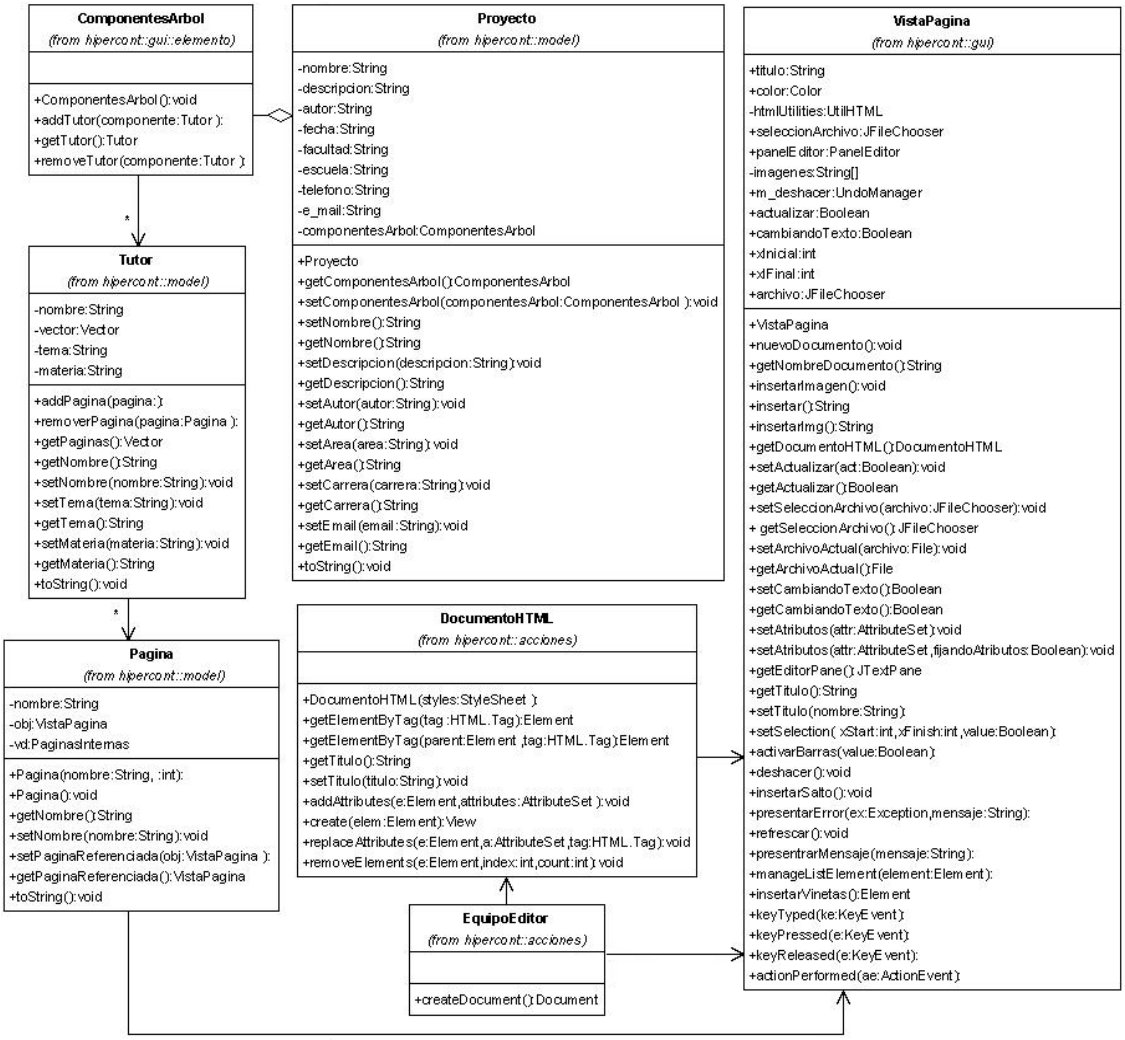


Figura 2.1: **Modelo de dominio del sistema Hipercont**.

2.3 Modelado y descripción del use case del sistema

2.3.1 Diagrama de casos de usos del sistema Hipercont.

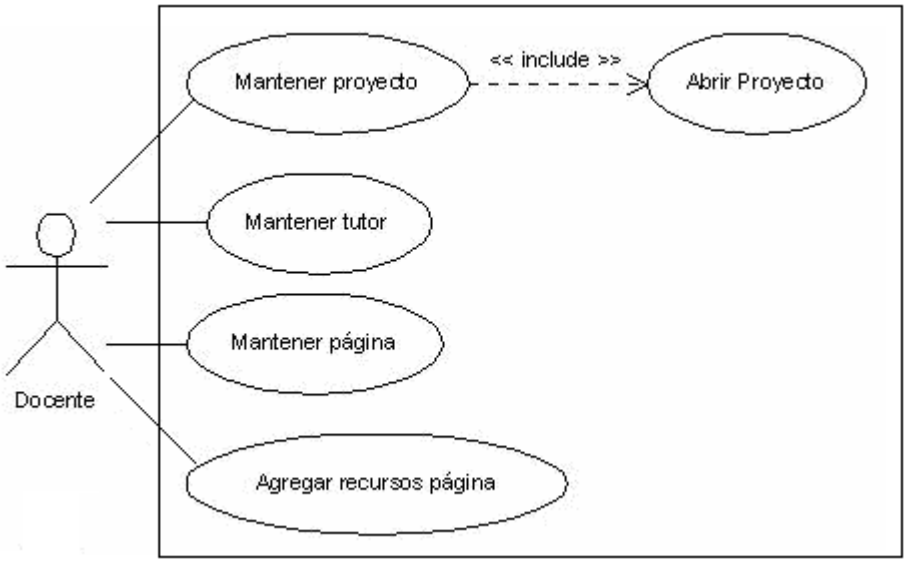


Figura 2.2: ***Diagrama de casos de usos.***

2.3.2 Descripción del use case de hipercont

- Mantener proyecto
- Mantener tutor
- Mantener página
- Agregar recursos página

Abrir proyecto

docente

UseCase Mantener proyecto **Include:** Abrir proyecto

Name	Mantener proyecto
Autor	Docente
Brief Description	Modificar las propiedades de un proyecto, el sistema presenta la ventana **"PropiedadesProyecto"**, el docente realiza los cambios y el sistema los actualiza y almacena. Crear un Nuevo proyecto, el sistema verifica si el proyecto actual ha sido almacenado de no ser así el sistema pide la confirmación de la acción. Si el docente acepta. El sistema lo guarda en la ruta especificada y crea un Nuevo Proyecto.
Referencia	R01, R02, R03, R04, R07.
Pre-conditions	Se debe haber ingresado al sistema. Estar trabajando en un proyecto. Haber guardado un proyecto.
Post- conditions	Las propiedades se modifican y quedan almacenadas. El proyecto queda listo para trabajar en él.

	El docente decide crear un tutor. El docente decide abrir un proyecto existente. Las propiedades del proyecto no se modifican se cancela.

Flow of Events		
Actor input		**System response**
0	El docente escoge el proyecto en **“ArbolProyecto”** y con click derecho presiona el botón **"Modificar"** propiedades.	
1		El sistema presenta la ventana **"PropiedadesProyecto".**
2	El docente modifica las propiedades del proyecto y presiona el botón **"Aceptar".**	
3		El sistema valida la información modificada.
4		El sistema guarda las propiedades del proyecto en **“Proyecto”** y actualiza el **“AdministradorProyecto”** y el **“ArbolProyecto”.**

Cursos Alternos		
0	a) El docente desea crear un nuevo proyecto y presiona **Nuevo Proyecto** en el menú de la "**VentanaPrincipal**". El sistema presenta un mensaje para "Guardar" el proyecto actual en la "**VentanaConfirmación**". El docente confirma la acción y el sistema lo guarda en la dirección especificada y continua con el paso 1. b) El docente presiona Abrir un proyecto. Llama al use case **Abrir proyecto.**	
3	Las propiedades del proyecto ingresadas no son validas. Se continúa con el paso 1.	

Prototipos Mantener Proyecto

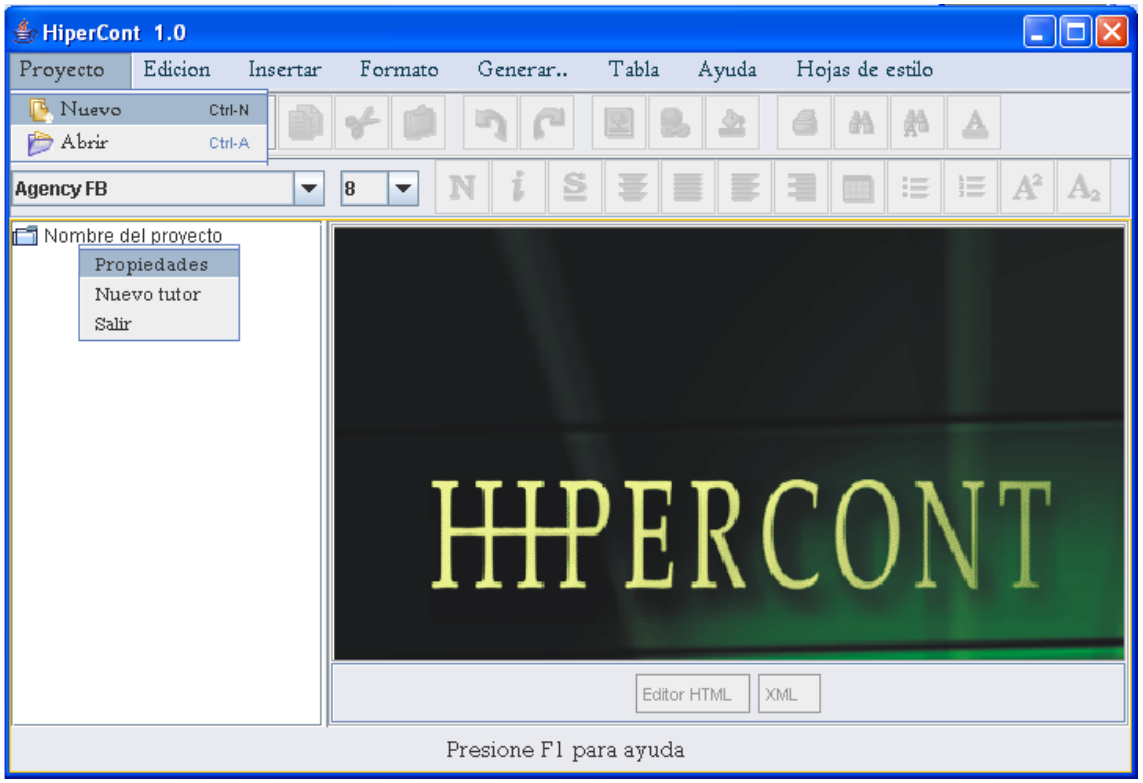


Figura 2.3: ***Prototipo Mantener Proyecto***

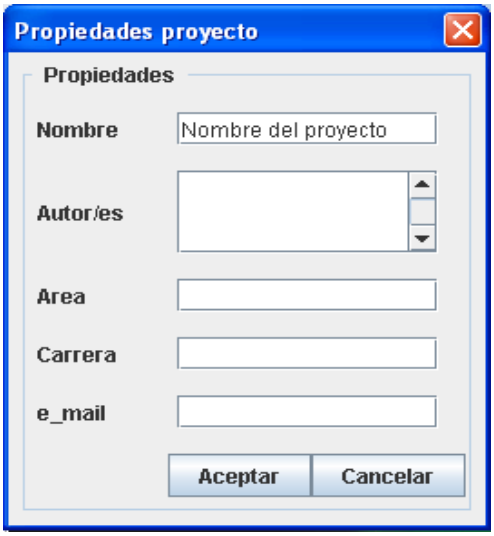


Figura 2.4: **Ventana Propiedades Proyecto**

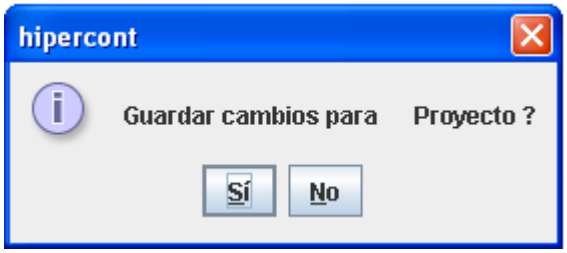


Figura 2.5: **Ventana Confirmación "Guardar"**

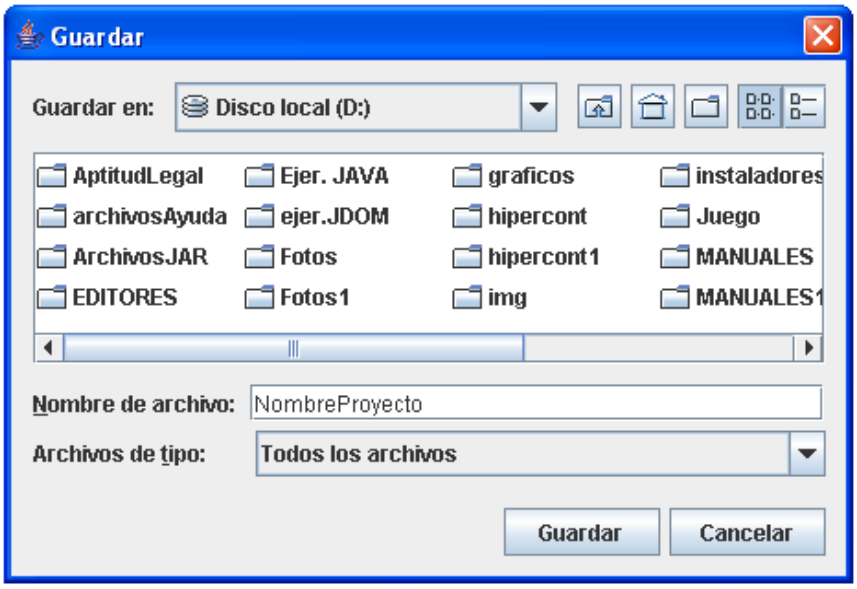


Figura 2.6: **Ventana "Guardar".**

Diagrama de colaboración **"Mantener Proyecto"**

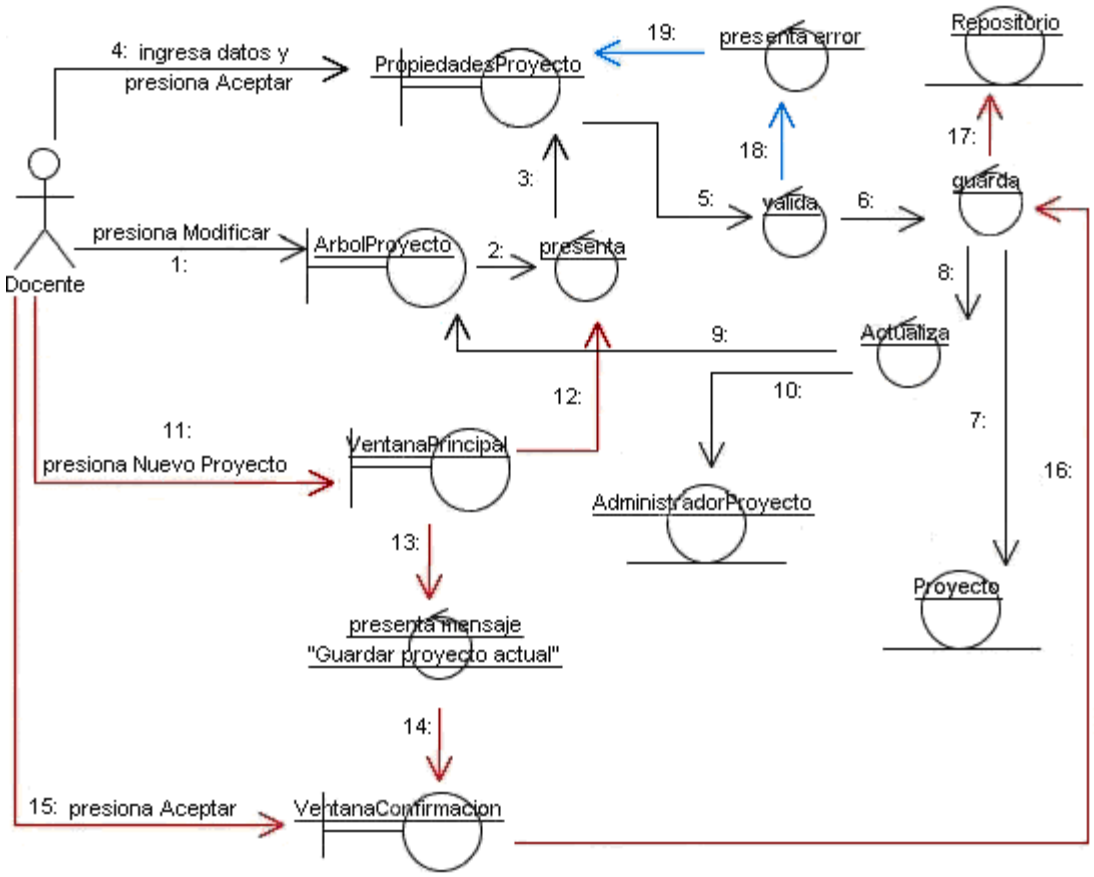


Figura 2.7: **Mantener Proyecto.**

Diagrama de secuencia **"Mantener Proyecto"**

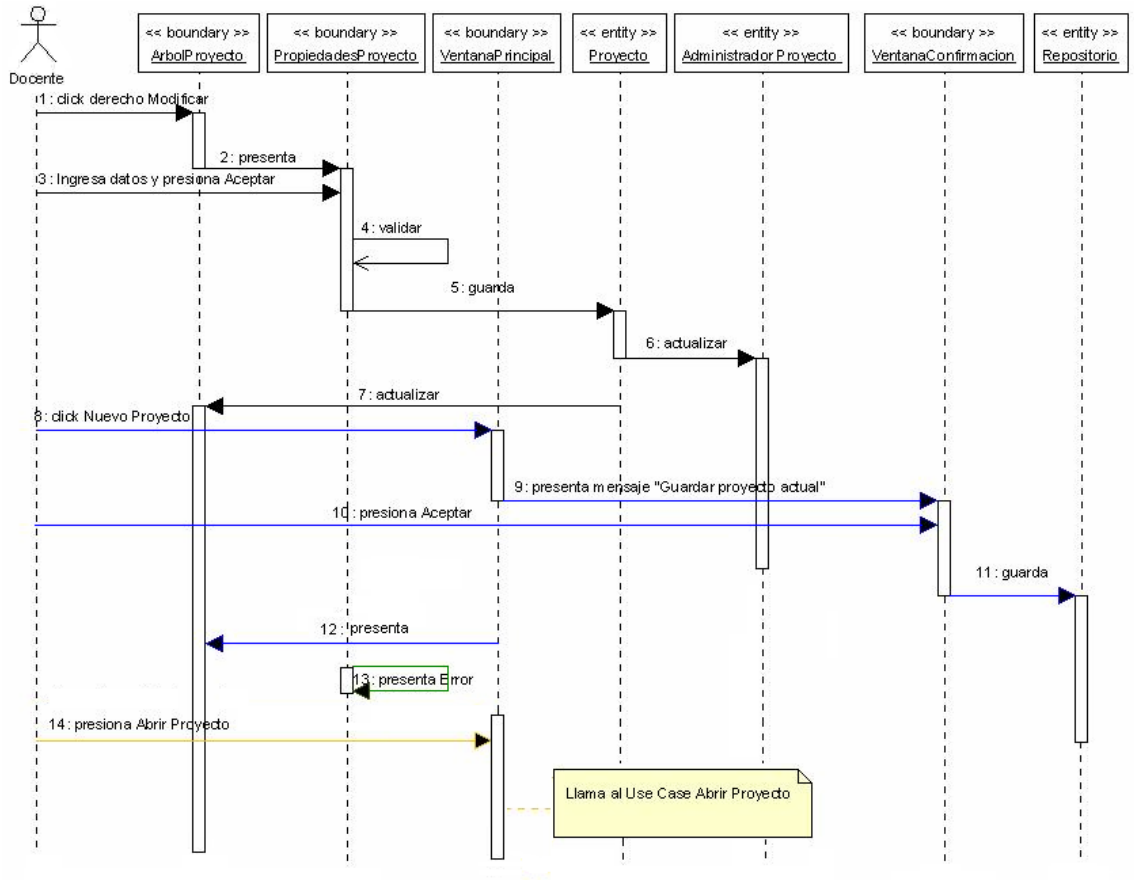


Figura 2.8: **Mantener Proyecto.**

UseCase Mantener tutor

Name	Mantener tutor.
Autor	Docente.
Brief Description	Permite al docente modificar las propiedades, crear y eliminar tutores en el proyecto que se este trabajando. Para realizar cambios en las propiedades de un tutor. El sistema presenta la ventana **"PropiedadesTutor".** El docente realiza los cambios. El sistema los actualiza y guarda. Al crear un Nuevo tutor. El sistema presenta la ventana **"PropiedadesTutor".** El docente ingresa los datos. El sistema los guarda. Cuando se va ha Eliminar un tutor**.** El sistema pide la confirmación de la acción. El docente confirma y el sistema elimina el componente seleccionado y actualiza el proyecto**.**
Referencia	R07.
Pre-conditions	Estar trabajando dentro de un proyecto. Haber creado un tutor.
Post-conditions	El tutor queda listo para trabajar en él. El docente crea un nuevo tutor. El docente decide eliminar el tutor.

Flow of Events		
	Actor input	**System response**
0	El docente en el **“ArbolProyecto”** elige un tutor de un proyecto y con click derecho selecciona **"Modificar"** propiedades del tutor.	
1		El sistema presenta la ventana **"PropiedadesTutor".**
2	El docente ingresa la nueva información y presiona el botón **"Aceptar".**	
3		El sistema valida los datos ingresados.
4		El sistema guarda las propiedades del tutor en **“Proyecto"** y actualiza el **“AdministradorProyecto"** y el **“ArbolProyecto”.**
Cursos alternos		

0	a) El docente decide crear un Nuevo tutor en un proyecto. Con un click derecho en el **“ArbolProyecto”** selecciona “Nuevo tutor” y continúa con el paso 1. b) El docente selecciona un tutor dentro de un proyecto y presiona con click derecho **“Eliminar”** tutor en **“ArbolProyecto”.** El sistema pide la confirmación de la acción en la **"VentanaConfirmación"**. El docente confirma y el sistema elimina el componente seleccionado en “**Proyecto”** y finalmente se actualiza el **"AdministradorProyecto"** y **“ArbolProyecto”.**
3	Las propiedades del Tutor no son válidas y continúan con el paso 1.

<u>Prototipos Mantener Tutor</u>

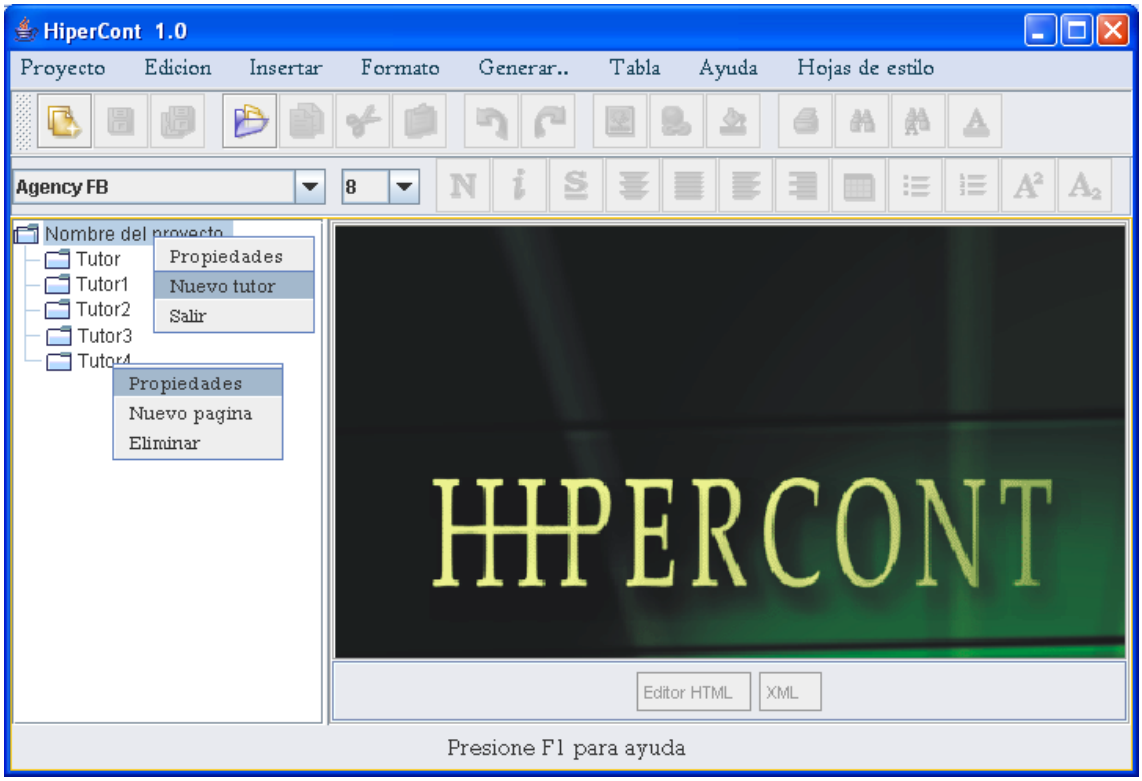


Figura 2.9: **Prototipo Mantener Tutor**

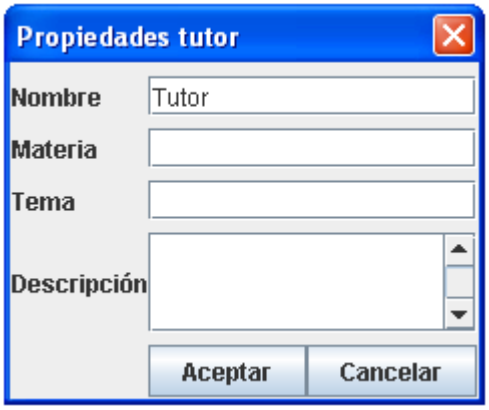


Figura 2.10: **Ventana Propiedades Tutor**

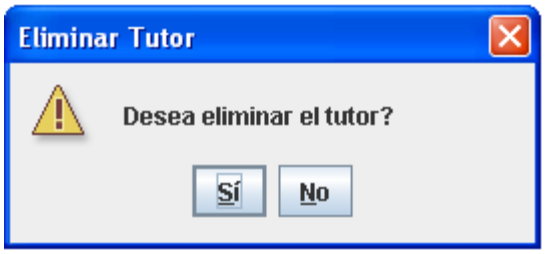


Figura 2.11: **Ventana Confirmación "Eliminar Tutor"**

Diagrama de colaboración *"Mantener Tutor"*

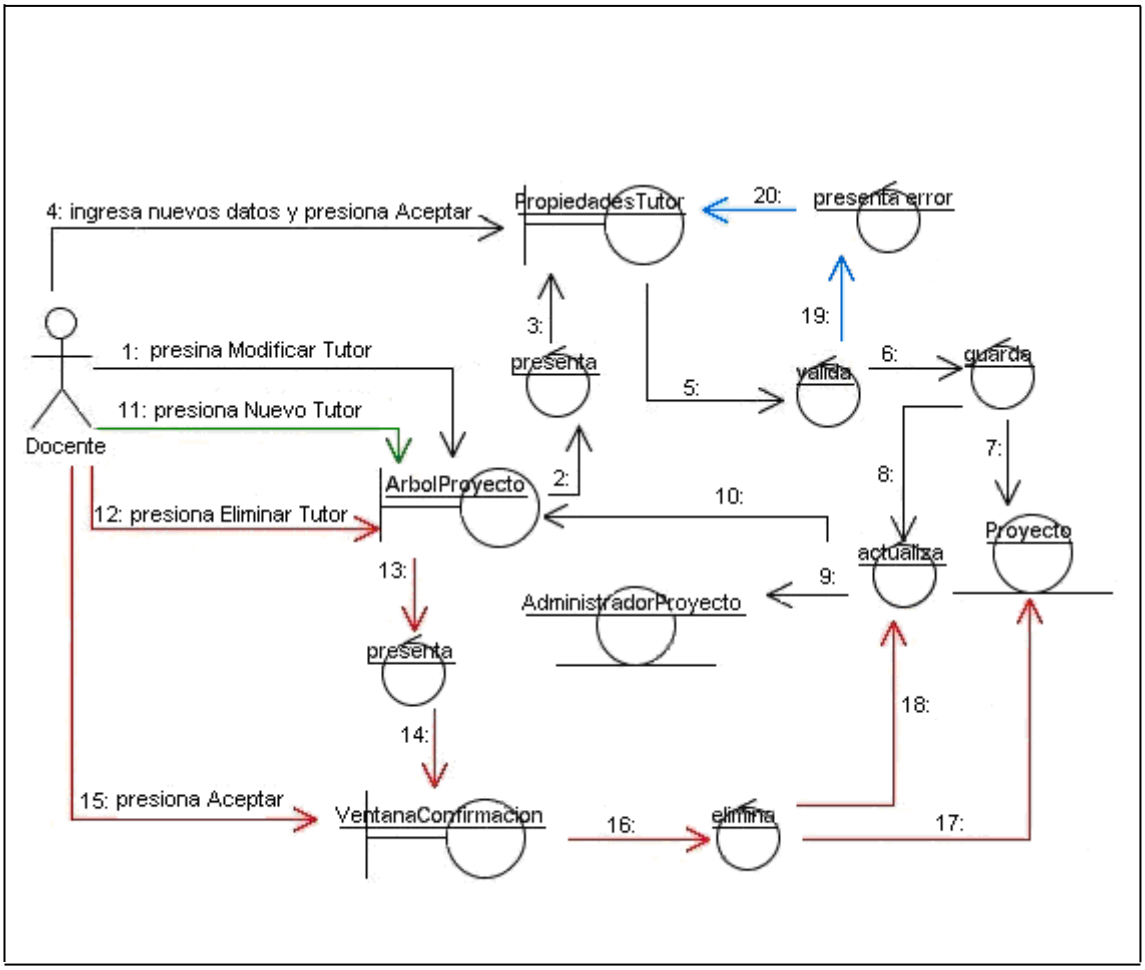


Figura 2.12: **Mantener Tutor.**

***Diagrama de secuencia* "Mantener tutor"**

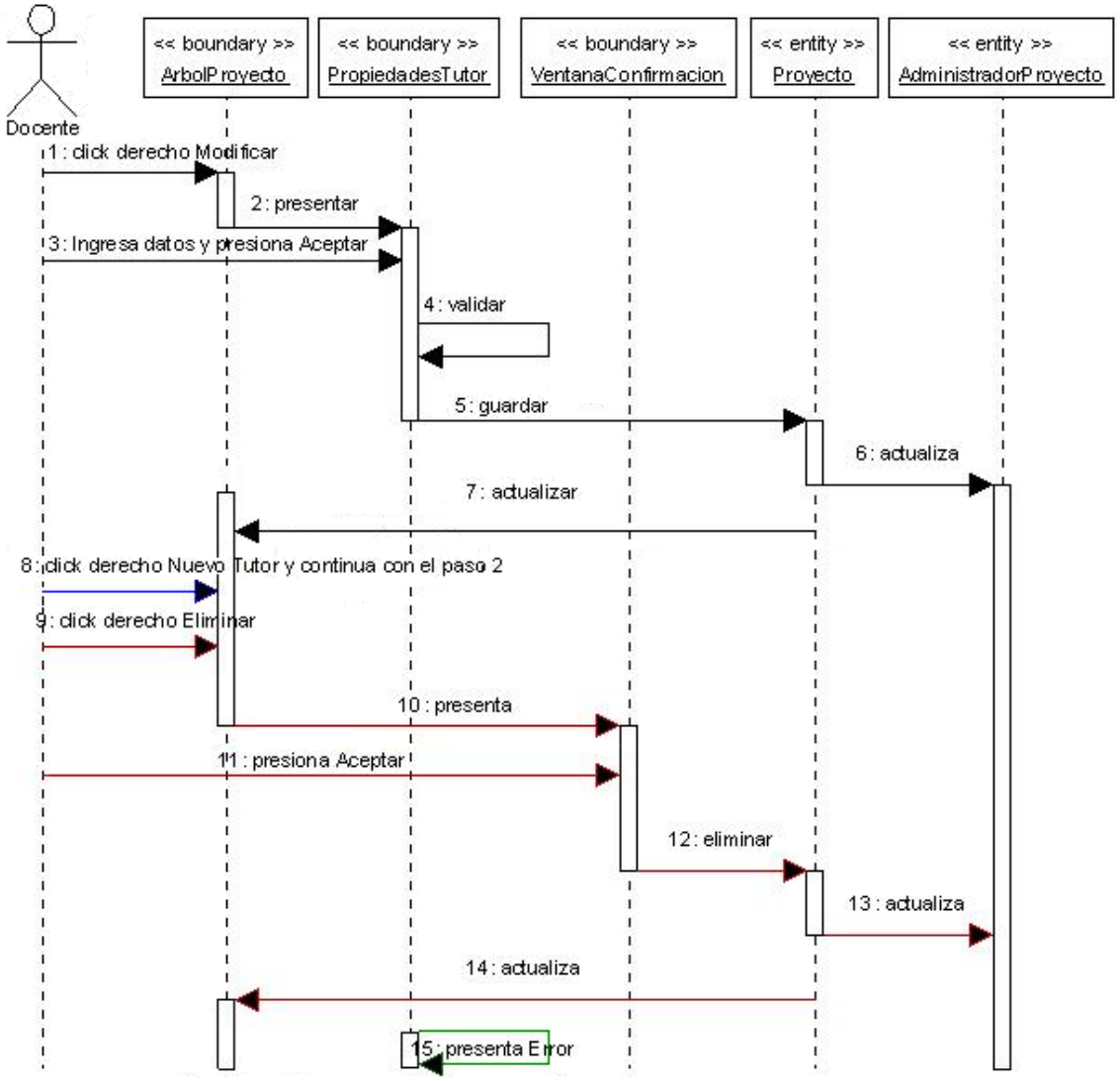


***Figura 2.13:* Mantener Tutor.**

UseCase Mantener página

Name	Mantener página
Author	Docente.
Brief Description	El docente puede cambiar el nombre, crear y eliminar una página. En caso de cambiar el nombre de una página. El sistema solicita el nuevo nombre. El docente lo ingresa y el sistema verifica que el nombre no se repita en otras páginas, guarda y actualiza. Para crear una nueva página. El sistema solicita el nombre. El docente ingresa el dato. El sistema valida, guarda y actualiza. Si se elimina una página del tutor. El sistema pide la confirmación de la acción. Cuando el docente ratifica el sistema elimina la página seleccionada y actualiza el tutor.
Referencia	R05, R06.
Pre-conditions	Se debe haber seleccionado un tutor. Se debe haber creado una página. Estar trabajando en una página.
Post-conditions	La página queda lista para trabajar en ella. El docente crea una nueva página. El docente decide eliminar la página.

Flow of Events		
Actor input		**System response**
0	El docente selecciona una página del **“ArbolProyecto”** y presiona click derecho en **"Modificar"**.	
1		El sistema despliega una ventana **“PropiedadesPagina".**
2	El docente ingresa el nuevo nombre de la página y presiona **"Aceptar".**	
3		El sistema valida el dato ingresado.
4		El sistema guarda los cambios de la página en **Tutor** y actualiza el "**AdministradorProyecto**" y el “**ArbolProyecto**”.
Cursos alternos		
0	a) El docente decide crear nueva página, selecciona un tutor en el **“ArbolProyecto”** y presiona un click derecho **Nueva página** y continúa	

	con el paso 1. b) El docente decide eliminar una página del tutor. El docente presiona un click derecho en "**Eliminar**" página en "**ArbolProyecto**". El sistema pide la confirmación de la acción en la "**Ventana Confirmación**". Cuando el docente ratifica el sistema elimina la página seleccionada del tutor en el que se esta trabajando y actualiza el entorno de trabajo.
3	El sistema no guarda el dato ingresado y continúa con el paso 1.

Prototipos Mantener Página

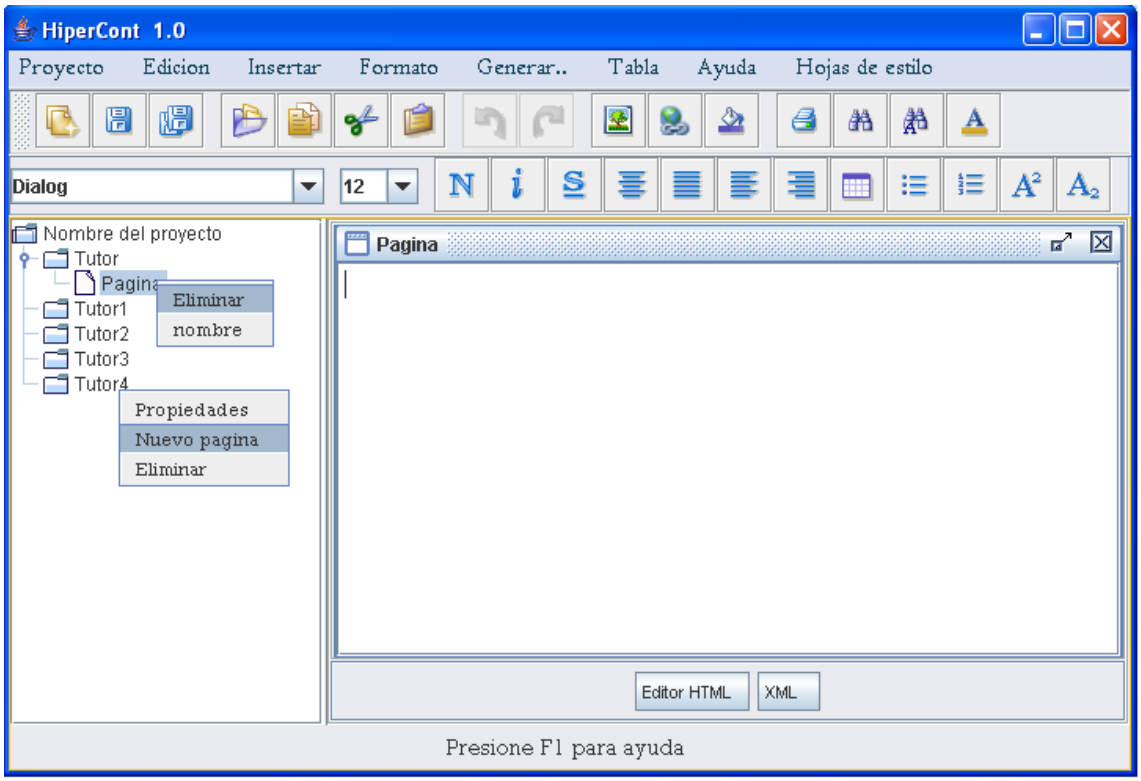


Figura 2.14: **Prototipo Mantener Página**

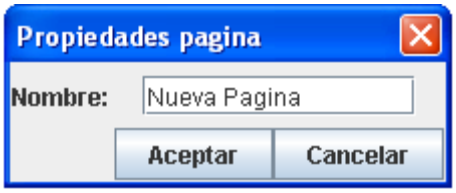


Figura 2.15**: Ventana Propiedades Página**

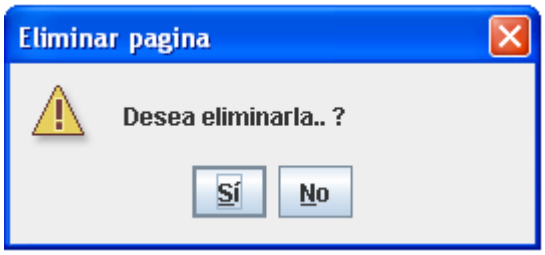


Figura 2.16: **Ventana Confirmación "Eliminar Página".**

Diagrama de colaboración *"Mantener página"*

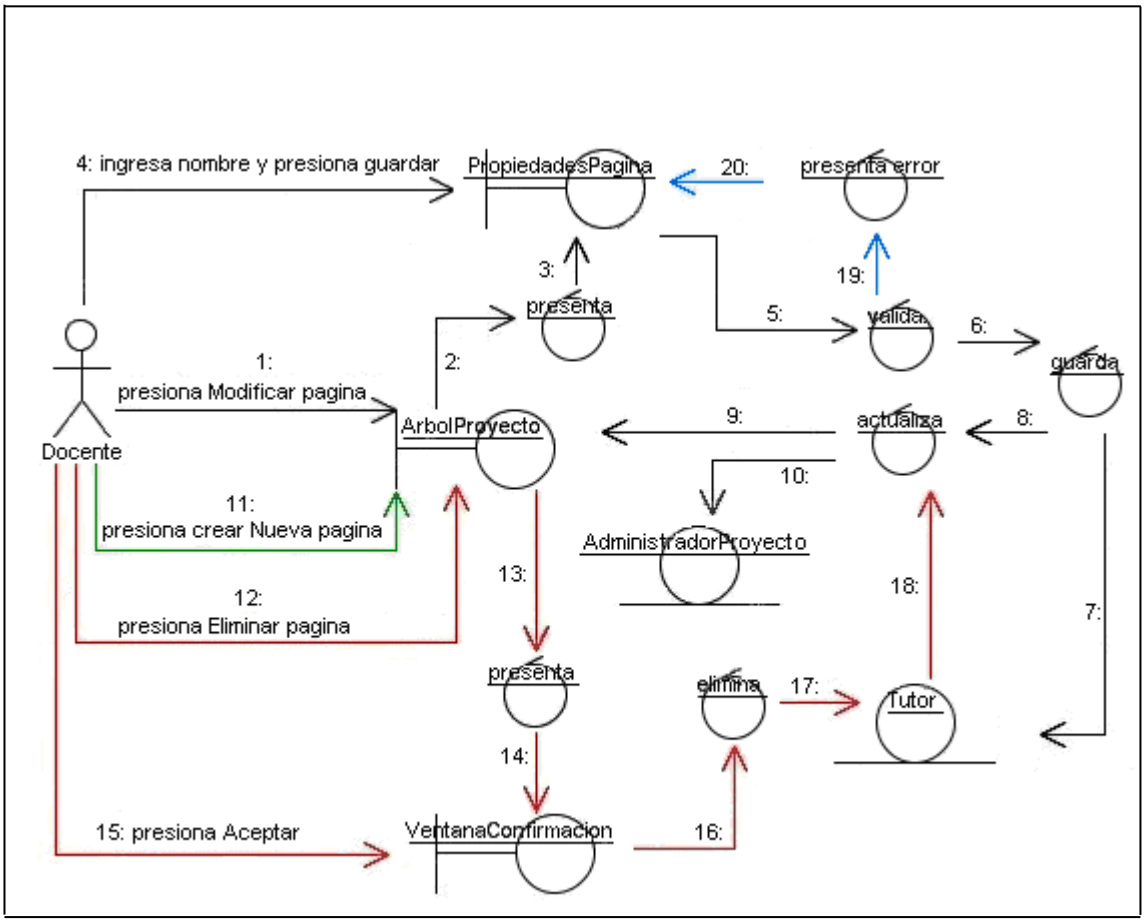


Figura 2.17: **Mantener Página.**

Diagrama de secuencia **"Mantener página"**

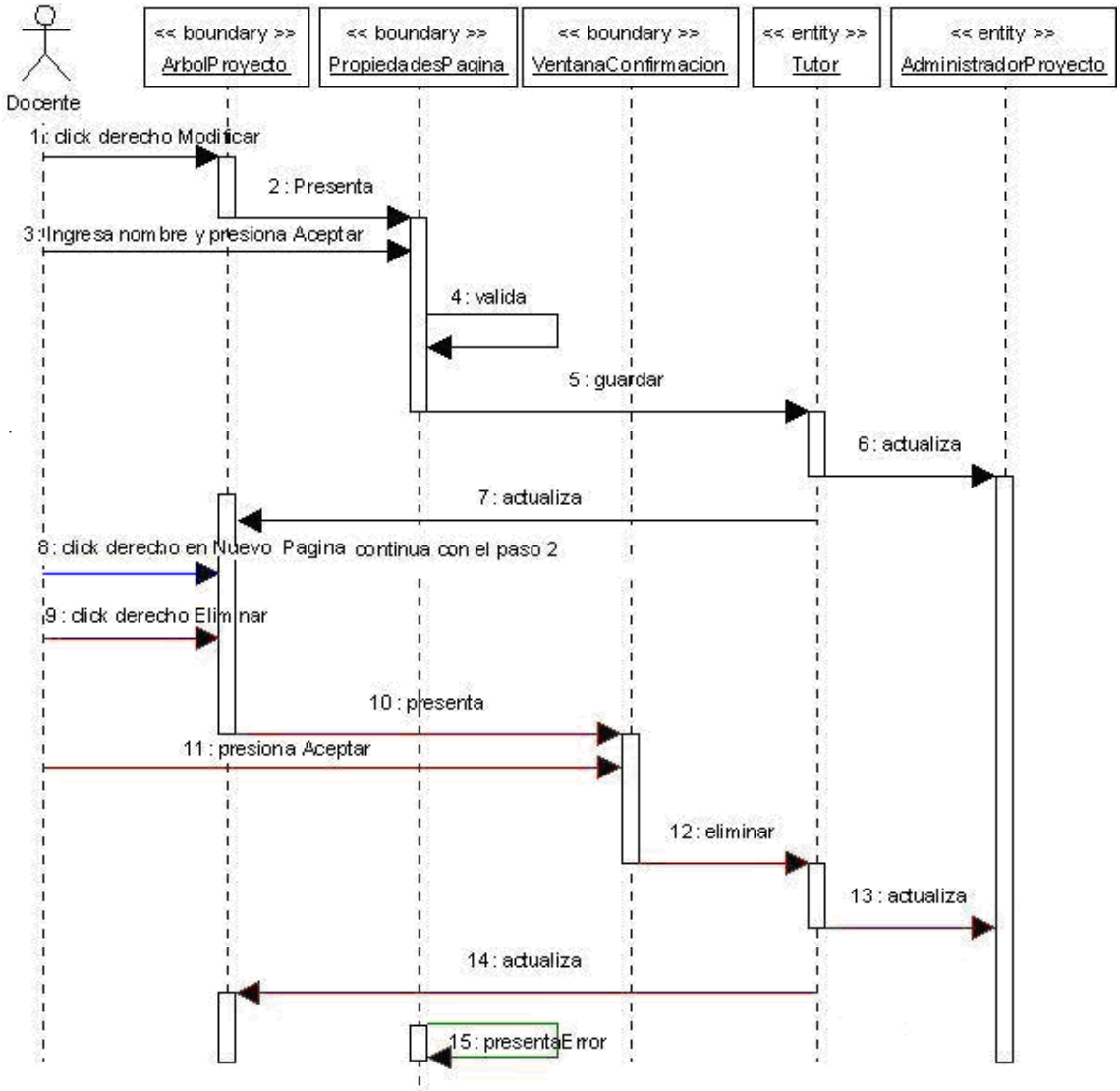


Figura 2.18: **Mantener Página.**

UseCase Agregar recursos página

Name	Agregar recursos página
Author	Docente
Brief Description	El docente decide insertar recursos hipermedia en la página. El sistema pide la ruta y el nombre del archivo a insertar. El docente ingresa los datos del archivo y el sistema inserta el archivo en la página seleccionada.
Referencia	R08, R09, R10, R11.
Pre-conditions	Estar trabajando en una página.
Post-conditions	El elemento queda insertado en la página.

Flow of Events

	Actor input	**System response**
0	El docente presiona "Insertar" (Hipervínculo, imagen) en la Barra menú de la “**VentanaPrincipal**”.	
1		El sistema presenta la ventana **"InsertarArchivo**" para ingresar la dirección y el nombre del elemento.
2	El docente selecciona el elemento a	

	insertar y presiona "**Insertar**".	
3		El sistema verifica la existencia del archivo en el **“Repositorio”**.
4		El sistema carga el archivo seleccionado y lo despliega en la "**VistaPagina**".
Cursos alternos		
0	El docente selecciona “Tabla” en la **“BarraMenu”.** El sistema presenta la ventana insertar tabla. El docente selecciona los datos y presiona Insertar. El sistema despliega la tabla en **“VistaPagina”.**	
2	El docente escribe la ruta del elemento y presiona “Aceptar” y continúa con el paso 3.	
3	a) El sistema presenta un mensaje de error "Formato Desconocido". b) El sistema presenta un mensaje de error "El nombre del archivo no existe".	

<u>Prototipos Agregar Recursos Página</u>

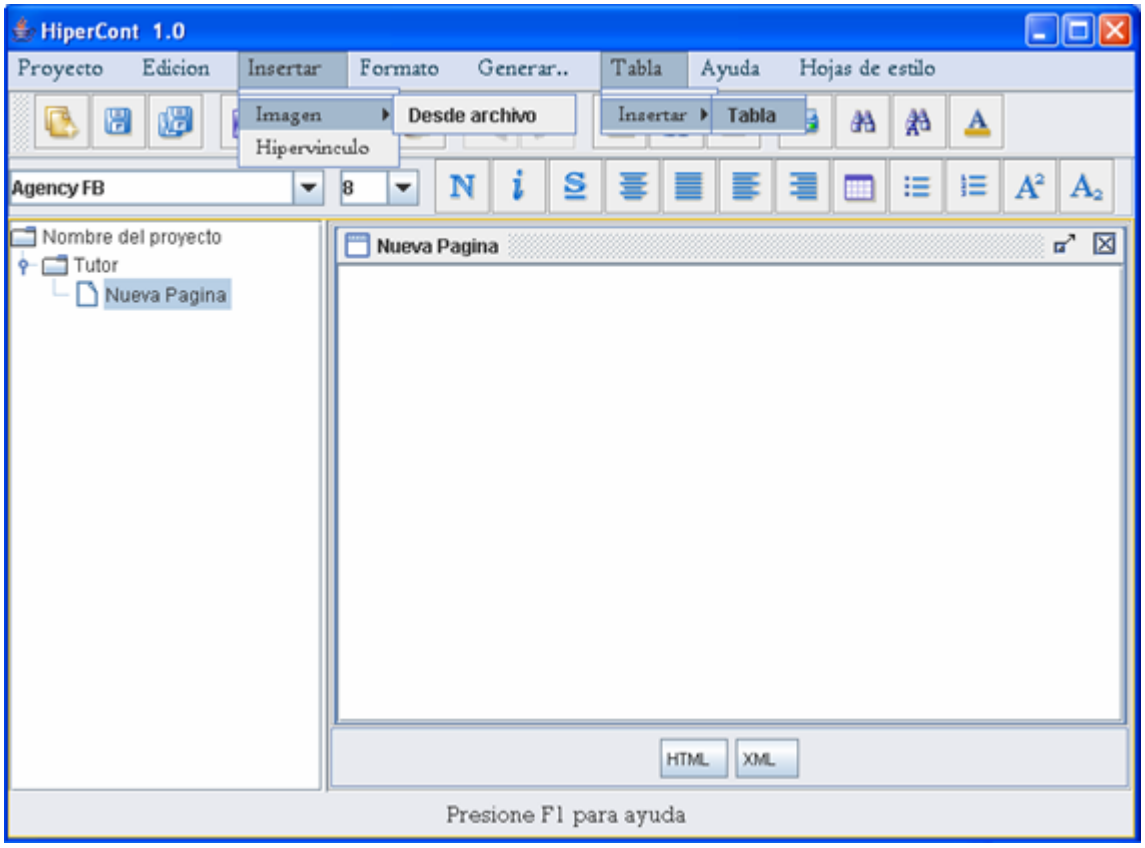


Figura 2.19: **Prototipo Agregar Recursos Página**

Figura 2.20: **Ventana Insertar "Tabla".**

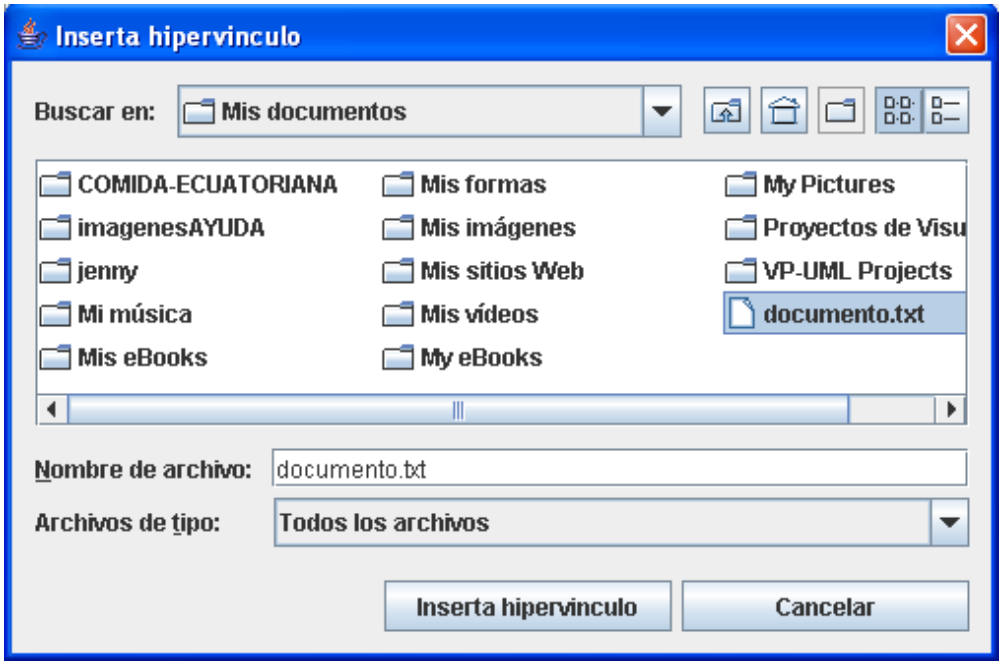


Figura 2.21: **Ventana Insertar "Hipervínculo".**

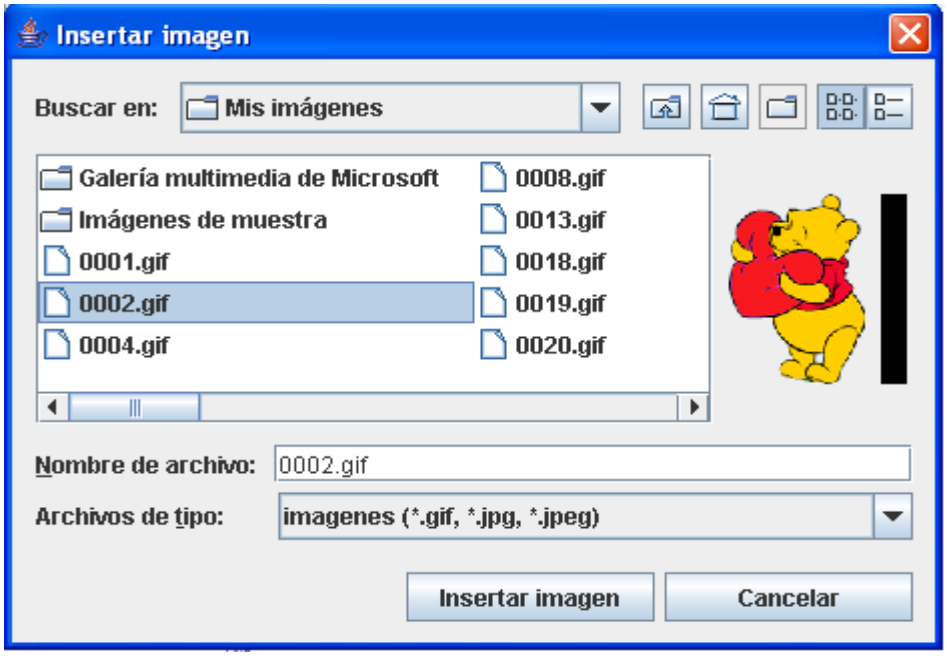


Figura 2.22: **Ventana Insertar "Imagen".**

Diagrama de colaboración **"Agregar recursos página"**

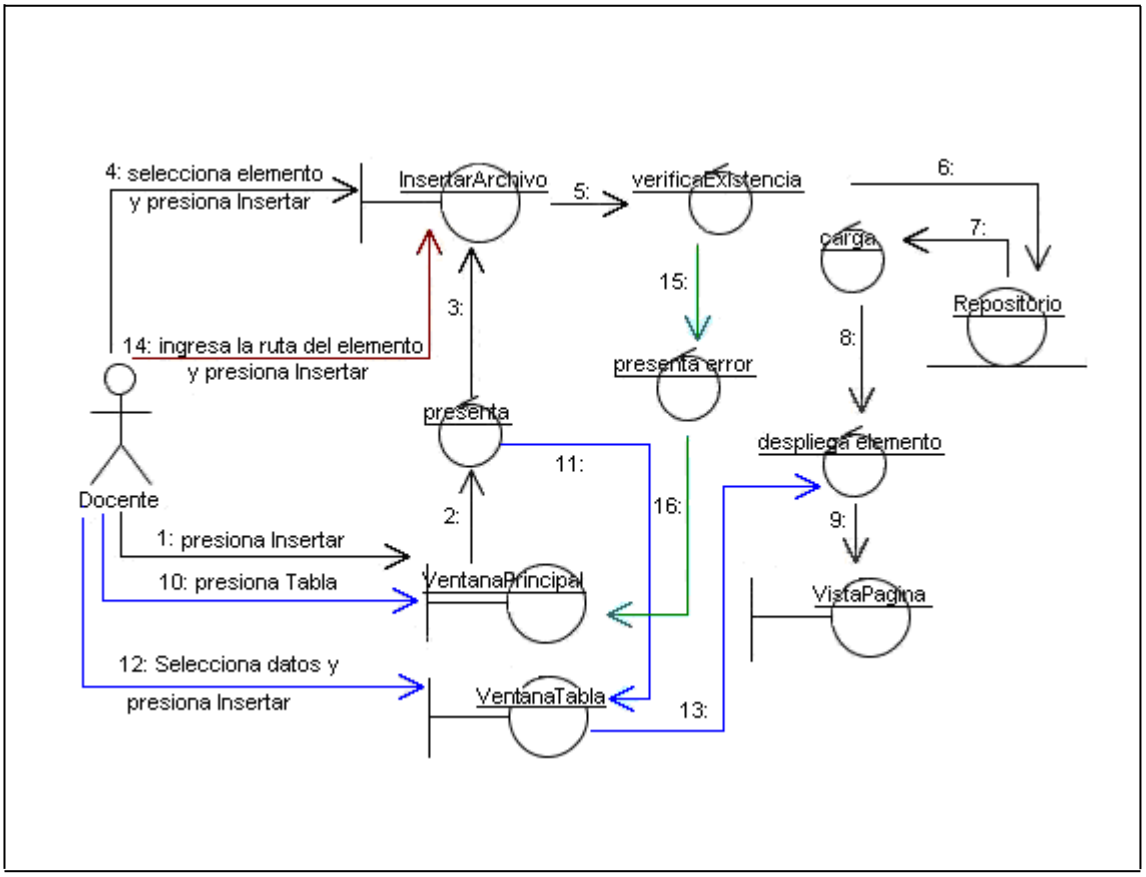


Figura 2.23: **Agregar Recursos Página.**

Diagrama de secuencia **"Agregar recursos página"**

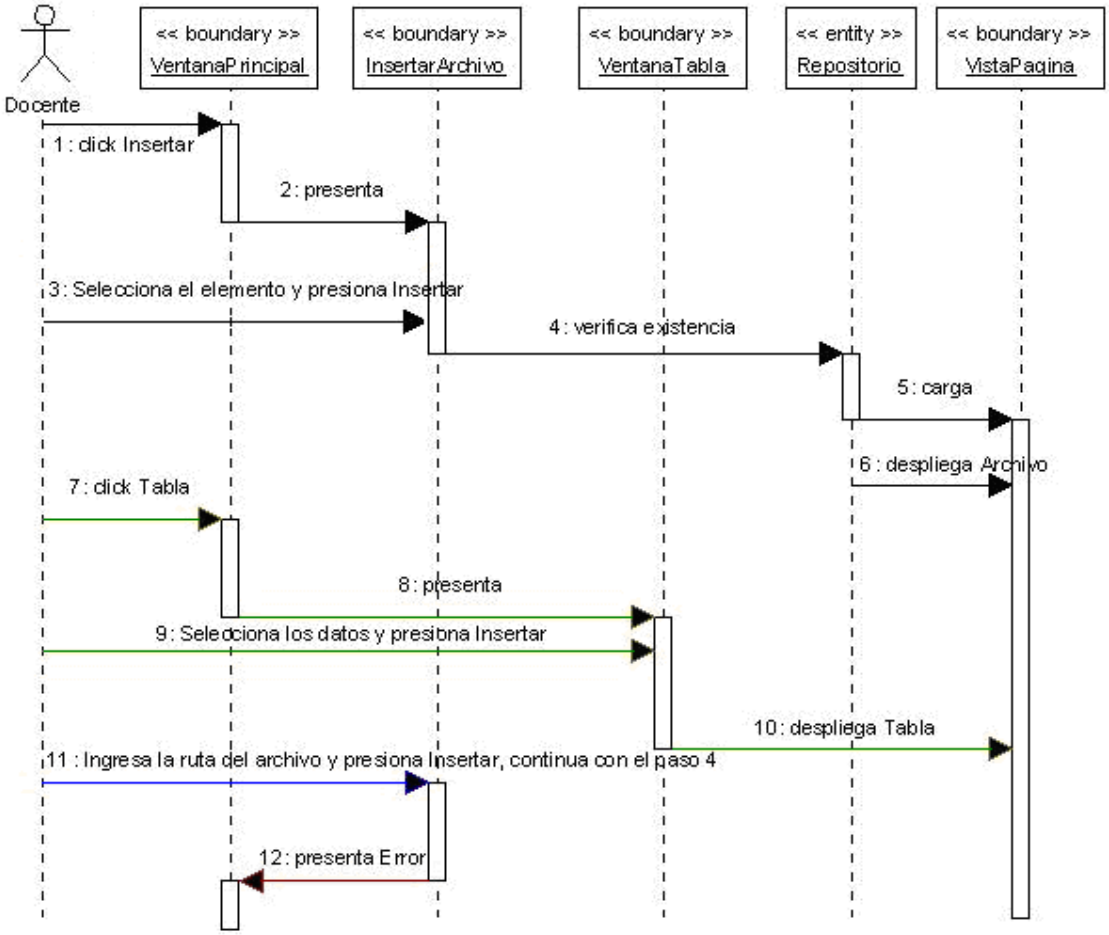


Figura 2.24: **Agregar Recursos Página.**

UseCase Abrir proyecto **Include by:** Mantener proyecto

Name	Abrir proyecto
Author	Docente
Brief Description	El docente decide abrir un proyecto creado previamente. El sistema solicita la ruta y el nombre del archivo. El docente ingresa los datos, confirma la acción y el sistema carga y despliega en la pantalla el archivo seleccionado.
Referencia	R02, R12.
Pre-conditions	Haber ingresado en el sistema. Haber creado un proyecto.
Post-conditions	El proyecto queda listo para trabajar en él.

Flow of Events

Actor input		**System response**
0	El docente elige "**Abrir proyecto**" en **Proyecto** de la Barra menú de la **"VentanaPrincipal".**	

1		El sistema presenta la ventana **“VentanaAbrir”** para ingresar el nombre y la ruta del proyecto.
2	El docente selecciona el archivo y presiona **"Abrir**".	
3		El sistema verifica la existencia del archivo en el **“Repositorio”** y carga el archivo seleccionado en **"Proyecto".**
4		El sistema actualiza el **"AdministradorProyecto"** y el **“ArbolProyecto".**
Cursos alternos		
2	El docente ingresa la ruta del archivo XML y presiona **“Abrir”** y continúa con el paso 3.	
3	El sistema presenta error "No se ha encontrado el archivo".	

Prototipo Abrir Proyecto

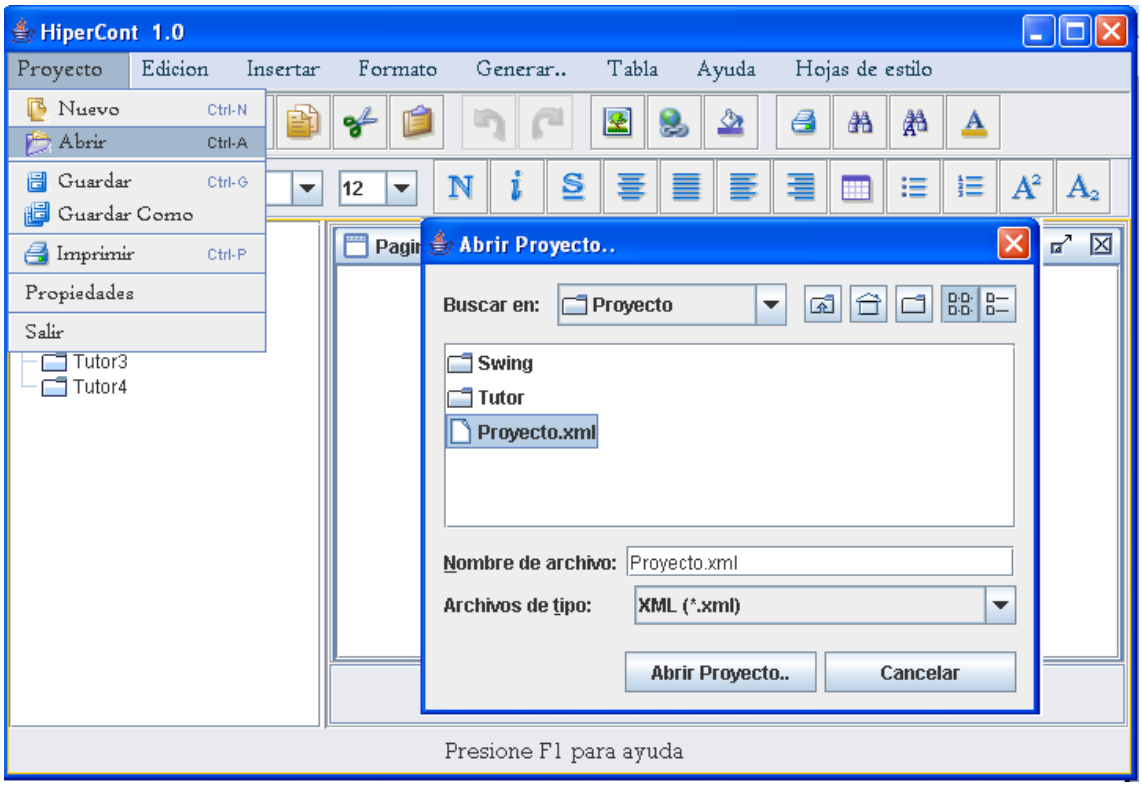


Figura 2.25: **Prototipo Abrir Proyecto.**

Diagrama de colaboración **"Abrir proyecto"**

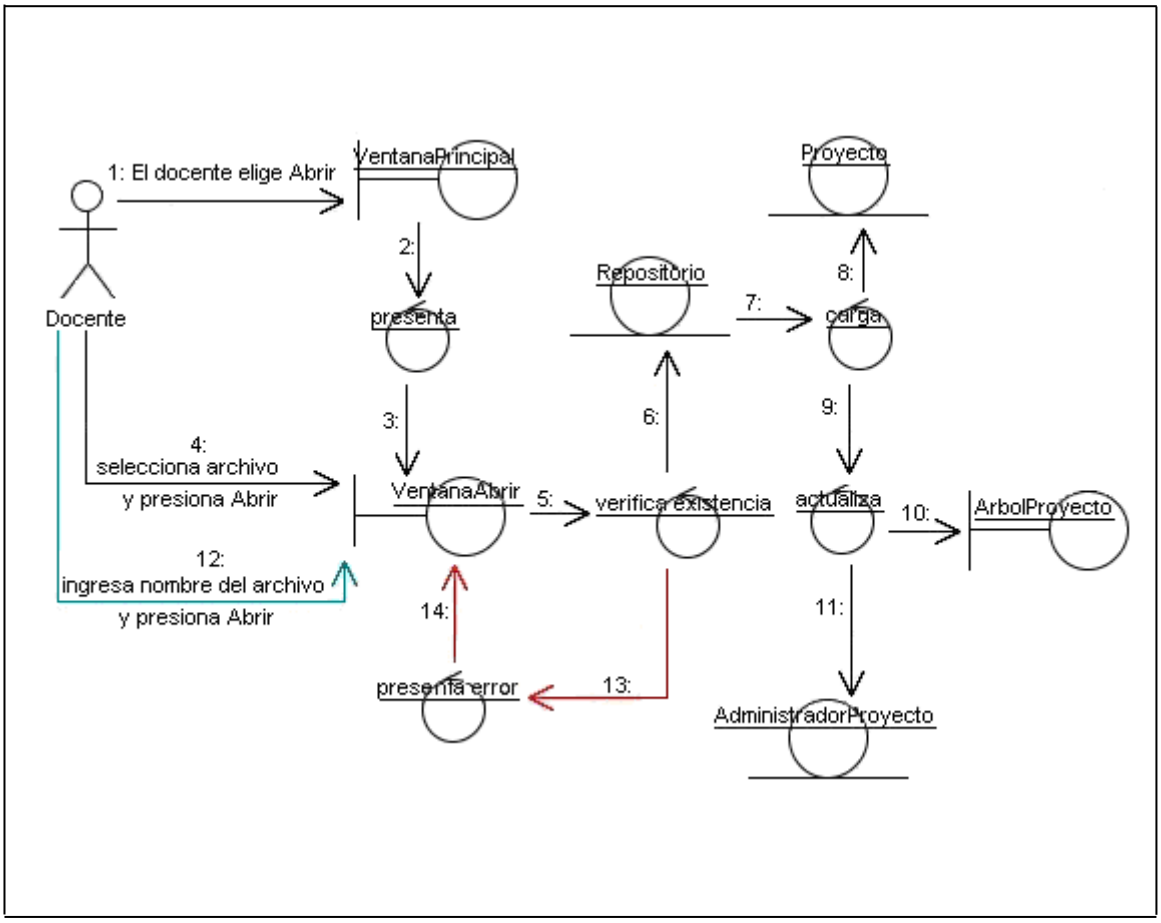


Figura 2.26: **Abrir Proyecto**

Diagrama de secuencia **"Abrir proyecto"**

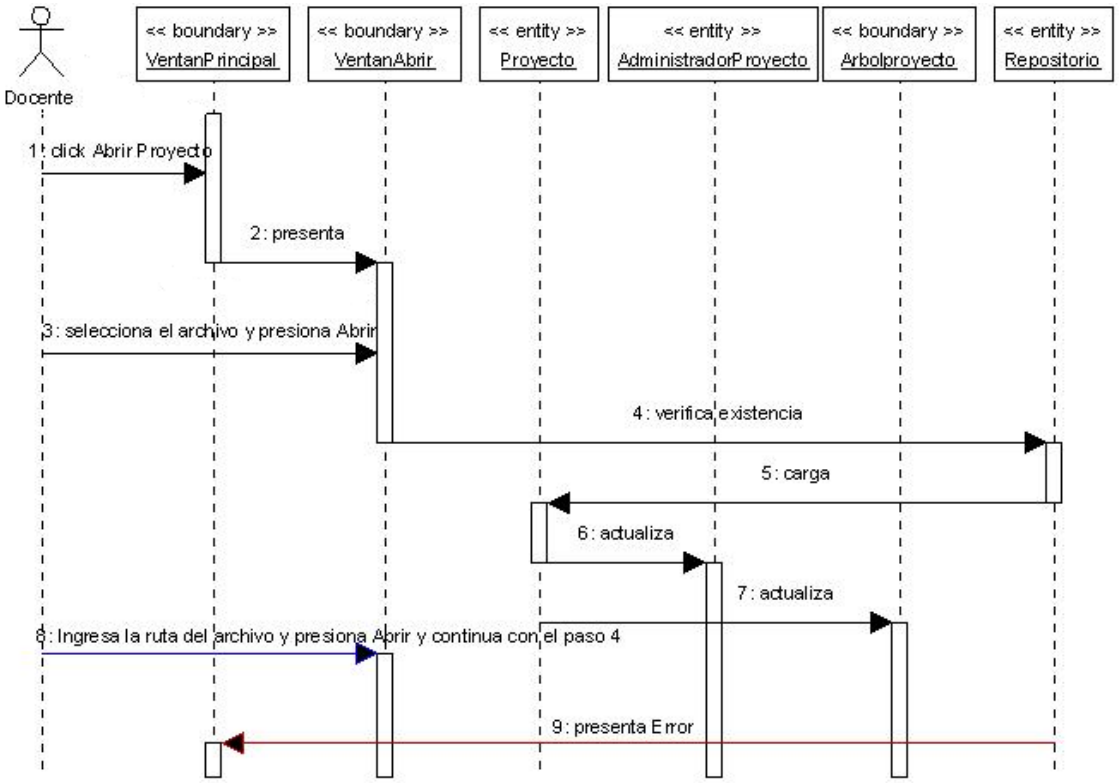


Figura 2.27: **Abrir Proyecto.**

2.2.3 Diagrama de Paquetes del sistema Hipercont

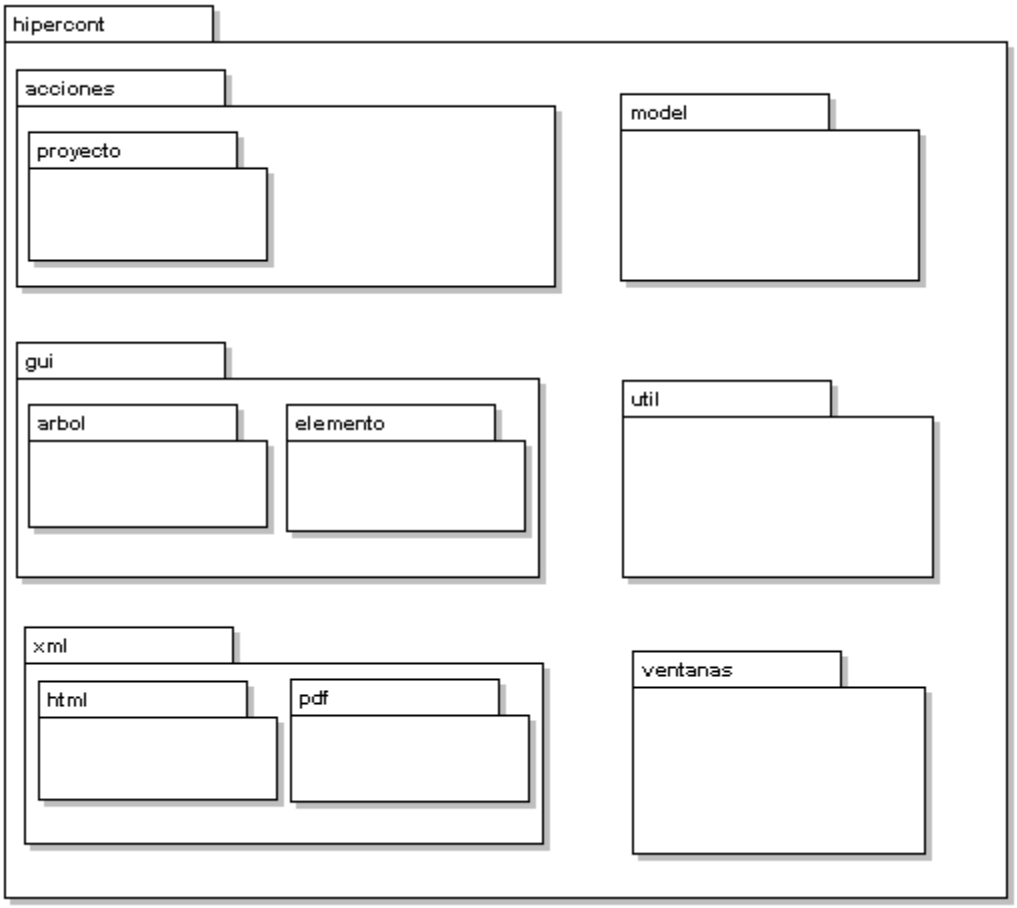


Figura 2.28: **Diagrama de Paquetes.**

2.3.4 Diagrama de clase del Modelo del Árbol del Proyecto.

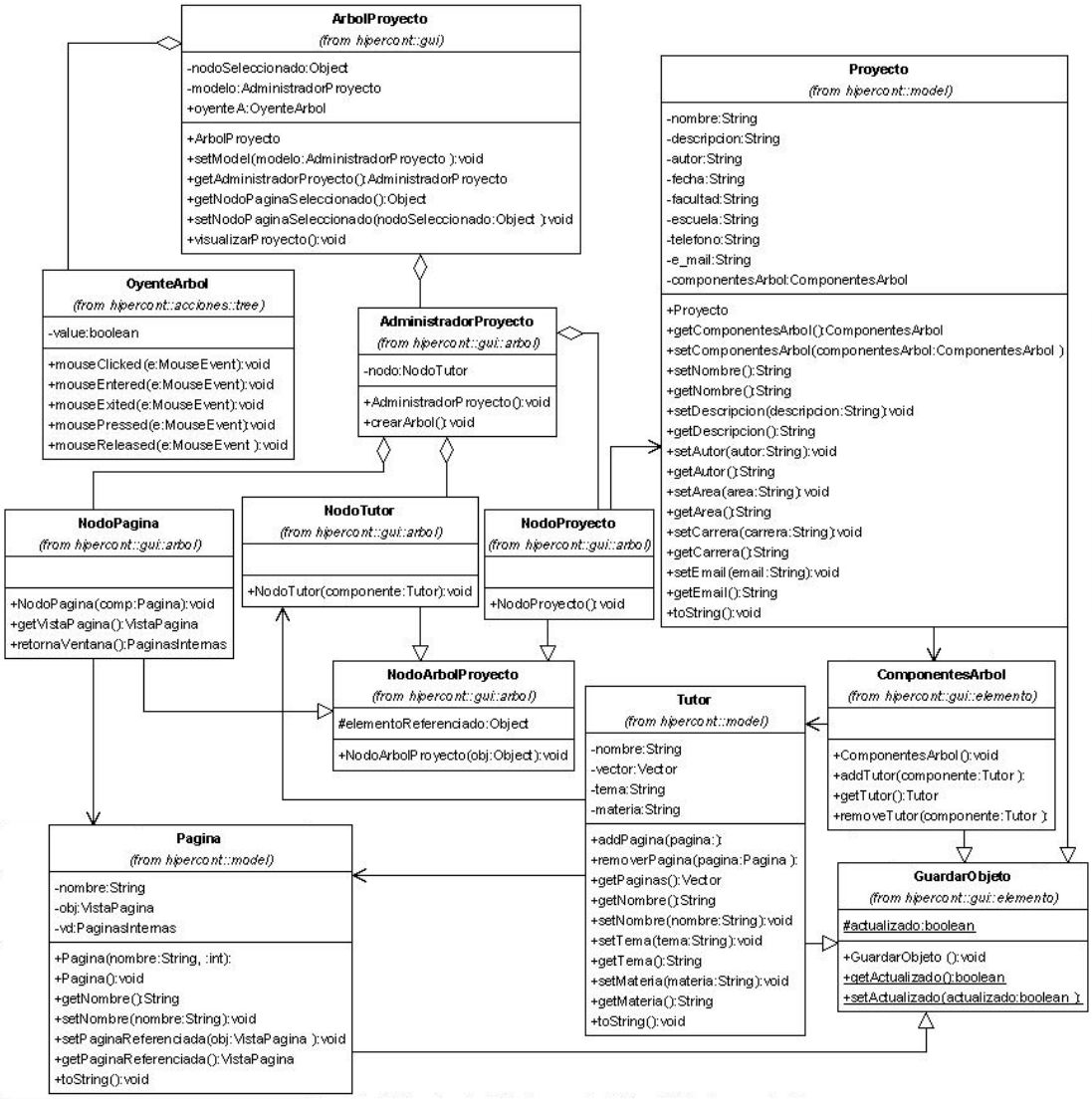


Figura 2.29: **Modelo del Árbol del Proyecto.**

2.3.5 Diagrama de clase de las propiedades de los elementos de un proyecto.

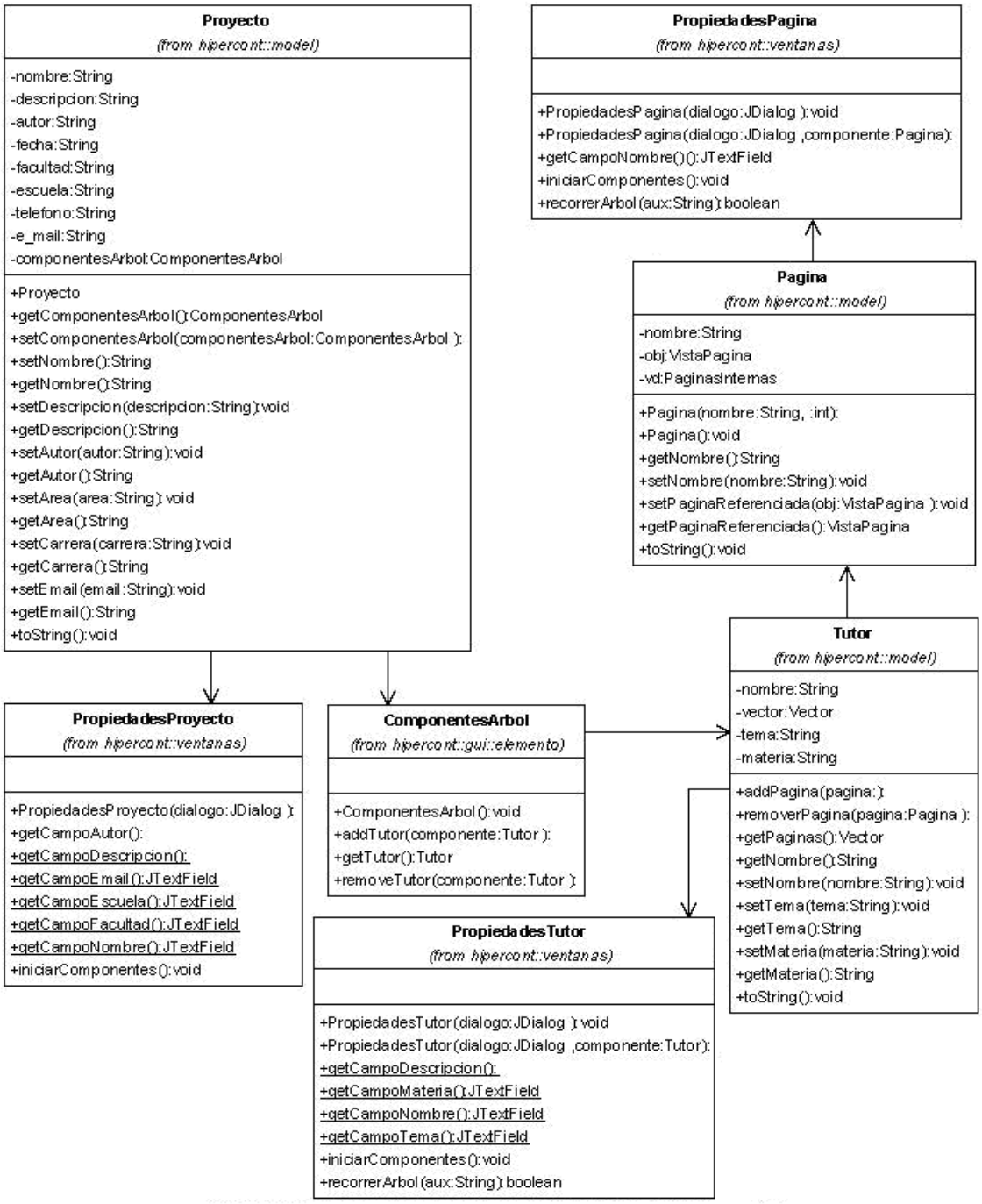


Figura 2.30: **Propiedades de los Elementos de un Proyecto**

2.3.6 Diagrama de clase de la Estructura de la Interfaz Gráfica Usuario.

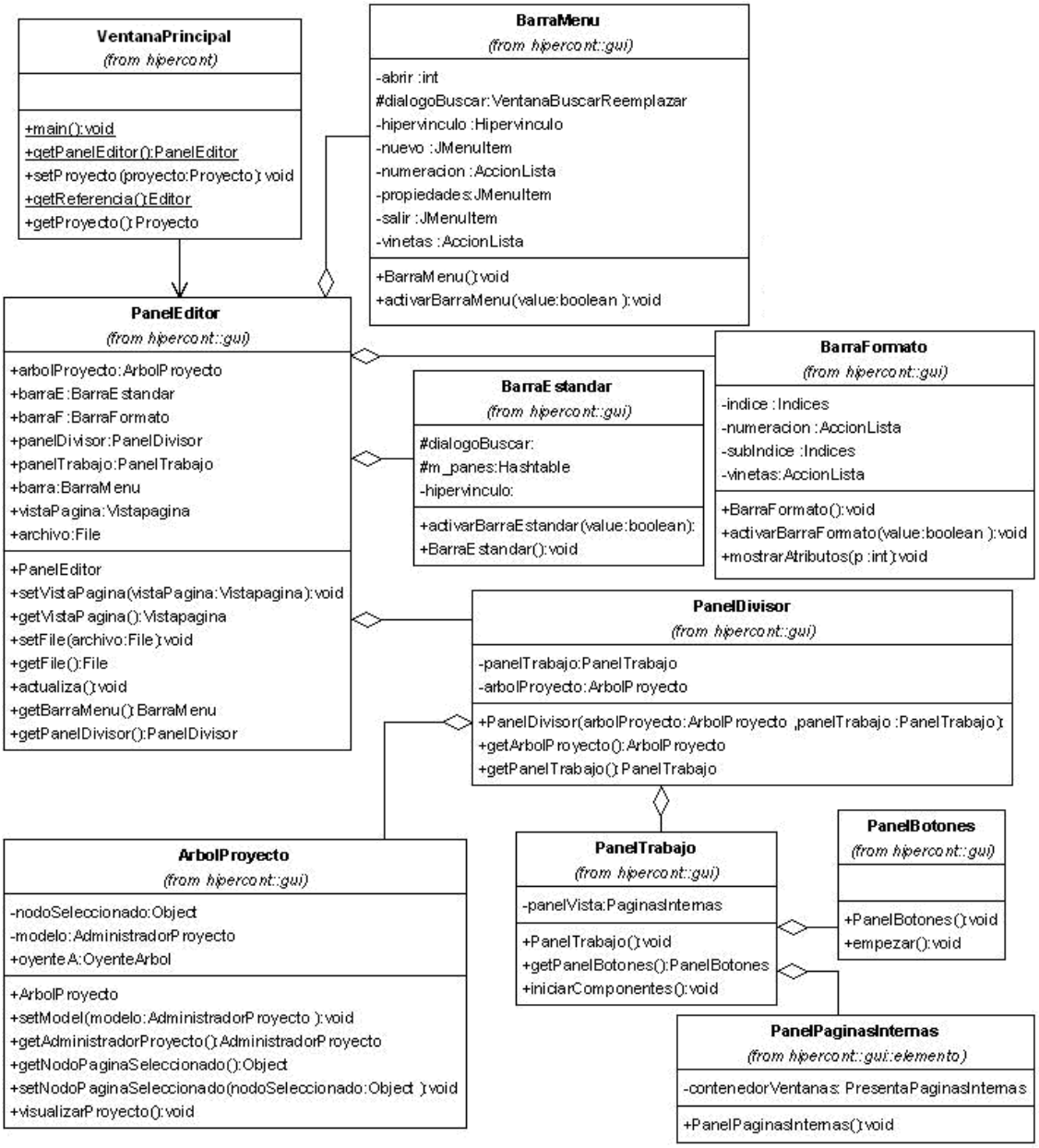


Figura 2.31: **Estructura de la Interfaz Gráfica Usuario**

2.3.7 Diagrama de clase Estructura de la Interfaz Gráfica Editor.

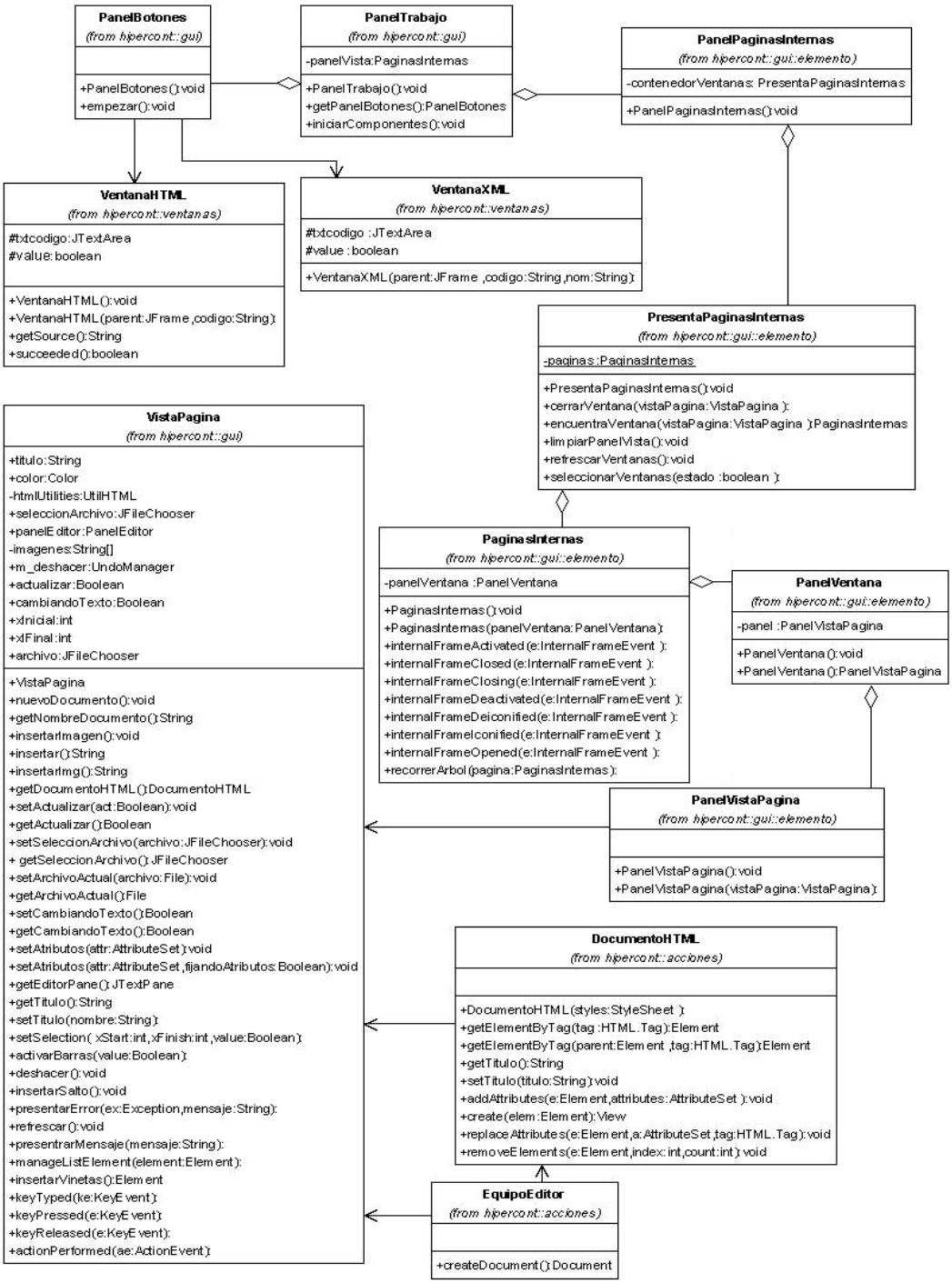


Figura 2.32: ***Estructura de la Interfaz Gráfica Editor.***

2.3.8 Diagrama de Componentes de sistema Hipercont

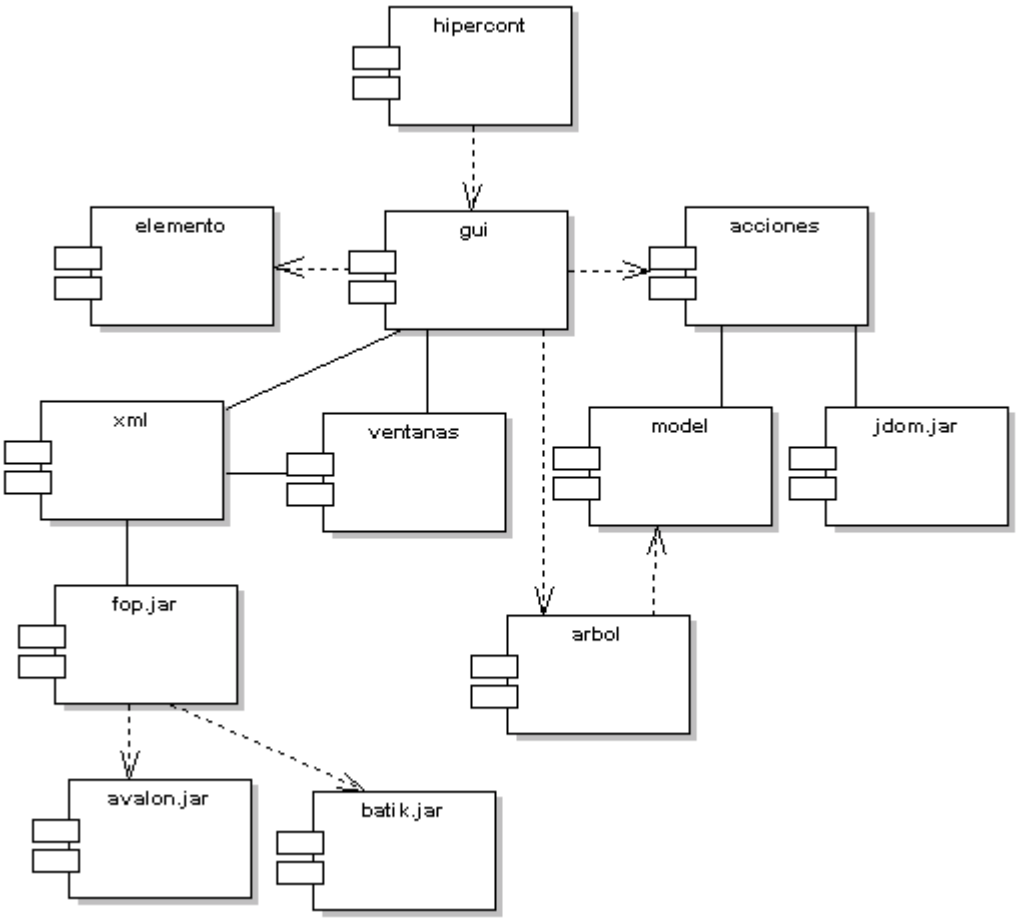


Figura 2.33: ***Diagrama de Componentes.***

CAPITULO III

GENERACIÓN DE DOCUMENTOS HTML Y PDF

3.1 Documento XML

“Todo documento XML posee una estructura lógica y una física. Físicamente, el documento está compuesto de unidades llamadas entidades. Una entidad puede hacer referencia a otras entidades para que éstas sean incluidas en el documento. Un documento comienza en una "raíz" o entidad documento. Lógicamente, el documento está compuesto de declaraciones, elementos, comentarios, referencias a caracteres e instrucciones de procesamiento, todos los cuales se indican en el documento mediante marcas explícitas.“[6] Las estructuras lógica y física deben anidarse de manera correcta.

El siguiente código presenta un ejemplo sencillo de este tipo de documentos.

```
<?xml version="1.0" encoding= "iso-8859-1"?>
<tutorial xsi:schemaLocation="http://www.wrox.com/support
hipercont.xsd"
 nombre-archivo="hipercont"
 zona="XML"
 salir= " http://www.wrox.com/support"
 xmlns="http://www.wrox.com/support"
 xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'>
      <cuerpo bgcolor = "#ccccff">
            <parrafo alineacion = "center"><negrita><fuente color
            ="#0066ff" tamanio ="5pt">HIPERCONT </fuente></negrita>
            </parrafo>
            <parrafo alineacion = "center"></parrafo>
            <parrafo><cursiva>Este es el primer tutor hecho en
            hipercont </cursiva>
            </parrafo>
            <parrafo><negrita>Este es el primer tutor hecho en
            hipercont</negrita>
```

[6] http://face.el.uma.es/imasd/xml/xml.htm

```
            </parrafo>
            <parrafo><subrayado>Este es el primer tutor hecho en
            hipercont</subrayado>
            </parrafo>
            <parrafo>   </parrafo>
            <vinetas><lista>viñeta 1</lista><lista>viñeta 2</lista>
            </vinetas>
            <parrafo> <imagen src ="file:/D:/img/Dibujo.gif" height
            ="106" width ="105"/>
            </parrafo>
            <parrafo><hipervinculo  href =
            "D:/img/copiar.gif">Hipervinculo</hipervinculo>
            </parrafo>
            <parrafo>   </parrafo>
            <tabla border = "2pt" bgcolor = "#3333ff">
            <columna_tabla   width= "100%"/>
            <columna_tabla   width= "100%"/>
            <fila_tabla bgcolor ="#ffcccc">
            <celda_tabla> tabla</celda_tabla>
            <celda_tabla> tabla</celda_tabla>
            </fila_tabla>
            <fila_tabla bgcolor ="#ffcccc">
            <celda_tabla> celda</celda_tabla>
            <celda_tabla> celda</celda_tabla>
            </fila_tabla>
            </tabla>
            <parrafo>   </parrafo>
            <numeracion></numeracion>
            <parrafo>   </parrafo>
      </cuerpo>
</tutorial>
```

Para generar los documentos HTML y PDF a partir de documentos XML se ha utilizado hojas de estilo (XSLT) y código java.

3.2 XML-Schema

Un "schema XML" define qué elementos puede contener un documento XML, cómo están organizados, y que atributos y de qué tipo pueden tener sus elementos.

Un Esquema de XML define:

- Elementos que pueden aparecer en un documento
- Atributos que pueden aparecer en un documento
- Qué elementos son los elementos hijos.
- El orden de elementos hijos.
- El número de elementos hijos.
- Si un elemento está vacío o puede incluir texto.
- Los tipos de los datos para los elementos y atributos.
- Valor por defecto y los valores fijos por los elementos y atributos.

El Esquema de XML presentado a continuación se ha desarrollado para el control de los documentos XML que se generen en el sistema.

```
<?xml version="1.0" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
           targetNamespace="http://www.wrox.com/support"
            xmlns:st="http://www.wrox.com/support"
            elementFormDefault="qualified">
```

```
<xs:element name="tutorial">
    <xs:complexType>
          <xs:sequence>
                <xs:element ref="st:cuerpo" />
          </xs:sequence>
          <xs:attribute name="zona" use="required">
                <xs:simpleType>
                      <xs:restriction base= "xs:NMTOKEN">
                            <xs:enumeration value="Java" />
                            <xs:enumeration value="Linux" />
                            <xs:enumeration value="Security" />
                            <xs:enumeration value="Unicode" />
                            <xs:enumeration value="Web" />
                            <xs:enumeration value="XML" />
                      </xs:restriction>
                </xs:simpleType>
          </xs:attribute>
          <xs:attribute name="salir"  type="xs:anyURI"/>
          <xs:attribute name="nombre-archivo"
                        type="xs:string"/>
    </xs:complexType>
</xs:element>
<xs:element name="indice">
      <xs:complexType>
          <xs:sequence>
              <xs:choice minOccurs="0" maxOccurs="unbounded">
                <xs:element ref="st:negrita"/>
                <xs:element ref="st:cursiva"/>
              </xs:choice>
          </xs:sequence>
            <xs:attribute name="bgcolor"  type="xs:string"/>
    </xs:complexType>
</xs:element>
<xs:element name="subIndice">
      <xs:complexType>
```

```
            <xs:sequence>
                <xs:choice minOccurs="0" maxOccurs="unbounded">
                  <xs:element ref="st:negrita"/>
                  <xs:element ref="st:cursiva"/>
                </xs:choice>
            </xs:sequence>
              <xs:attribute name="bgcolor"  type="xs:string"/>
      </xs:complexType>
 </xs:element>
<xs:element name="subrayado">
        <xs:complexType>
            <xs:sequence>
                <xs:choice minOccurs="0" maxOccurs="unbounded">
                  <xs:element ref="st:negrita"/>
                   <xs:element ref="st:cursiva"/>
                </xs:choice>
             </xs:sequence>
             <xs:attribute name="bgcolor"  type="xs:string"/>
      </xs:complexType>
 </xs:element>
 <xs:element name="cuerpo">
   <xs:complexType>
      <xs:sequence>
            <xs:element ref="st:columna_texto"/>
      </xs:sequence>
              <xs:attribute name="style"  type="xs:string"/>
              <xs:attribute name="bgcolor"  type="xs:string"/>
        </xs:complexType>
 </xs:element>
 <xs:element name="parrafo" >
      <xs:complexType  mixed="true">
            <xs:choice minOccurs="0" maxOccurs="unbounded">
                  <xs:element ref="st:negrita"/>
                  <xs:element ref="st:salto"/>
                  <xs:element ref="st:linea"/>
```

```
                <xs:element ref="st:hipervinculo"/>
                <xs:element ref="st:subrayado"/>
                <xs:element ref="st:indice"/>
                <xs:element ref="st:subIndice"/>
                <xs:element ref="st:cursiva"/>
                <xs:element ref="st:fuente"/>
                <xs:element ref="st:imagen"/>
                <xs:element ref="st:numeracion"/>
                <xs:element ref="st:vinetas"/>
            </xs:choice>
                <xs:attribute name="alineacion" >
                <xs:simpleType>
                    <xs:restriction base="xs:NMTOKEN">
                        <xs:enumeration value="left" />
                        <xs:enumeration value="center" />
                        <xs:enumeration value="right" />
                        <xs:enumeration value="justify"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
        </xs:complexType>
    </xs:element>
    <xs:element name="columna_texto">
      <xs:complexType>
        <xs:sequence>
                <xs:choice minOccurs="0" maxOccurs="unbounded">
                <xs:element ref="st:linea"/>
                <xs:element ref="st:salto"/>
                <xs:element ref="st:numeracion"/>
                <xs:element ref="st:vinetas"/>
                <xs:element ref="st:parrafo"/>
                <xs:element ref="st:tabla"/>
                </xs:choice>
        </xs:sequence>
      </xs:complexType>
```

```
</xs:element>
  <xs:element name="fuente">
      <xs:complexType  mixed="true">
                 <xs:attribute name="tipo"   type="xs:string"/>
                 <xs:attribute name="color"  type="xs:string"/>
                 <xs:attribute name="tamanio" type="xs:string"/>
      </xs:complexType>
</xs:element>
<xs:element name="imagen">
      <xs:complexType>
              <xs:attribute name="src"  type="xs:anyURI"/>
              <xs:attribute name="height"  type="xs:string"/>
              <xs:attribute name="width"  type="xs:string"/>
      </xs:complexType>
</xs:element>
<xs:element name="negrita">
      <xs:complexType mixed="true">
            <xs:choice minOccurs="0" maxOccurs="unbounded">
                  <xs:element ref="st:cursiva"/>
                  <xs:element ref="st:subrayado"/>
            </xs:choice>
      </xs:complexType>
</xs:element>
<xs:element name="salto">
      <xs:complexType/>
</xs:element>
<xs:element name="linea">
      <xs:complexType/>
</xs:element>
<xs:element name="cursiva">
      <xs:complexType mixed="true">
            <xs:choice minOccurs="0" maxOccurs="unbounded">
                  <xs:element ref="st:negrita" />
                          <xs:element ref="st:subrayado"/>
                          <xs:element ref="st:indice"/>
```

```
                        <xs:element ref="st:subIndice"/>
                  </xs:choice>
      </xs:complexType>
</xs:element>
<xs:element name="tabla">
      <xs:complexType>
         <xs:sequence>
            <xs:element ref="st:columna_tabla"
      maxOccurs="unbounded" />
            <xs:element ref="st:fila_tabla"
      maxOccurs="unbounded" />
         </xs:sequence>
            <xs:attribute name="border" type ="xs:string"/>
            <xs:attribute name="bgcolor" type ="xs:string"/>
            <xs:attribute name="height"  type="xs:string"/>
            <xs:attribute name="width"  type="xs:string"/>
      </xs:complexType>
</xs:element>
<xs:element name="celda_tabla">
      <xs:complexType mixed="true">
            <xs:choice minOccurs="0" maxOccurs="unbounded">
                  <xs:element ref="st:hipervinculo" />
                  <xs:element ref="st:negrita" />
                  <xs:element ref="st:salto" />
                  <xs:element ref="st:cursiva" />
                  <xs:element ref="st:subrayado"/>
                  <xs:element ref="st:imagen" />
                  <xs:element ref="st:numeracion" />
                  <xs:element ref="st:tabla"/>
                  <xs:element ref="st:vinetas" />
                  <xs:element ref="st:fuente" />
            </xs:choice>
              <xs:attribute name="alineacion" >
                  <xs:simpleType>
                        <xs:restriction base="xs:NMTOKEN">
                              <xs:enumeration value="left" />
```

```
                                <xs:enumeration value="center" />
                                <xs:enumeration value="right" />
                            </xs:restriction>
                        </xs:simpleType>
                    </xs:attribute>
                </xs:complexType>
            </xs:element>
            <xs:element name="columna_tabla">
                <xs:complexType>
                        <xs:attribute name="width" type="xs:string">
                        </xs:attribute>
                </xs:complexType>
            </xs:element>
            <xs:element name="titulo_tabla">
                <xs:complexType>
                    <xs:sequence>
                        <xs:element ref="st:celda_tabla"
                    maxOccurs="unbounded" />
                    </xs:sequence>
                </xs:complexType>
            </xs:element>
            <xs:element name="fila_tabla">
                <xs:complexType>
                    <xs:sequence>
                    <xs:element ref="st:celda_tabla"
                    maxOccurs="unbounded"/>
                    </xs:sequence>
                </xs:complexType>
            </xs:element>


            <xs:element name="vinetas">
                <xs:complexType>
                    <xs:sequence >
                        <xs:element ref="st:lista"
                        maxOccurs="unbounded"/>
                    </xs:sequence>
```

```
        </xs:complexType>
    </xs:element>
         <xs:element name="hipervinculo">
        <xs:complexType  mixed="true">
              <xs:choice minOccurs="0" maxOccurs="unbounded" >
                    <xs:element ref="st:negrita"/>
                    <xs:element ref="st:cursiva"/>
              </xs:choice>
              <xs:attribute name="href"  type="xs:anyURI" />
              <xs:attribute name="style"  type="xs:string" />
              <xs:attribute name="target">
                    <xs:simpleType>
                          <xs:restriction base="xs:string">
                          <xs:enumeration value="espacio_blanco" />
                          </xs:restriction>
                    </xs:simpleType>
              </xs:attribute>
        </xs:complexType>
    </xs:element>
    <xs:element name="lista">
        <xs:complexType mixed="true">
              <xs:choice minOccurs="0" maxOccurs="unbounded">
                    <xs:element ref="st:negrita" />
                    <xs:element ref="st:salto" />
                    <xs:element ref="st:cursiva" />
                    <xs:element ref="st:subrayado"/>
                    <xs:element ref="st:indice"/>
                    <xs:element ref="st:subIndice"/>
                    <xs:element ref="st:numeracion" />
                    <xs:element ref="st:parrafo" />
                    <xs:element ref="st:hipervinculo"/>
                    <xs:element ref="st:vinetas" />
              </xs:choice>
        </xs:complexType>
    </xs:element>
```

```
    <xs:element name="numeracion">
        <xs:complexType>
            <xs:sequence maxOccurs="unbounded">
                <xs:element ref="st:lista" />
            </xs:sequence>
        </xs:complexType>
    </xs:element>
</xs:schema>
```

3.3 XSLT (hojas de estilo)

"XSLT es un lenguaje de programación que forma parte de la ***trilogía transformadora*** de XML, compuesta por:

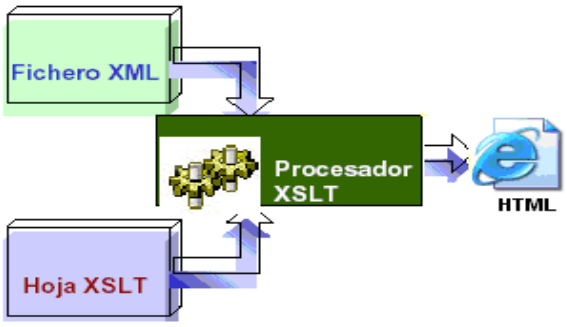
- *CSS (hojas de estilo en cascada), que permite dar una apariencia en el navegador determinada a cada una de las etiquetas XML.*
- *XSLT, lenguaje de transformación basado en hojas de estilo.*
- *XSL: FO, (objetos de formateo), o transformaciones para fotocomposición, o, en general, para cualquier cosa que no sea XML."*[7]

XSLT es un lenguaje que se usa para convertir documentos XML en otros documentos XML; ó puede convertir a "formatos finales", tales como WML, PDF o HTML.

[7] http://geneura.ugr.es/~jmerelo/XSLT

3.3.1 Hoja de estilos (XSLT) para generar HTML

Al documento XML, se lo debe convertir en HTML para que el navegador pueda presentarlo.



Este proceso de transformación se llevará a cabo utilizando la siguiente hoja de estilos.

```
<?xml version="1.0" encoding="iso-8859-1"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
version="1.0">
    <!--<xsl:output method="html"/> -->
    <xsl:template match="/">
        <xsl:apply-templates/>
    </xsl:template>
    <xsl:template match="*">

        <xsl:variable name="nodo" select="name()"/>
        <xsl:variable name="tag">
            <xsl:choose>
                <xsl:when test="$nodo='negrita'">b</xsl:when>
                <xsl:when test="$nodo='parrafo'">p</xsl:when>
                <xsl:when test="$nodo='cursiva'">i</xsl:when>
                <xsl:when test="$nodo='tutorial'">html</xsl:when>
                <xsl:when test="$nodo='fuente'">font</xsl:when>
                <xsl:when test="$nodo='cuerpo'">body</xsl:when>
```

```
        <xsl:when test="$nodo='tabla'">table</xsl:when>
        <xsl:when test="$nodo='subIndice'">sub</xsl:when>
        <xsl:when test="$nodo='indice'">sup</xsl:when>
        <xsl:when test="$nodo='titulo_tabla'">th</xsl:when>
        <xsl:when test="$nodo='fila_tabla'">tr</xsl:when>
        <xsl:when test="$nodo='imagen'">img</xsl:when>
        <xsl:when test="$nodo='celda_tabla'">td</xsl:when>
        <xsl:when test="$nodo='salto'">br</xsl:when>
        <xsl:when test="$nodo='linea'">hr</xsl:when>
        <xsl:when test="$nodo='subrayado'">u</xsl:when>
        <xsl:when test="$nodo='lista'">li</xsl:when>
        <xsl:when test="$nodo='hipervinculo'">a</xsl:when>
        <xsl:when test="$nodo='numeracion'">ol</xsl:when>
        <xsl:when test="$nodo='vinetas'">ul</xsl:when>

        <xsl:otherwise>desconocido</xsl:otherwise>
    </xsl:choose>
</xsl:variable>

<xsl:element name="{$tag}">
    <xsl:if test="string-length(@bgcolor) > 0">
          <xsl:attribute name="bgcolor">
                <xsl:value-of select="@bgcolor"/>
          </xsl:attribute>
    </xsl:if>
    <xsl:if test="string-length(@cols) > 0">
          <xsl:attribute name="cols">
                <xsl:value-of select="@cols"/>
          </xsl:attribute>
    </xsl:if>
    <xsl:if test="string-length(@background) > 0">
        <xsl:attribute name="background">
            <xsl:value-of select="@background"/>
        </xsl:attribute>
    </xsl:if>
```

```
<xsl:if test="string-length(@color) > 0">
    <xsl:attribute name="color">
         <xsl:value-of select="@color"/>
    </xsl:attribute>
</xsl:if>
<xsl:if test="string-length(@style) > 0">
    <xsl:attribute name="style">
            <xsl:value-of select="@style"/>
    </xsl:attribute>
</xsl:if>
<xsl:if test="string-length(@alineacion) > 0">
    <xsl:attribute name="align">
            <xsl:value-of select="@alineacion"/>
    </xsl:attribute>
</xsl:if>
<xsl:if test="string-length(@tamanio) > 0">
    <xsl:attribute name="size">
            <xsl:value-of select="@tamanio"/>
    </xsl:attribute>
</xsl:if>
<xsl:if test="string-length(@src) > 0">
    <xsl:attribute name="src">
            <xsl:value-of select="@src"/>
    </xsl:attribute>
</xsl:if>
<xsl:if test="string-length(@start) > 0">
    <xsl:attribute name="start">
            <xsl:value-of select="@start"/>
    </xsl:attribute>
</xsl:if>
<xsl:if test="string-length(@dynsrc) > 0">
    <xsl:attribute name="dynsrc">
            <xsl:value-of select="@dynsrc"/>
    </xsl:attribute>
</xsl:if>
```

```
            <xsl:if test="string-length(@tipo) > 0">
                <xsl:attribute name="face">
                        <xsl:value-of select="@tipo"/>
                </xsl:attribute>
            </xsl:if>
            <xsl:if test="string-length(@border) > 0">
                <xsl:attribute name="border">
                        <xsl:value-of select="@border"/>
                </xsl:attribute>
            </xsl:if>
            <xsl:if test="string-length(@width) > 0">
                <xsl:attribute name="width">
                        <xsl:value-of select="@width"/>
                </xsl:attribute>
            </xsl:if>
            <xsl:if test="string-length(@height) > 0">
                <xsl:attribute name="height">
                    <xsl:value-of select="@height"/>
                </xsl:attribute>
            </xsl:if>
            <xsl:if test="string-length(@href) > 0">
                <xsl:attribute name="href">
                        <xsl:value-of select="@href"/>
                </xsl:attribute>
            </xsl:if>
            <xsl:apply-templates/>
        </xsl:element>
    </xsl:template>
</xsl:stylesheet>
```

Una vez generada, la página debe verse así en su navegador:

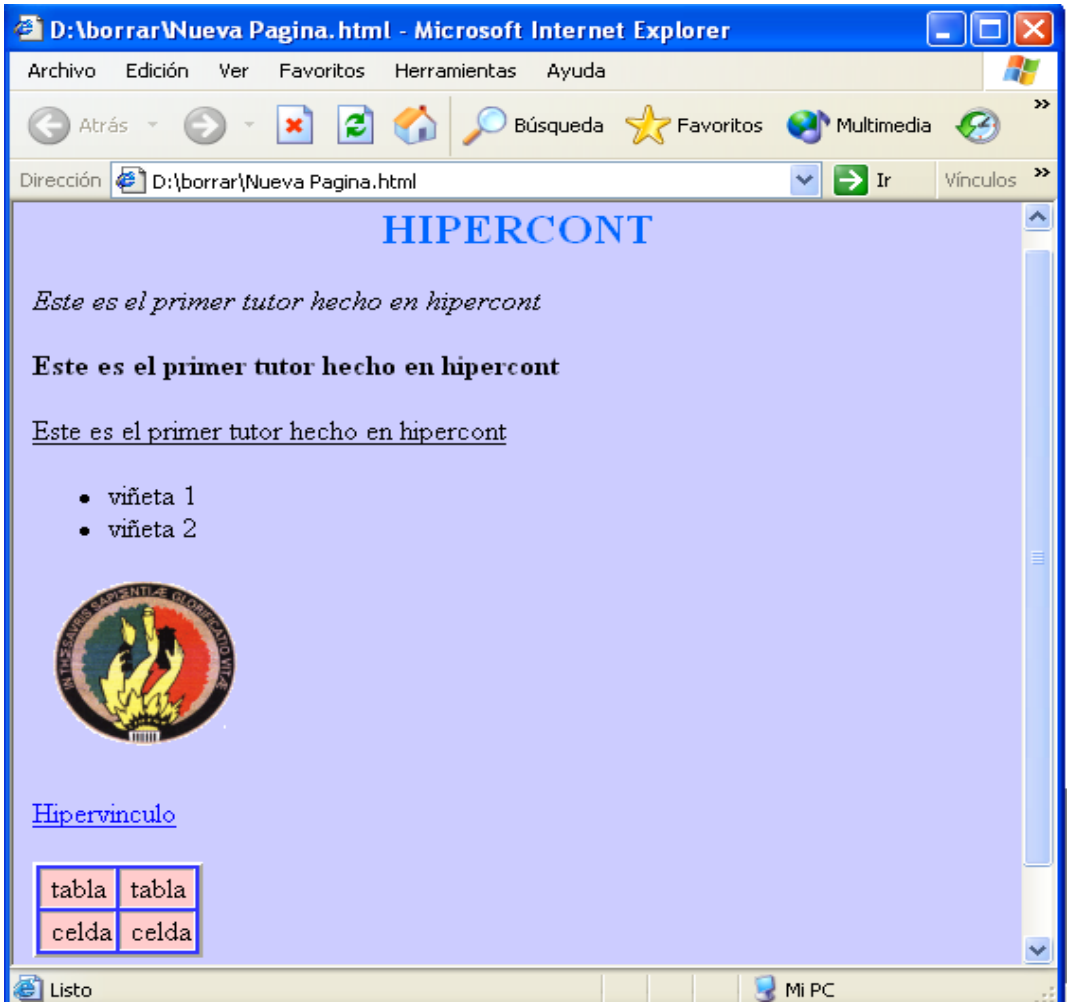
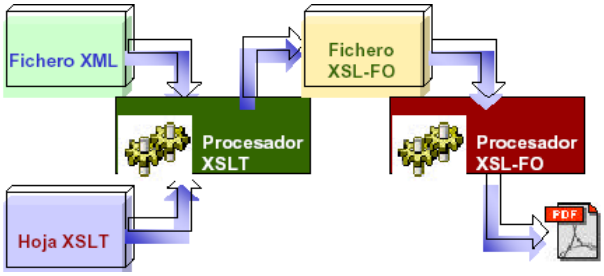


Figura 3.1: **Salida HTML.**

3.3.2 Hoja de estilo (XSLT) para generar PDF

La entrada del proceso de generación de PDF es un documento XML. Después de esto, el XML es transformado en un documento XML FO utilizando la hoja de estilos XSL. Una vez creado el documento FO, se pasa a un agente de presentación FO, para generar la salida PDF.

El proceso Objetos de formato (FO) XSL se esquematiza en el siguiente diagrama.




```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.1"
 xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
 xmlns:fo="http://www.w3.org/1999/XSL/Format"
 xmlns:svg="http://www.w3.org/2000/svg"
 exclude-result-prefixes="fo">
  <xsl:output method="xml" version="1.0"
                    omit-xml-declaration="no" indent="yes"/>

  <xsl:template match="tutorial">
    <fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
      <fo:layout-master-set>
            <fo:simple-page-master master-name="simpleA4"
                                   page-height="29.7cm"
                                   page-width="21cm"
                                   margin-top="2cm"
                                   margin-bottom="2cm"
                                   margin-left="2cm"
```

```
                              margin-right="2cm">
                <fo:region-body margin-top="2cm"/>
                <fo:region-before extent="6cm"/>
                <fo:region-after extent="6cm"/>
            </fo:simple-page-master>
        </fo:layout-master-set>

        <fo:page-sequence master-reference="simpleA4">
          <fo:static-content flow-name="xsl-region-before">
             <fo:block text-align="end"
                       font-size="12pt"
                       font-family="serif"
                       line-height="14pt"
                       color="blue"
                       font-style="italic">
                       HiperCont
            </fo:block>
                  <!--numero de pagina-->
            <fo:block text-align="end"
                       font-size="10pt"
                       font-family="serif"
                       line-height="14pt">
                       page:<fo:page-number/>
            </fo:block>
          </fo:static-content>
                <!--BODY-->
          <fo:flow flow-name="xsl-region-body" >
              <xsl:apply-templates />
          </fo:flow>
        </fo:page-sequence>
      </fo:root>
    </xsl:template>

    <xsl:template match="parrafo" >
      <fo:block space-after.optimum="14pt" line-height="20pt">
```

```
        <xsl:variable name= "align"  select="@alineacion" />
                  <xsl:attribute name="text-align">
                   <xsl:choose>
                        <xsl:when test="$align='center'">
                              <xsl:text>center</xsl:text>
                        </xsl:when>
                        <xsl:when test="$align='right'">
                              <xsl:text>right</xsl:text>
                        </xsl:when>
                        <xsl:when test="$align='left'">
                              <xsl:text>left</xsl:text>
                        </xsl:when>
                        <xsl:when test="$align='justify'">
                              <xsl:text>justify</xsl:text>
                        </xsl:when>
                        <xsl:otherwise>
                              <xsl:text>justify</xsl:text>
                        </xsl:otherwise>
                   </xsl:choose>
                  </xsl:attribute>
           <xsl:apply-templates select="*|text()"/>
        </fo:block>
 </xsl:template>

<xsl:template match="subrayado" >
    <fo:inline text-decoration="underline">
      <xsl:apply-templates select="*|text()" />
    </fo:inline>
  </xsl:template>

 <xsl:template match="cursiva" >
    <fo:inline font-style="italic">
      <xsl:apply-templates select="*|text()" />
    </fo:inline>
  </xsl:template>
```

```
 <xsl:template match="negrita" >
   <fo:inline font-weight="bold">
     <xsl:apply-templates select="*|text()" />
   </fo:inline>
 </xsl:template>

<xsl:template match="fuente" >
  <fo:inline>
      <xsl:variable name= "tam"  select="@tamanio" />
      <xsl:attribute name="font-size">
           <xsl:choose>
                 <xsl:when test="$tam='1pt'">
                       <xsl:text>8pt</xsl:text>
                 </xsl:when>
                 <xsl:when test="$tam='2pt'">
                       <xsl:text>10pt</xsl:text>
                 </xsl:when>
                 <xsl:when test="$tam='3pt'">
                       <xsl:text>12pt</xsl:text>
                 </xsl:when>
                 <xsl:when test="$tam='4pt'">
                       <xsl:text>14pt</xsl:text>
                 </xsl:when>
                 <xsl:when test="$tam='5pt'">
                       <xsl:text>18pt</xsl:text>
                 </xsl:when>
                 <xsl:when test="$tam='6pt'">
                       <xsl:text>22pt</xsl:text>
                 </xsl:when>
                 <xsl:when test="$tam='7pt'">
                       <xsl:text>28pt</xsl:text>
                 </xsl:when>
                 <xsl:otherwise>
                       <xsl:text>12pt</xsl:text>
```

```
                </xsl:otherwise>
            </xsl:choose>
        </xsl:attribute>
        <xsl:attribute name="font-family">
                <xsl:value-of select="@tipo"/>
        </xsl:attribute>
        <xsl:attribute name="color">
                <xsl:value-of select="@color"/>
        </xsl:attribute>
       <xsl:apply-templates select="*|text()" />
      </fo:inline>
  </xsl:template>

  <xsl:template match="imagen" >
    <fo:external-graphic space-after.optimum="14pt"
                                    text-align-last="center">
      <xsl:attribute name="src">
                <xsl:value-of select="@src"/>
       </xsl:attribute>
    </fo:external-graphic>
  </xsl:template>

<xsl:template match="vinetas" >
    <fo:list-block provisional-distance-between-starts="0.4cm"
      provisional-label-separation="0.2cm">
      <xsl:attribute name="space-after.optimun">
            <xsl:choose>
                <xsl:when test="ancestor::vinetas or
ancestor::numeracion">
                      <xsl:text>0pt</xsl:text>
                </xsl:when>
                <xsl:otherwise>
                      <xsl:text>14pt</xsl:text>
                </xsl:otherwise>
            </xsl:choose>
      </xsl:attribute>
```

```
      <xsl:apply-templates select="*"/>
   </fo:list-block>
</xsl:template>

 <xsl:template match="numeracion" >
   <fo:list-block provisional-distance-between-starts="1.5cm"
       provisional-label-separation="0.2cm">
     <xsl:attribute name="space-after.optimun">
          <xsl:choose>
                <xsl:when test="ancestor::vinetas or
                          ancestor::numeracion">
                      <xsl:text>0pt </xsl:text>
                </xsl:when>
                <xsl:otherwise>
                      <xsl:text>14pt</xsl:text>
                </xsl:otherwise>
          </xsl:choose>
    </xsl:attribute>
    <xsl:apply-templates select="*"/>
   </fo:list-block>
 </xsl:template>

 <xsl:template match="lista" >
   <xsl:choose>
     <xsl:when test="parent::numeracion">
       <fo:list-item space-after.optimum="3pt">
         <fo:list-item-label end-indent="label-end()">
           <fo:block text-align="end">
                 <xsl:number value="position()" format="1. "/>
           </fo:block>
         </fo:list-item-label>
         <fo:list-item-body start-indent="body-start()">
           <fo:block>
                 <xsl:apply-templates select="*|text()" />
           </fo:block>
```

```
          </fo:list-item-body>
        </fo:list-item>
      </xsl:when>
      <xsl:otherwise>
        <fo:list-item space-after.optimum="3pt">
          <fo:list-item-label end-indent="label-end()">
            <fo:block>
                  <fo:inline font-family="Symbol">&#x2022;</fo:inline>
            </fo:block>
          </fo:list-item-label>
          <fo:list-item-body start-indent="body-start()">
            <fo:block>
                  <xsl:apply-templates select="*|text()" />
            </fo:block>
          </fo:list-item-body>
        </fo:list-item>
      </xsl:otherwise>
    </xsl:choose>
  </xsl:template>

  <xsl:template match="indice" >
    <fo:inline vertical-align="super" font-size="6pt">
      <xsl:apply-templates select="*|text()" />
    </fo:inline>
  </xsl:template>

  <xsl:template match="subIndice" >
    <fo:inline vertical-align="sub" font-size="6pt">
      <xsl:apply-templates select="*|text()"/>
    </fo:inline>
  </xsl:template>

<xsl:template match="hipervinculo">
    <xsl:choose>
      <xsl:when test="@href">
```

```
      <fo:inline font-style="italic">
        <fo:basic-link color="blue">
              <xsl:attribute name="external-destination">
                <xsl:value-of select="@href"/>
            </xsl:attribute>
          <xsl:apply-templates select="*|text()"/>
        </fo:basic-link>
      </fo:inline>
    </xsl:when>
    <!-- <xsl:otherwise>
      <xsl:apply-templates select="*|text()"/>
    </xsl:otherwise>-->
  </xsl:choose>
</xsl:template>
<xsl:template match="tabla">
 <fo:table space-after.optimum="2pt" table-layout="fixed"  >
  <xsl:variable name= "color"  select="@bgcolor"/>
          <xsl:attribute name="border-width">
                      <xsl:value-of select="@borde"/>
          </xsl:attribute>
          <xsl:attribute name="border-style">
                      <xsl:text>solid</xsl:text>
          </xsl:attribute>
          <xsl:attribute name="height">
                      <xsl:value-of select="@height"/>
          </xsl:attribute>
          <xsl:attribute name="border-color">
                      <xsl:value-of select="@bgcolor"/>
          </xsl:attribute>

          <xsl:for-each select="columna_tabla">
                <fo:table-column>
                      <xsl:attribute name="width">
                            <xsl:value-of select="@width"/>
                      </xsl:attribute>
```

```
                </fo:table-column>
            </xsl:for-each>
    <fo:table-body>
       <xsl:for-each select="fila_tabla">
         <fo:table-row>
        <xsl:variable name= "color_borde"  select="@bgcolor"/>
           <xsl:for-each select="celda_tabla">
             <fo:table-cell  >
                <xsl:attribute name="border-style">
                      <xsl:text>solid</xsl:text>
                </xsl:attribute>
                <xsl:attribute name="background-color">
                   <xsl:value-of select="$color_borde"/>
                </xsl:attribute>
                <xsl:attribute name="border-color">
                      <xsl:value-of select="$color"/>
                </xsl:attribute>
                <xsl:attribute name="border-width">
                    <xsl:text>1pt</xsl:text>
                </xsl:attribute>
               <fo:block>
                 <xsl:apply-templates select="*|text()" />
               </fo:block>
             </fo:table-cell>
           </xsl:for-each>
         </fo:table-row>
       </xsl:for-each>
     </fo:table-body>
   </fo:table>
 </xsl:template>
  <xsl:template match="salto">
     <fo:block space-after.optimum="8pt">
     </fo:block>
</xsl:template>
<xsl:template match="linea">
```

```
      <fo:block >
         <fo:leader leader-pattern ="rule" space-after.optimun ="6pt"/>
      </fo:block>
   </xsl:template>
</xsl:stylesheet>
```

Si se tiene Adobe Acrobat instalado, el documento PDF generado se devolverá con el formato aquí mostrado.

Figura 3.2: **Salida PDF.**

3.4 WAP WML

WAP es un protocolo basado en los estándares de Internet que ha sido desarrollado para permitir a teléfonos celulares navegar a través de Internet. Con la tecnología WAP se pretende que desde cualquier teléfono celular WAP se pueda acceder a la

información que hay en Internet así como realizar operaciones de comercio electrónico.

WAP es una serie de tecnologías que consisten en: WML, que es el lenguaje de etiquetas, WMLScript es un lenguaje de script, lo que vendría a ser JavaScript y el Wireless Telephony Application Interface (WTAI).

WML es el lenguaje de etiquetas similar al HTML. WML es compatible con XML. Las páginas WML son llamadas barajas ya que están compuestas por cartas, un navegador WAP, solo puede mostrar una carta al mismo tiempo.

Las características principales de WML son:

1. Soporte para imágenes y texto, con posibilidad de texto con formato.

2. Tarjetas agrupadas en barajas. Una página WML es como una página HTML en la que hay una serie de cartas, al conjunto de estas cartas se les suele llamar baraja.

3. Posibilidad de navegar entre cartas y barajas de la misma forma que se navega entre páginas Web.

4. Manejo de variables y formularios para el intercambio de información entre el teléfono celular y el servidor.

Los rasgos más importantes de WML son:

- Todos los elementos de WML son sensibles a mayúsculas/minúsculas, esto incluye las etiquetas, los atributos, los identificadores, las variables...

- El conjunto de caracteres definido por defecto es el ISO/IEC-10646 que es el mismo que el Unicode 2.0 WAP soporta los siguientes subconjuntos de Unicode:

  1. UTF-8
  2. ISO-8859-1 o ISO Latin-1
  3. UCS-2

  Se definen en la etiqueta <?xml version="1.0" encoding="UTF-8"?>

- Todas las etiquetas en WML se escriben en minúsculas. Hay dos tipos de etiquetas, las contienen elementos, para lo cual hay una etiqueta de inicio y otra de fin. Los atributos de las etiquetas han de ir siempre en la etiqueta de inicio.

  <etiqueta> Inicio

  </etiqueta> Fin

  Y las etiquetas que no contienen elementos tienen el siguiente formato: <etiqueta/>

- Los comentarios al igual que en HTML tienen el siguiente formato:

  <!-- Comentario -->

Ejemplo de WML:

```
<?xml version="1.0" encoding= "iso-8859-1"?>
```

```
<?xml-stylesheet type="text/xsl" href="../xslt/wml/doc.xsl"?>
<!DOCTYPE wml PUBLIC  " -//WAPFORUM//DTD WML 1.1//EN" "
http://www.wapforum.org/DTD/wml_1.1.xml">
<wml>
      <template>
            <do name ="anterior" label = "Anterior" type = "prev">
                  <prev/>
            </do>
      </template>
      <card id = "0">
            <p align = "center">HIPERCONT </p>
            <p align = "center"></p>
            <p>Este es el primer tutor hecho en hipercont</p>
            <p>Este es el primer tutor hecho en hipercont
            <do name =" anterior" type = "prev">
            <noop/>
            </do>
            <do name ="Aceptar" type = "accept">
                   <go href="#1" />
            </do>       </p>
      </card>
      <card id = "1">
            <p>Este es el primer tutor hecho en hipercont</p>
            <p>         </p>
                  viñeta 1 viñeta 2
            <p>
                  <img height ="106" src ="file:/D:/img/Dibujo.gif"
                  width ="106"/>
            </p>
            <p>
            <a href = "D:/img/copiar.gif">Hipervinculo</a>
            <do name =" anterior" type = "prev">
                   <noop/>
            </do>
            <do name ="Aceptar" type = "accept">
                   <go href="#2" />
```

```
            </do>
            </p>
            </card>
            <card id = "2">
                  <p>         </p>
                  <table columns = "2">               <tr>
                        <td> tabla        </td>
                        <td> tabla        </td>
                        </tr>
                        <tr>              <td> celda  </td>
                        <td> celda  </td>
                        </tr>
                  </table>
            <p>         </p>
            <p>
                  <do name =" anterior" type = "prev">
                   <noop/>
                  </do>
                  <do name ="Aceptar" type = "accept">
                   <go href="#3" />
                  </do>
            </p>
      </card>
</wml>
```

Utilizando el simulador **Openwave V7 Simulator** se obtiene la siguiente salida.

Figura 3.3: **Salida WML.**

CONCLUSIONES Y RECOMENDACIONES

Conclusiones

- Hipermedia es un enfoque flexible y muy eficiente, para la gestión de información, facilita la navegación a través de la información distribuida, dando lugar a sistemas útiles y fáciles de usar.

- El lenguaje de modelo de datos orientado a objetos UML, es uno de los más apropiados para documentar, especificar y visualizar un sistema gracias a que modela los procesos en forma ordenada y eficiente.

- La programación orientada a objetos permite llegar a la etapa de implementación con un diseño lo suficientemente explícito, debido al refinamiento sucesivo en la construcción de las aplicaciones y a la ejemplarización del mundo real a través de clases.

- Esta herramienta se ha desarrollado con XML y Java que tienen características complementarias constituyendo una plataforma muy potente para la compartición y el procesamiento de datos y documentos.

- Hipercont es una herramienta amigable al usuario que permite una adecuada organización de los proyectos así como de los documentos que se vayan generando.

- XML-Shema describe la estructura y controla los datos del documento XML que se genera en el editor de texto.

- Las hojas de estilo XSLT como modelo de procesamiento XML poseen un nivel muy superior a SAX y DOM.

- Esta herramienta permite generar código HTML, PDF y WML, gracias a los estándares establecidos en el esquema XML y a las plantillas XSLT y XSL desarrolladas previamente.

- Los estudiantes podrán realizar una lectura no secuencial de los documentos hipermedia, acceder de diferentes modos a la información distribuida, de manera que se adapte a las diferencias individuales de cada uno.

- El sistema permite que los docentes tengan un extenso control sobre la generación de todos los aspectos de los contenidos y las relaciones que entre ellos considere oportunas.

- Los tutores generados por el sistema pueden considerarse como material complementario de los contenidos que se imparten en el aula, permitiendo a los alumnos profundizar los conocimientos sin restricciones de tiempo ni espacio.

Recomendaciones

- Se recomienda el uso del sistema Hipercont como apoyo complementario en el proceso de enseñanza-aprendizaje.

- Para el óptimo funcionamiento del sistema Hipercont, se recomienda que el editor HTML sea manipulado por usuarios que tengan conocimientos sobre este lenguaje, para evitar así errores en su desempeño.

- Se sugiere aplicar UML para visualizar, especificar y documentar cada una de las partes que comprende el desarrollo de software.

- Se propone recurrir a sistemas hipermedia para la construcción del conocimiento que permitan complementar las crecientes demandas de los estudiantes.

- Se recomienda utilizar Java y XML como herramientas potentes de programación en el procesamiento de datos y documentos.

- Se propone a los desarrolladores de software que para la construcción de sistemas utilicen nuevas tecnologías para apoyar la investigación y lograr un avance en la deficiencia tecnológica existente en nuestro medio.

- Se sugiere usar la tecnología orientada a objetos en la construcción del sistema para escribir, mantener y reutilizar los procesos.

R E S U M E N

Debido a la incorporación de nuevas herramientas en el mundo educativo. El interés de esta investigación se centra en la hipermedia como técnica para la construcción del conocimiento. Tecnología educacional capaz de competir con el concepto de enseñanza en una realidad tangible donde la demanda de una educación personalizada es cada vez más creciente.

Es por esta razón que se hace necesario el desarrollo un sistema hipermedia que permita a los usuarios la creación y manipulación de Hiperdocumentos que representen un apoyo fundamental para lograr una enseñanza de calidad.

Se ha utilizado los lenguajes Java y XML para el desarrollado de un sistema que permita el manejo de documentos a través de un Editor de Texto, el mismo que ofrece a los usuarios el ingreso de texto, imágenes y creación de hipervínculos. Este sistema cuenta también con un administrador de proyectos para una adecuada organización de los documentos que se van generando.

Para la generación de documentos en otros formatos, se procede ha convertir los elementos ingresados por el usuario a través del editor de texto, a un nuevo árbol de resultados XML mediante el uso de un esquema de validación de datos XML-Schema.

La gran parte de la potencia de XML no esta en el propio XML sino en las herramientas que lo rodean, se ha elaborado hojas de estilo XSL y XSLT para

crear salidas flexibles en formatos HTML y PDF, siendo posible extender esto a otro formato como el WML.

B I B L I O G R A F Í A

- **AKIF, BRODHEAD, CIOROIANU, HART, WRITZ, Java y XML, Anaya Multimedia / Wrox, Madrid, 2002.**
- **BIRBECK y otros, Profesional XML, II Edición, Wrox Press, U.S.A, 2001.**
- **BURKE Eric, Java and XSLT, O' Reilly & Associates, U.S.A, 2001.**
- **HALL Marty, BROWN Larry, Core WEB Programming, Sun Microsystems Press, U.S.A, 2001.**
- **LARMAN, Craig. UML y Patrones, Adisson Wesley, Traducción Luz María Hernandez. México. 1999.**
- **VLIST Eric van der, XML Schema, O' Reilly & Associates, U.S.A, 2002.**
- **http://www.jdom.org Información sobre JDOM y la API actual para la última versión beta.**
- http://www.libros-electronicos.net Diseño orientado a objetos con UML.
- http://www.terra.es/personal/wapfacil Manual de WML.
- http://xml.apache.org/fop/index.html Agente de presentación FOP de Apache.
- http://www.webestilo.com/wml Información sobre la tecnología WAP.

ANEXOS

Cuestionario para la evaluación del sistema hipercont
Perfil de usuario

Nombre:

Ocupación:

Características del sistema.

Por favor, marque las casillas que mejor represente su opinión sobre la utilización del sistema.

Sistema Hipercont	
La distribución de los elementos de la aplicación (barras de desplazamiento, zona de contenido, botones, etc.) es..	Excelente ____ Buena ____ Regular ____ Mala ____
La manipulación de la aplicación es...	Fácil ____ Regular ____ Compleja ____
La ayuda del sistema resuelve fácilmente las dudas del usuario.	Si ____ No ____
La velocidad de funcionamiento de la aplicación, considerando el tipo de tarea que exige, es...	Excelente ____ Buena ____ Regular ____ Mala ____
El sistema permite una adecuada organización de la información a través del manejo de proyectos.	Si ____ No ____
La presentación de los documentos generados (HTML y PDF) es aceptable.	Si ____ No ____
La incorporación de imágenes texto, tablas, hipervínculos a través del editor de texto es...	Fácil ____ Regular ____ Compleja ____
Para la creación de documentos en el sistema, necesitó tener conocimientos previos de lenguaje XML.	Si ____ No ____

Observaciones:.....
.....

.....

Firma