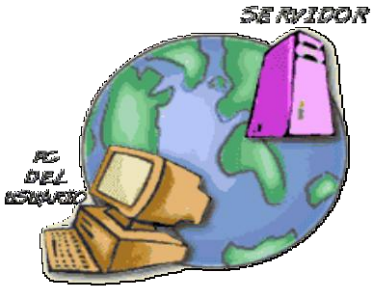
Universidad de las Ciencias Informáticas

FACULTAD 5

Trabajo de Diploma para optar por el Título de

Ingeniero en Ciencias Informáticas

Análisis, diseño e implementación de un software con tecnología multimedia para la simulación del funcionamiento de arquitecturas de Firewall



AUTOR: Freddy Ibargollín Gavilán

TUTORA: Ing. Ruth Yurina Vega Cutiño

Ciudad de La Habana, junio del 2009

"Año del 50 Aniversario del triunfo de la Revolución"

Resumen

El presente trabajo está enmarcado en el objetivo de realizar el análisis, diseño e implementación del software educativo con tecnología multimedia para la simulación del funcionamiento de arquitecturas de Firewall, relacionado con el departamento de Sistemas Digitales, con el cual se pretende enseñar, de forma amena e interactiva la importancia de estos sistemas para la seguridad de los sistemas informáticos, como configurarlos, identificarlos, visualizar su funcionamiento y otros temas de interés. Este documento recoge los resultados del trabajo realizado, en donde se hace un estudio exhaustivo de los principales modelos, metodologías y estándares para el desarrollo de este tipo de software, así como de las tendencias y tecnologías actuales, permitiendo seleccionar las más adecuadas que apoyen la solución del problema y las herramientas de desarrollo a emplear en la producción del mismo. Se cumplió el objetivo propuesto a partir del empleo de la metodología RUP (Proceso Unificado de Desarrollo de Software), utilizando además OMMMA-L como extensión de UML para el modelado, lográndose identificar las funcionalidades y el modo en que estas deben ser implementadas para obtener el modelado que facilitará, viabilizará y apoyará el proceso de producción del software. Como conclusiones generales, se determina que los objetivos planteados han sido cumplidos de manera satisfactoria, y como complemento se proponen una serie de recomendaciones para posteriores mejoras.

Agradecimientos

A mi madre, mi padre, mi familia y amigos cercanos de Holguín que crecieron conmigo y que me brindaron su apoyo en los momentos más difíciles.

A mis primos y el resto de mi familia que siempre me apoyaron durante todo este tiempo. A mi madrina que me ayudó mucho en estos 5 años.

A mi grupo de 1er año y 2do año, a mis compañeros de cuarto y todas esas amistades que estuvieron ahí cuando realmente hacían falta.

Por último, un especial agradecimiento a mi novia por estar a mi lado y apoyarme en todo lo que necesité para la realización de este trabajo.

A todos ustedes.

Muchas Gracias.

Dedicatoria

A mi familia, especialmente a mis padres por ayudarme durante estos 5 años de universidad en todo lo que necesitaba.

A mi novia que me ayudo mucho en la realización de este trabajo, brindándome su apoyo en todo momento.

A mi tutora y a todas esas personas que intervinieron en un momento u otro en la realización de este trabajo.

Índice

Índice de tablas

Índice de figuras

Introducción

En la actualidad, las organizaciones son cada vez más dependientes de sus redes informáticas y un problema que las afecte, por mínimo que sea, puede llegar a comprometer la continuidad de las operaciones. La seguridad de la información se ha convertido en un área clave en el mundo interconectado de hoy para evitar los ataques de los Hackers, programas malignos y otros peligros tecnológicos, que cada vez están más organizados, por lo que van adquiriendo día a día habilidades más especializadas, obteniendo así mayores beneficios.

Una de las tecnologías más comunes para ello son los Firewalls, medios imprescindibles en la seguridad de las redes que constituyen la mejor manera de proteger el ordenador personal contra los ataques maliciosos que existen actualmente en Internet. Debido a esto, se hace necesario el conocimiento de su funcionamiento para cualquier sistema informático, por lo que su estudio y desarrollo evoluciona constantemente en todas partes del mundo.

Cuba inserta estos temas en el proceso de enseñanza de los estudiantes universitarios, pero las nuevas tecnologías de la información forman parte de la vida cotidiana y muchas veces, constituye un impedimento la existencia de pocos software que contengan información al respecto, que serían un medio esencial para lograr un aprendizaje mucho más rápido, no existiendo así la posibilidad de un adecuado entendimiento acerca de como probar, configurar y comprender el funcionamiento de los Firewalls y los equipos de red de alta tecnología.

El desarrollo de software para la educación se ha convertido en una necesidad insoslayable para todo sistema educativo, de manera que puedan ser explotados, de la forma más eficiente posible, las Tecnologías de la Información y las Comunicaciones que como resultado del desarrollo tecnológico, están introduciéndose en el entorno educacional de muchos países del mundo. Una de las mayores aplicaciones de la informática es la producción de software educativo. En la Universidad de las Ciencias Informáticas, reconocido centro cubano de altos estudios que ha acumulado una vasta experiencia en la creación de software, mejorando y perfeccionando con el paso del tiempo la calidad en el proceso de desarrollo, debido al crecimiento descontrolado de amenazas en las redes, se desarrollan esfuerzos encaminados a lograr tal propósito, decidiéndose así la creación de una aplicación que aborde dichos temas.

Teniendo en cuenta lo anteriormente expuesto se plantea como ***Problema científico la:*** Inexistencia de materiales audiovisuales que sirvan de apoyo para el proceso de enseñanza en el aprendizaje del funcionamiento de los Firewalls en la Informática.

El problema científico se enmarca en el ***objeto de estudio:*** Abordando acerca del proceso de enseñanza-aprendizaje de los estudiantes universitarios esencialmente dentro de nuestra universidad.

Para resolver el problema se plantea el siguiente ***objetivo general*****:** Desarrollar un software educativo con tecnología multimedia que sirva de apoyo al aprendizaje sobre el funcionamiento de los Firewalls.

Situándose como ***campo de acción:*** Proceso de desarrollo de Software educativo mediante la aplicación de las Tecnologías de Información y las Comunicaciones (TIC)

Para lograr el objetivo se plantean las siguientes ***tareas de investigación***:

1. Documentarse sobre el funcionamiento de los Firewalls.

2. Analizar las características de aplicaciones informáticas que simulen el funcionamiento de los equipos de la red.

3. Documentarse sobre el funcionamiento de las herramientas de autor: Flash.

4. Analizar posibles bases de datos a utilizar que sean compatibles con la herramienta.

5. Diseñar una interfaz grafica que cumpla con los objetivos establecidos.

6. Realizar las pruebas pertinentes a la aplicación para asegurar y comprobar su correcto funcionamiento.

El documento se encuentra estructurado en: la presente introducción, 4 capítulos, conclusiones, recomendaciones, referencias bibliográficas, bibliografía, glosario de términos y anexos.

En el **Capítulo 1: Fundamentación teórica** se presenta un análisis de las tendencias actuales del progreso del software con tecnología multimedia, así como de las metodologías de desarrollo y herramientas de autor más comunes con el objetivo de seleccionar las más adecuadas o idóneas para ser empleadas en la solución del problema.

En el **Capítulo 2: Descripción de la Solución Propuesta** se hace referencia a la solución que se propone para el desarrollo del software. Se presentan diferentes modelos y diagramas que

fundamentan el proceso de desarrollo.

En el **Capítulo 3: Construcción de la Solución Propuesta** se realiza la construcción de la solución propuesta, exponiendo los artefactos esenciales del modelo de diseño. Se presentan, fundamentando el proceso de desarrollo, los diagramas de presentación y navegación y queda plasmado el hardware necesario para el funcionamiento de la aplicación, además del modelo de implementación fundamentado con la realización de los diagramas de componentes.

En el **Capítulo 4: Estudio de factibilidad** se lleva a cabo un estudio y análisis completo de la factibilidad, teniendo en cuenta aspectos que influyen en la ejecución de una aplicación como la planificación, el tiempo de desarrollo, los costos y los beneficios tangibles e intangibles que reportará el software.

Capítulo 1. Fundamentación del Tema

1.1 Introducción del capítulo

En este capítulo se realiza un estudio y valoración sobre el desarrollo actual del software y su evidente impacto hoy en día, haciendo énfasis en el software educativo y su creación usando tecnología multimedia. Se expone el resultado del análisis de algunas metodologías de desarrollo de software más comúnmente usadas, además de varias tecnologías y herramientas de autor afines, teniendo en cuenta sus características, importancia, ventajas y desventajas, para seleccionar cuales utilizar para el modelado y confección del producto.

1.2 Software Educativo

A lo largo de la última década, como consecuencia del avance tecnológico, los medios de enseñanza han evolucionado, facilitando la inclusión de los medios informáticos en el aula, que ayudan a favorecer el aprendizaje, pero: “No es la tecnología la que debe salir a la búsqueda de una aplicación educativa. Por el contrario, la idea es partir de la educación, la que ante una necesidad busca un soporte tecnológico para que sea socio en la construcción significativa del aprender” [1]

La necesidad de introducir la tecnología en los procesos docentes se ha hecho evidente, dentro de los cuales juega un papel fundamental el software educativo.

“Software educativo, programas educativos y programas didácticos son programas para ordenador creados con una finalidad específica de ser utilizados como medios didácticos, es decir, para facilitar los procesos de enseñanza y de aprendizaje.” [2]

Un software educativo no es más que una herramienta que facilitará impartir el conocimiento a una audiencia, además de adaptarlo al ritmo de trabajo y las necesidades de cada educando, para lograr diversas alternativas de aprendizaje en él y generar procesos cognitivos en su intelecto. En la actualidad, las tendencias actuales de software educativo se inclinan hacia la preferencia de presentación de contenidos que utilizan las técnicas hipertexto, multimedia o hipermedia, esto ocurre debido a la inclusión de las Tecnologías de la Informática y las Comunicaciones (TIC) que ha propiciado un marco ideal para que estas se desarrollen.

1.2.1 Tecnología Multimedia: una nueva tecnología de comunicación e información

La tecnología multimedia, con un gran potencial para transformar aspectos importantes, se ha convertido en el más reciente fenómeno tecnológico y cultural. Ha encontrado aplicaciones de una manera muy rápida en diferentes campos como la educación, la capacitación y la instrucción, así como en la publicidad y marketing, mostrando grandes beneficios aceptados por la sociedad.

"Multimedia es el uso de texto y gráficas, recursos tradicionales en una computadora, combinados con el video y sonido, nuevos elementos integrados bajo el control de un programa que permite crear aplicaciones enfocadas básicamente a la capacitación y el ofrecimiento de servicios… es una tendencia de mezclar diferentes tecnologías de difusión, de información, impactando varios sentidos a la vez, para lograr un efecto mayor en la comprensión del mensaje" [3]

Los sistemas con esta tecnología permiten un activo intercambio de información a través de sistemas informáticos y pueden resultar muy útiles en los contextos educativos, no sólo posibilitando ver y oír, sino también interactuar con el objeto en cuestión. Son materiales que integran diversos elementos textuales (secuenciales e hipertextuales) y audiovisuales (voz, sonido, música, video, animación, fotografía, imagen), además de la capacidad de ser un medio interactivo, dinámico y lleno de estímulos, donde el usuario experimenta con el sistema, sintiéndose participante y no un simple espectador.

"... la multimedia representa una ventaja como SOPORTE al proceso educativo, pues presenta y manipula la información en un lenguaje contemporáneo, que además permite a maestros y alumnos jugar con su estructura para lograr diferentes objetivos pedagógicos." [4]

La tecnología multimedia permite navegar por el programa y buscar la información sin tener que recorrerlo todo, se puede aplicar como medio de aprendizaje e informativo, ofreciéndonos diferentes productos con actividades que facilitan la interacción tales como: preguntas, ejercicios, simulaciones, juegos, programas de aprendizaje y materiales de referencia, entre otros.

1.2.2 Antecedentes y desarrollo del software con tecnología multimedia

La tecnología multimedia tiene sus antecedentes desde muchos años atrás con el lanzamiento de la Macintosh por Apple Computer porque contaba con un ambiente de trabajo que permitía el diseño gráfico y la edición.

La tecnología multimedia cobra gran auge en el año 1992 con el surgimiento de los videos-juegos cuando se logran integrar todos los elementos al mismo tiempo. Tenían el fin desde entonces, de

buscar información, navegar, e interactuar sin necesidad de recorrer toda la multimedia. En este mismo año se presentó en las Vegas, en la feria Consumer Electronics Show (CES), en un CD que contenía varias funciones como sonido, imágenes, animaciones y videos en diferentes combinaciones, que permitían a su vez la interacción multimedia-usuario, comenzando a desarrollar habilidades de creatividad y facilidad de aprendizaje.

Debido a que el desarrollo de la tecnología multimedia no ha dejado de avanzar, se le han trazado una serie de normas tecnológicas y metodológicas que debe cumplir para lograr establecer una generalización de compatibilidad y transferencia. De la misma manera se han definido estándares para la construcción de equipos multimedia que requieren tener en cuenta la capacidad y velocidad de procesamiento, la capacidad de almacenamiento masivo de información, la posibilidad de almacenar y reproducir información diferenciada y de diferente naturaleza y con el ambiente en que se trabaja la información. [3]

1.3 Modelos, metodologías y herramientas para la construcción de sistemas

Debido al auge del software educativo durante los últimos años, gran parte de ellos han sido realizados en forma desorganizada y poco documentada. La diversidad y complejidad de las actividades, trae aparejado que la elaboración del producto sea difícil y riesgoso de controlar y si no somos capaces de llevar una guía de por medio, el resultado será la insatisfacción del cliente.

Para el desarrollo de un software con calidad se necesita de un plano en el que apoyarse, de un previo análisis de los aspectos generales. Habitualmente nos olvidamos de estas cuestiones de vital importancia y utilizamos herramientas de autor a nivel de implementación, donde no se recoge información alguna, haciendo complicados la calidad y el mantenimiento.

“Para desarrollar un proyecto de software es necesario establecer un enfoque disciplinado y sistemático. Las metodologías de desarrollo influyen directamente en el proceso de construcción y se elaboran a partir del marco definido por uno o más ciclos de vida”. [5]

1.3.1 Metodologías de desarrollo

Una metodología de desarrollo establece la necesidad de considerar un diseño previo a la construcción del sistema y ofrece vías para recoger las especificaciones. Constituye un proceso para la producción organizada del software, empleando una serie de pasos, con técnicas y notaciones asociadas a cada uno.

…el software debe ser desarrollado con una metodología que contemple los aspectos pedagógicos y fundamentalmente hay que evaluarlo en un contexto similar al de uso… [6]

Las metodologías proporcionan información sobre las estrategias y técnicas didácticas, así como las herramientas que se utilizan, estas han probado ser realmente útiles y brindar un valor agregado al desarrollo del software. En los últimos años ha existido una tendencia al proceso de ingeniería, por lo que ya se han propuesto diferentes metodologías, cada una de las cuales cuenta con sus propias ventajas y desventajas, las que debemos tener en cuenta, analizarlas y determinar su alcance, para luego llegar a la conclusión de cual usar que se adapte más a nuestro medio y a nuestra aplicación. A continuación se hace una valoración sobre algunas comúnmente usadas.

"…ninguna metodología hace el trabajo por sí sola, pero te podrá ayudar". [7]

Programación Extrema (XP)

Esta metodología ha recibido gran atención convirtiéndose en una de las más exitosas en la actualidad. Se debe destacar su flexibilidad, ya que muestra efectividad cuando los requisitos son muy cambiantes. Todo proyecto que la siga se ha de dividir en iteraciones de aproximadamente 3 semanas de duración, tiene como principal objetivo producir el software a corto plazo, es una metodología liviana, ágil y está orientada más a las personas por sobre las herramientas, asumiendo las características de cambio acelerado y reemplazando la documentación detallada por la comunicación cara a cara. Se emplea para proyectos de un grupo de personas pequeño. Hacen poco énfasis en la arquitectura de software, tiene pocos roles y artefactos, como particularidad debe tener el cliente como parte del equipo de desarrollo.

Consta de cuatro fases:

- *Planificación*: Definir las historias de usuario, crear un plan de publicaciones, iteraciones y reuniones bastante frecuentes, estimar la velocidad, XP aconseja la programación en parejas pues incrementa la productividad y la calidad del software desarrollado.

- *Diseño*: Se sugieren diseños simples y sencillos, elaborar glosarios de términos, identificar riesgos en pareja y reducirlo al máximo, revisar de nuevo estos códigos para procurar optimizar su funcionamiento.

- *Desarrollo o codificación*: El cliente es la parte más importante del equipo de desarrollo, su presencia es indispensable en las distintas fases de XP. La codificación debe hacerse ateniendo a

estándares de codificación ya creados. Se sugiere un modelo de trabajo colectivo usando repositorios de código.

- *Pruebas*: Se realizan pruebas a los principales procesos, de tal manera que adelantándose en algo hacia el futuro, se puedan hacer pruebas de las fallas que pudieran ocurrir, o sea obtener los posibles errores.

XP plantea la planificación como un permanente diálogo entre los empresarios y desarrolladores, en la que los primeros decidirán: [8]

1. *Alcance*: ¿Qué es lo que el software debe de resolver para que genere valor?

2. *Prioridad*: ¿Qué debe ser hecho en primer lugar?

3. *Composición de las versiones*: ¿Cuánto es necesario hacer para saber si el negocio va mejor con software que sin él?

4. *Fechas de versiones*: ¿Cuáles son las fechas en la presencia del software?

5. *Consecuencias*: Informar sobre las consecuencias de la toma de decisiones.

6. *Procesos*: ¿Cómo se organiza el trabajo y el equipo?

7. *Programación detallada*: Dentro de una versión ¿Qué problemas se resolverán primero?

Extreme Programming, como es llamada en inglés, consiste en una programación rápida o extrema, cuya particularidad es tener como parte del equipo al usuario final, pues es uno de los requisitos para llegar al éxito del proyecto. Además sostiene que el código es el único entregable que realmente importa, desplazando el rol del análisis y diseño durante la creación de software.

Metodología de Administración de Relaciones (RMM)

RMM se define como un proceso de análisis y diseño, basado en conceptos del Modelo de diseño de Hipertexto (HDM) y desarrollo de aplicaciones de estructura estable. Se centra en el diseño, desarrollo y fases de construcción. Impone una disciplina de trabajo sobre el proceso de desarrollo de software, con el objetivo de asegurar la obtención de una aplicación que satisfaga los requerimientos del usuario y reúna estándares aceptables de calidad.

RMM constituye una metodología tentadora para el desarrollo del proceso por el desglose de las fases de la producción y la incorporación de diagramas para el diseño de la presentación, el comportamiento dinámico y la estructura de la navegación. [9]

Esta metodología, comúnmente llamada en inglés: Relationship Management Methodology, está basada en el tradicional Modelo Entidad - Interrelación (MER), donde se modela el dominio de información a considerar; y el diseño de M-Slices, estructuras conceptuales que permiten modelar las unidades de presentación de la aplicación.

Es una metodología que asume un ciclo de vida completo, compuesto de siete etapas, en las que se va modelando la estructura de la aplicación, antes de estas etapas se determinan objetivos, se analizan necesidades y tipos de navegación. [10]

1*. Viabilidad*: se determinan los objetivos y el análisis de necesidades.

2*. Análisis de los requerimientos de navegación*: Se determinan los tipos de navegación que se utilizarán en el proyecto.

En la etapa de diseño, la más importante, se recogen los tres primeros pasos:

1*. Diseño del Diagrama Entidad Relación*: realizada una vez elaborado el diagrama de burbujas y normalización de las entidades.

2*. Diseño del Diagrama de Slices*: se determina la forma en que la información en las entidades elegidas se presentará a los usuarios, y la forma en que pueden acceder a él.

3*. Diseño del Diagrama RMDM o diseño de navegación*: se diseñan los caminos que permitan la navegación de hipertexto. Cada relación asociativa que aparecen en el diagrama ER se analiza y se indica el tipo de navegación entre las entidades.

4*. Diseño del protocolo de conversión:* Cada elemento del diagrama RMDM se transforma en objetos.

5*. Diseño de la interfaz de usuario*: Se desarrollan las pantallas para mostrar información al usuario.

6*. Diseño del funcionamiento en runtime:* Se determinan que botones de la pantalla serán visibles y cuales invisibles.

7*. Construcción*: Se desarrolla la aplicación sobre la base del diagrama RMDM en un lenguaje seleccionado.

Al final se realizan pruebas y evaluaciones experimentales para corregir errores encontrados.

"...En el lado opuesto del espectro, un trabajo artístico puede tener una estructura bastante difusa

en la cual no se observen cambios frecuentes a través del tiempo, haciendo de RMM poco aplicable." [11]

Marco de Solución de Microsoft (MSF)

Esta es una metodología flexible e interrelacionada con una serie de conceptos, modelos y prácticas de uso, que controlan la planificación, el desarrollo y la gestión de proyectos tecnológicos. [12]

Los modelos de los que se compone MSF son los encargados de planificar las diferentes partes implicadas en el desarrollo del proyecto, estos son: Modelo de arquitectura, Modelo de equipo, Modelo de proceso, Modelo de gestión del riesgo, Modelo de diseño de proceso y Modelo de aplicación.

Esta metodología, más conocida por su nombre en inglés: Microsoft Solution Framework, se centra en los modelos de proceso y de equipo dejando en un segundo plano las elecciones tecnológicas.

Consta de 4 fases: [12]

1. *Estrategia y alcance:* Elaboración y aprobación del documento de alcance y estrategia definitivo, formación del equipo de trabajo y distribución de competencias y responsabilidades, elaboración del plan de trabajo, de la matriz de riesgos y del plan de contingencia.

2. *Planificación y Prueba de concepto:* Documento de planificación y Diseño de arquitectura, Documento de plan de laboratorio, Prueba de concepto

3. *Estabilización:* Selección del entorno de prueba piloto, Gestión de Incidencias, Revisión de la documentación final de arquitectura, Elaboración de la documentación de formación y operaciones, Elaboración del plan de despliegue, Elaboración del plan de formación.

4. *Despliegue:* Se llevarán a cabo los planes diseñados en la anterior, principalmente el de despliegue y el de formación: registro de mejoras y sugerencias, incluyendo mejoras aportadas por los fabricantes de software, revisión de las guías y manuales de usuario, rectificación de errores y obtención de los documentos de formación definitivos, entrega de los documentos definitivos, revisión, establecimiento de los estándares de calidad, entrega del proyecto y cierre del mismo, en base a la información y experiencia obtenidos.

MSF tiene las siguientes características:

- *Adaptable***:** es parecido a un compás, usado en cualquier parte como un mapa, del cual su uso es limitado a un específico lugar.
- *Escalable***:** puede organizar equipos tan pequeños entre 3 ó 4 personas, así como también, proyectos que requieren 50 personas a más.
- *Flexible***:** puede ser utilizada en el ambiente de desarrollo de cualquier cliente.
- *Tecnología Agnóstica***:** puede ser usada para desarrollar soluciones basadas sobre cualquier tecnología.

RUP (Proceso Unificado de Software)

El proceso unificado de desarrollo constituye la metodología estándar más utilizada para el análisis, diseño, implementación y documentación de sistemas orientados a objetos. Proporciona una aproximación disciplinada a la asignación de tareas y responsabilidades. RUP actúa como modelo y puede ser adaptado y extendido, requiere un equipo de trabajo capaz de administrar un proceso complejo que basa su desarrollo en ciclos que al concluir originan una versión del producto. [13]

RUP toma en cuenta las mejores prácticas en el modelo de desarrollo de software:

1. Desarrollo de software en forma iterativa (repite una acción)
2. Manejo de requerimientos
3. Utiliza arquitectura basada en componentes
4. Modela el software visualmente
5. Verifica la calidad del software
6. Controla los cambios

Cada ciclo RUP consta de cuatro fases que llevan a cabo el cumplimiento de los objetivos propuestos y en su culminación el alcance de un hito siendo respectivamente: objetivos del ciclo de vida, arquitectura del ciclo de vida, funcionalidad operativa inicial y la versión del producto, estas son:

1. Concepción o inicio: Comprender los requisitos y determinar visión y alcance del proyecto.

2. Elaboración: Asignar recursos, especificar las características y definir la arquitectura.

3. Construcción: Implementación, construir el producto operacional.

4. Transición: Hacerlo operativo para los usuarios, nivel correcto de calidad para entregar.

En RUP se han agrupado las actividades en grupos lógicos definiéndose nueve flujos de trabajo principales. Los seis primeros son los flujos de ingeniería, estos son:

1. Modelamiento de negocio: se entienden las necesidades del negocio.

2. Requerimientos: se trasladan las necesidades del negocio a un sistema automatizado.

3. Análisis y diseño: se trasladan los requerimientos dentro de la arquitectura de software.

4. Implementación: se crea un software que se ajuste a la arquitectura y que tenga el comportamiento deseado.

5. Prueba (Testeo): se debe asegurar que el comportamiento requerido es el correcto y que todo lo solicitado está presente.

6. Instalación del software

Los tres últimos son de apoyo:

7. Administración del proyecto: para el control de los horarios y recursos.

8. Administración de configuración y cambios: permite guardar todas las versiones del proyecto.

9. Ambiente: se realiza la administración el ambiente de desarrollo.

RUP es un proceso y en su modelación define como sus principales elementos a: los trabajadores, que serán las personas involucradas en cada proceso, los flujos de actividades, que definen cuando se realizará cada secuencia de actividades, las actividades que son los procesos que se llegan a determinar en cada iteración y los artefactos que se generan en cada flujo, pueden ser un documento, modelo o un elemento de modelo. (Ver Anexo 1)

El ciclo de vida de RUP se caracteriza por: [14]

1*. Dirigido por casos de uso*: Los casos de uso reflejan lo que los usuarios futuros necesitan y desean, lo cual se capta cuando se modela el negocio y se representa a través de los requerimientos. A partir de aquí los casos de uso guían el proceso de desarrollo y representarán su realización los diferentes modelos que se obtienen, como resultado de los diferentes flujos de trabajo.

2*. Centrado en la arquitectura:* La arquitectura muestra la visión común del sistema completo en la que el equipo de proyecto y los usuarios deben estar de acuerdo, por lo que describe los elementos del modelo que son más importantes para su construcción, los cimientos del sistema que son necesarios como base para comprenderlo, desarrollarlo y producirlo económicamente.

3*. Iterativo e Incremental:* RUP propone que cada fase se desarrolle en iteraciones. Una iteración involucra actividades de todos los flujos de trabajo, aunque desarrolla fundamentalmente algunos más que otros. Es práctico dividir el trabajo en miniproyectos, y cada uno de ellos es una iteración planificada que resulta en un incremento.

Es recomendable clasificar las iteraciones y ordenarlas según su prioridad y que cada una se convierte en un entregable al cliente, trayendo como beneficio la retroalimentación. Una particularidad de esta metodología es que se hace exigente el uso de artefactos, siendo por este motivo, una de las más importantes para alcanzar un grado de certificación en el desarrollo del software.

1.3.1.1 Metodología a utilizar

Luego del estudio realizado anteriormente, donde se pudieron analizar las metodologías seleccionadas y comparar sus diferentes características y particularidades de empleo en varias situaciones, se determinó cuales no se ajustan al proceso en cuestión, en el caso de XP a pesar de su utilidad, tiene como principal objetivo producir el software a corto plazo sin importar la documentación, está más orientada a las personas que a las herramientas y plantea que el usuario debe ser un integrante más del equipo de desarrollo, lo cual sería muy difícil en este caso. RMM por su parte se centra en el diseño, desarrollo y fase de construcción, impidiéndonos así su uso, pues para el cumplimiento del objetivo de esta propuesta se deben hacer estudios previos al diseño. Mientras MSF, además de proponer la realización de gran cantidad de documentación no aplicable para la solución de este problema, se centra en el modelo de equipo dejando en un segundo plano las elecciones tecnológicas.

Se seleccionó la metodología RUP como la más adecuada, ya que además de definir ciclos de trabajo y permitir describir lo que se espera del software, permite un desarrollo centrado en una arquitectura sólida. Es adaptable para proyectos a largo plazo, proporciona un trabajo organizado por partes pequeñas que permiten una mejor calidad del producto final, a través del control de riesgos y la aplicación de pruebas.

1.3.2 Lenguajes de modelado

La metodología del Proceso Unificado (RUP) se aplica a una buena cantidad de productos y procesos de software en el mundo, pero para lograr su aplicación adecuada se hace necesario un soporte o base, con el que se alcance la compatibilidad en el análisis y diseño orientado a objetos. Es una metodología con un enfoque basado en modelos y para tal fin necesita un lenguaje bien definido para realizar el modelado del sistema a implementar; por lo que se decide realizar un análisis, basado en un estudio y evaluación comparativos, de variantes más difundidas en la actualidad, para de esta forma seleccionar el más idóneo.

A través de la historia se han desarrollado varios modelos de proceso de software, es por ello que existen diversas propuestas de lenguajes. En los últimos años, su construcción se ha convertido en una herramienta habitual en distintos ámbitos tales como el de la ingeniería de requisitos y el del modelado de procesos en organizaciones.

Un lenguaje de modelado es un conjunto de elementos o notaciones que se disponen de modo que ayuden a modelar parte de una aplicación. No todos cubren los mismos aspectos de un sistema, suelen estar ligados a metodologías de desarrollo y/o paradigmas para avanzar de una especificación inicial a un plan de implementación que debe conocer todo el equipo de desarrolladores. A continuación se muestra una breve descripción de importantes propuestas, dando así a conocer sus características principales.

UML (Lenguaje Unificado de Modelado)

Es el lenguaje de modelado de sistemas de software más conocido y utilizado por los modeladores en la actualidad. No tiene propietario y está basado en el común acuerdo de gran parte de la comunidad informática. Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema de software, no pretende ser un método de desarrollo completo, pues no incluye un proceso de desarrollo paso a paso. El UML está compuesto por diversos elementos gráficos que se combinan para conformar diagramas y cuenta con reglas para combinar tales elementos.

Este lenguaje propone un vocabulario que incluye tres categorías: elementos, relaciones y diagramas (Ver anexo 2). Los elementos son abstracciones de cosas reales o ficticias (objetos, acciones), las relaciones van a ser, como su nombre lo indica, las que enlacen los elementos entre sí y los diagramas son colecciones de elementos y sus relaciones.

UML consta de varias áreas conceptuales estas son: estructura estática, comportamiento dinámico, construcciones de implementación, organización del modelo y mecanismos de extensión. A su vez, consta de ocho vistas: estática, de casos de uso, de implementación, de despliegue, de máquinas de estado, de actividad, de interacción y de gestión del modelo; para la modelación de los productos. [15]

El lenguaje unificado está diseñado a través de un lenguaje de diagramas y artefactos, distribuidos por cada una de las vistas, fácilmente ajustables para especificar aspectos distintivos de un sistema a modelar. Dichos diagramas se agrupan en cuatro categorías: de caso de uso, estructurales, de comportamiento y de implementación, siendo el segundo y el tercero los que interactúan directamente con las descripciones de los modelos estáticos estructurales y de comportamiento dinámicos.

Diagramas de estructura estática:

- *Diagrama de clases:* Clases, interfaces y colaboraciones; así como sus colaboraciones.
- *Diagrama de objetos*: Objetos y sus relaciones.
- *Diagrama de casos de uso*: Casos de uso, actores y sus relaciones.

Diagramas de comportamiento:

- *Diagramas de interacción (secuencia y colaboración):* Objetos y sus relaciones, incluyendo los mensajes que pueden ser enviados entre ellos.
- *Diagrama de estados:* Muestra una máquina de estado que consta de estados, transiciones, eventos y actividades.
- *Diagrama de actividad:* Es un tipo especial de diagrama de estados que muestra el flujo de actividades dentro de un sistema.

Diagramas de implementación:

- *Diagrama de componentes:* Organización y las dependencias entre un conjunto de componentes.
- *Diagrama de despliegue:* Configuración de nodos de procesamiento en tiempo de ejecución y los componentes que residen en ellos.

UML incluye todos los conceptos que se consideran necesarios para utilizar un proceso moderno

iterativo, se basa en construir una sólida arquitectura para resolver requisitos dirigidos por casos de uso. Incluye aspectos conceptuales tales como procesos de negocios y funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y componentes de software reutilizables. Se puede aplicar en gran variedad de formas para dar soporte a una metodología de desarrollo pero no especifica en sí mismo qué metodología o proceso usar. [16]

Es importante resaltar que UML es un "lenguaje" para especificar y no para describir métodos o procesos. Además no es una guía para realizar el análisis y diseño, sino es un lenguaje que permite la modelación de sistemas con tecnología orientada a objetos.

UML no soporta todos los planteamientos de las aplicaciones con tecnología multimedia para modelar aspectos de la interfaz de usuario, sus características no se aplican en los entornos multimedia. Fue por esta y otras razones, gracias a la riqueza de UML y a sus facilidades de extensión, que se toma como base en el necesario desarrollo de un lenguaje para este tipo de productos: OMMMA-L, que se presenta como algo eficientemente realizable y facilita un gran rango de aspectos significativos.

Lenguaje orientado a objetos para la modelación de aplicaciones multimedia OMMMA-L

OMMMA-L se encuentra sustentado en su modelación en cuatro vistas fundamentales, donde cada una se asocia a un tipo de diagrama en particular, modifica los diagramas originales de UML de: clases, secuencia y estado. Añade como parte de la vista de presentación espacial un nuevo diagrama: el diagrama de presentación, para la representación espacial de los elementos visuales del futuro software multimedia. [17]

1. *Vista Lógica:* modelada a través del diagrama de clases, utilizando las mismas notaciones del de UML, pero incorporando las clases correspondientes a las medias. Divide en dos áreas dicho diagrama: una para la jerarquía de los tipos de media y otra para la modelación de la estructura lógica del dominio de la aplicación.

2. *Vista de Presentación espacial:* modelada a través de los diagramas de presentación, los cuales tienen el propósito de declarar las interfaces de usuario con un conjunto de estructuras delimitadas en tamaño y área, dividiéndose en objetos de visualización e interacción.

3. *Vista de Comportamiento temporal predefinido:* modelada por el diagrama de secuencia, el cual modela una secuencia de una presentación predefinida dentro de una escena, donde todos los

objetos se relacionan al mismo eje del tiempo.

4. *Vista de Control Interactivo:* modelado a través del diagrama de estado, sintácticamente igual al de UML, mas con la diferencia semántica de que el comportamiento especificado por el diagrama de secuencia se provoca automáticamente cuando se entra al estado correspondiente donde se hace referencia.

En fin con OMMMA - L se puede modelar la estructura a través de diagramas de objetos y clases, mientras que el comportamiento puede ser descrito en los diagramas de interacción, estado y actividad, además integra el comportamiento interactivo con el de procedimientos temporales para lograr la descripción de aplicaciones que reaccionan ante eventos externos y producen ejecuciones dinámicas predecibles en tiempo de ejecución.

En la última década se ha logrado en la modelación de entornos educativos, la incorporación de lenguajes de propósito general como UML, o sus extensiones como OMMMA - L. [18]

1.3.2.1 Lenguaje a utilizar

Luego de realizar un análisis sobre algunos lenguajes de modelado de software se llega a la conclusión que UML, a pesar de ser el lenguaje más conocido y utilizado por los modeladores en la actualidad, no soporta, con sus vistas y diagramas, todos los aspectos de las aplicaciones multimedia de una forma adecuada. Por lo que se escoge para el modelado del software el Lenguaje orientado a objetos para la modelación de aplicaciones multimedia OMMMA-L, presentada como algo eficientemente realizable que facilita un gran rango de aspectos de aplicaciones multimedia interactivas; no podemos decir que este es un lenguaje nuevo, sino una extensión del UML que permite operar con productos de este tipo de tecnología, que se desarrollen en ambientes orientados a objetos y no es necesario aprenderlo, sino centrarse en la lógica de funcionamiento de este tipo de software e interpretar sus características extendidas.

1.3.3 Herramientas de ingeniería asistida por computadoras (Computer Aided Software Engineering (CASE))

Con el actual aumento de la complejidad en la producción de sistemas informáticos se hace necesario documentar toda la información relacionada con el proceso de desarrollo y, aparejado a esto, el conocimiento de herramientas de modelado visual que nos ayuden a comprender mejor la aplicación que estamos desarrollando; usarlas hace más portable la documentación a generar, proporciona la reutilización de componentes y permite generar automáticamente el código base.

En este caso se utilizará Racional Rose, teniendo en cuenta las características de la metodología a aplicar, debido al conocimiento previo de la misma, ya que estudiarla atrasaría el proceso de desarrollo.

Rational como soporte a la metodología

Rational es una herramienta de modelado visual de sistemas orientados a objetos, su vista de explorador está compuesta por cuatro de las vistas arquitectónicas: Vista de casos de uso, Vista lógica, Vista de componentes y Vista de despliegue, permitiendo así agrupar en paquetes los elementos de los modelos. Facilita la generación de código en lenguajes como C++, Visual Basic, Java y Ada. Soporta realizar ingeniería inversa por lo que se puede obtener un diseño a partir del código de un programa. Permite varias personas trabajando a la vez en un proceso iterativo controlado, para ello posibilita que cada desarrollador opere en un espacio de trabajo privado que contiene el modelo completo.

Rational Rose cubre todo el ciclo de vida de un proyecto: concepción y formalización del modelo, construcción de los componentes, transición a los usuarios y certificación de las distintas fases y entregables. Propone la utilización de cuatro tipos de modelos para realizar un diseño del sistema, utilizando una vista estática y otra dinámica de los modelos del sistema, uno lógico y otro físico." [14]

Esta herramienta enmarca los requerimientos y variables del diseño, permitiendo crear un ciclo de desarrollo más pequeño, robusto y adaptable. Evidenciando así las características:

1. Reusabilidad: Mediante el modelado visual se pueden crear componentes (modelos), que al salvarlos pueden ser compartidos y reutilizados por varios proyectos, permitiendo que los cambios sean fácilmente incorporados a proyectos existentes.

2. Desarrollo basado en componentes: El desarrollo basado en componentes se ha convertido en el proceso de diseño más efectivo, los usuarios de Rational pueden modelar sus componentes e interfaces sólo haciendo un "arrastrar y soltar" de los componentes del sistema hacia el diagrama de componentes.

En fin, esta herramienta permite hacer análisis y diseño para un producto de software ayudando a los equipos de desarrollo a construir, integrar, extender, modernizar e implantar.

1.4 Herramientas de autor

Existen múltiples herramientas que manejan elementos de media asociados a la programación con el objetivo de lograr una funcionalidad didáctica, brindan el marco esencial para organizar y editar elementos de un producto interactivo. Estas son utilizadas para diseñar las interfaces del usuario, con el fin de presentar un proyecto en pantalla con un ambiente integrado del contenido y las funciones, según la decisión del cliente o el desarrollador en algunos casos.

Se pueden encontrar herramientas que son verdaderos editores de obras de este tipo, con un arsenal de recursos para la integración y el manejo de los diferentes tipos de información incluyendo las animaciones. Algunas proporcionan funciones para establecer enlaces hipertextuales e hipermediales y dotan a la obra de una alta interactividad empleando múltiples y diversos modos de navegación. [19]

Algunas de las más potentes y utilizadas en el mundo para hacer software interactivo, que combinan las ventajas de la edición visual de la obra y la libertad y poder de la programación, son: Macromedia Flash, Director, Toolbook, entre otros, los cuales son utilizados según la decisión del cliente o el desarrollador en algunos casos.

Macromedia Director

Director es una poderosa herramienta, de fácil manejo y proporciona desarrollar aplicaciones sin apenas tener que programar. Permite la combinación de texto, gráficos, sonido, animación y vídeo en un documento que se reproduce en el ordenador y que es presentado con múltiples detalles. Incorpora un rango de nuevas capacidades para satisfacer las necesidades evolutivas del desarrollador actual. Incluye nuevas y mejores eficiencias en el flujo de trabajo y la habilidad de crear contenido accesible para que sus presentaciones enriquecidas puedan ser disfrutadas por personas con discapacidades.

La filosofía seguida por este programa es la de una línea de tiempo durante el cual irán sucediendo diferentes acontecimientos según se vayan necesitando. Este proceso no tiene necesariamente por qué ser lineal ni continuo sino que permite detenerse en un punto del tiempo y saltar a otro en esa línea temporal. El lenguaje de programación orientado a objetos de Director (Lingo) agiliza los tiempos de desarrollo y ayuda a integrar a sus producciones una interactividad única y de alto nivel. [19]

Esta herramienta posee la capacidad para responder rápidamente a los cambios de los requisitos

de trabajo para reducir los cambios complejos, ayudando a hacer cambios rápidos; son esenciales para ayudar a las empresas a ser más ágiles y controlar los costes de desarrollo. Director a pesar de ser muy usada y permitir la combinación de medias requiere del uso de programas exteriores para crearlos, característica que en ocasiones impide su utilización, además de que no es multiplataforma.

Macromedia Flash

Desde hace ya algunos años la aparición de Flash, con sus diferentes versiones, ha traído hasta nuestras pantallas todo un mundo de interactividad, movimiento y color. Esta herramienta resulta fácil de aprender y sus archivos están creados para reproducirse rápidamente. Además posee grandes posibilidades para lograr presentaciones de animación y manejo del sonido, texto, video y gráficos.

Los archivos de Flash aparecen muy a menudo como animaciones en páginas Web, pues consigue dar dinamismo mediante animaciones y crear aplicaciones interactivas que permiten al usuario ver la web como algo atractivo, no estático. En la actualidad Macromedia ha ampliado Flash más allá de las animaciones simples, esto lo convierte en una herramienta de desarrollo completa, para crear principalmente elementos multimedia. Una de las versiones más utilizadas es Flash 8, que cuenta con varias prestaciones para incorporar un servicio de video mucho más integrado. [19]

Flash no es sólo un programa para crear gráficos sino que es un lenguaje de programación, pues mediante su lenguaje script orientado a objetos: ActionScript se pueden crear programas que, por ejemplo, busquen en una base de datos o interactúen con un programa en otro lenguaje. Además nos brinda la posibilidad de exportar la película para ser vista en diferentes sistemas operativos.

ToolBook

Toolbook ofrece un ambiente de programación orientada a objeto para construir proyectos, o libros, a fin de presentar gráficamente información, como dibujos, imágenes digitalizadas a color, textos, videos, sonidos, gráficos y animaciones, lo que hace de este programa una herramienta muy útil para la elaboración de software con tecnología multimedia. Su interfaz es muy fácil ya que cuenta con la apariencia de un programa de dibujo y utiliza como lenguaje de programación el Openscript, basado en el lenguaje script.

ToolBook tiene dos niveles de trabajo: el lector y el autor y se ejecutan los guiones a nivel de

lector. A nivel autor se utilizan órdenes para crear nuevos libros, crear y modificar objetivo en las páginas y escribir guiones. ToolBook ofrece opciones de vinculación para botones y palabras claves, de forma que se pueda crear guiones de navegación identificando la página a la que debe ir. [20]

Las aplicaciones creadas con Toolbook se asemejan a la estructura de un libro (book), donde cada una de sus pantallas se corresponde con las diferentes páginas del mismo. Además, Toolbook permite asignar un mismo fondo (background) en la aplicación (mediante una imagen, color o conjunto de objetos) para optimizar los recursos tanto de la aplicación como del propio ordenador. [21]

1.4.1 Herramientas seleccionadas para el desarrollo del producto

Como resultado del anterior estudio comparativo de algunas herramientas de autor seleccionadas se puede resumir que estas presentan características que:

1. La herramienta Director aunque permite la combinación de texto, gráficos, sonido, animación y vídeo, requiere del uso de programas exteriores para crearlos. Los editores presentes en Director son muy limitados y principalmente sirven para crear objetos muy sencillos, o para retocar los creados externamente. Necesita de un cierto grado de programación para dar cierta interactividad a la aplicación que se desee desarrollar, además de que los programadores deben tener conocimiento del lenguaje de programación por la complejidad del mismo.

2. Toolbook, por su parte, es una herramienta que no cuenta prácticamente con editores de recursos, únicamente se incluye un editor de mapas de bits que, por otro lado, es muy básico. No es recomendable usarla en el caso de que los programadores no tengan un mínimo de conocimiento de la misma y si no se desea perder demasiado tiempo en el aprendizaje del lenguaje de programación, pues requiere de muchas líneas de código para llevar a cabo el proceso de implementación.

Teniendo en cuenta las particularidades de cada una se selecciona para desarrollar el software con tecnología multimedia Macromedia Flash 8, la cual brinda la posibilidad de integrar el sonido de una manera sencilla, su aspecto gráfico es superior a otros entornos, su reproductor está considerado como uno de los más rápidos en iniciarse, soporta características avanzadas para la carga de datos, sonido mp3, imágenes JPEG y otras películas de Flash a través del lenguaje de marcas extensibles (*XML)*, las animaciones son fluidas, emplea gráficos vectoriales, por lo que los

archivos tienen un menor tamaño y consumen menos ancho de banda al ser transmitidos que las imágenes en mapa de bits.

Para llevar a cabo el proceso de implementación del software educativo, el equipo de desarrollo se apoya en XML puesto que permite agilizar el proceso de implementación del producto, tiene gran utilidad para almacenar los contenidos de textos y las medias del software, permite además poder cargar de forma dinámica los contenidos de la multimedia sin tener en cuenta el volumen del mismo.

XML (Lenguaje de Marcas Extensibles)

XML se propone como un estándar para el intercambio de información estructurada entre diferentes plataformas. No es realmente un lenguaje en particular, sino una manera de definir lenguajes para diferentes necesidades.

XML es una tecnología sencilla que tiene a su alrededor otras que la complementan y la hacen mucho más grande y con unas posibilidades mucho mayores. Tiene un papel muy importante en la actualidad ya que permite la compatibilidad entre sistemas para compartir la información de una manera segura, fiable y fácil. [22]

Es extensible, lo que quiere decir que una vez diseñado un lenguaje y puesto en producción, igual es posible extenderlo con la adición de nuevas etiquetas de manera de que los antiguos consumidores de la vieja versión todavía puedan entender el nuevo formato. El analizador (código que permite traducir el XML a un objeto reconocible por el computador) es un componente estándar, no es necesario crear un analizador específico para cada lenguaje, esto posibilita el empleo de uno de los tantos disponibles, de esta manera se evitan errores y se acelera el desarrollo de la aplicación. Si un tercero decide usar un documento creado en XML, es sencillo entender su estructura y procesarlo. Mejora la compatibilidad entre aplicaciones. [22]

1.5 Conclusiones del capítulo

En este capítulo se realizó un estudio, exponiendo elementos teóricos del software educativo e implicaciones de la informática en la educación actual, así como de la tecnología multimedia. Para el desarrollo de la propuesta se seleccionó RUP como la metodología a utilizar, entre otras analizadas a profundidad, fundamentando las particularidades necesarias para realizar el análisis y diseño del software que la distinguen como propuesta ideal en este proceso; conjuntamente con OMMMA-L, pues como extensión de UML ofrece sus mismas comodidades añadiéndole o

modificándole diagramas que permiten un mejor modelado para este tipo de aplicaciones. Además se utilizará para el modelado Rational, como soporte a la metodología y para la elaboración de la multimedia la herramienta de autor Macromedia Flash.

Capítulo 2. Descripción de la solución propuesta

2.1 Introducción del capítulo

En este capítulo se comienza a realizar el análisis del producto, con el apoyo del Proceso Unificado de Desarrollo de Software. Se modela el entorno donde se usará el software y se identifican y describen sus principales conceptos asociados, reflejados mediante la realización de un modelo de dominio. Se definen los requerimientos funcionales y no funcionales por los que se regirá el sistema propuesto, que generan especificaciones correctas que describan con claridad, sin ambigüedades y en forma consistente las necesidades del cliente.

2.2 Soluciones existentes

Internet, red de redes, es la solución inmediata y actual a la búsqueda de bibliografía información, sin embargo no está al alcance de todos, por lo que es desarrollo de software educativo se ha convertido en una solución factible. Actualmente en Cuba el proceso de conocimiento a desarrollar por nuestros universitarios se hace complejo (entre otros aspectos) por la escasez de recursos y medios informáticos que se ponen a disposición como apoyo a las actividades docentes educativas. Por ello de plantea la construcción de un software con tecnología multimedia, que sirva de apoyo al estudio, abordando el tema de las redes, específicamente del funcionamiento de arquitecturas de Firewall esto resulta una solución inmediata a la barrera del alcance tecnológico que presentan algunas instituciones el país.

2.3 Identificación de la audiencia

Cuando se decide realizar un producto software de alta calidad que cumpla con los objetivos para los que se concibió, una de las consideraciones más importantes es identificar correctamente y tener en cuenta el público al que va dirigido.

Es de suma importancia determinar las características del cliente a quien va dirigida la aplicación y encaminarse a satisfacer sus necesidades. Es solamente mediante una detallada comprensión de la audiencia que es posible concebir y construir mensajes y acciones que lleguen a esas personas y que generen en su pensamiento y en sus acciones la respuesta deseada. [23]

Este software educativo con tecnología multimedia está concebido para estudiantes

universitarios, pretendiendo que ellos puedan contar con un producto, que de una forma amena e instructiva visualice el funcionamiento de las distintas topologías de *Firewalls,* simulando la trayectoria de los paquetes de una red a la otra. Hay que tener en cuenta que aun siendo estudiantes universitarios la información se deberá mostrar de la forma más sencilla posible de manera facilitando así el rápido aprendizaje por parte del alumno. Debido a las dificultades que ha atravesado nuestro país por causa del que se nos ha impuesto ha tenido efecto en la poca existencia de software relacionados con la seguridad. Basándose en esto se debe poner atención hasta qué punto los estudiantes tienen conocimiento acerca de los elementos que se hacen referencia en el software.

El contenido que se abordara en el software con tecnología multimedia para la simulación de Arquitecturas de Firewall se pretende distribuir por medio de un soporte digital, para lo cual fue seleccionada la plataforma de aprendizaje EVA, por su facilidad de acceso, y así los usuarios podrán consultarla cada vez que sea necesario por los mismos.

2.4 Aspectos generales

Antes de implementar este software educativo es importante destacar por qué se decide hacer o qué objetivos se persiguen con el mismo:

1. Obtener una aplicación que permita la retroalimentación del estudiante para comprender el funcionamiento de las distintas arquitecturas de *Firewalls*.

2. Desarrollar un software con una interfaz cómoda e interactiva.

3. Permitir cierto grado de flexibilidad en el estudiante para definir la topología y la selección del tipo de *Firewall*.

4. Proporcionar configuración predefinida y permitirle al estudiante definir una propia configuración.

5. Visualizar el funcionamiento de la topología de *Firewalls,* simulando la trayectoria de los paquetes de una red a la otra.

2.5 Solución propuesta

La solución más idónea para resolver el problema, es la creación de un software con tecnología multimedia sobre la simulación de arquitecturas de Firewall, en el cual se presenta inicialmente la introducción e información sobre el personal que se relaciona con el desarrollo del producto,

se pretende que este sirva de apoyo al desarrollo docente educativo de nuestras universidades, en las cuales se imparten clases de este tipo mediante el uso de las nuevas tecnologías de la Información y las Comunicaciones (TIC). Aquí se adapta el *texto* al espacio de la pantalla electrónica, pues el *texto,* o más genéricamente el *contenido,* en el espacio digital son los textos propiamente dichos, pero también las imágenes, los vídeos, los sonidos, los elementos que representan el esqueleto, la forma y la estética del producto multimedia. Se mostrara una pequeña presentación de inicio para luego aparecer los distintos botones de acceso al contenido. Las arquitecturas se muestran en una tabla de contenidos donde se hace referencia a las distintas arquitecturas desarrolladas, donde el usuario puede seleccionar diferentes tipos de arquitectura predefinidas, mostrándose una pequeña descripción de la misma con sus principales características. Estas arquitecturas predefinidas serán cuatro y estarán divididas de la siguiente forma: Filtrado de Paquetes, Dual Homed Gateway, Screened Host y Screened Subnet. A lo largo del cuerpo del tema definido existirán palabras de difícil comprensión las cuales encontrarán su significado dentro de las páginas del glosario. En la sección arquitecturas predefinidas el usuario encontrará la posibilidad de enviar un paquete a través de un Firewall de cada tipo de las arquitecturas previamente mencionadas. Las cuales abarcarán de cierta forma todo lo necesario para hacer comprender al usuario cómo funcionan las mismas, permitiendo realizar una simulación con distintos datos ofrecidos por el usuario. Dentro de la galería se mostrarán diferentes imágenes o videos que harán más completa la información que se ofrecen en las secciones de contenido para que el usuario pueda comprender mejor los distintos temas tratados. También poseerá un vínculo que dará control sobre la música de fondo, la que permitirá durante el trabajo con la multimedia deshabilitarla o habilitarla a petición del usuario para su propia comodidad. También se ofrecerá la ayuda en la que el usuario encontrará a modo de información como navegar a lo largo del software, además de las opciones de retorno a la pantalla de inicio de la multimedia y de la solicitud de cierre, que finalmente concluya con la aplicación.

2.6 Modelo de Dominio

El análisis de sistemas implica determinar las necesidades del cliente para poder especificar los aspectos que sirvan como base para el desarrollo de un software siendo el “que” de un producto informático. Para capturar correctamente los requisitos se necesita tener un firme conocimiento del funcionamiento del objeto de estudio y lograr de esta forma construir una aplicación

adecuada.

En este caso, existe poca estructuración de lo que se desea hacer, por lo que no se puede considerar que exista un negocio bien definido y consistente que permita determinar las responsabilidades, entidades y funcionamiento de la situación a modelar. Se decide representar los conceptos que se manejan en la situación existente a través de un Modelo de dominio, el cual es una de las alternativas que brinda RUP para la identificación de requisitos y la comprensión del contexto cuando existe poca estructuración en los procesos de negocio. Así los involucrados en el desarrollo tendrán a su alcance un vocabulario común que permita el entendimiento del contexto del sistema.

En este modelo se representan las clases conceptúales que pueden intervenir en el sistema y sus asociaciones, así como los objetos más importantes en el mismo. Estos objetos del dominio representan "cosas" que existen o los eventos que acontecen en el medio en el que se desenvuelve la aplicación. [24]

2.6.1 Identificación de los conceptos del dominio

Arquitectura: Jerarquía más alta del contenido del software, esta contendrá dentro de sí todo lo referente acerca de las diferentes arquitecturas de Firewalls. La cual a su vez contendrá el subtema.

Arquitecturas Predefinidas que serán los contenidos específicos acerca de cada una de ellas.

Usuario: Cualquier persona que quiera interactuar con el software en busca de aprendizaje e información.

Tema: Contenido que aborda cada arquitectura predefinida.

Introducción: Breve resumen que da preámbulo a una arquitectura determinada.

Medias: Diferentes imágenes, animaciones o videos que se mostrarán para hacer más completa la información que se ofrece al usuario.

Palabras calientes: Términos de raro entendimiento que se mostrarán con su significado para ser conocido por el usuario.

2.6.2 Diagrama de clases del modelo del dominio

El modelo de dominio de la aplicación a crear, mostrado en la figura 1, se describe con un

diagrama de clases conceptuales significativas, brindando una mayor comprensión de los eventos que suceden en el entorno que se desea modelar.

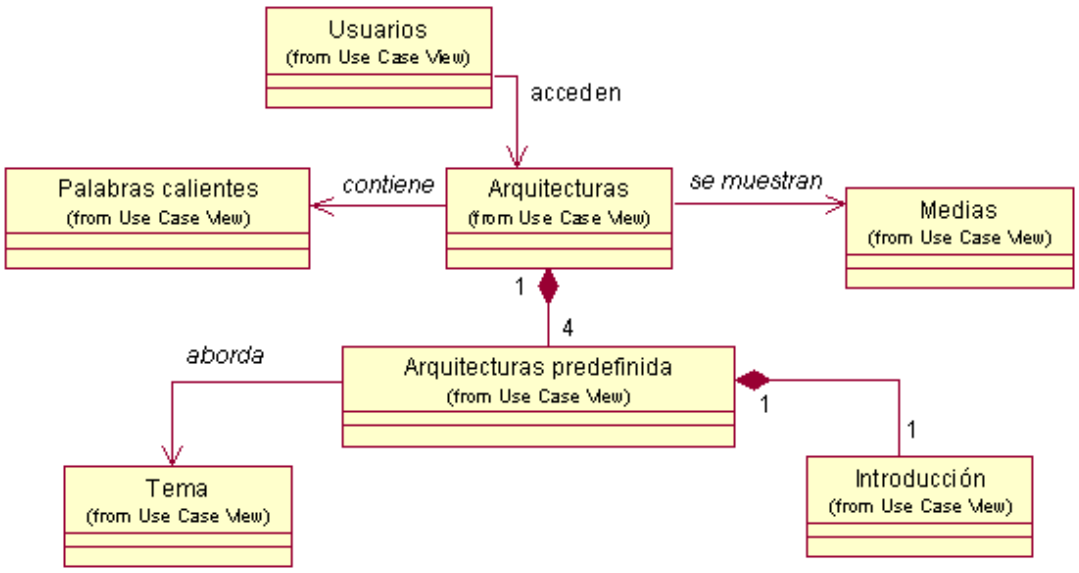


Figura 1: Diagrama de clases del Modelo de Dominio

2.7 Descripción de la funcionalidad

A través de los años de ha podido constatar que los requerimientos son la pieza fundamental en un proyecto de desarrollo de software, ya que marcan el punto de partida para actividades como la planeación, contribuyen a la identificación de las funcionalidades, el comportamiento de entrada y salida del sistema además generan especificaciones correctas que describen con claridad, sin ambigüedades y en forma consistente las características o cualidades que debe poseer la futura aplicación.

La especificación de requerimientos es la base que permite verificar si se alcanzaron o no los objetivos establecidos en el proyecto ya que estos son un reflejo detallado de las necesidades de los clientes o usuarios del sistema y es contra lo que se va a estar verificando si se están cumpliendo las metas trazadas. [24].

Los requisitos de software pueden dividirse en dos categorías: funcionales y no funcionales, en este caso como resultado de las peticiones del cliente se definen 9 de los primeros y 4 de los segundos.

2.7.1 Requerimientos funcionales

A continuación se muestran los requerimientos funcionales del sistema, los cuales especifican acciones que el sistema debe ser capaz de realizar, sin considerar restricciones físicas.

Estos describen las transformaciones a realizar sobre las entradas para producir salidas. [25] Al tiempo que avanza el proyecto estos se convierten en los algoritmos, la lógica y gran parte del código del sistema.

Lista de Requisitos Funcionales

R#	**Descripción de la Función**
R 1	Mostrar presentación del producto
R 2	Mostrar pantalla introductoria
R 3	Seleccionar tema de contenido
R 3.1	Mostrar contenido referente del tema seleccionado
R 4	Seleccionar Glosario
R 4.1	Mostrar información acerca del Glosario
R 5	Mostrar Créditos
R 6	Mostrar arquitecturas de Firewalls
R 6.1	Seleccionar arquitectura predefinida
R 6.2	Mostrar Información acerca de la arquitectura seleccionada
R 7.	Realizar envío de paquete a través del Firewall
R 7.1	Permitir al usuario definir parámetros del Paquete
R 7.2	Permitir al usuario crear las reglas del Firewall
R 8	Controlar sonido
R 8.1	Habilitar o deshabilitar sonido
R 9	Mostrar Ayuda
R 9.1	Seleccionar tema de la Ayuda
R 10	Permitir la salida desde cualquier pantalla
R 10.1	Permitir la salida de la aplicación

2.7.2 Requerimientos no funcionales

A continuación se muestran los requerimientos no funcionales del sistema, estos expresan las características o propiedades que de una u otra forma puedan limitar el sistema, que hacen al producto atractivo, usable, rápido o confiable. En muchos casos estos requisitos son fundamentales en el éxito del producto.

Los Requisitos no funcionales no alteran la funcionalidad del producto, esto quiere decir que los funcionales se mantienen invariables sin importarle con que cualidades se relacionen. Sin embargo la razón fundamental de que esta funcionalidad sea parte del producto es brindarle a este las características deseadas. [17]

1. Requerimientos de Software: El producto se deberá correr sin dificultad en el sistema operativo Windows XP, incluyéndose Linux aunque por no poseer una herramienta de este tipo capaz de visualizar la multimedia se podrá visualizar la misma mediante cualquier navegador web.
2. Restricciones en el diseño y la implementación:
    - Implementar el producto utilizando el lenguaje de programación ActionScript_2.0 y XML.
3. Requerimientos de apariencia o interfaz externa**:**

El producto deberá ser:

- Legible
- Simple de usar
- Autoritario, para que los usuarios se sientan confiados
- Discreto para que los usuarios no lo noten
- Interactivo
- Resolución: 800 x 600

2.8 Modelo de casos de uso del sistema

El modelado de casos de uso es la técnica más efectiva para modelar los requisitos del sistema, pues los casos de uso se utilizan para representar el funcionamiento o como el cliente desea que funcione el sistema. Utilizando las facilidades del lenguaje, se representan los requisitos

funcionales capturados mediante un diagrama de casos de uso, definiendo primeramente los actores que van a interactuar.

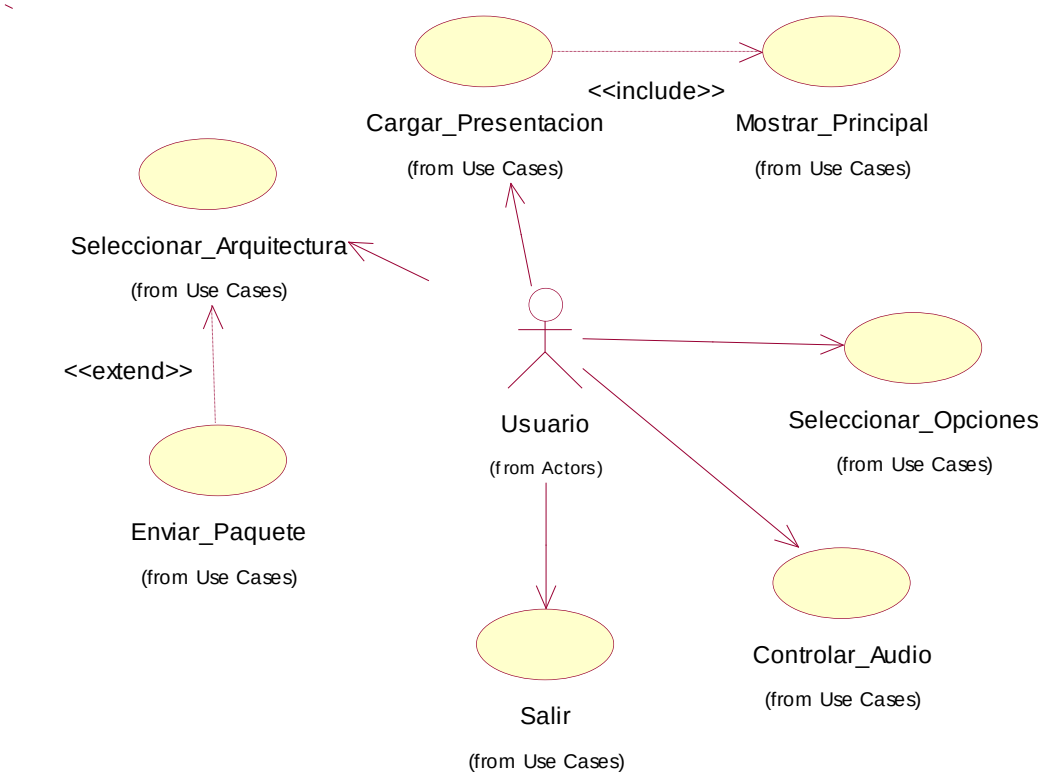


Figura 2: Diagrama de Casos de Uso

2.8.1 Actores

Los actores son el entorno del sistema, constituyen entidades que representan roles que son desempeñados por personas, otros sistemas o hardware externo que de alguna manera participan en la historia de un caso de uso o que interactúan con el sistema por decirlo de otra manera. En este caso el actor es un usuario o persona que va a utilizar el sistema en busca de conocimiento e información.

2.8.2 Casos de uso

- ✓ Caso de Uso Cargar_Presentación
- ✓ Caso de Uso Mostrar_Principal
- ✓ Caso de Uso Seleccionar_Opciones

- ✓ Caso de Uso Seleccionar_Arquitectura
- ✓ Caso de Uso Enviar_Paquete
- ✓ Caso de Uso Controlar_Audio
- ✓ Caso de Uso Salir

2.8.2.1 Descripciones textuales de los Casos de Uso

Para el logro de un mejor resultado, se describen explícitamente los casos de uso, documentándolos para un mejor entendimiento por parte de clientes y desarrolladores, los que tendrán en sus manos un artefacto esencial y significativo para su trabajo.

2.8.2.1.1 Caso de Uso para la presentación de la multimedia

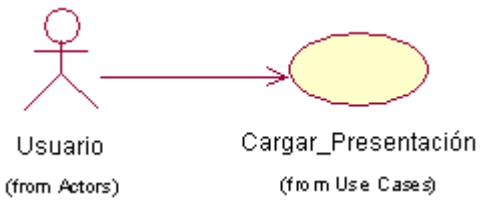


Tabla 1. Caso de Uso Cargar_Presentación

Caso de Uso	**Cargar_Presentación**
Actores	Usuario
Resumen	El caso de uso inicia cuando el usuario decide acceder a la multimedia, y se muestra una animación como presentación.
Propósito	Mostrar al usuario la presentación del sistema.
Referencias	RF1
Flujo normal de los eventos	
Acción de Actor	**Respuesta del Sistema**
1. El usuario ejecuta la aplicación	2. El sistema carga la pantalla Principal de la multimedia

Curso alterno	
Poscondiciones	

2.8.2.1.2 Caso de Uso para la Pantalla Principal de la Multimedia

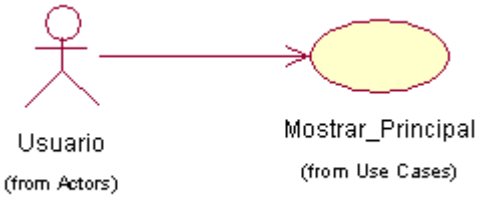


Tabla 2. Caso de Uso Mostrar_Principal.

Caso de Uso	**Mostrar_Principal**
Actores	Usuario
Resumen	El caso de uso inicia cuando culmina la animación de presentación de la multimedia y se muestra la pantalla principal con los botones de interacción disponibles, para que el usuario comience a interactuar
Propósito	Mostrar al usuario la pantalla introductoria de la multimedia.
Referencias	RF2
Flujo normal de los eventos	
Acción de Actor	**Respuesta del Sistema**
	1. El sistema carga la pantalla Principal de la multimedia con diferentes opciones de navegación. Permite ver información sobre las arquitecturas de Firewall, acceder a las opciones de Galería, Glosario y Créditos, además de poder controlar el audio y salir de

2. El usuario realiza su selección	la aplicación. 3. Se muestra la pantalla correspondiente.
Curso alterno	

2.8.2.1.3 Caso de uso para seleccionar otras opciones de contenido

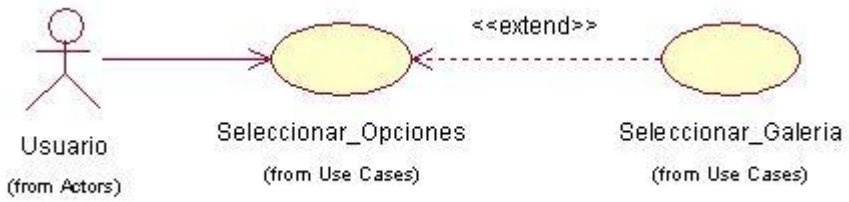


Tabla 3. Caso de uso Seleccionar_Opciones

Caso de Uso	**Seleccionar_Opciones**
Actores	Usuario
Resumen	El caso de uso se inicia cuando el usuario quiere más información general del ambiente de la multimedia y selecciona alguna opción.
Propósito	Permitirle al usuario obtener información adicional
Referencias	RF 3, RF 3.1,RF4, RF4.1, RF 5, RF 9, RF 9.1
Flujo normal de los eventos	
Acción de Actor	**Respuesta del Sistema**
1. El usuario selecciona el botón correspondiente a la opción deseada.	2. El sistema carga el archivo XML correspondiente a la información que el usuario selecciona. 3. El sistema muestra la pantalla con la información correspondiente.

Curso alterno	Sección Galería
2.1 El usuario selecciona ver la Galería	2.2 Carga la presentación correspondiente a la galería 2.3 Muestra imágenes y videos relacionados.
Poscondiciones	

2.8.2.1.4 Caso de uso para seleccionar una arquitectura predefinida

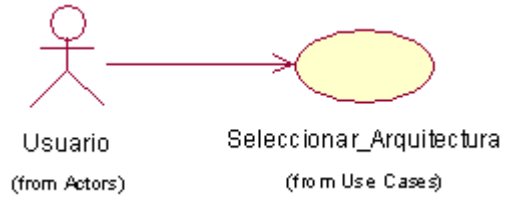


Tabla 4. Caso de uso Seleccionar_Arquitectura

Caso de Uso	**Seleccionar _Arquitectura**
Actores	Usuario
Resumen	El caso de uso se inicia cuando el usuario selecciona una de las arquitecturas de Firewall y se mostrarán los archivos con la información correspondiente.
Propósito	Permitir al usuario seleccionar una arquitectura predefinida por el sistema
Referencias	RF 6.1, RF 6.2
Flujo normal de los eventos	
Acción de Actor	**Respuesta del Sistema**
1. El usuario decide seleccionar una arquitectura de Firewall	2. El sistema carga el archivo XML y muestra información acerca de la arquitectura seleccionada. 3. El sistema brinda la opción de entender el

	funcionamiento de la arquitectura seleccionada observando el envío de un paquete a través del Firewall.
Curso alterno	
Poscondiciones	

2.8.2.1.5 Caso de uso para la simulación del envío de paquetes

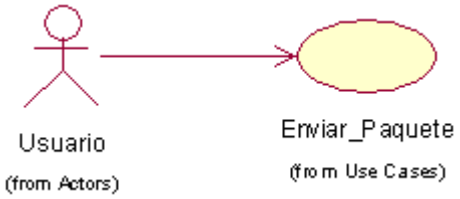


Tabla 5. Caso de uso Enviar_Paquete

Caso de Uso	**Enviar_Paquete**
Actores	Usuario
Resumen	El caso de uso se inicia después de que el usuario ha seleccionado una arquitectura de Firewall predefinida y se dispone a enviar el paquete desde su origen hasta su destino.
Propósito	Permitir al usuario realizar el envío de un paquete a través de la arquitectura seleccionada
Referencias	RF 7, RF 7.1, RF 7.2
Flujo normal de los eventos	
Acción de Actor	**Respuesta del Sistema**
1. Decide realizar el envío del paquete.	2. Solicita los parámetros necesarios para realizar la operación

7. Introduce los datos requeridos	3. Solicita los parámetros del paquete 4. Solicita puerto origen 5. Solicita puerto destino 6. Solicita el protocolo del paquete 8. Procesa los datos 9. Realiza el envío del Paquete 10. Analiza según las reglas del Firewall si permite o no el paso del paquete 11. Realiza la animación correspondiente
Curso alterno	
	8.1 En caso de error muestra un mensaje para que el usuario verifique los datos de entrada 10.1 En caso de no cumplir con las reglas mostrará la animación correspondiente
Requerimientos no funcionales	
Poscondiciones	

2.8.2.1.6 Caso de Uso para el control del audio de la multimedia

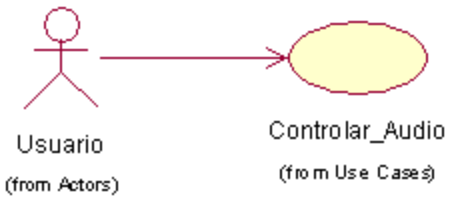


Tabla 6. Caso de Uso Controlar_Audio

Caso de Uso	**Controlar_Audio**

Actores	Usuario
Resumen	El caso de uso se inicia cuando el usuario desea controlar el audio de la pantalla de presentación.
Propósito	Permitirle al usuario controlar el audio que el sistema ofrece en la pantalla de presentación.
Referencias	RF8 , RF 8.1
Flujo normal de los eventos	
Acción de Actor	**Respuesta del Sistema**
1. El usuario desea controlar el audio 3. El usuario decide detener el audio 5. El usuario decide reproducir el audio	2. El sistema muestra un botón con un icono que representa el estado del audio(encendido o apagado) 4. El sistema detiene el audio 6. El sistema reproduce el audio
Curso alterno	
Requerimientos no funcionales	
Poscondiciones	

2.8.2.1.7 Caso de uso para salir de la multimedia

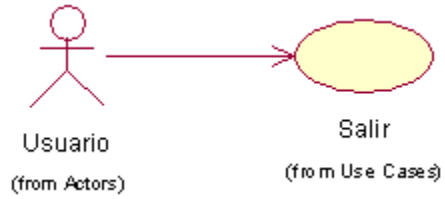


Tabla 7. Caso de uso Salir

Caso de Uso	**Salir**
Actores	Usuario
Resumen	El caso de uso se inicia cuando el usuario desea salir del sistema, para ello aparecerá una ventana con el icono de salir del sistema.
Propósito	Permitirle al usuario salir del sistema
Referencias	RF10, RF10.1
Flujo normal de los eventos	
Acción de Actor	**Respuesta del Sistema**
1. El usuario desea salir del sistema a través del botón cerrar	2. El sistema se cierra
Curso alterno	
Requerimientos no funcionales	
Poscondiciones	

2.9 Conclusiones del capítulo

Al finalizar este capítulo se obtuvo toda la información perteneciente al Modelo de dominio del entorno, mostrando sus conceptos fundamentales. Luego de un análisis del problema se definieron 10 requerimientos funcionales y 4 no funcionales que se deben cumplir para obtener un producto más efectivo, además de la definición de 7 casos de uso, de los cuales se le realizaron las correspondientes descripciones textuales, que contribuyen al comienzo de la construcción del sistema, tratando de que se cumplan todas las funciones que se han considerado necesarias para darle cumplimiento a la solución propuesta.

Capítulo 3. Construcción de la solución propuesta

3.1 Introducción del capítulo

En este capítulo se exponen los artefactos esenciales para el desarrollo que guiarán la implementación del software con tecnología multimedia, para su modelado, apoyado en el flujo de trabajo de diseño, se ha tomado como base el lenguaje de modelado OMMMA-L, que ha sido lanzado como una propuesta extensiva de UML. Se incluye el diagrama de navegación y los diagramas de presentación y se presenta el modelo de implementación fundamentado con la realización de los diagramas de componentes que se obtienen. Además del Modelo de despliegue, plasmando el hardware necesario para el funcionamiento de la aplicación.

3.2 Diagrama de navegación

El sistema de navegación general de una multimedia es el que se utiliza a partir del Índice Principal o entre sus secciones principales. Tal como muestra en la figura 3 se realizó un diagrama de navegación general, en el que han sido representadas las diferentes pantallas y las vías de acceso entre las mismas. El objetivo es facilitarle al usuario la comprensión y asimilación del sistema.

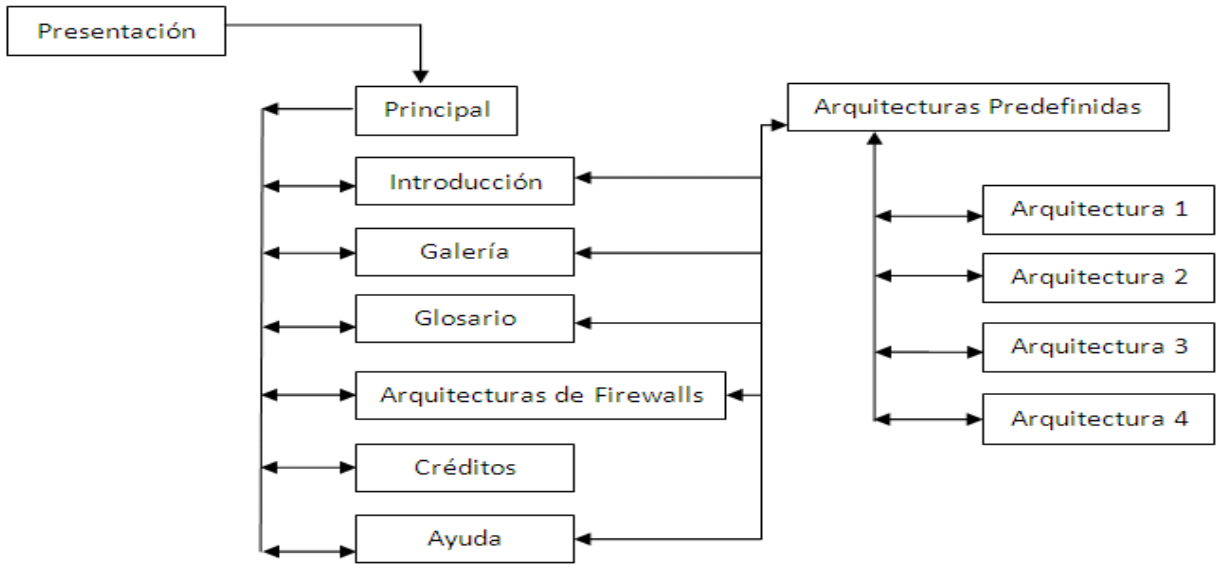


Figura 3: Diagrama de Navegación

3.3 Diagramas de interacción

Los diagramas de interacción muestran el comportamiento de varios objetos en un caso de uso. . Existen los tipos de diagramas de interacción de secuencia y de colaboración se utilizan para modelar los aspectos dinámicos de un módulo de un sistema, mostrando las relaciones del conjunto de objetos que participan, incluyendo los mensajes que intercambian entre ellos.

3.3.1 Diagramas de secuencia

Los diagramas de secuencia resaltan el ordenamiento temporal de los mensajes, presenta un conjunto de objetos y los mensajes enviados y recibidos por ellos. Los objetos suelen ser instancias con nombre o anónimas de clases, pero también pueden representarse instancias de otros elementos.

Para el modelado guiado por el lenguaje de modelado OMMMA-L, se modelan los diagramas de secuencia (ver figuras 4 - 9) pertenecientes a la Vista de Comportamiento, surgen extendido a partir del de UML. Se modela una secuencia de una presentación predefinida dentro de una escena, donde todos los objetos se relacionan al mismo eje del tiempo, haciendo un refinamiento de dicho eje con la introducción de marcas de tiempo a través de diferentes tipos de intervalos; marcas de inicio y fin de ejecución.

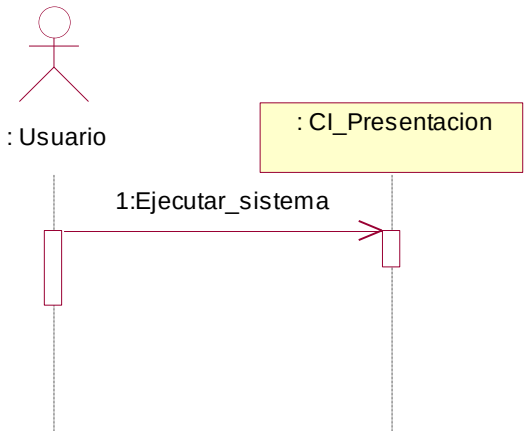


Figura 4: Diagramas de Secuencia Cargar Presentación

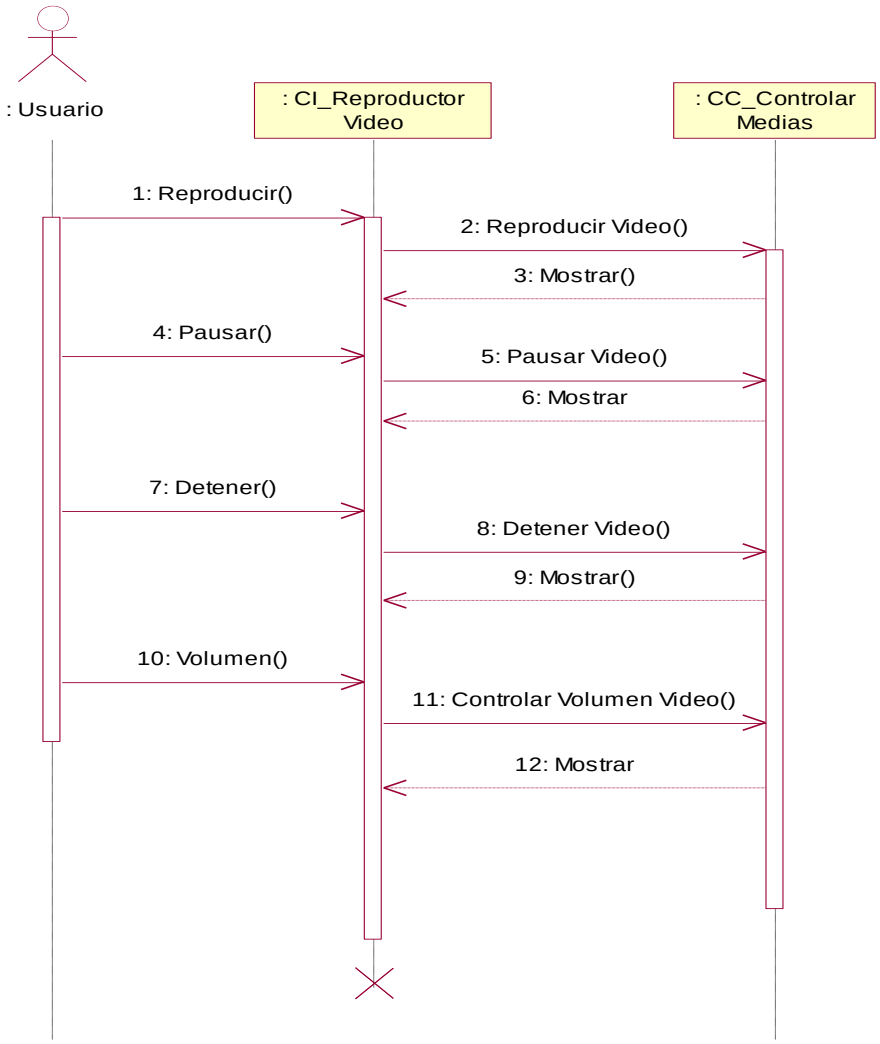


Figura 5: Diagrama de Secuencia Reproducir Video

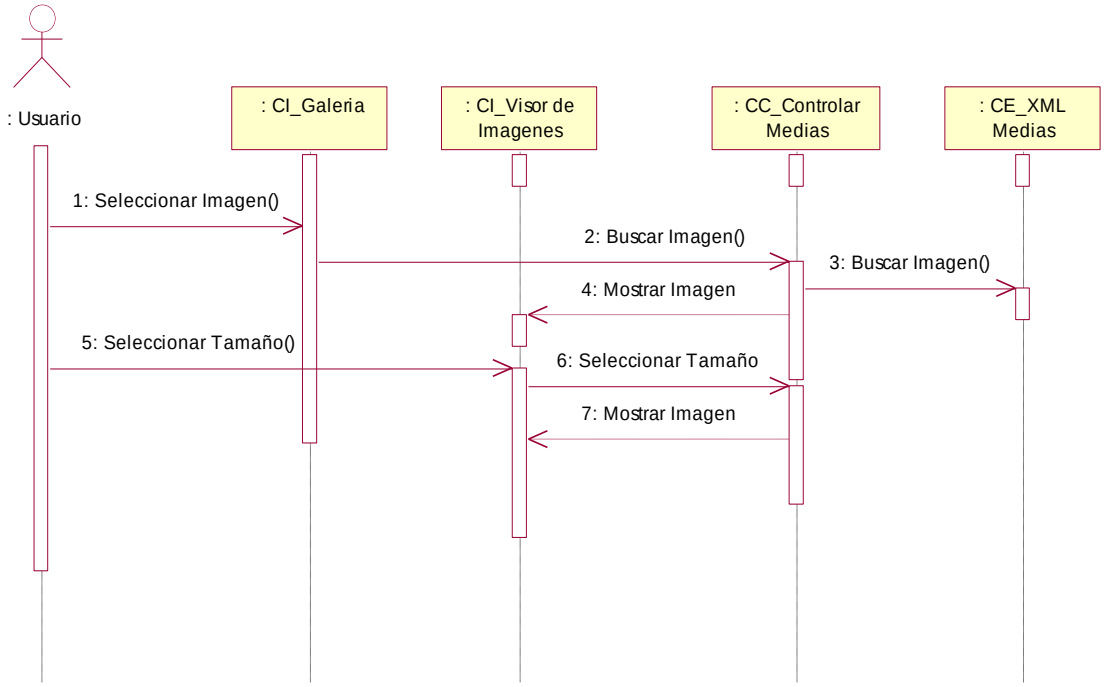


Figura 6: Diagrama Secuencia Imagen

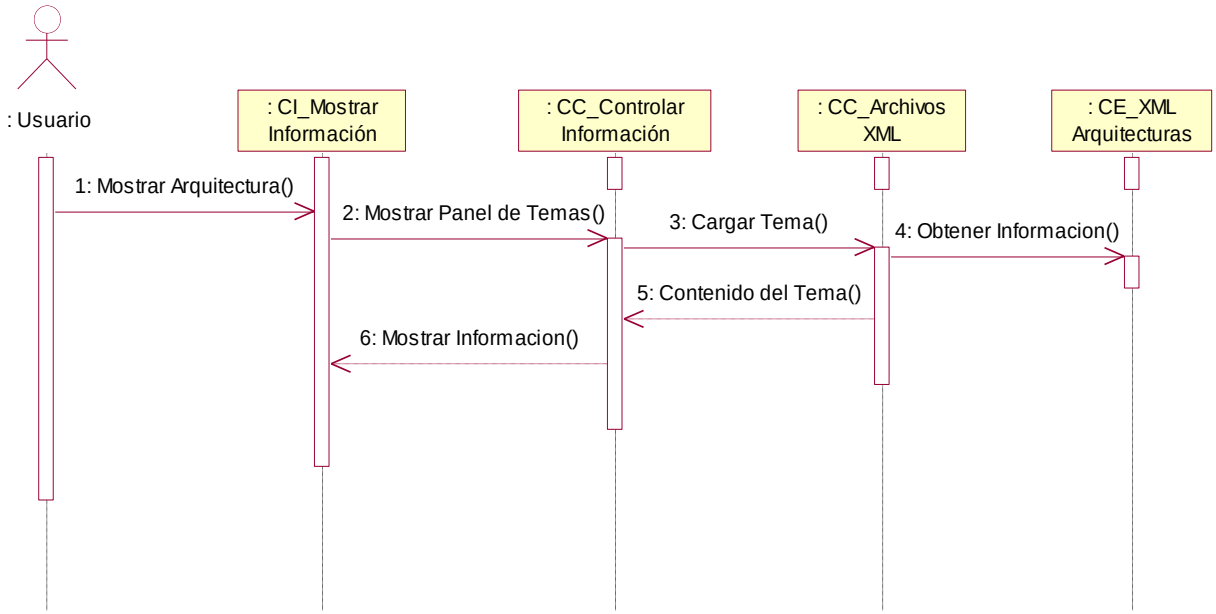


Figura 7: Diagrama Secuencia Seleccionar Arquitecturas

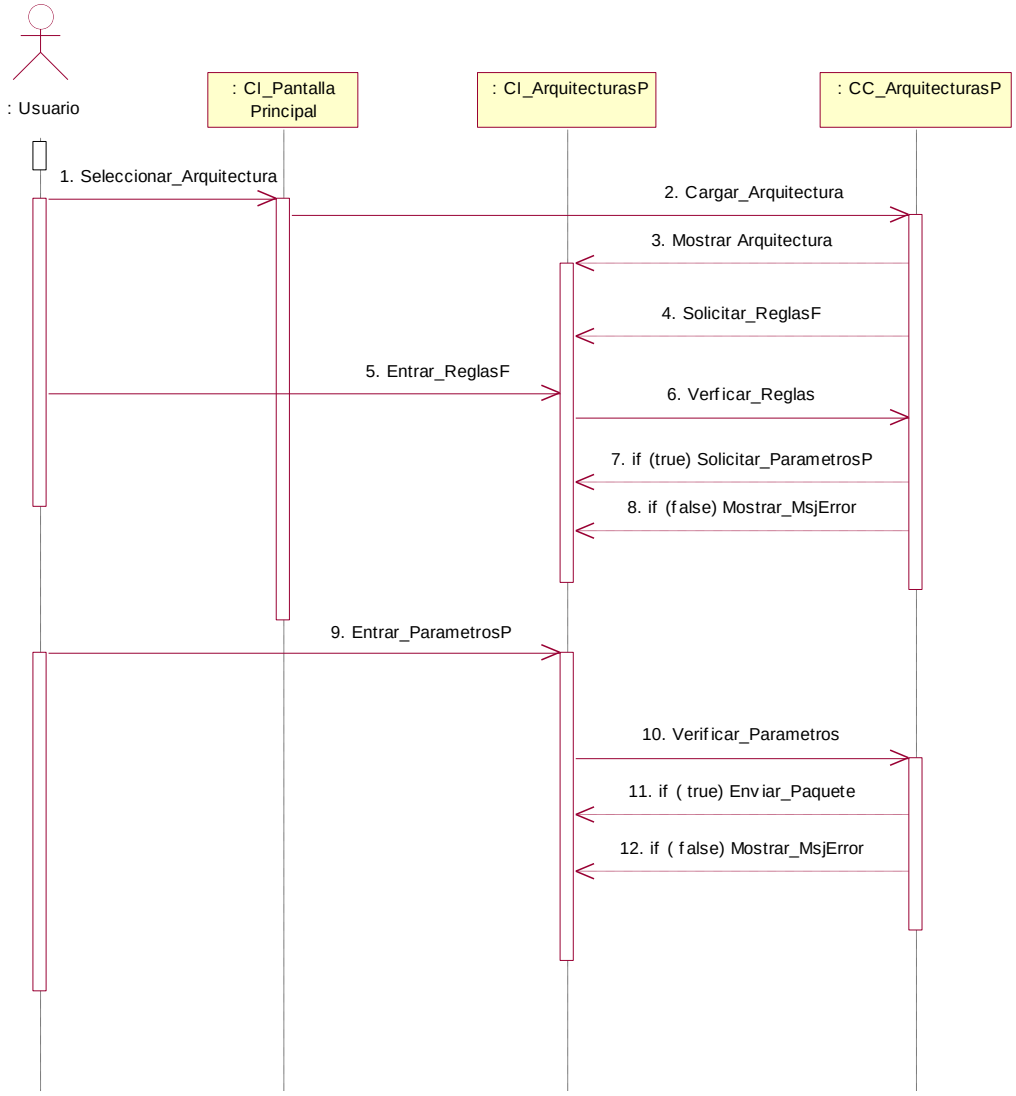


Figura 8: Diagrama Secuencia Enviar Paquete

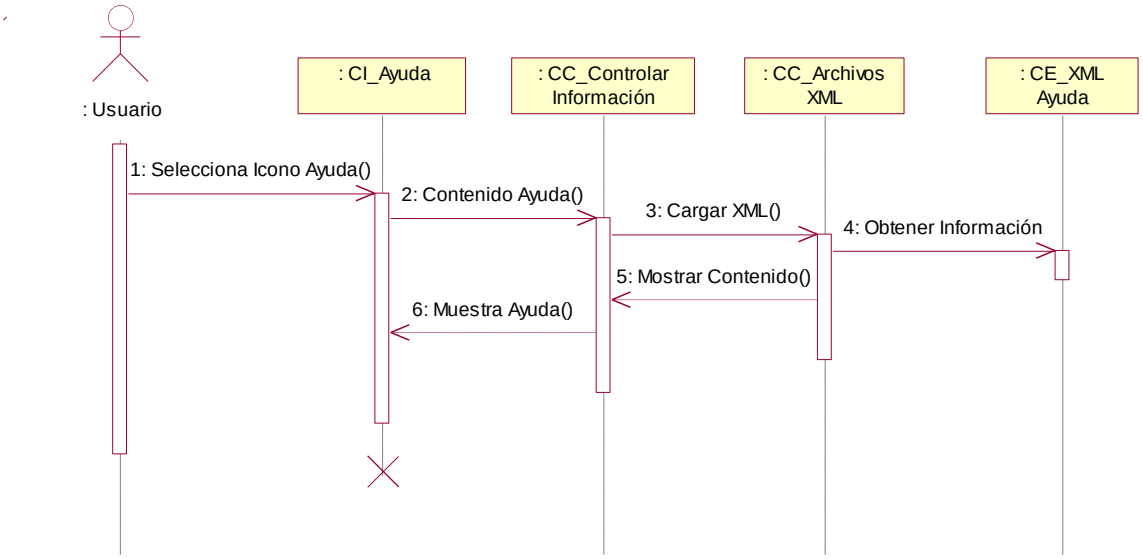


Figura 9: Diagrama Secuencia Ayuda

3.3 Modelo de diseño

En el modelo de diseño es un modelo de objetos que se centra en cómo los requisitos funcionales y no funcionales, junto con otras restricciones del entorno de implementación, tienen impacto en el sistema que se desarrolla. [24]

Este modelo debe ser mantenido durante todo el ciclo de vida del software, sirve de abstracción de la implementación del sistema y es, de ese modo, utilizado como una entrada fundamental de las actividades de este flujo.

3.3.1 Diagramas de presentación

El diagrama de presentación es un artefacto nuevo dentro del lenguaje UML, es específico de OMMMA – L, y sirve, como se explicó en la fundamentación teórica, para describir la parte estática del modelo a través de una descripción intuitiva de la distribución espacial de objetos visuales de la interfaz de usuario. Aunque UML especifica propuesta de interfaz de usuario en sus requisitos no funcionales, no es un aspecto de fuerte medición, ni consideración en el análisis de la arquitectura del software [25]

En estos diagramas se representan las especificaciones de cada pantalla del sistema. Es decir, respondiendo la construcción de la solución propuesta, se simbolizan los elementos estáticos

que componen las pantallas, entre los que aparecen botones, áreas de contenido texto, imágenes y animaciones.

3.3.1.1 Diagrama de presentación del Caso de Uso Cargar Presentación

Figura 10: Pantalla Presentación

3.3.1.2 Diagrama de presentación del Caso de Uso Mostrar Principal

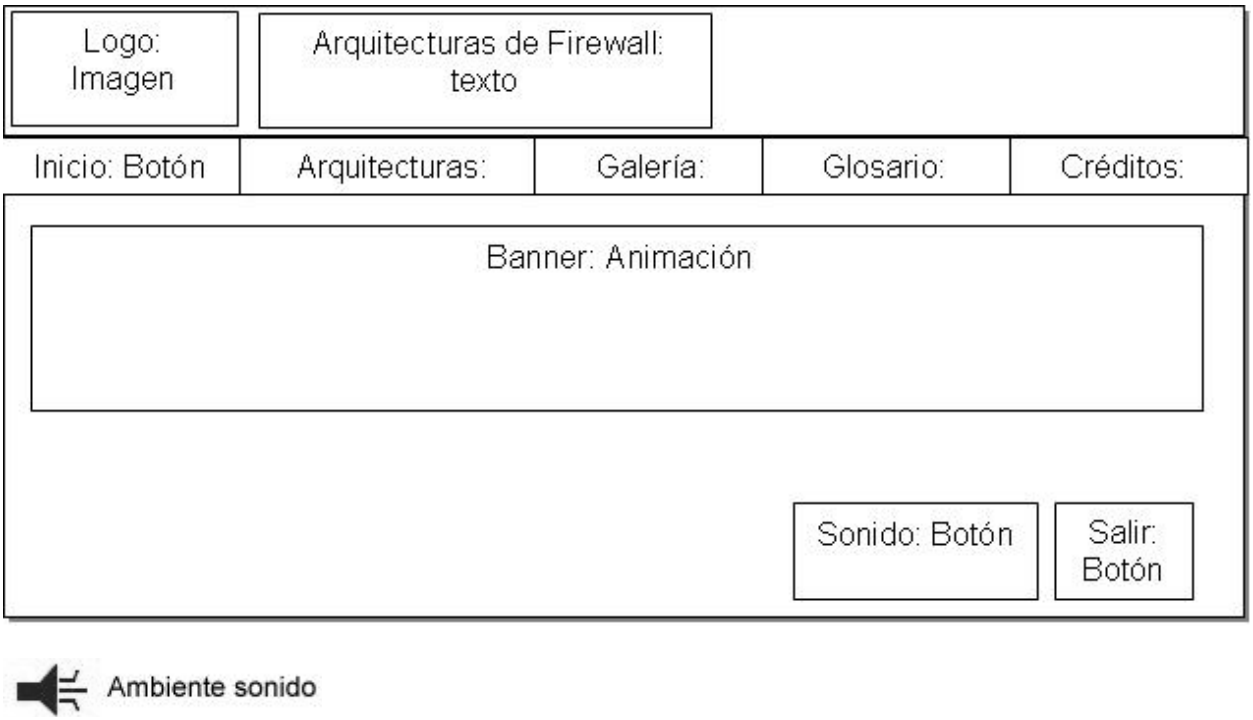


Figura 11: Pantalla Principal

3.3.1.3 Diagrama de presentación del Caso de Uso Galería

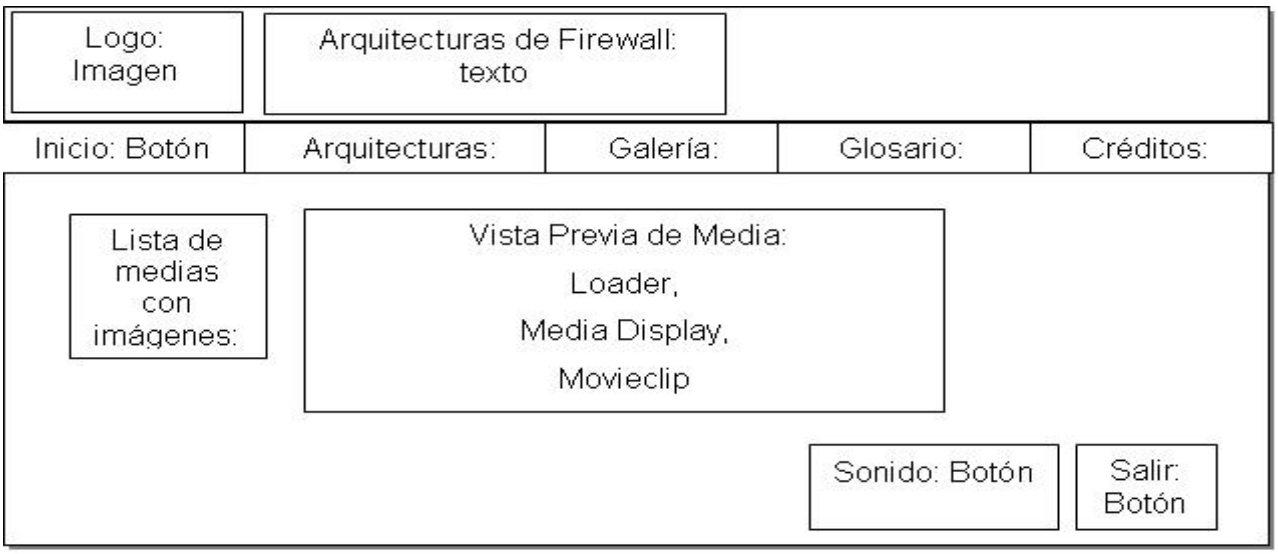


Figura 12: Pantalla Galería

3.3.1.3 Diagrama de presentación para el Caso de Uso Seleccionar Opciones

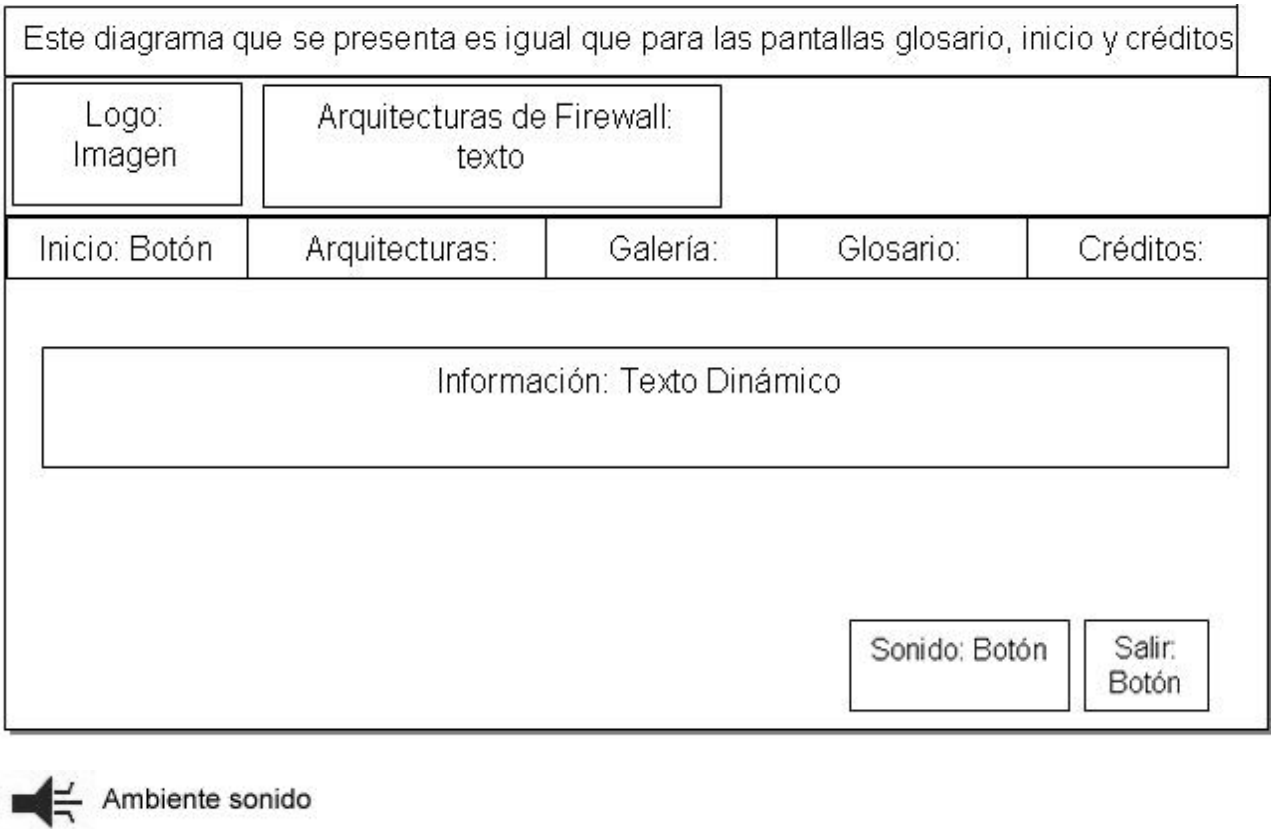


Figura 13: Contenido general

3.3.1.4 Diagrama de presentación para el Caso de Uso Seleccionar_Arquitectura

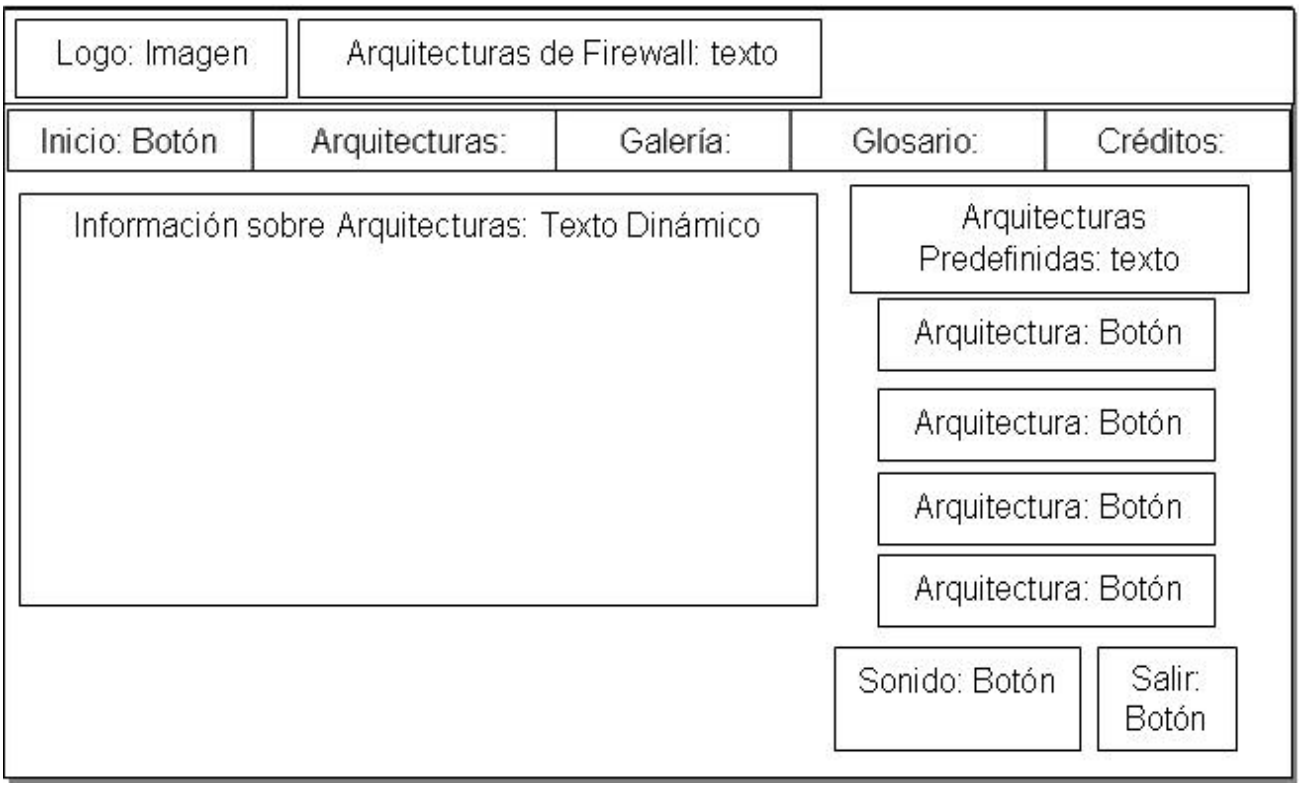


Figura 14: Pantalla Arquitecturas

3.3.1.5 Diagrama de presentación para el Caso de Uso Enviar Paquete

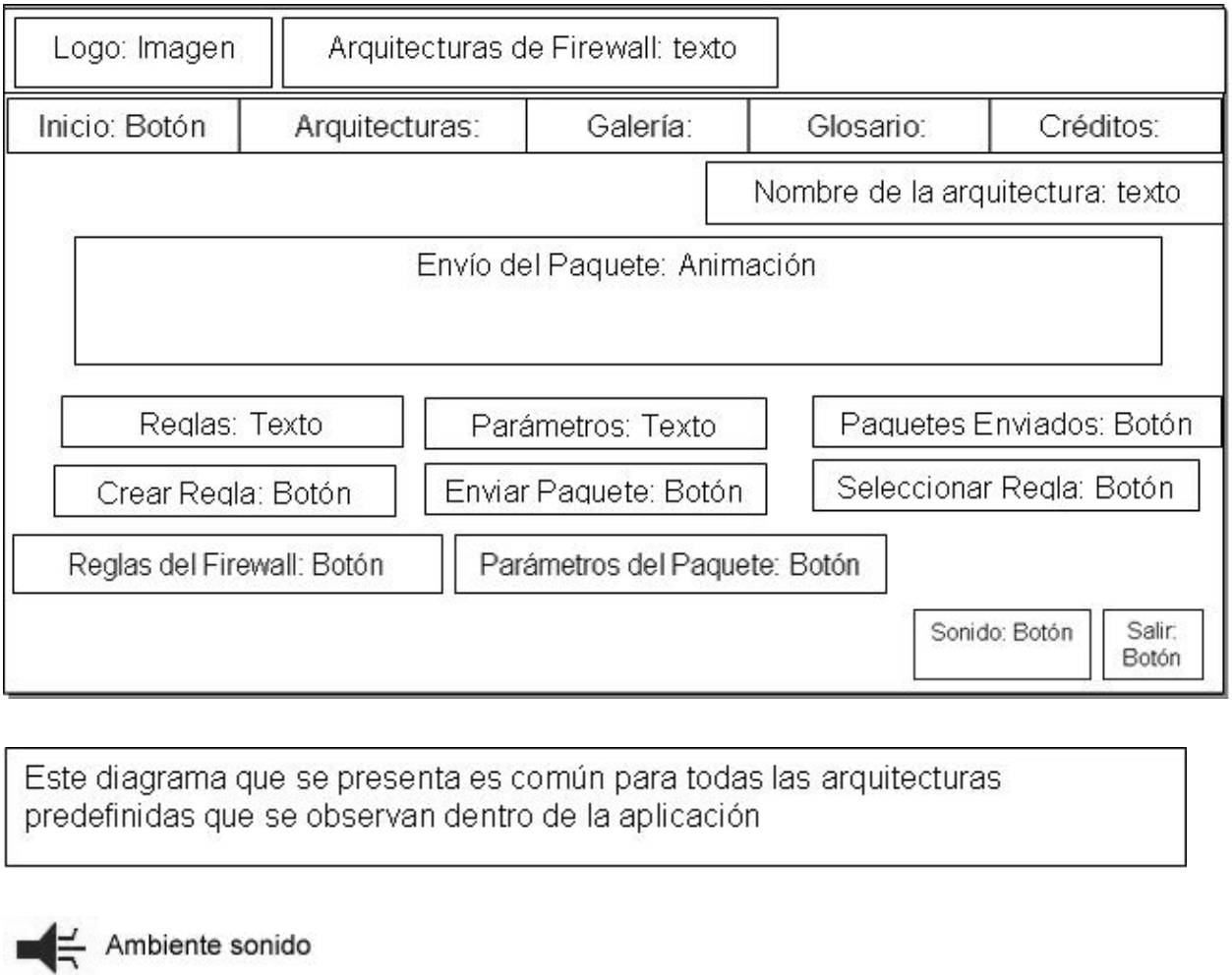


Figura 15: Pantalla de Simulación

3.3.1.6 Diagrama de presentación para el Caso de Uso Salir

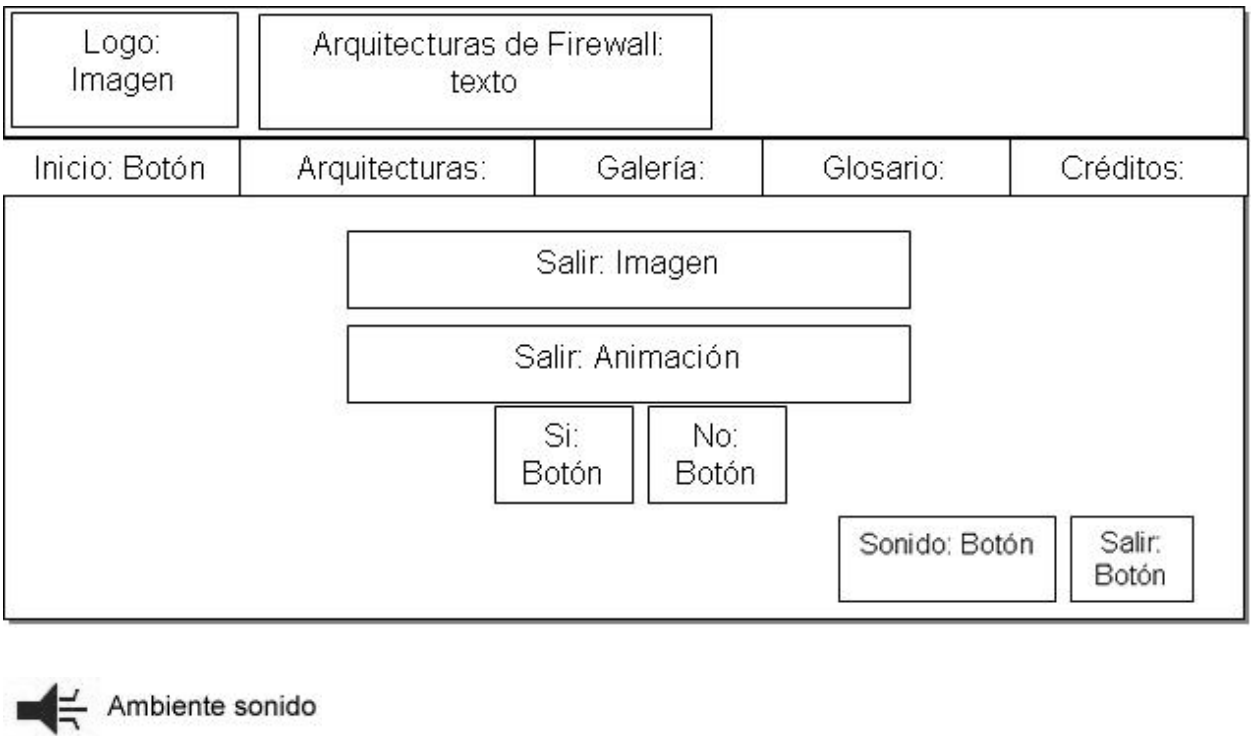


Figura 16: Pantalla Salir

3.4 Modelo de despliegue

El modelo de despliegue se utiliza como entrada fundamental en las actividades de diseño e implementación debido a que la distribución del sistema tiene una influencia principal en su diseño. En este se realiza el diagrama de despliegue, el cual modela la arquitectura en tiempo de ejecución de un sistema, mostrando la configuración de los elementos de hardware (nodos) y cómo los elementos y artefactos del software se relacionan con ellos. El diagrama de despliegue para este software educativo, mostrado en la figura 17, corresponde a un procesador (ordenador o PC) que no necesita estar en red, solo una unidad lectura de CD y debe contar con los dispositivos señalados en los requerimientos no funcionales de hardware.

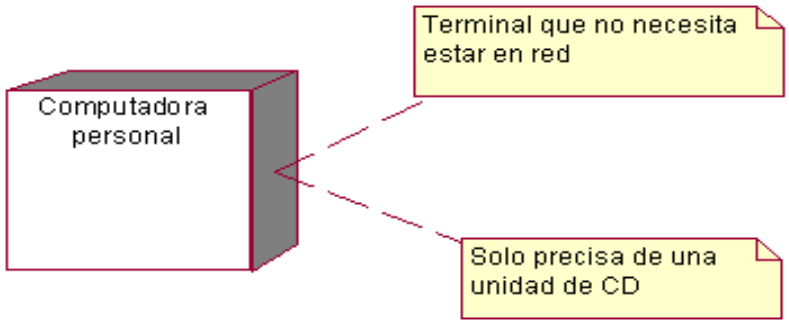


Figura 17: Diagrama de despliegue

3.5 Modelo de Implementación

El modelo de implementación denota la implementación actual del sistema en términos de subsistemas y componentes de implementación (como ficheros de código fuente y ejecutables). Este describe también como se organizan los componentes de acuerdo con los mecanismos de estructuración y modularización disponible en el entorno de implementación y en el lenguaje o lenguajes de programación utilizados, y como dependen los componentes unos de otros.

Durante el flujo de trabajo de implementación se desarrolla todo lo necesario para obtener un sistema ejecutable: componentes ejecutables, componentes ficheros (código fuente, guiones shell, etc.), componentes de tabla (elementos de la base de datos), etc. Un componente es una parte física y reemplazable del sistema que cumple y proporciona la realización de un conjunto de interfaces.

3.5.1 Diagramas de Componentes

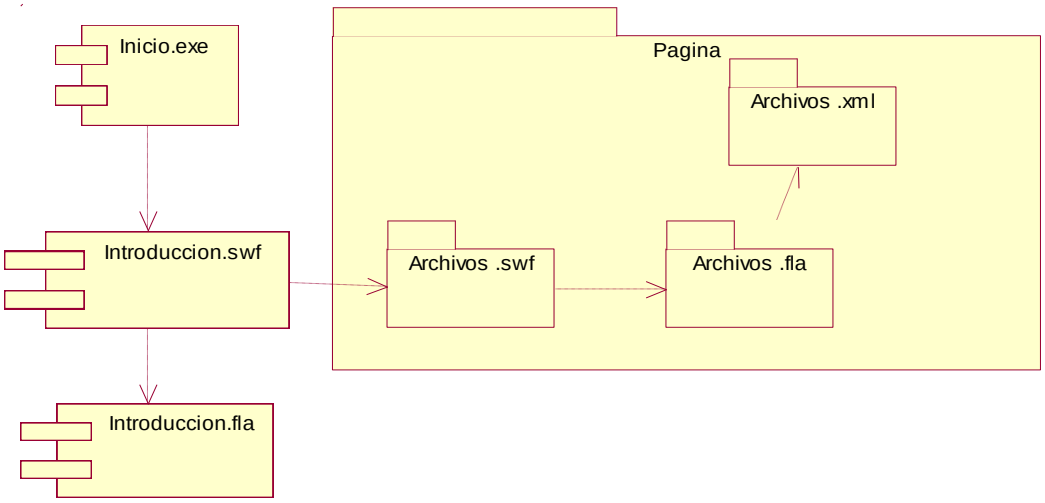


Figura 18: Diagrama de Componentes del Sistema

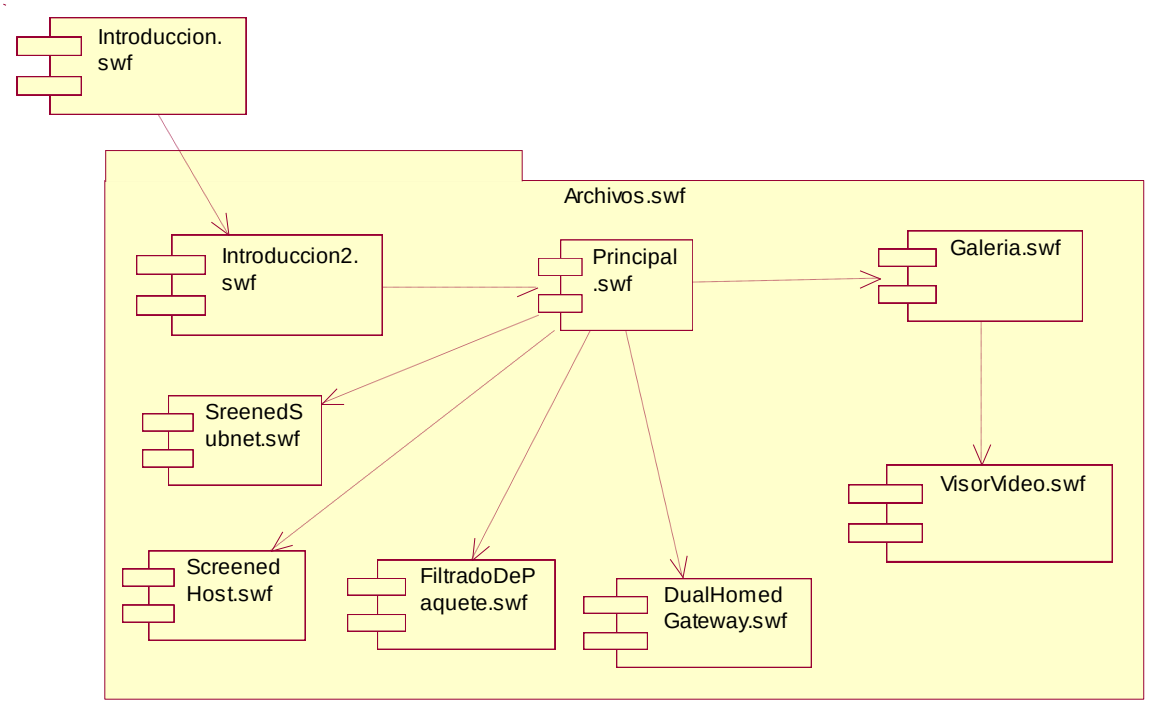


Figura 19: Diagrama de Archivos SWF

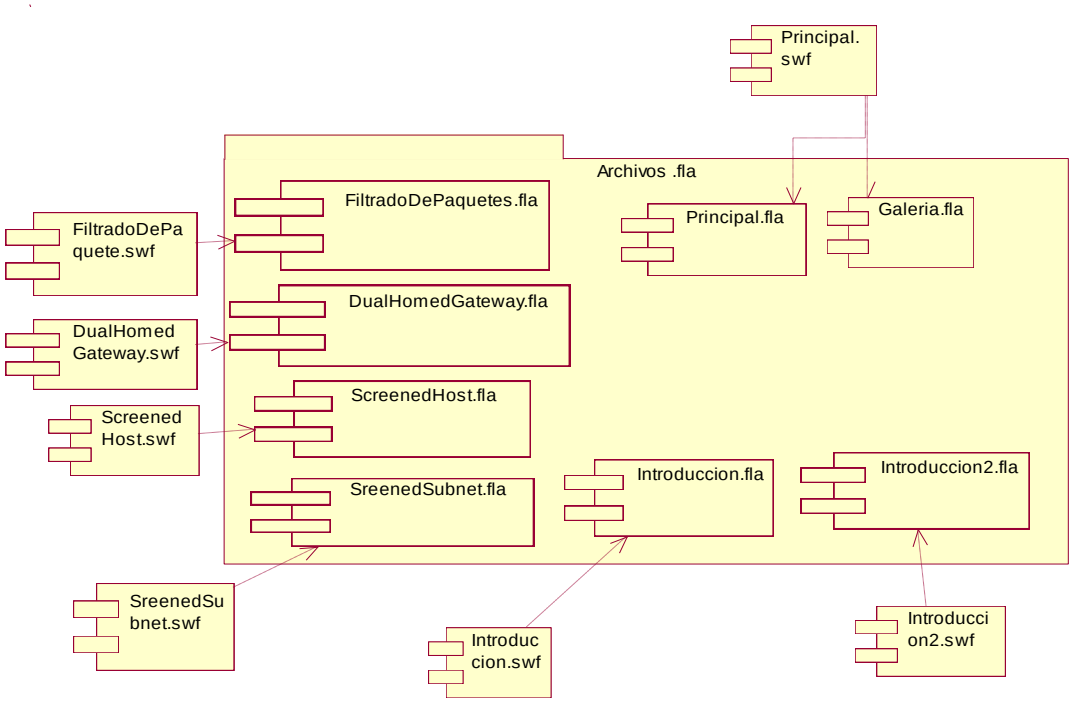


Figura 20: Diagrama de Componentes FLA

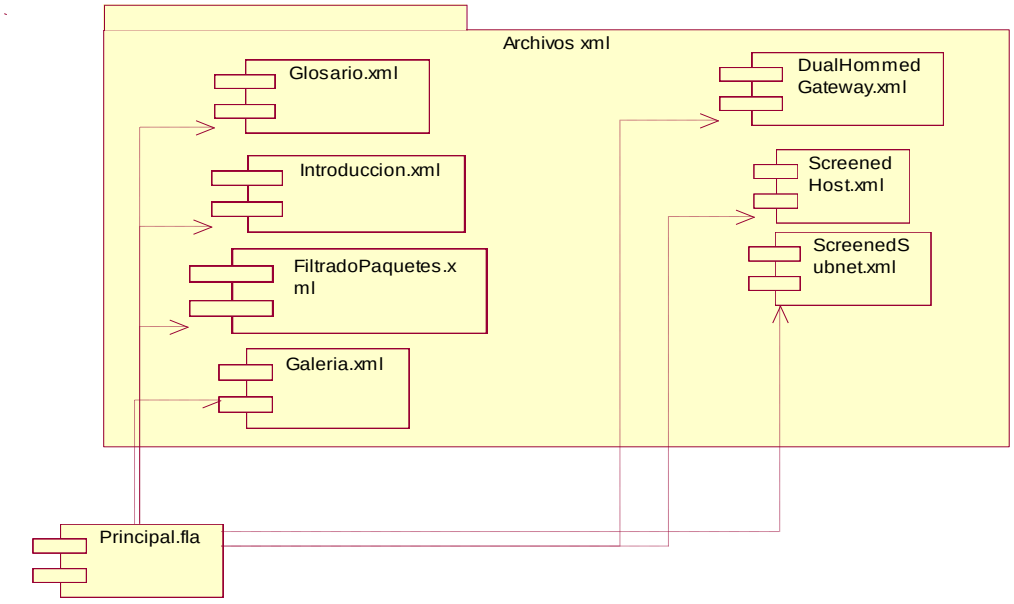


Figura 21: Diagrama de Componentes XML

3.6 Estándares de Interfaz

La interfaz de aplicación es una sola para todo el sistema y se utilizará una resolución de 800 x 600, pantalla completa. Se propone un diseño para una fácil navegación, en la que los colores que prevalezcan sean distintas tonalidades desde el blanco hasta el azul oscuro. Los logotipos de las distintas Arquitecturas de Firewall que se propone presentar. Para representar el texto, la fuente que se propone es: Verdana con estilo fuente normal y tamaño 12 para garantizar una visualización agradable a la vista del usuario, cuyo color tenga un contraste adecuado con el color de fondo(blanco), por lo que es recomendable utilizar el tono más oscuro que se empleará.

3.7 Conclusiones

Al finalizar este capítulo se plantea que se realizó la construcción de la solución propuesta, mostrando los diagramas de presentación, como un artefacto nuevo del lenguaje UML organizados de acuerdo a cada una de las pantallas funcionales. Se incluyen los diagramas de secuencia en los cuales se modela una secuencia de una presentación predefinida dentro de una escena, además del diagrama de despliegue mostrando los requerimientos de hardware necesarios para el funcionamiento de la aplicación y como parte del flujo de trabajo de implementación, se muestran como se organizan los componentes de software, mediante la definición de los diagramas de componentes.

Capítulo 4. Estudio de factibilidad

4.1 Introducción del capítulo

Para la realización de un proyecto es de suma importancia el análisis del costo y los beneficios que reportará, para obtener el tiempo de desarrollo en meses y la cantidad de personas que se necesitan para desarrollarlo, así como realizar una valoración acerca de si se puede llevar a cabo o no. En este capítulo se procede a realizar dicho análisis, con el objetivo de conocer además de las herramientas de desarrollo, la metodología utilizada, el problema que se desea resolver, las ventajas que proporciona el software, etc., otros recursos que se usan, como el tiempo de desarrollo, el recurso humano empleado y los gastos ocasionados. La intención de este capítulo es dejar demostrado la factibilidad del software de acuerdo al costo.

4.2 Estimación de tiempo de desarrollo. Costos y esfuerzo

Para el desarrollo de este capítulo centramos el trabajo en el uso de una de las técnicas más difundidas recientemente denominada Análisis de los Puntos de Caso de Uso, que permite estimar el tiempo de desarrollo, costo y esfuerzo en el desarrollo de un software, usando como punto de referencia los Casos de Uso que se proponen. El Análisis por Puntos de Caso de Uso es un método propuesto por Gustav Karner, pero que ha estado expuesto a diversas modificaciones por otros autores desde su surgimiento. Consiste en asignarle un valor denominado “peso” a diversos factores por los que se verán afectados y de esta manera poder determinar el tiempo de desarrollo estimado para el software.

Para dar inicio al cálculo del costo se procede a realizar los pasos según lo planteado en el método Análisis de los Puntos de Caso de Uso:

Cálculo de Puntos de Casos de Uso sin ajustar

El primer paso para la estimación consiste en el cálculo de los Puntos de Casos de Uso sin ajustar. Este valor, se calcula a partir de la siguiente ecuación:

UUCP = UAW + UUCW

UUCP: Puntos de Casos de Uso sin ajustar

UAW: Factor de Peso de los Actores sin ajustar

UUCW: Factor de Peso de los Casos de Uso sin ajustar

Factor de Peso de los Actores sin ajustar (UAW)

Este valor se calcula mediante un análisis de la cantidad de Actores presentes en el sistema y la complejidad de cada uno de ellos. La complejidad de los Actores se establece teniendo en cuenta en primer lugar si se trata de una persona o de otro sistema, y en segundo lugar, la forma en la que el actor interactúa con el sistema. Los criterios se muestran en la siguiente tabla:

Tabla 8. Factor de Peso de Actores sin Ajustar

Tipo de Actor	Descripción	Factor de Peso
Simple	Otro sistema que interactúa con el sistema a desarrollar mediante una interfaz de programación (API, Application Programming Interface).	1
Medio	Otro sistema que interactúa con el que se va a desarrollar mediante un protocolo o interfaz basada en texto.	2
Complejo	Una persona que interactúa con el sistema mediante una interfaz gráfica.	3

De acuerdo a los criterios expuestos en la tabla anterior, se selecciona la opción 3, donde el tipo de actor es complejo y su factor de peso es 3. Se multiplica por 1 debido a que no habrá concurrencia al interactuar con el sistema, por tanto el factor de peso de los actores sin ajustar se calcula:

UAW = 1 x 3

UAW = 3

Factor de Peso de los Casos de Uso sin ajustar (UUCW)

Este valor se calcula mediante un análisis de la cantidad de Casos de Uso presentes en el sistema y la complejidad de cada uno de ellos. La complejidad de los Casos de Uso se establece teniendo en cuenta la cantidad de transacciones efectuadas en el mismo, donde una transacción se entiende como una secuencia de actividades atómica, es decir, se efectúa la secuencia de actividades completa, o no se efectúa ninguna de las actividades de la secuencia. Los criterios se muestran en la siguiente tabla:

Tabla 9. Factor de Peso de CU sin Ajustar

Tipo de Caso de Uso	Descripción	Factor de Peso
Simple	El Caso de Uso contiene de 1 a 3 transacciones.	5
Medio	El Caso de Uso contiene de 4 a 7 transacciones.	10
Complejo	El Caso de Uso contiene más de 8 transacciones.	15

Este coeficiente se calcula mediante la cuantificación de un conjunto de factores que determinan la complejidad técnica del sistema. Cada uno de los factores se cuantifica con un valor de 0 a 5, donde 0 significa un aporte irrelevante y 5 un aporte muy importante. En la siguiente tabla se muestra el significado y el peso de cada uno de éstos factores:

Tabla 10. Factor de complejidad técnica

Factor	Descripción	Peso	Valor asignado
T1	Sistema distribuido.	2	0
T2	Objetivos de performance o tiempo de respuesta.	1	2
T3	Eficiencia del usuario final.	1	1
T4	Procesamiento interno complejo.	1	1
T5	El código debe ser reutilizable.	1	5
T6	Facilidad de instalación.	0.5	1
T7	Facilidad de uso.	0.5	4
T8	Portabilidad.	2	4
T9	Facilidad de cambio.	1	4
T10	Concurrencia.	1	0
T11	Incluye objetivos especiales de seguridad.	1	0
T12	Provee acceso directo a terceras partes.	1	0
T13	Se requieren facilidades especiales de entrenamiento a usuarios.	1	0

Factor de ambiente (EF)

Las habilidades y el entrenamiento del grupo involucrado en el desarrollo tienen un gran impacto en las estimaciones de tiempo. Estos factores son los que se contemplan en el cálculo del Factor de ambiente. El cálculo del mismo es similar al cálculo del Factor de complejidad técnica, es decir, se trata de un conjunto de factores que se cuantifican con valores de 0 a 5. En la siguiente tabla se muestra el significado y el peso de cada uno de éstos factores.

Tabla 11. Factor de ambiente

Factor	Descripción	Peso	Valor asignado
E1	Familiaridad con el modelo de proyecto utilizado.	1.5	4
E2	Experiencia en la aplicación.	0.5	2
E2	Experiencia en orientación a objetos.	1	2
E4	Capacidad del analista líder.	0.5	5
E5	Motivación.	1	5
E6	Estabilidad de los requerimientos.	2	1
E7	Personal part-time.	-1	3
E8	Dificultad del lenguaje de programación.	-1	1

De los Puntos de Casos de Uso a la estimación del esfuerzo

- Se contabilizan cuántos factores de los que afectan al Factor de ambiente están por debajo del valor medio (3), para los factores E1 a E6.

- Se contabilizan cuántos factores de los que afectan al Factor de ambiente están por encima del valor medio (3), para los factores E7 y E8.

- Si el total es 2 o menos, se utiliza el factor de conversión 20 horas-hombre/Punto de Casos de Uso, es decir, un Punto de Caso de Uso toma 20 horas-hombre.

- Si el total es 3 o 4, se utiliza el factor de conversión 28 horas-hombre/Punto de Casos de Uso, es decir, un Punto de Caso de Uso toma 28 horas-hombre.

- Si el total es mayor o igual que 5, se recomienda efectuar cambios en el proyecto, ya que se considera que el riesgo de fracaso del mismo es demasiado alto.

Luego, para calcular el Esfuerzo (E) en horas-hombre, se tiene la siguiente ecuación:

E = UCP x CF

Donde

E: esfuerzo estimado en horas-hombre

UCP: Puntos de Casos de Uso ajustados

CF: factor de conversión

De acuerdo a los criterios planteados anteriormente se define que el factor de conversión que se va a emplear es 28, por tanto al calcular **E** se tiene:

E = 44.8 x 28

E = 1254.4 horas-hombres

De acuerdo al valor porcentual que representa este valor del esfuerzo con respecto al esfuerzo total del proyecto, se realiza la siguiente tabla:

Tabla 12. Estimación del Esfuerzo

Actividad	**Porcentaje**	**Porciento**
Análisis	10%	125.44
Diseño	20%	250.88
Programación	40%	1254.4
Prueba	15%	188.16
Sobre carga (otras actividades)	15%	188.16
Total	100%	2007.04

TDEV = E / CH

Donde

TDEV: tiempo de desarrollo

E: valor del esfuerzo (calculado anteriormente)

CH: cantidad de hombre que van trabajar en el proyecto

Por tanto al sustituir en por los valores en la formula de TDEV se tiene:

TDEV = 13.938 hombres-mes / 1 hombre

TDEV = 13.938 meses

Teniendo ya todos los elementos necesarios para calcular el costo del proyecto, se efectúa de la manera siguiente:

Costo = CH x SM x TDEV

Donde

SM: salario mínimo del programador

CH y **TDEV**, han sido definidos anteriormente

Por tanto al sustituir los valores de **SM**, **CH** y **TDEV** se tiene:

Costo = 100 pesos x 1 hombres x 13.938 meses

Costo = $ 1393.8

Por tanto el software se valora para un costo de producción de $1393.8

4.3 Beneficios tangibles

Como beneficios tangibles luego de cumplido los objetivos propuestos, se obtiene como resultado la documentación requerida para la implementación del software. Además se cuenta con la ventaja de la reusabilidad del código al realizar cada nivel semejante a los anteriores, creando así una guía para el futuro desarrollo de aplicaciones similares en los cuales se podría aumentar el nivel de interactividad entre el usuario y el software.

4.4 Beneficios intangibles

El análisis y diseño realizado y la documentación obtenida facilitan el trabajo, reduciendo el tiempo en el proceso de implementación y se logra una buena calidad del producto final. El usuario podrá aumentar su conocimiento acerca de las distintas arquitecturas de Firewalls. La cual contribuirá a un aumento en el estudio y desarrollo de estas aplicaciones.

4.5 Análisis de los costos y beneficios

Se da solución al problema de la investigación: se cuenta con una multimedia educativa, que se utiliza de apoyo al proceso de aprendizaje sobre el funcionamiento de los Firewall. La universidad cuenta con un software educativo que facilita el aumento de los conocimientos y la cultura respecto a la seguridad. La documentación que se realiza no es extensa y si de fácil manejo, por lo que será factible utilizarla como guía y material de estudio, y no ocasionará gastos por conceptos de adiestramientos. De acuerdo a lo expuesto anteriormente se considera que el desarrollo del software es factible y que el esfuerzo de desarrollo empleado está justificado.

4.5 Conclusiones

Con el desarrollo de este capítulo, se ha hecho un análisis de la factibilidad y de los beneficios que genera el proyecto sobre arquitecturas de Firewall mediante la técnica del Análisis de los Puntos de Caso de Uso, que permite estimar el tiempo de desarrollo, costo y esfuerzo en el desarrollo de un software. Finalmente se concluye que el desarrollo del proyecto es factible debido a los beneficios económicos-sociales que proporciona

Conclusiones

Como resultado de haber cumplido el propósito del presente trabajo, luego de confeccionar un documento de apoyo a la comprensión de la situación problémica existente y su solución, a partir de la propuesta de diferentes artefactos que permitieron la correcta implementación de un software educativo sobre temas relacionados con las arquitecturas de Firewall.

- Se realizó un estudio sobre las tendencias, tecnologías y metodologías más usadas en la actualidad para la construcción de sistemas informáticos similares y se seleccionó como la metodología idónea para cumplimentar los procesos de análisis y diseño, al Proceso Unificado de Desarrollo (RUP), conjuntamente con OMMMA – L, que surge como extensión de UML para la integración de especificaciones de sistemas con tecnología multimedia.
- Se realizó la identificación de las funcionalidades que deberá poseer el producto, definiendo 10 requisitos funcionales y 4 no funcionales.
- Se construyó el modelo de dominio, lográndose una correcta comprensión del problema, así mismo para un trabajo más organizado y un mayor entendimiento de las características funcionales y visuales del software, incluyendo además diagramas de secuencia, de componentes y de presentación como artefactos nuevos de OMMMA-L. Se realizó la descripción de su contexto productivo a partir de la identificación de 7 casos de uso, referenciadas en sus descripciones por los requerimientos funcionales definidos.
- Se modelaron los artefactos requeridos para guiar el proceso de implementación, exponiendo en cada caso el comportamiento y la interacción de los elementos que conforman cada entorno del software.
- Se comprobó la factibilidad del software a partir de la realización del estudio de factibilidad.
- Se expresó claramente la ventaja que implica la implementación de esta aplicación, ya que la misma permite ahorrar recursos humanos.

De esta forma se puede afirmar que se alcanzó, satisfactoriamente el objetivo propuesto, creándose la modelación e implementación de un software educativo con tecnología multimedia.

Recomendaciones

Para finalizar este trabajo se sugieren algunas ideas:

- ✓ Continuar el estudio de las herramientas de autor con el objetivo de encontrar nuevas funcionalidades para refinar e implementar una herramienta más completa y general.
- ✓ Profundizar en el uso de las alternativas de carga dinámica dado que elevan la eficiencia y rapidez de ejecución de los productos multimedia.
- ✓ Continuar en el desarrollo de aplicaciones similares para cultivar un mayor conocimiento de estas aplicaciones.

Referencias bibliográficas

[1] Condori, C. *Impacto de las nuevas tecnologías en educación.* 2004.

[Disponible en:] http://www.ilhn.com/periodismo/archives/cat_trabajo_final.php.

[2] Marqués, P. *El software Educativo.* Universidad Autónoma de Barcelona, 1995.

[Disponible en:]

http://www.karisma.org.co/documentos/softwareredp/clasif_software_educativo_de_pere.doc

[3] Díaz, C. C. *La tecnología Multimedia: una nueva tecnología de comunicación e información. Características, concepciones y aplicaciones*, 1994. México.

[Disponible en:] http://iteso.mx/~carlosc/pagina/documentos/multidef.htm#guia

[4] Kimera. *La multimedia como facilitador en el proceso educativo.* 1999.

[Disponible en:] http://www.kimera.com/articulos/multimedia.html

[5] Piattini. *Análisis y Diseño Detallado de Aplicaciones Informáticas de Gestión*. 1996. Rama. Madrid.

[6] Bork A. *El ordenador en la enseñanza. Análisis y perspectivas de futuro.* 1986.

[7] Calero, S, M. *Una explicación de la programación extrema (XP).* 2003.

[Disponible en:] http://www.apolosoftware.com/

[8] Pérez, Y, H. *Metodologías para el desarrollo de software educativo: Un estudio comparativo.* 2006.

[9] Jiménez, S. V. *Propuesta de una estructura organizacional del proceso de producción de un software multimedia a través de RUP y una extensión de UML para hipermedia.* 2006. Universidad de las Ciencias Informáticas, Ciudad Habana.

[10] Isakowitz, T, Stohr E, A. *RMM: a methodology of structured hypermedia design.*1995. New York University

[Disponible en:] http://jmis.bentley.edu/rmm/papers/rmd.pdf

[11] Isakowitz, T, S, Edgard, A, Balasubramanian, P. *RMM: Metodología para el Diseño*

Estructurado de Hipermedios.

[12] Mendoza, M, S. *Metodologías de desarrollo de software.* 2004. Perú.

[Disponible en:] http://www.informatizate.net-MetodologíasDeDesarrolloDeSoftware.htm

[13] Jacobson, I. Booch, G y Rumaugh, J. *El proceso unificado de desarrollo de software vol. 1.* Cap. 9. Editorial Félix Varela. 2004. Ciudad de la Habana.

[14] Colectivo de autores., *Introducción a la Ingeniería de Software. El Proceso Unificado de Desarrollo (RUP)*, Dpto. de Ingeniería y Gestión de Software 2007 Universidad de las Ciencias Informáticas, Ciudad Habana.

[15] Booch, G; Jacobson, I; y Rumbaugh, J. *El Lenguaje Unificado de Modelado. Manual de Referencia.* 1ra edición. Madrid: Addison-Wesley. 2000

[16] Lorente A, E; Martínez, Y y Díaz, A. *Plantilla para el montaje dinámico de los productos de la colección multisaber.* 2006. Ciudad Habana.

[17] Souchay, D, F*. Multimedia aprendarte, centralización de información.* 2007*.*

[Disponible en:] http://www.ilustrados.com/publicaciones/EEAkulVyEVyKTwmnEi.php

[18] Ciudad, F, R. *ApEM - L como una nueva solución a la modelación de aplicaciones educativas multimedia en la UCI*. 2007. Universidad de las Ciencias Informáticas, Ciudad Habana.

[19] Bermúdez, H, E. *Multimedia Historia Universal Volumen I*. 2007. Universidad de las Ciencias Informáticas, Ciudad Habana.

[20] Avalos, F, P. *Multimedia Educativa para los niños de la Enseñanza Primaria con disgrafía escolar.* 2007. Universidad de las Ciencias Informáticas, Ciudad Habana.

[21] REGAÑA, C. B. and L. M. T. BARZABAL. *Atención a la diversidad y multimedia: el diseño de materiales curriculares un reto al alcance de todo.* 2004. [Disponible en:]

http://www.monografias.com/trabajos903/atencion-diversidad-multimedia/atencion-diversidad-multimedia2.shtml

[22] Baeza, M*. Interoperatividad y XML* 2007. [Disponible en:]

http://www.gestionpublica.cl/biblioteca/documentos/67112INTEROPERATIVIDADYXML.pdf.

[23] Frascara, J. *Definiendo audiencia.*

[Disponible en:] http://www.wolkoweb.com.ar/apuntes/textos/definiendo_audiencia.rtf.

[24] Arias, M. *La ingeniería de requerimientos y su importancia en el desarrollo de proyectos de software.* 2005. Universidad de las Ciencias Informáticas, Ciudad Habana.

[25] OBREGÓN, R. F. and Y. R. TAMAYO. *Multimedia Educativa Constitución Bolivariana Volumen II* Universidad de las Ciencias Informáticas. 2004.

Bibliografía

1. Albalat, V. H; Peris, A; Esparza, A. *Macromedia Flash*. 1993. [Disponible en:]

http://www.lsi.uji.es/asignatura/obtener.php?letra=5&codigo=44&fichero=1115202606544

2. ANÓNIMO. *Contenidos Educativos en Tecnología de Información y Comunicación - CETIC*, 2005. [Disponible en:] http://www.cetic.cnti.gob.ve/software2005.htm

3. Díaz Catalá, S, Solenzal Fernández, G. *Multimedia Auto-Aprende*. 2006. Universidad de las Ciencias Informáticas, Ciudad Habana.

4. Enríquez, B. A. M. *El desarrollo de sistemas de información empleando el lenguaje de modelado unificado UML*, 2005.

[Disponible en:] http://www.monografias.com/trabajos16/lenguaje-modelado-unificado/lenguaje-modelado-unificado.shtml

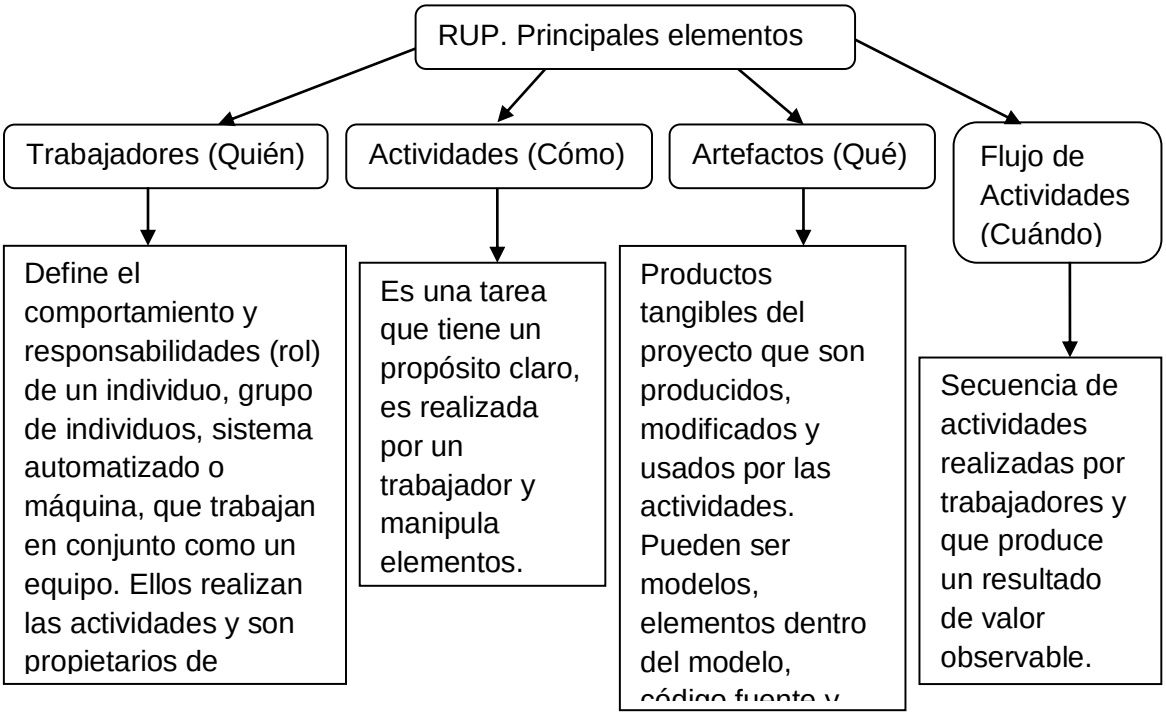
5. GRAELLS, D. P. M. *Multimedia educativo: clasificación, funciones, ventajas, e inconvenientes.* 2004. [Disponible en:] http://dewey.uab.es/pmarques/funcion.htm

6. Hernández, R. L. y Coello, S, G. *El paradigma cuantitativo de la investigación científica.* 2002. Universidad de las Ciencias Informáticas, Ciudad Habana.

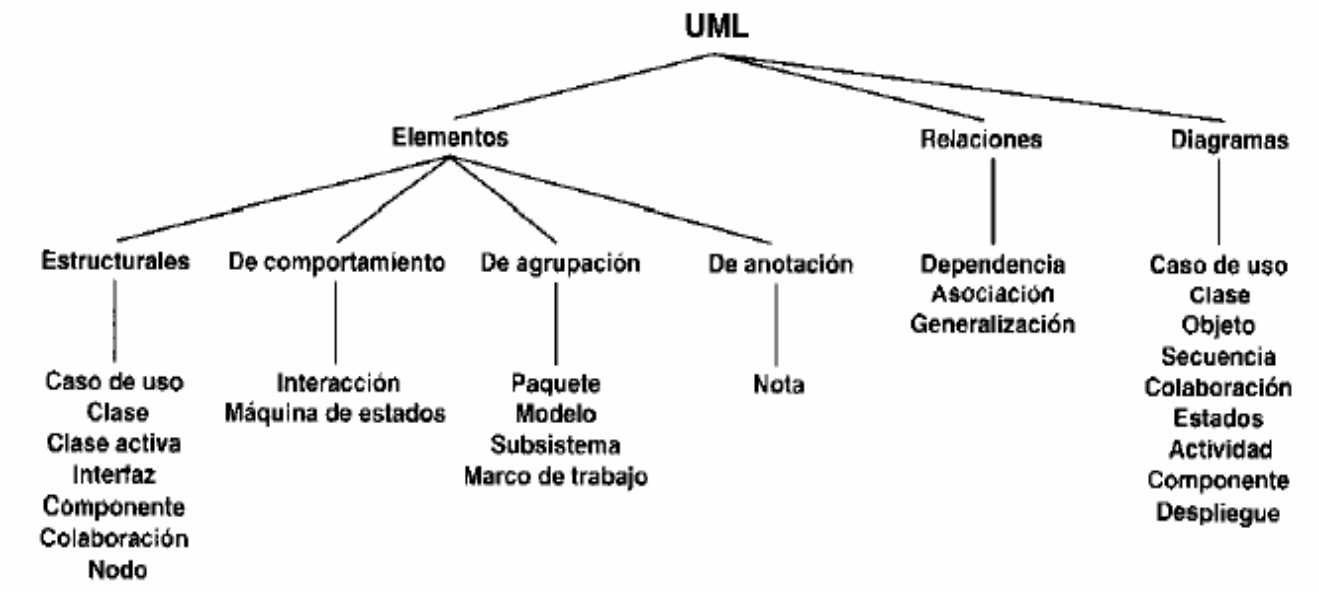
7. Pascual, J. *Herramientas profesionales para la creación de aplicaciones multimedia*, 1998. Barcelona, [Disponible en:]

http://209.85.215.104/search?q=cache:AEm7WXJXfs4J:roble.pntic.mec.es/~sblanco1/pagina_n.htm+desventajas+de+ToolBook&hl=es&ct=clnk&cd=13&gl=cu

Anexos



Anexo 1. Principales elementos del RUP



Anexo 2. Vocabulario de UML

Glosario de términos

Animación: Representación sucesiva de una secuencia de imágenes que produce la sensación de estar viendo imágenes en movimiento. Para ello, a cada imagen de una animación se le modifica un pequeño detalle para mantener el movimiento tan fluido como sea posible.

Artefacto: Es una pieza tangible que crean, modifican y usan las personas encargadas de realizar determinadas actividades durante el proceso de desarrollo del software.

Capítulo: Se le llama capítulo a la primera división que tiene el contenido. El contenido está agrupado en capítulos y estos a su vez en epígrafes.

CD-ROM: Es un disco compacto pregrabado, del que se puede leer información pero no borrarla ni modificarla.

Fases: Período de tiempo entre dos hitos principales de un proceso de desarrollo

Flujo de trabajo: Realización de un caso de negocio o parte de él. Puede describirse en términos de diagramas de actividad, que incluyen a los trabajadores participantes, las actividades que realizan y los artefactos que producen.

Herramientas de autor: Las herramientas de autor son aplicaciones que permiten un trabajo multimedia y constructivista para generar un entorno de aprendizaje dinámico, dentro de las funcionalidades que este tipo de herramientas presentan se puede destacar la posibilidad de crear actividades o pequeñas aplicaciones desde la misma herramienta.

Hipertexto: Un hipertexto es un documento digital o no, que se puede leer de manera no secuencial. Un hipertexto tiene los siguientes elementos: secciones, enlaces o hipervínculos y anclajes.

Hipermedia: En contextos específicos, se identifica hipermedia como extensión del término Hipertexto, en el cual audio, video, texto e hipervínculos generalmente no secuenciales, se entrelazan para formar un continuo de información, que puede considerarse como virtualmente infinito desde la perspectiva de Internet.

Internet: Sistema de redes de computación ligadas entre sí, con alcance mundial, que facilita servicios de comunicación de datos como registro remoto, transferencia de archivos, correo electrónico y grupos de noticias. Internet es una forma de conectar las redes de computación

existentes que amplía en gran medida el alcance de cada sistema participante.

Medias**:** Es el término que representa información multimedia como texto, imagen, sonido, animación, video.

Proceso de Desarrollo del software: Conjunto de actividades necesarias para transformar los requisitos en un conjunto consistente de artefactos que representan un producto software y para transformar cambios en dichos requisitos en nuevas versiones del producto software.

Software: Es un término genérico que designa al conjunto de programas de distinto tipo (sistema operativo y aplicaciones diversas) que hacen posible operar con el ordenador.

Tecnologías de la Información y las Comunicaciones (TIC): Conjunto de avances tecnológicos que proporciona la informática, las telecomunicaciones y las tecnologías audiovisuales, que comprenden los desarrollos relacionados con los ordenadores, Internet, la telefonía, los "mas media", las aplicaciones multimedia y la realidad virtual. Estas tecnologías básicamente proporcionan información, herramientas para su proceso y canales de comunicación.

Tecnología: Conjunto de teorías y de técnicas que permiten el aprovechamiento práctico del conocimiento científico.

UML: Lenguaje estándar para el modelado de software - lenguaje para visualizar, especificar, construir y documentar los artefactos de un sistema con gran cantidad de software. Permite a los desarrolladores visualizar el producto de su trabajo en esquemas o diagramas estandarizados.