UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

FACULTAD 9

APLICACIÓN DE LA NOTACIÓN OMMMA-L PARA SOFTWARE EDUCATIVO

TRABAJO DE DIPLOMA PARA OPTAR POR EL TÍTULO DE INGENIERÍA EN CIENCIAS INFORMÁTICAS

Autor: Yadira Mislay López Hernández

Tutor: Ing. Yaneisis Pérez Heredia

Ciudad de la Habana, Julio 2007

"Año 49 de la Revolución"

La enseñanza que deja huella

No es aquella que se hace de cabeza a cabeza

Es la que se hace de corazón a corazón

Félix Varela

DEDICATORIA

A mis familiares

AGRADECIMIENTOS

A mis padres que durante todo el tiempo transcurrido en La Universidad de las Ciencias Informáticas, han hecho posible mi estancia en la misma. Ellos que han sido mi sostén y aliento para seguir adelante, quiero agradecerles el haberme ayudado a mantenerme firme en el camino tan difícil que se mostró al principio de la carrera, cuando pensaba que el fin nunca iba a llegar. Ellos que se han esforzado por hacerme cada día más llevadero para que solo me concentrara en mi carrera. Pero sobre todo quiero agradecer a mi mamá que me lleno de consejos, que me guió, que me condujo sabiamente por el camino correcto.

Quiero agradecer además a mi hermano ya que gracias a el me he esforzado por ser mejor y he tratado de ser ejemplo a seguir, para que un día el me tome como referencia y trate de hacer las cosas bien en la vida, que se esfuerce por ser cada día mejor el también.

A mis abuelos que con su cariño han sembrado la semilla del triunfo para que yo con mis logros la regara cada día y, por que no, recoger frutos en un futuro cercano. A mi tía que durante todo este tiempo ha permanecido a mi lado y que fue la primera persona con quien pude compartir las experiencias iniciales de lo que luego fuera la UCI, cuando solo podíamos llamarle ”Proyecto Futuro”. Quiero además dar gracias a toda mi familia que de manera directa o no han sido parte esencial de esta etapa de mi vida.

A mi esposo que ha sido mi mejor aliado en este trance, él que ha sabido compensar cada día de ausencia con amor, dedicación, entrega, apoyo. El que con absoluta paciencia me ha sabido comprender como pocas personas, que me ha apoyado en los estudios y demás tareas, que ha sido ejemplo, amigo, que me ha enseñado a disfrutar el logro que es aprender algo cada día.

Quiero dar gracias a mi tutora que ha tenido la suficiente paciencia de cada día lidiar con las inexperiencias, los problemas y demás situaciones que se han presentado en el transcurso de la elaboración del presente trabajo. Vale resaltar el papel tan importante que ha jugado al hacerse participe de cada situación, de cada suceso importante. Quiero agradecer al tiempo que me ha dedicado, aún teniendo gran contenido de trabajo por

realizar. Ella que ha sido incansablemente preocupada, que a cada momento estaba dispuesta a aclarar una duda, a “dar una mano” en lo que fuera necesario.

Quiero dar gracias a mis compañeros de grupo que me han hacho más amena la vida dentro de La Universidad y que han sido durante todo este tiempo nuestra familia mas cercana, esos con los que hemos podido contar en todo momento, esos que siempre han estado para bien o para mal, a todos ellos gracias.

Pero sobre todo quiero dar gracias a La Revolución y a nuestro Comandante en Jefe Fidel Castro por haber tenido la suficiente confianza en nosotros para habernos aceptado como creadores de este proyecto que unió a personas de toda la nación para hacer realidad un sueño de muchos de nosotros y darnos la oportunidad de graduarnos en una escuela que consta del mejor y mas comprometido claustro que hayamos podido tener.

DECLARACION DE AUTORIA

Por este medio se declara que soy la única autora de este trabajo y se autoriza a la Universidad de las Ciencias Informáticas para que haga el uso que estime pertinente con el mismo.

Para que así conste firmamos la presente a los ____ días del mes de ____ de 2007.

_____

Yadira Mislay López Hernández.

_____

Yaneisis Pérez Heredia

OPINIÓN DEL TUTOR DEL TRABAJO DE DIPLOMA

Título: “Aplicación de la notación OMMMA-L para Software Educativo”.

Autor(es): Yadira Mislay López Hernández

Como tutora del Trabajo de Diploma “Aplicación de la notación OMMMA-L para Software Educativo”, luego de haber culminado la realización del mismo, considero que la autora Yadira Mislay López Hernández ha desarrollado un conjunto de habilidades que les permitirán darle solución adecuadamente a cualquier tipo de necesidad de informatización que se presente en su vida profesional.

Durante la realización del presente trabajo la estudiante ha trabajado organizadamente dando muestras de poseer responsabilidad y compromiso en la realización de su tesis. Su desempeño ha manifestado que ha desarrollado un valioso nivel de asimilación de nuevas notaciones, llegando a alcanzar un profundo conocimiento y una gran capacidad para la toma de decisiones correctas.

La originalidad, la elegancia en el trabajo y la independencia, han sido cualidades dignas de destacar a lo largo de la realización del trabajo realizado. El documento está muy bien redactado y no presenta faltas de ortografía. La estudiante manifestó laboriosidad a lo largo del cumplimiento de las tareas programadas, logrando resultados satisfactorios.

Por otra parte, el elemento investigativo del documento, estuvo desde el inicio muy bien orientado y estructurado, basado en una gran cantidad de bibliografía actualizada. Cada contenido se ha expuesto con claridad y aporta grandes conocimientos al lector.

Por todo lo anteriormente planteado, considero que la diplomante está apta para ejercer como Ingeniero Informático; y propongo al tribunal que se le otorgue al Trabajo de Diploma la máxima calificación.

_____ _____

Firma Fecha

RESUMEN

El presente trabajo ha sido realizado con el objetivo de ampliar los conocimientos a cerca del Lenguaje de Modelado para Aplicaciones Multimedia OMMMA-L, debido a que este tipo de software es en estos momentos uno de los mas utilizados en el ámbito nacional ya que es constituye una solución a los problemas existentes para acceder a Internet. Para demostrar las ventajas que esta notación ofrece se planteo su aplicación a un caso tomado de la CUJAE para la asignatura “Topografía” de la carrera de Ingeniería Civil.

Para ello fue necesario realizar un profundo estudio de los conceptos de software, aplicaciones multimedia, lenguajes de modelado, además se investigo sobre las diferentes tipologías de Software existentes, se da a conocer que es la Ingeniería de Software y que es un Proceso de Ingeniería de Software Educativo, así como las diferentes clasificaciones que estos tienen, sus ventajas e inconvenientes. De las aplicaciones multimedia también se muestran sus diferentes características y los aspectos (positivos y negativos) más relevantes de las mismas.

Con respecto al lenguaje de modelado OMMMA-L, tema que ocupa fundamentalmente el desarrollo del presente trabajo, se hizo necesario realizar una investigación que ayudó a comprender de manera más eficiente el funcionamiento de esta notación así como sus características fundamentales. De esta notación se supo que era una extensión de UML para modelar Aplicaciones Multimedia y que a diferencia de UML, que posee 8 vistas, este lenguaje de modelado esta sustentado en 4 vistas fundamentalmente y que ellas están dadas por diferentes diagramas que se desarrollan en las fases de Análisis y Diseño del software multimedia. Esta notación brinda la posibilidad de realizar de una manera ágil y eficaz el desarrollo de este tipo de software.

Luego de estudiar detenidamente OMMMA-L pasamos a la realización del análisis y diseño del software multimedia educativo “Topografía”. Esto constituye un ejemplo de cómo utilizar esta notación así como las facilidades que ella brinda.

TABLA DE CONTENIDOS

INTRODUCCIÓN

El fin de la educación
No es hacer al hombre nulo,
por el desdén o el acomodo imposible
al país en que ha de vivir,
sino prepararlo para vivir bueno y útil en él.

José Martí.

Utilizar la informática como apoyo a los procesos de enseñanza/aprendizaje ha sido una inquietud que durante mucho tiempo ha sido investigada y probada por muchas instituciones y docentes. Su asimilación dentro de instituciones educativas, incluyendo el hogar, ha aumentado en los últimos años, por lo que la demanda de software educativo de calidad es cada vez mayor. Por lo tanto se investiga sobre los lenguajes de modelado que se puedan utilizar para desarrollar software educativo (multimedia) de calidad. Se conoce que en estas notaciones, se debe incorporar dentro de las fases de análisis y diseño, aspectos didácticos y pedagógicos.

Existen varios lenguajes de modelado para el desarrollo de aplicaciones hipermedia, aunque no todos están orientados al mismo tipo de aplicaciones. En los últimos años, la construcción de modelos orientados a objetos 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 [1, 2, 3].

Existen diversas propuestas de lenguajes para la construcción de modelos orientados a objetivos. Entre ellas destaca poderosamente la notación i* propuesta por Eric Yu en la primera mitad de la década de los 90 [4, 5]. Podemos encontrar además el GRL [15], así también existen notaciones tales como DFD, Notación Mutimed y UML, aunque cabe resaltar que esta ultima ha sido de gran importancia para el desarrollo de muchos de los productos de software desarrollados en la actualidad.

El Proceso Unificado de Software (RUP) es un proceso de desarrollo creado por la Corporación “Multimed Software”, ahora una división de IBM, como una plataforma adaptable de procesos para describir cómo crear productos efectivos a través de técnicas de alta fidelidad. Aunque RUP abarca un determinado número de actividades diferentes,

está diseñado para poder ajustarse en la selección de procesos específicos destinados a un proyecto u organización de desarrollo en particular y es reconocida en medio de grandes equipos de trabajo que llevan a cabo el manejo de complicadas aplicaciones de software.

Los creadores de este proceso, se basaron en los diagnósticos de las fallas de diferentes proyectos de software, identificaron las causas matrices, los procesos de ingeniería de software y las soluciones propuestas, construyendo un sistema basado en el conjunto de todas las formas óptimas de trabajo y modelando el proceso de desarrollo con las mismas técnicas de modelado de software, a través del paradigma Orientado a Objetos y el Lenguaje Unificado (UML).

RUP se aplica a una buena cantidad de productos y procesos de software en el mundo. No es específico para diseño hipermedia, sin embargo a través de la extensión de UML para multimedia, conocida por OMMMA – L, se presenta como algo eficientemente realizable. En el presente trabajo se eligió el lenguaje de modelado OMMMA-L para hacer un estudio profundo de la misma y proponer su aplicación en el desarrollo de SWE, dicha metodología fue desarrollada por Stefan ulti y Gregor Engels, profesores e investigadores del Departamento de Matemática y Ciencias de la Computación de la Universidad de Paderborn, Alemania, OMMMA-L 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, dando una muestra sólida de la integración temporal y la sincronización de diferentes objetos de media.

En cuanto a lenguajes de modelado, varios autores han tratado el tema, por ejemplo Jaime Proluskys o Alvaro Galvís [Galvis, 94]. De éste último: “Ingeniería de Software Educativo”, es una referencia bastante completa y es una buena guía para el desarrollo del software.

Dentro de los aportes en ingeniería de SWE destacan los trabajos de Galvis (2000), Hinostroza, Hepp y Straub (1996), los enfocados a orientación a objetos (Mónica, 1993; Gómez, Galvís y Mariño, 1998; Ovalle y Padilla, 1998) y los de reutilización de componentes (DiGiano y Roschelle 2000; Roschelle y Kaput, 1996; Neilson y Thomas 1996; Roschelle, Kaput y Kahn, 1998; García, González, Mandado, Pérez, Baltasar y Valdés, 2002).

El aprendizaje multimedia se adopta en Cuba y en gran parte del mundo como una alternativa a la barrera del alcance tecnológico a Internet, así como las posibilidades del conocimiento residente y atractivo, mucho más pedagógico e influyente por el poder captativo de la interactividad. Este conocimiento residente, de fácil distribución y amplia accesibilidad, unido a la variedad de temas que modela, hace de la multimedia educativa un arma importante para la preparación de la sociedad en cualquier espacio productivo, cultural o social.

Nuestro país en los últimos años se ha visto enfrascado en una revolución tecnológica donde la esfera de educación no ha estado exenta en toda una serie de inversiones que ha realizado el Estado Cubano en equipos y tecnología. Se han instalado en todas las escuelas computadoras y otros equipos para que se haga uso de las Tecnologías de la Informática y las Comunicaciones (TIC) en los procesos de enseñanza-aprendizaje.

En nuestro país el Departamento Nacional de Software Educativo del Ministerio de Educación que forma parte de la Dirección de Computación Educacional tiene dentro de sus funciones dirigir, coordinar, organizar y controlar la producción e introducción en la práctica de software educativo para la escuela cubana desde el ámbito de la red de Centros de Estudio de Software Educativo de los Institutos Superiores Pedagógicos, bajo un esquema de investigación-producción.

Se han creado proyectos de desarrollo de software en los Institutos Superiores Pedagógicos donde trabajan Master en Ciencias y Licenciados junto a programadores y técnicos en Ciencias de la Informática entre otros especialistas. Con el objetivo de satisfacer las necesidades de SWE como medio de enseñanza-aprendizaje en la escuela cubana, específicamente software educativo multimedia.

Como resultado de estos proyectos en los últimos años se han desarrollado más de 78 productos, otros que se están desarrollando y algunos que están en proyecto para un futuro no muy lejano. Nuestras escuelas cuentan con una diversidad de SWE que los alumnos y docentes lo utilizan de forma satisfactoria en su preparación.

Actualmente se trabajo con “Multisaber”, colección que tiene un enfoque curricular y multidisciplinario por su relación con los contenidos de los programas de cada asignatura del currículo de estudio de la Educación Primaria.

- ”El Navegante”, colección de 10 software para la Educación Secundaria.
- “Futuro”, colección de 19 software para la Educación Preuniversitaria.

- "Jugar y Aprender". Este nuevo software es el resultado de un proyecto de investigación que se lleva a cabo en nuestro país por un equipo de especialistas relacionado con la inclusión de la computación en las edades preescolares.

Además se han desarrollado varios software multimedia monotemáticos de diferentes educaciones que son utilizados según el tema específico para el cual fueron diseñados e implementados.

Estos proyectos no hacen uso de un lenguaje de modelado que guié el proceso de desarrollo de los SWE que en ellos se realizan y si analizamos que un proceso de desarrollo de software requiere, por un lado un conjunto de conceptos, una metodología y un lenguaje propio. Podemos decir que la notación a utilizar es parte fundamental dentro del proceso de desarrollo del mismo; pero como cada software puede tener sus características específicas, debemos que elegir para cada tipo de software la notación que nos brinde mejor eficiencia en su desarrollo.

Por tanto **el problema científico** a resolver seria:

Como lograr mayor eficiencia en la representación estructural así como funcional del software educativo a través del uso de la notación OMMMA-L.

Tendríamos así como **objeto de estudio**:

- Notaciones para la representación del SWE.
- Representación estructural y funcional del SWE en nuestro país.

Se persigue como **objetivo general** la aplicación de la notación OMMMA-L al caso de estudio Multimedia "Topografía".

Para dar cumplimiento al objetivo planteado se ha propuesto cumplir las siguientes **tareas**.

1. Estudio de las notaciones internacionales para el desarrollo de software educativos.
2. Estudio del software educativo a nivel internacional.
3. Estudio de las condiciones actuales en la cuales se desarrolla el software educativo en Cuba.

4. Estudio del software educativo “Topografía” y de la metodología OMMMA-L aplicable a este software.
5. Aplicar la metodología OMMMA-L al caso de prueba “Topografía”

Para el desarrollo de nuestro trabajo se utilizaron como **métodos científicos de la investigación** los que a continuación hacemos referencia.

Se empleó como **métodos teóricos** el histórico-lógico con el fin de investigar sobre las aplicaciones informáticas multimedia desarrolladas Cuba y el resto del mundo que aplican OMMMA-L como notación para su realización. Su uso en el ámbito de la educación, así como las ventajas y desventajas de este lenguaje de modelado. El hipotético-deductivo en la formulación del planteamiento hipotético lo que nos permitió establecer predicciones a partir del sistema de conocimientos mediante la aplicación de reglas lógicas de la deducción.

Después de darle cumplimiento a todas las tareas planteadas en función a resolver el problema antes expuesto esperamos que nuestra investigación sirva como guía o como material de estudio para futuros sistemas con fines similares en otras instituciones.

Para lograr una mejor comprensión el presente documento se estructura en tres capítulos de contenidos donde se incluye todo lo relacionado con el trabajo investigativo realizado.

Estructuración del contenido en una breve explicación de sus partes.

Capitulo I

El capitulo 1 constituye la fundamentación teórica del presente trabajo, este es el que refleja los principales conceptos que serán abordados. En el se describen aspectos tales como ¿Qué es Software?, se trata un sin numero de aspectos que dan cuerpo y funcionalidad tanto a los SWE como a las aplicaciones multimedia, así como sus ventajas y desventajas y por ultimo se trata de dar una visión actual de algunos de los lenguajes de modelado orientados a objeto que existen en el mundo.

Capitulo II

En este se tratara fundamentalmente la Notación OMMMA-L (Object-oriented Modeling of ultimedia Applications – the Language), lenguaje de modelado orientado al desarrollo de aplicaciones Multimedia, extensión de UML que surge para brindar los aspectos a la hora de modelar las aplicaciones multimedia que se obvian en UML . Se verán además algunos aspectos del Modelo Vista Controlador debido a que es con el que trabaja OMMMA-L.

Capitulo III

Esta etapa del presente trabajo tiene como principal aspecto el desarrollo de un Caso de Estudio que tendrá como objetivo mostrar las facilidades que tiene OMMMA-L para desarrollar aplicaciones multimedia y dejará a la vista los nuevos aspectos (4 vistas que se sustentan en diferentes diagramas) que esta notación incluye, así como la factibilidad que estas ofrecen a la hora de desarrollar la aplicación.

CAPÍTULO I

Fundamentación Teórica

Educar no es fabricar adultos según un modelo
Sino liberar en cada hombre lo que le impide ser el mismo,
Permitirle realizarse según su genio.

Oliver Reboul.

En este primer capítulo del presente trabajo de diploma se hace alusión a los conceptos fundamentales y se describen de forma general los aspectos relacionados con el objeto de estudio y el campo de acción en que se trabaja. Este capítulo es la base teórica para la comprensión del trabajo que se desarrolla y los aspectos fundamentales son el proceso de desarrollo de software educativo, la importancia que tiene la ingeniería de software, que es un software educativo y el aspecto fundamental, las metodologías de desarrollo que se aplican en el proceso de desarrollo de software.

1.1 El proceso de Ingeniería de Software

1.1.1 Software

Se denomina software, programática, equipamiento lógico o soporte lógico a todos los componentes intangibles de un ordenador o computadora, es decir, al conjunto de programas y procedimientos necesarios para hacer posible la realización de una tarea específica, en contraposición a los componentes físicos del sistema (hardware). Esto incluye aplicaciones informáticas tales como un procesador de textos, que permite al usuario realizar una tarea, y software de sistema como un sistema operativo, que permite al resto de programas funcionar adecuadamente, facilitando la interacción con los componentes físicos y el resto de aplicaciones. [1]

Software es la rama de la ingeniería que aplica los principios de la ciencia de la computación y las matemáticas para lograr soluciones costo-efectivas (eficaces en costo

o económicas) a los problemas de desarrollo de software", es decir, "permite elaborar consistentemente productos correctos, utilizables y costo-efectivos" [Cota 1994].

Probablemente la definición más formal de software es la atribuida a la IEEE en su estándar 729: «la suma total de los programas de cómputo, procedimientos, reglas documentación y datos asociados que forman parte de las operaciones de un sistema de cómputo» [Lewis 1994]. Bajo esta definición, el concepto de software va más allá de los programas de cómputo en sus distintas formas: código fuente, binario o ejecutable, además de su documentación: es decir, todo lo intangible.

El término «software» fue usado por primera vez en este sentido por John W. Tukey en 1957. En las ciencias de la computación y la ingeniería de software, el software es toda la información procesada por los sistemas informáticos: programas y datos. El concepto de leer diferentes secuencias de instrucciones de la memoria de un dispositivo para controlar cálculos fue inventado por Charles Babbage como parte de su máquina diferencial. La teoría que forma la base de la mayor parte del software moderno fue propuesta por vez primera por Alan Turing en su ensayo de 1936, Los números computables, con una aplicación al problema de decisión.

El software se ha convertido en el elemento clave de la evolución de los sistemas y productos informáticos. En las pasadas cuatro décadas, el software ha pasado de ser una resolución de problemas especializadas y una herramienta de análisis de información, a ser una industria por si misma.

1.1.2 Ingeniería de Software

La Ingeniería de Software es una tecnología multicapa en la que, según Pressman, se pueden identificar: los métodos (indican cómo construir técnicamente el software), el proceso (es el fundamento de la Ingeniería de Software, es la unión que mantiene juntas las capas de la tecnología) y las herramientas (soporte automático o semiautomático para el proceso y los métodos) (ver figura 1).

Ingeniería de Software es la rama de la ingeniería que aplica los principios de la ciencia de la computación y las matemáticas para lograr soluciones costo-efectivas (eficaces en costo o económicas) a los problemas de desarrollo de software", es decir, "permite elaborar consistentemente productos correctos, utilizables y costo-efectivos" [2].

1.1.3 Proceso de Ingeniería de Software.

El proceso de ingeniería de software se define como "un conjunto de etapas parcialmente ordenadas con la intención de logra un objetivo, en este caso, la obtención de un producto de software de calidad" [Jacobson 1998].

El proceso de desarrollo de software "es aquel en que las necesidades del usuario son traducidas en requerimientos de software, estos requerimientos transformados en diseño y el diseño implementado en código, el código es probado, documentado y certificado para su uso operativo". Concretamente "define quién está haciendo qué, cuándo hacerlo y cómo alcanzar un cierto objetivo" [2].

El proceso de desarrollo de software requiere de un conjunto de conceptos, una metodología y un lenguaje propio. A este proceso también se le llama el ciclo de vida del software que comprende cuatro grandes fases: concepción, elaboración, construcción y transición. La concepción define el alcance del proyecto y desarrolla un caso de negocio. La elaboración define un plan del proyecto, especifica las características y fundamenta la arquitectura. La construcción crea el producto y la transición transfiere el producto a los usuarios.

En otros documentos se dice que el proceso de construcción de un sistema de software está formado por etapas que son: la obtención de los requisitos, el diseño del sistema, la codificación y las pruebas del sistema. Desde la perspectiva del producto, se parte de una necesidad, se especifican los requisitos, se obtiene el diseño del mismo, el código respectivo y por último el sistema de software.

Algunos autores sostienen que el nombre ciclo de vida ha sido relegado en los últimos años, utilizando en su lugar proceso de software, cambiando la perspectiva de producto a proceso. En todas las etapas del ciclo de vida del software o proceso de software es necesario definir los procesos, las actividades y las tareas a desarrollar.

El proceso de software puede describirse sintéticamente como: la obtención de los requisitos del software, el diseño del sistema de software (diseño preliminar y diseño detallado), la implementación, las pruebas, la instalación, el mantenimiento y la ampliación o actualización del sistema.

A veces se habla de ciclo de desarrollo para denominar al subconjunto del ciclo de vida que empieza en el análisis y finaliza en la entrega del producto.

1.1.4 Proceso de Ingeniería de Software Educativo

Según Alvaro Galvis de la Universidad de los Andes, es muy coherente y pertinente considerar la producción de software educativo dentro de la Ingeniería de software.

¿Qué tan valida es la propuesta de Alvaro Galvis, donde nos habla de la ingeniería del software, se puede considerar una Ingeniería de software educativo aparte de la Ingeniería del Software o es un caso particular donde el análisis del dominio del problema es educativo?

Se considera que la Ingeniería del Software educativo es una adecuación al área de las ciencias informáticas enfocadas a la creación de software que facilite el proceso de aprendizaje-enseñanza como herramienta pedagógica.

En el proceso de software educativo se tiene como uno de las fundamentales tareas la inclusión de los aspectos pedagógicos, siguiendo las pautas de las teorías educativas y de la comunicación subyacente.

1.2 El Software Educativo

Por la importancia que se brinda al SWE en este trabajo se le dedicará el presente epígrafe para mostrar todos los aspectos fundamentales sobre este tipo de software que encontramos dentro de la tipología de software de aplicación, porque se puede distinguir el software arbitrariamente de la siguiente forma:

1.2.1 Tipologías de Software

- Software de sistema, que permite funcionar al hardware. Su objetivo es aislar tanto como sea posible al programador de aplicaciones de los detalles del computador particular que se use, especialmente de las características físicas de la memoria, dispositivos de comunicaciones, impresoras, pantallas, teclados, etcétera. Incluye entre otros:
    - Sistemas operativos
    - Controladores de dispositivo

- Herramientas de diagnóstico
- Servidores
- Sistemas de ventanas
- Utilidades

- Software de programación, que proporciona herramientas para ayudar al programador a escribir programas informáticos y a usar diferentes lenguajes de programación de forma práctica. Incluye entre otros:

  - Editores de texto
  - Compiladores
  - Intérpretes
  - Enlazadores
  - Depuradores

- Software de aplicación, que permite a los usuarios llevar a cabo una o varias tareas más específicas, en cualquier campo de actividad susceptible de ser automatizado o asistido, con especial énfasis en los negocios. Incluye entre otros:

  - Aplicaciones de automatización industrial.
  - Aplicaciones ofimáticas.
  - Software educativo.
  - Software médico.
  - Bases de datos.
  - Videojuegos.

1.2.3 Concepto de Software Educativo

El software educativo se puede definir como entornos de trabajo en formato digital orientado temático y metodológicamente al proceso de formación, los avances tecnológicos han enriquecido enormemente las posibilidades de trabajo al integrar elementos multimedia y nuevas concepciones pedagógicas. El apoyo de estos programas a la labor educativa puede ser catalogado como diverso dependiendo por un lado de las posibilidades ofertadas por el software y por otro la iniciativa metodológica del docente.

Otra definición de Software Educativo es que son "Programas para computadora elaborados con fines didácticos". Dicho de otra manera, es aquello que convierte a la computadora común, de una máquina de propósito general a una máquina para fines educativos.

Sirve para:

- Elevar el desempeño académico.
- Enriquecer el menú de recursos de enseñanza que el educador tiene
- Ofrecer un medio ágil para la consulta.
- Guiar al educando en su proceso de aprendizaje.
- Ayudar al monitoreo del desarrollo académico.
- Motivar al alumno
- Evaluar.
- Servir de herramienta para la investigación.
- Solventar insuficiencia de recursos humanos para la enseñanza.

Ing. Héctor A. Silva Sánchez

El software educativo se ha enfocado principalmente en dos polos:

El software educativo de tipo algorítmico que es aquel en donde predomina el aprendizaje por transmisión de conocimientos, dentro de estos se pueden considerar los denominados tutoriales, entrenadores y libros electrónicos.

El software educativo de tipo heurísticos que es donde el estudiante descubre el conocimiento interactuando con el ambiente de aprendizaje que le permite llegar a él, aquí se encuentran los simuladores, juegos educativos, sistemas expertos y sistemas tutoriales inteligentes.

Cada uno de estos software tienen sus cualidades y limitaciones que se deben tener en cuenta a la hora de seleccionar uno de ellos, dada una necesidad educativa.

El software educativo es uno de los pilares del sistema de educación a distancia y se perfila como la herramienta base de las próximas generaciones de educandos.

Dentro de una gran variedad de software este trabajo se enfoca en el desarrollo de software educativo el cual se define como un programa automatizado diseñado con una intencionalidad educativa para ser utilizado en el proceso de aprendizaje, utiliza procedimientos para que el estudiante aprenda, se fomenta el análisis de problemas,

facilita el trabajo en grupo, provee soporte en actividades docentes y en el sentido más amplio, mejora las habilidades del pensamiento y la resolución de problemas.

1.2.4 Clasificación de los Software Educativos

El software educativo son la base fundamental de la enseñanza aprendizaje en la actualidad, estos se presentan de diferentes formas unos se presentan como juegos, otros como laboratorios o libros etc. Para resolver esto Pérez Marques [2] plantea 7 clasificaciones según el tipo, en este trabajo se abordaran tres de ellas, las cuales son:

Atendiendo a su estructura, los materiales didácticos multimedia se pueden clasificar en programas tutoriales, de ejercitación, simuladores, bases de datos, constructores, programas herramienta..., presentando diversas concepciones sobre el aprendizaje y permitiendo en algunos casos (programas abiertos, lenguajes de autor) la modificación de sus contenidos y la creación de nuevas actividades de aprendizaje por parte de los profesores y los estudiantes. Con más detalle, la clasificación es la siguiente:

- **Materiales formativos directivos**. En general siguen planteamientos conductistas. Proporcionan información, proponen preguntas y ejercicios a los alumnos y corrigen sus respuestas.
- **Programas de ejercitación**. Se limitan a proponer ejercicios autocorrectivos de refuerzo sin proporcionar explicaciones conceptuales previas

Su estructura puede ser: lineal (la secuencia en la que se presentan las actividades es única o totalmente aleatoria), ramificada (la secuencia depende de los aciertos de los usuarios) o tipo entorno (proporciona a los alumnos herramientas de búsqueda y de proceso de la información para que construyan la respuesta a las preguntas del programa).

- **Programas tutoriales**. Presentan unos contenidos y proponen ejercicios autocorrectivos al respecto. Si utilizan técnicas de Inteligencia Artificial para personalizar la tutorización según las características de cada estudiante, se denominan tutoriales expertos.
- **Bases de datos**. Presentan datos organizados en un entorno estático mediante unos criterios que facilitan su exploración y consulta selectiva para resolver problemas, analizar y relacionar datos, comprobar hipótesis, extraer

conclusiones... Al utilizarlos se pueden formular preguntas del tipo: *¿Qué características tiene este dato? ¿Qué datos hay con la característica X? ¿Y con las características X e Y?.*

- **Programas tipo libro o cuento**. Presenta una narración o una información en un entorno estático como un libro o cuento.
- **Bases de datos convencionales**. Almacenan la información en ficheros, mapas o gráficos, que el usuario puede recorrer según su criterio para recopilar información.
- **Bases de datos expertas**. Son bases de datos muy especializadas que recopilan toda la información existente de un tema concreto y además asesoran al usuario cuando accede buscando determinadas respuestas.
- **Simuladores**. Presentan modelos dinámicos interactivos (generalmente con animaciones) y los alumnos realizan aprendizajes significativos por descubrimiento al explorarlos, modificarlos y tomar decisiones ante situaciones de difícil acceso en la vida real (pilotar un avión, VIAJAR POR LA Historia A través del tiempo...). Al utilizarlos se pueden formular preguntas del tipo: *¿Qué pasa al modelo si modifico el valor de la variable X? ¿Y si modifico el parámetro Y?*
- **Modelos físico-matemáticos**. Presentan de manera numérica o gráfica una realidad que tiene unas leyes representadas por un sistema de ecuaciones deterministas. Incluyen los programas-laboratorio, trazadores de funciones y los programas que con un convertidor analógico-digital captan datos de un fenómeno externo y presentan en pantalla informaciones y gráficos del mismo.
- **Entornos sociales**. Presentan una realidad regida por unas leyes no del todo deterministas. Se incluyen aquí los juegos de estrategia y de aventura.
- **Constructores o talleres creativos**. Facilitan aprendizajes heurísticos, de acuerdo con los planteamientos constructivistas. Son entornos programables (con los interfaces convenientes se pueden controlar pequeños robots), que facilitan unos elementos simples con los cuales pueden construir entornos complejos. Los alumnos se convierten en profesores del ordenador. Al utilizarlos se pueden formular preguntas del tipo: *¿Qué sucede si añado o elimino el elemento X?*

- **Constructores específicos**. Ponen a disposición de los estudiantes unos mecanismos de actuación (generalmente en forma de órdenes específicas) que permiten la construcción de determinados entornos, modelos o estructuras.
- **Lenguajes de programación**. Ofrecen unos "laboratorios simbólicos" en los que se pueden construir un número ilimitado de entornos

Hay que destacar el lenguaje LOGO, creado en 1969 por Seymour Papert, un programa constructor que tiene una doble dimensión: proporciona a los estudiantes entornos para la exploración y facilita el desarrollo de actividades de programación, que suponen diseñar proyectos, analizar problemas, tomar decisiones y evaluar los resultados de sus acciones.

- **Programas herramienta**. Proporcionan un entorno instrumental con el cual se facilita la realización de ciertos trabajos generales de tratamiento de la información: escribir, organizar, calcular, dibujar, transmitir, captar datos, etc.
- **Programas de uso general**. Los más utilizados son programas de uso general (procesadores de textos, editores gráficos, hojas de cálculo...) que provienen del mundo laboral. No obstante, se han elaborado versiones "para niños" que limitan sus posibilidades a cambio de una, no siempre clara, mayor facilidad de uso.
- **Lenguajes y sistemas de autor**. Facilitan la elaboración de programas tutoriales a los profesores que no disponen de grandes conocimientos informáticos.

Atendiendo a su concepción sobre el aprendizaje, en los materiales didácticos multimedia podemos identificar diversos planteamientos :la perspectiva conductista (B.F.Skinner), la teoría del procesamiento de la información (Phye), el aprendizaje por descubrimiento (J. Bruner), el aprendizaje significativo (D. Ausubel, J. Novak), el enfoque cognitivo (Merrill, Gagné, Solomon...), el constructivismo (J.Piaget), el socio-constructivismo (Vigotsky):

- **La perspectiva conductista**. Desde la perspectiva conductista, formulada por B.F.Skinner hacia mediados del siglo XX y que arranca de Wundt y Watson, pasando por los estudios psicológicos de Pavlov sobre condicionamiento y de los trabajos de Thorndike sobre el refuerzo, intenta explicar el aprendizaje a partir de unas leyes y mecanismos comunes para todos los individuos.
- **Condicionamiento operante**. Formación de reflejos condicionados mediante mecanismos de estímulo-respuesta-refuerzo. Aprendizaje = conexiones entre estímulos y respuestas.

- **Ensayo y error con refuerzos y repetición**: las acciones que obtienen un refuerzo positivo tienden a ser repetidas.
- **Asociacionismo**: los conocimientos se elaboran estableciendo asociaciones entre los estímulos que se captan. Memorización mecánica.
- **Enseñanza programada**: Resulta especialmente eficaz cuando los contenidos están muy estructurados y secuenciados y se precisa un aprendizaje memorístico. Su eficacia es menor para la comprensión de procesos complejos y la resolución de problemas no convencionales. Los primeros ejemplos están en las máquinas de enseñar de Skinner y los sistemas ramificados de Crowder. En muchos materiales didácticos multimedia directivos (ejercitación, tutoriales) subyace esta perspectiva.
- **Teoría del procesamiento de la información** (Phye): La teoría del procesamiento de la información, influida por los estudios cibernéticos de los años cincuenta y sesenta, presenta una explicación sobre los procesos internos que se producen durante el aprendizaje. Sus planteamientos básicos, en líneas generales, son ampliamente aceptados. Considera las siguientes fases principales:
  - Captación y filtro de la información a partir de las sensaciones y percepciones obtenidas al interactuar con el medio.
  - Almacenamiento momentáneo en los registros sensoriales y entrada en la memoria a corto plazo, donde, si se mantiene la actividad mental centrada en esta información, se realiza un reconocimiento y codificación conceptual.
  - Organización y almacenamiento definitivo en la memoria a largo plazo, donde el conocimiento se organiza en forma de redes. Desde aquí la información podrá ser recuperada cuando sea necesario.
  - Aprendizaje por descubrimiento. La perspectiva del aprendizaje por descubrimiento, desarrollada por J. Bruner, atribuye una gran importancia a la actividad directa de los estudiantes sobre la realidad.
  - Experimentación directa sobre la realidad, aplicación práctica de los conocimientos y su transferencia a diversas situaciones.

- Aprendizaje por penetración comprensiva. El alumno experimentando descubre y comprende lo que es relevante, las estructuras.
- Práctica de la inducción: de lo concreto a lo abstracto, de los hechos a las teorías.
- Utilización de estrategias heurísticas, pensamiento divergente.
- Currículum en espiral: revisión y ampliación periódica de los conocimientos adquiridos.

Esta perspectiva está presente en la mayoría de los materiales didácticos multimedia no directivos (simuladores, constructores...)

- Aprendizaje significativo (D. Ausubel, J. Novak) postula que el aprendizaje debe ser significativo, no memorístico, y para ello los nuevos conocimientos deben relacionarse con los conocimientos previos que posea el aprendiz. Frente al aprendizaje por descubrimiento de Bruner, defiende el aprendizaje por recepción donde el profesor estructura los contenidos y las actividades a realizar para que los conocimientos sean significativos para los estudiantes.
- Condiciones para el aprendizaje:
  - Significabilidad lógica (se puede relacionar con conocimientos previos).
  - Significabilidad psicológica (adecuación al desarrollo del alumno).
  - Actitud activa y motivación.
- Relación de los nuevos conocimientos con los conocimientos previos. La mente es como una red proposicional donde aprender es establecer relaciones semánticas.
- Utilización de organizadores previos que faciliten la activación de los conocimientos previos relacionados con los aprendizajes que se quieren realizar.
- Diferenciación-reconciliación integradora que genera una memorización comprensiva.
- Funcionalidad de los aprendizajes, que tengan interés, se vean útiles.

Esta perspectiva está presente en la mayoría de los materiales didácticos multimedia.

- Enfoque cognitivo. Psicología cognitivista. El cognitivismo (Merrill, Gagné, Solomon...), basado en las teorías del procesamiento de la información y recogiendo también algunas ideas conductistas (refuerzo, análisis de tareas) y del aprendizaje significativo, aparece en la década de los sesenta y pretende dar una explicación más detallada de los procesos de aprendizaje, distingue:
  - El aprendizaje es un proceso activo. El cerebro es un procesador paralelo, capaz de tratar con múltiples estímulos. El aprendizaje tiene lugar con una combinación de fisiología y emociones. El desafío estimula el aprendizaje, mientras que el miedo lo retrae.

El estudiante representará en su mente simbólicamente el conocimiento, que se considera (igual que los conductistas) como una realidad que existe externamente al estudiante y que éste debe adquirir. El aprendizaje consiste en la *adquisición y representación exacta del conocimiento externo*. La enseñanza debe facilitar la transmisión y recepción por el alumno de este conocimiento estructurado.

Posteriormente cuando se haga una pregunta al estudiante se activarán las fases: *recuerdo, generalización o aplicación* (si es el caso) y *ejecución* (al dar la respuesta, que si es acertada dará lugar a un *refuerzo*).

- Condiciones internas que intervienen en el proceso: *motivación, captación y comprensión, adquisición, retención*.
- Condiciones externas: son las circunstancias que rodean los actos didácticos y que el profesor procurará que favorezcan al máximo los aprendizajes.

En muchos materiales didácticos multimedia directivos (ejercitación, tutoriales) subyace esta perspectiva.

- Constructivismo. J. Piaget, en sus estudios sobre epistemología genética, en los que determina las principales fases en el desarrollo cognitivo de los niños, elaboró un modelo explicativo del desarrollo de la inteligencia y del aprendizaje en general a partir de la consideración de la adaptación de los individuos al medio.

Considera tres estadios de desarrollo cognitivo universales: sensoriomotor, estadio de las operaciones concretas y estadio de las operaciones formales. En todos ellos la actividad es un factor importante para el desarrollo de la inteligencia.

- Construcción del propio conocimiento mediante la interacción constante con el medio. Lo que se puede aprender en cada momento depende de la propia capacidad cognitiva, de los conocimientos previos y de las interacciones que se

pueden establecer con el medio. En cualquier caso, los estudiantes comprenden mejor cuando están envueltos en tareas y temas que cautivan su atención. El profesor es un mediador y su metodología debe promover el cuestionamiento de las cosas, la investigación, etc.

- Reconstrucción de los esquemas de conocimiento. El desarrollo y el aprendizaje se produce a partir de la secuencia*: equilibrio - desequilibrio – reequilibrio* (que supone una adaptación y la construcción de nuevos esquemas de conocimiento).

Aprender no significa ni reemplazar un punto de vista (el incorrecto) por otro (el correcto), ni simplemente acumular nuevo conocimiento sobre el viejo, sino más bien transformar el conocimiento. Esta transformación, a su vez, ocurre a través del pensamiento activo y original del aprendiz. Así pues, la educación constructivista implica la experimentación y la resolución de problemas y considera que los errores no son antitéticos del aprendizaje sino más bien la base del mismo.

El constructivismo considera que *el aprendizaje es una interpretación personal del mundo* (el conocimiento no es independiente del alumno), de manera que da sentido a las experiencias que construye cada estudiante. Este conocimiento se consensúa con otros, con la sociedad.

Esta perspectiva actualmente está presente en muchos materiales didácticos multimedia de todo tipo, especialmente en los no tutoriales.

- Socio-constructivismo. Basado en muchas de las ideas de Vigotski, considera también los aprendizajes como un proceso personal de construcción de nuevos conocimientos a partir de los conocimientos previos (actividad instrumental), pero inseparable de la situación en la que se produce. Tiene lugar conectando con la experiencia personal y el conocimiento base del estudiante y se sitúa en un contexto social donde él construye su propio conocimiento a través de la interacción con otras personas (a menudo con la orientación del docente). Enfatiza en los siguientes aspectos:
    - ❖ Importancia de la interacción social y de compartir y debatir con otros los aprendizajes. Aprender es una experiencia social donde el contexto es muy importantes y el lenguaje juega un papel básico como herramienta mediadora, no solo entre profesores y alumnos, sino también entre estudiantes, que así aprenden a explicar, argumentar... Aprender significa "aprender con otros", recoger también sus puntos de vista. La socialización se va realizando con "otros" (iguales o expertos).

- ❖ Incidencia en la zona de desarrollo próximo, en la que la interacción con los especialistas y con los iguales puede ofrecer un "andamiaje" donde el aprendiz puede apoyarse.

Actualmente el aprendizaje colaborativo y el aprendizaje situado, que destaca que todo aprendizaje tiene lugar en un contexto en el que los participantes negocian los significados, recogen estos planteamientos. El aula debe ser un campo de interacción de ideas, representaciones y valores. La interpretación es personal, de manera que no hay una realidad compartida de conocimientos. Por ello, los alumnos individualmente obtienen diferentes interpretaciones de los mismos materiales, cada uno construye (reconstruye) su conocimiento según sus esquemas, sus conocimientos y experiencias previas su contexto.

Esta perspectiva actualmente está presente en algunos materiales didácticos multimedia no tutoriales.

Otras clasificaciones. Además de considerar la "estructura", los materiales didácticos multimedia se pueden clasificar según múltiples criterios:

- ➢ Según los contenidos (temas, áreas curriculares...)
- ➢ Según los destinatarios (criterios basados en niveles educativos, edad, conocimientos previos...)
- ➢ Según los medios que integra: convencional, hipertexto, multimedia, hipermedia, realidad virtual.
- ➢ Según su "inteligencia": convencional, experto (o con inteligencia artificial).
- ➢ Según los objetivos educativos que pretende facilitar: conceptuales, procedimentales, actitudinales (o considerando otras taxonomías de objetivos).
- ➢ Según las actividades cognitivas que activa: control psicomotriz, observación, memorización, evocación, comprensión, interpretación, comparación, relación (clasificación, ordenación), análisis, síntesis, cálculo, razonamiento (deductivo, inductivo, crítico), pensamiento divergente, imaginación, resolución de problemas, expresión (verbal, escrita, gráfica…), creación, exploración, experimentación, reflexión metacognitiva, valoración, etc.
- ➢ Según el tipo de interacción que propicia: recognitiva, reconstructiva, intuitiva/global, constructiva *(Kemmis)*
- ➢ Según su función en el aprendizaje: instructivo, revelador, conjetural, emancipador. *(Hooper y Rusbhi)*
- ➢ Según su comportamiento tutor, herramienta, aprendiz. *(Taylor)*

- Según el tratamiento de errores: tutorial (controla el trabajo del estudiante y le corrige), no tutorial.
- Según sus bases psicopedagógicas sobre el aprendizaje: conductista, cognitivista, constructivista *(Begoña Gros)*
- Según su función en la estrategia didáctica: entrenar, instruir, informar, motivar, explorar, experimentar, expresarse, comunicarse, entretener, evaluar, proveer recursos (calculadora, comunicación telemática)
- Según su diseño: centrado en el aprendizaje, centrado en la enseñanza, proveedor de recursos. *(Hinostroza, Mellar, Rehbein, Hepp, Preston)*
- Según el soporte: disco, web

1.2.4.1 Programas Tutoriales

Son programas que en mayor o menor medida dirigen, autorizan, el trabajo de los alumnos. Pretenden que, a partir de unas informaciones y mediante la realización de ciertas actividades previstas de antemano, los estudiantes pongan en juego determinadas capacidades, y aprendan o refuercen conocimientos y habilidades. Estos tienen cuatro clasificaciones:

- Programas lineales, que presentan al alumno una secuencia de información ó ejercicios (siempre la misma o determinada aleatoriamente) con independencia de la corrección o incorrección de sus respuestas. Herederos de la enseñanza programada, transforman el ordenador en una máquina de enseñar transmisora de conocimientos y adiestradora de habilidades. No obstante, su interactividad resulta pobre y el programa se hace largo de recorrer.
- Programas ramificados, basados inicialmente también en modelos conductistas, siguen recorridos pedagógicos diferentes según el juicio que hace el ordenador sobre la corrección de las respuestas de los alumnos o según su decisión de profundizar más en ciertos temas. Ofrecen mayor interacción, más opciones, pero la organización de la materia suele estar menos compartimentada que en los programas lineales y exigen un esfuerzo más grande al alumno. Pertenecen a éste grupo los programas multinivel, que estructuran los contenidos en niveles de dificultad y previenen diversos caminos, y los programas ramificados con dientes de sierra, que establecen una diferenciación entre los conceptos y las preguntas de profundización, que son opcionales.

- Entornos tutoriales. En general están inspirados en modelos pedagógicos cognitivitas, y proporcionan a los alumnos una serie de herramientas de búsqueda y de proceso de la información que pueden utilizar libremente para construir la respuesta a las preguntas del programa. Este es el caso de los entornos de resolución de problemas, "problem solving", donde los estudiantes conocen parcialmente las informaciones necesarias para su resolución y han de buscar la información que falta y aplicar reglas, leyes y operaciones para encontrar la solución.
- Sistemas tutoriales expertos, como los Sistemas Tutores Inteligentes (Intelligent Tutoring Systems), que, elaborados con las técnicas de la Inteligencia Artificial y teniendo en cuenta las teorías cognitivas sobre el aprendizaje, tienden a reproducir un diálogo auténtico entre el programa y el estudiante, y pretenden comportarse como lo haría un tutor humano: guían a los alumnos paso a paso en su proceso de aprendizaje, analizan su estilo de aprender y sus errores y proporcionan en cada caso la explicación o ejercicio más conveniente.

1.2.4.2 Simuladores

Presentan un modelo o entorno dinámico (generalmente a través de gráficos o animaciones interactivas) y facilitan su exploración y modificación a los alumnos, que pueden realizar aprendizajes inductivos o deductivos mediante la observación y la manipulación de la estructura subyacente; de esta manera pueden descubrir los elementos del modelo, sus interrelaciones, y pueden tomar decisiones y adquirir experiencia directa delante de unas situaciones que frecuentemente resultarían difícilmente accesibles a la realidad (control de una central nuclear, contracción del tiempo, pilotaje de un avión...).

También se pueden considerar simulaciones ciertos videojuegos que, al margen de otras consideraciones sobre los valores que incorporan (generalmente no muy positivos) facilitan el desarrollo de los reflejos, la percepción visual y la coordinación psicomotriz en general, además de estimular la capacidad de interpretación y de reacción ante un medio concreto.

Al igual que la anterior esta tiene dos clasificaciones:

- Modelos físico-matemáticos: Presentan de manera numérica o gráfica una realidad que tiene unas leyes representadas por un sistema de ecuaciones deterministas. Se incluyen aquí los programas-laboratorio, algunos trazadores de funciones y los programas que mediante un convertidor analógico-digital captan datos analógicos de un fenómeno externo al ordenador y presentan en pantalla un modelo del fenómeno estudiado o informaciones y gráficos que van asociados. Estos programas a veces son utilizados por profesores delante de la clase a manera de pizarra electrónica, como demostración o para ilustrar un concepto, facilitando así la transmisión de información a los alumnos, que después podrán repasar el tema interactuando con el programa.
- Entornos sociales: Presentan una realidad regida por unas leyes no del todo deterministas. Se incluyen aquí los juegos de estrategia y de aventura, que exigen una estrategia cambiante a lo largo del tiempo.

1.2.4.3 Programas de Ejercitación o entrenadores

Su finalidad es que el estudiante practique mediante una repetición de preguntas y ejercicios. Responden a la necesidad de aprender destrezas específicas sencillas. Los entrenadores se diseñan con diferentes niveles de complejidad, en dependencia del fin que se persiga con el mismo (de aplicación reproductiva o productiva). Su principal objetivo es la adquisición por parte del estudiante de habilidades que lo conduzcan implícitamente a la reafirmación o consolidación de conocimientos.

Durante el proceso de diseño de los programas de ejercitación deben tomarse decisiones en torno al nivel, contenido y estructura de las tareas a realizar. También debe decidirse el control del progreso en función del número de aciertos obtenidos en cada nivel.

Se puede decir que en el presente trabajo se propone un software educativo que clasifica como un programa de ejercitación, de acuerdo a las características que se han concebido para la asignatura de Contabilidad y Finanzas

Este mismo autor da diversos criterios de concepción de aprendizaje estos se presenta a continuación:

- Conductismo: Las principales características que presentas es la formación de reflejos condicionados mediante mecanismos de estímulo- respuesta, ensayo y error con refuerzos y repetición, ley del efecto o del resultado de la acción Memorización mecánica. Los principales programas donde subyace esta perspectiva son los programas tutoriales y ejercitación.
- Aprendizaje por descubrimiento: se presenta como experimentación directa sobre la realidad, aplicación práctica de los conocimientos y su transferencia a diversas situaciones.
- Aprendizaje por penetración comprensiva: El alumno experimentando descubre y comprende lo que es relevante, las estructuras.
- Práctica de la inducción: de lo concreto a lo abstracto, de los hechos a las teorías. Utilización de estrategias heurísticas: pensamiento divergente.
- Currículum en espiral: revisión y ampliación periódica de los conocimientos adquiridos.

Dentro de esta se encuentran los simuladores y constructors.

- Aprendizaje significativo:
- Relación con las estructuras cognitivas previas y funcionalidad
- Utilización de organizadores previos.
- Diferenciación-reconciliación integradora que genera una memorización comprensiva.

Todos los software educativo tratan de promover aprendizajes significativos.

Cognitivismo: Consideración de diversas etapas en el proceso de aprendizaje.

Consideración de las interacciones: estudiante - sistema simbólico de los medios.

Estos se encuentran fundamentalmente en los programas tutoriales

Constructivismo: Las principales características son construcción del propio conocimiento mediante la interacción constante con el medio. Equilibrio - desequilibrio - reequilibrio: adaptación y construcción de nuevos esquemas de conocimiento Atención al desarrollo cognitivo. Estos se presentan en los programas tutoriales.

1.2.5 Funciones del Software Multimedia Educativos

Los materiales multimedia educativos, como los materiales didácticos en general, pueden realizar múltiples funciones en los procesos de enseñanza y aprendizaje. Las principales funciones que pueden realizar los recursos educativos multimedia son las siguientes [9]:

- Informativa: Estos materiales, a través de sus actividades, presentan unos contenidos que proporcionan información estructurada de la realidad, a los estudiantes.
- Instructiva Entrenadora: Todos los software educativos orientan y regulan el aprendizaje de los estudiantes ya que, explícita o implícitamente, promueven determinadas actuaciones de los mismos encaminadas a este fin. Además, mediante sus códigos simbólicos, estructuración de la información e interactividad condicionan los procesos de aprendizaje.
- Motivadora: La interacción con el ordenador suele resultar por sí misma motivadora.

Estos programas incluyen además elementos para captar la atención de los alumnos, mantener su interés y focalizarlo hacia los aspectos más importantes.

- Evaluadora: La posibilidad de "feed back" inmediato a las respuestas y acciones de los alumnos, hace adecuados a los programas para evaluarles. Esta evaluación puede ser:
    - Implícita: el estudiante detecta sus errores, se evalúa a partir de las respuestas que le da el ordenador.
    - Explícita: el programa presenta informes valorando la actuación del alumno.
- Explorar Experimentar: Les presentan a los estudiantes interesantes entornos donde explorar, experimentar, investigar, buscar determinadas informaciones, cambiar los valores de las variables de un sistema, etc.
- Innovadora: Los programas educativos pueden desempeñar esta función ya que utilizan una tecnología actual y, en general, suelen permitir muy diversas formas de uso. Esta versatilidad abre amplias posibilidades de experimentación didáctica e innovación educativa en el aula.

1.2.6 Características de los buenos Software Educativos

Los buenos Software Educativos formativos son eficaces, facilitan el logro de sus objetivos, y ello es debido, supuesto un buen uso por parte de los estudiantes y profesores, a una serie de características que atienden a diversos aspectos funcionales, técnicos y pedagógicos, y que se comentan a continuación:

1.- **Facilidad de uso e instalación**. Con el abaratamiento de los precios de los ordenadores y el creciente reconocimiento de sus ventajas por parte grandes sectores de la población, para que los programas puedan ser realmente utilizados por la mayoría de las personas es necesario que sean agradables, fáciles de usar y autoexplicativos, de manera que los usuarios puedan utilizarlos inmediatamente sin tener que realizar una exhaustiva lectura de los manuales ni largas tareas previas de configuración.

En cada momento el usuario debe conocer el lugar del programa donde se encuentra y tener la posibilidad de moverse según sus preferencias: retroceder, avanzar... Un sistema de ayuda on-line solucionará las dudas que puedan surgir.

Por supuesto la instalación del programa en el ordenador también será sencilla, rápida y transparente. También será de apreciar la existencia de una utilidad desinstaladora para cuando llegue el momento de quitar el programa del ordenador.

2.- **Versatilidad (adaptación a diversos contextos)**. Otra buena característica de los programas, desde la perspectiva de su funcionalidad, es que sean fácilmente integrables con otros medios didácticos en los diferentes contextos formativos, pudiéndose adaptar a diversos:

- Entornos (aula de informática, clase con un único ordenador, uso doméstico...)
- Estrategias didácticas (trabajo individual, grupo cooperativo o competitivo,,,)
- Usuarios (circunstancias culturales y necesidades formativas)

Para lograr esta versatilidad conviene que tengan unas características que permitan su adaptación a los distintos contextos. Por ejemplo:

- Que sean programables, que permitan la modificación de algunos parámetros: grado de dificultad, tiempo para las respuestas, número de usuarios simultáneos, idioma, etc.

- Que sean abiertos, permitiendo la modificación de los contenidos de las bases de datos.
- Que incluyan un sistema de evaluación y seguimiento (control) con informes de las actividades realizadas por los estudiantes: temas, nivel de dificultad, tiempo invertido, errores, itinerarios seguidos para resolver los problemas...)
- Que permitan continuar los trabajos empezados con anterioridad.
- Que promuevan el uso de otros materiales (fichas, diccionarios...) y la realización de actividades complementarias (individuales y en grupo cooperativo)

3.- **Calidad del entorno audiovisual.** El atractivo de un programa depende en gran manera de su entorno comunicativo. Algunos de los aspectos que, en este sentido, deben cuidarse más son los siguientes:

- Diseño general claro y atractivo de las pantallas, sin exceso de texto y que resalte a simple vista los hechos notables.
- Calidad técnica y estética en sus elementos:
- Títulos, menús, ventanas, iconos, botones, espacios de texto-imagen, formularios, barras de navegación, barras de estado, elementos hipertextuales, fondo...
- Elementos multimedia: gráficos, fotografías, animaciones, vídeos, voz, música…
- Estilo y lenguaje, tipografía, color, composición, metáforas del entorno…
- Adecuada integración de medias, al servicio del aprendizaje, sin sobrecargar la pantalla, bien distribuidas, con armonía.

4.- **La calidad en los contenidos (bases de datos).** Al margen de otras consideraciones pedagógicas sobre la selección y estructuración de los contenidos según las características de los usuarios, hay que tener en cuenta las siguientes cuestiones:

- La información que se presenta es correcta y actual, se presenta bien estructurada diferenciando adecuadamente: datos objetivos, opiniones y elementos fantásticos.
- Los textos no tienen faltas de ortografía y la construcción de las frases es correcta

- No hay discriminaciones. Los contenidos y los mensajes no son negativos ni tendenciosos y no hacen discriminaciones por razón de sexo, clase social, raza, religión y creencias...
- La presentación y la documentación.

5.- **Navegación e interacción**. Los sistemas de navegación y la forma de gestionar las interacciones con los usuarios determinarán en gran medida su facilidad de uso y amigabilidad. Conviene tener en cuenta los siguientes aspectos:

- Mapa de navegación. Buena estructuración del programa que permite acceder bien a los contenidos, actividades, niveles y prestaciones en general.
- Sistema de navegación. Entorno transparente que permite que el usuario tenga el control. Eficaz pero sin llamar la atención sobre si mismo. Puede ser: lineal, paralelo, ramificado...
- La velocidad entre el usuario y el programa (animaciones, lectura de datos…) resulta adecuada.
- El uso del teclado. Los caracteres escritos se ven en la pantalla y pueden corregirse errores.
- El análisis de respuestas. Que sea avanzado y, por ejemplo, ignore diferencias no significativas (espacios superfluos...) entre lo tecleado por el usuario y las respuestas esperadas.
- La gestión de preguntas, respuestas y acciones...
- Ejecución del programa. La ejecución del programa es fiable, no tiene errores de funcionamiento y detecta la ausencia de los periféricos necesarios.

6.- **Originalidad y uso de tecnología avanzada**. Resulta también deseable que los programas presenten entornos originales, bien diferenciados de otros materiales didácticos, y que utilicen las crecientes potencialidades del ordenador y de las tecnologías multimedia e hipertexto en general, yuxtaponiendo dos o más sistemas simbólicos, de manera que el ordenador resulte intrínsecamente potenciador del proceso de aprendizaje, favorezca la asociación de ideas y la creatividad, permita la práctica de nuevas técnicas, la reducción del tiempo y del esfuerzo necesarios para aprender y facilite aprendizajes más completos y significativos.

La inversión financiera, intelectual y metodológica que supone elaborar un programa educativo sólo se justifica si el ordenador mejora lo que ya existe.

7.- **Capacidad de motivación**. Para que el aprendizaje significativo se realice es necesario que el contenido sea potencialmente significativo para el estudiante y que éste tenga la voluntad de aprender significativamente, relacionando los nuevos contenidos con el conocimiento almacenado en sus esquemas mentales.

Así, para motivar al estudiante en este sentido, las actividades de los programas deben despertar y mantener la curiosidad y el interés de los usuarios hacia la temática de su contenido, sin provocar ansiedad y evitando que los elementos lúdicos interfieren negativamente en los aprendizajes. También conviene que atraigan a los profesores y les animen a utilizarlos.

8.- Adecuación a los usuarios y a su ritmo de trabajo. Los buenos programas tienen en cuenta las características iniciales de los estudiantes a los que van dirigidos (desarrollo cognitivo, capacidades, intereses, necesidades…) y los progresos que vayan realizando. Cada sujeto construye sus conocimientos sobre los esquemas cognitivos que ya posee, y utilizando determinadas técnicas.

Esta adecuación se manifestará en tres ámbitos principales:

- Contenidos: extensión, estructura y profundidad, vocabulario, estructuras gramaticales, ejemplos, simulaciones y gráficos… Los contenidos deben ser significativos para los estudiantes y estar relacionados con situaciones y problemas de su interés.
- Actividades: tipo de interacción, duración, elementos motivacionales, mensajes de corrección de errores y de ayuda, niveles de dificultad, itinerarios, progresión y profundidad de los contenidos según los aprendizajes realizados (algunos programas tienen un pre-test para determinar los conocimientos iniciales de los usuarios)….
- Entorno de comunicación: pantallas, sistema de navegación, mapa de navegación...

9.- **Potencialidad de los recursos didácticos**. Los buenos programas multimedia utilizan potentes recursos didácticos para facilitar los aprendizajes de sus usuarios. Entre estos recursos se pueden destacar:

- Proponer diversos tipos de actividades que permitan diversas formas de utilización y de acercamiento al conocimiento.
- Utilizar organizadores previos al introducir los temas, síntesis, resúmenes y esquemas.
- Emplear diversos códigos comunicativos: usar códigos verbales (su construcción es convencional y requieren un gran esfuerzo de abstracción) y códigos icónicos (que muestran representaciones más intuitivas y cercanas a la realidad)
- Incluir preguntas para orientar la relación de los nuevos conocimientos con los conocimientos anteriores de los estudiantes.
- Tutorización las acciones de los estudiantes, orientando su actividad, prestando ayuda cuando lo necesitan y suministrando refuerzos

10.- **Fomento de la iniciativa y el autoaprendizaje**. Las actividades de los programas educativos deben potenciar el desarrollo de la iniciativa y el aprendizaje autónomo de los usuarios, proporcionando herramientas cognitivas para que los estudiantes hagan el máximo uso de su potencial de aprendizaje, puedan decidir las tareas a realizar, la forma de llevarlas a cabo, el nivel de profundidad de los temas y puedan autocontrolar su trabajo.

En este sentido, facilitarán el aprendizaje a partir de los errores (empleo de estrategias de ensayo-error) tutorizando las acciones de los estudiantes, explicando (y no sólo mostrando) los errores que van cometiendo (o los resultados de sus acciones) y proporcionando las oportunas ayudas y refuerzos.

Además estimularán el desarrollo de habilidades metacognitivas y estrategias de aprendizaje en los usuarios, que les permitirán planificar, regular y evaluar su propia actividad de aprendizaje, provocando la reflexión sobre su conocimiento y sobre los métodos que utilizan al pensar.

11.- **Enfoque pedagógico actual**. El aprendizaje es un proceso activo en el que el sujeto tiene que realizar una serie de actividades para asimilar los contenidos informativos que recibe. Según repita, reproduzca o relacione los conocimientos, realizará un aprendizaje repetitivo, reproductivo o significativo.

Las actividades de los programas convienen que estén en consonancia con las tendencias pedagógicas actuales, para que su uso en las aulas y demás entornos educativos provoque un cambio metodológico en este sentido.

Por lo tanto los programas evitarán la simple memorización y presentarán entornos heurísticos centrados en los estudiantes que tengan en cuenta las teorías constructivistas y los principios del aprendizaje significativo donde además de comprender los contenidos puedan investigar y buscar nuevas relaciones. Así el estudiante se sentirá constructor de sus aprendizajes mediante la interacción con el entorno que le proporciona el programa (mediador) y a través de la reorganización de sus esquemas de conocimiento.

Ya que aprender significativamente supone modificar los propios esquemas de conocimiento, reestructurar, revisar, ampliar y enriquecer las estructura cognitivas.

12. - **La documentación**. Aunque los programas sean fáciles de utilizar y autoexplicativos, conviene que tengan una información que informe detalladamente de sus características, forma de uso y posibilidades didácticas. Esta documentación (on-line o en papel) debe tener una presentación agradable, con textos bien legibles y adecuados a sus destinatarios, y resultar útil, clara, suficiente y sencilla. Podemos distinguir tres partes:

- Ficha resumen, con las características básicas del programa.
- El manual del usuario. Presenta el programa, informa sobre su instalación y explica sus objetivos, contenidos, destinatarios, modelo de aprendizaje que propone..., así como sus opciones y funcionalidades. También sugiere la realización de diversas actividades complementarias y el uso de otros materiales.
- La guía didáctica con sugerencias didácticas y ejemplos de utilización que propone estrategias de uso y indicaciones para su integración curricular. Puede incluir fichas de actividades complementarias, test de evaluación y bibliografía relativa del contenido.

13.- **Esfuerzo cognitivo**. Las actividades de los programas, contextualizadas a partir de los conocimientos previos e intereses de los estudiantes, deben facilitar aprendizajes significativos y transferibles a otras situaciones mediante una continua actividad mental en consonancia con la naturaleza de los aprendizajes que se pretenden.

Así desarrollarán las capacidades y las estructuras mentales de los estudiantes y sus formas de representación del conocimiento (categorías, secuencias, redes conceptuales, representaciones visuales...) mediante el ejercicio de actividades cognitivas del tipo: control psicomotriz, memorizar, comprender, comparar, relacionar, calcular, analizar, sintetizar, razonamiento (deductivo, inductivo, crítico), pensamiento divergente, imaginar, resolver problemas, expresión (verbal, escrita, gráfica...), crear, experimentar, explorar, reflexión metacognitiva (reflexión sobre su conocimiento y los métodos que utilizan al pensar y aprender).

1.2.7 Ventajas de los software educativos

- Enriquece el campo de la Pedagogía al incorporar la tecnología de punta que revoluciona los métodos de enseñanza - aprendizaje.
- Constituyen una nueva, atractiva, dinámica y rica fuente de conocimientos.
- Pueden adaptar el software a las características y necesidades de su grupo teniendo en cuenta el diagnóstico en el proceso de enseñanza - aprendizaje.
- Permiten elevar la calidad del proceso docente - educativo.
- Permiten controlar las tareas docentes de forma individual o colectiva.
- Muestran la interdisciplinariedad de las asignaturas.
- Marca las posibilidades para una nueva clase más desarrolladora.

1.2.8 Desventajas de los Software Educativos

- Adicción: El software educativo interactivo resulta motivador, pero un exceso de motivación puede provocar adicción.
- Distracción: Los alumnos a veces se dedican a jugar en vez de trabajar.
- Ansiedad: La continua interacción ante el ordenador puede provocar ansiedad en los estudiantes.

- Aprendizajes incompletos y superficiales: La libre interacción de los alumnos con estos materiales (no siempre de calidad) a menudo proporciona aprendizajes incompletos con visiones de la realidad simplista y poco profunda.
- Cansancio visual y otros problemas físicos: Un exceso de tiempo trabajando ante el ordenador o malas posturas pueden provocar diversas dolencias.
- Problemas con los ordenadores: A veces los alumnos desconfiguran o contaminan con virus los ordenadores.

1.3 Aplicaciones Multimedia

En el universo audio visual donde vive el hombre en las sociedades desarrolladas modernas, las técnicas multimedia se convierten cada día en un instrumento eficaz de comunicación y acceso a la información. La implementación de las capacidades multimedia en las computadoras es sólo el último episodio de una larga serie de avances en el desarrollo de la humanidad, que reflejan el deseo innato del hombre de crear herramientas para expresarse creativamente, de comunicarse más poderosamente y liberar ideas: pinturas rupestres, textos, manuscritos, imprenta, radio y televisión. [10]

La hipermedia surge como resultado de la fusión de dos tecnologías, el hipertexto y la multimedia. El hipertexto es la organización de una determinada información en diferentes nodos, conectados entre sí a través de enlaces. Los nodos pueden contener subelementos con entidad propia. Un hiperdocumento estaría formado por un conjunto de nodos conectados y relacionados temática y estructuralmente. La tecnología multimedia es la que permite integrar diferentes medios (sonido, imágenes, secuencias...) en una misma presentación. La hipermedia, por tanto, es la tecnología que permite estructurar la información de una manera no-secuencial, a través de nodos interconectados por enlaces. La información presentada en estos nodos podrá integrar diferentes medios tales como: Texto, sonido, gráficos, video, etc.

A partir de esta descripción se puede afirmar de forma sintética que una obra hipermedia es una obra de comunicación audiovisual interactiva, la cual no es sólo un producto informático cuya realización es totalmente tecnológica e ingenieril, sino también una labor creativa en la que se integran elementos estéticos y funcionales capaces de actuar sobre las emociones humanas de forma intensa, como pueden ser la literatura, el cine o la televisión, potenciada a su vez por las posibilidades interactivas y de multiplicidad de

canales de comunicación que posee tanto para la cultura como en la educación de las personas.

Estas hipermedias y multimedia pretenden resolver el problema del procesamiento lineal de la información por el receptor, como ocurre en el libro de texto. Por el contrario, la información se puede construir desde diferentes trayectorias y alternativas, y con diferentes tipos de códigos. Estas trayectorias pueden limitarse por el autor del programa, para evitar problemas de desorientación en el usuario [13], pudiendo soportar la autoría de documentos complejos y el procesamiento de ideas, especialmente en entornos de trabajo colaborativos.

1.3.1 Ventajas de las Aplicaciones Multimedia

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

1.3.2 Desventajas de las Aplicaciones Multimedia

- La desorientación: incapacidad del usuario para controlar la información en un inextricable espacio interconectado. Éste problema está íntimamente ligado al diseño del hiperdocumento y las soluciones pasan por el uso de metáforas en sustitución de los iconos, uso de índices, posibilidad de vuelta atrás, uso de mapas de situación y navegadores gráficos.
- La sobrecarga: se ha comprobado que comprender y utilizar las técnicas de recuperación de información en un hiperdocumento puede suponer un gran esfuerzo para el usuario. Las soluciones para minimizar éste problema pasa por el uso de un interfaz lo más intuitivo posible, huir de cualquier tipo de exceso como el

empleo masivo e innecesario de elementos multimedia así como la generación sin sentido de enlaces.

1.4 Lenguaje de Modelado

En todas las disciplinas de la Ingeniería se hace evidente la importancia de los modelos ya que describen el aspecto y la conducta de "algo". Ese "algo" puede existir, estar en un estado de desarrollo o estar, todavía, en un estado de planeación. Es en este momento cuando los diseñadores del modelo deben investigar los requerimientos del producto terminado y dichos requerimientos pueden incluir áreas tales como funcionalidad, performance y confiabilidad. Además, a menudo, el modelo es dividido en un número de vistas, cada una de las cuales describe un aspecto específico del producto o sistema en construcción.

El modelado sirve no solamente para los grandes sistemas, aun en aplicaciones de pequeño tamaño se obtienen beneficios de modelado, sin embargo es un hecho que entre más grande y más complejo es el sistema, más importante es el papel de que juega el modelado por una simple razón: "El hombre hace modelos de sistemas complejos porque no puede entenderlos en su totalidad".

1.4.1 Lenguaje de Modelado de Objetos

En los últimos años, la construcción de modelos orientados a objetivos 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

El lenguaje de modelado de objetos es un conjunto estandarizado de símbolos y de modos de disponerlos para modelar (parte de) un diseño de software orientado a objetos.

Algunas organizaciones los usan extensivamente en combinación con una metodología de desarrollo de software para avanzar de una especificación inicial a un plan de implementación y para comunicar dicho plan a todo un equipo de desarrolladores. El uso de un lenguaje de modelado es más sencillo que la auténtica programación, pues existen menos medios para verificar efectivamente el funcionamiento adecuado del modelo. Esto

puede suponer también que las interacciones entre partes del programa den lugar a sorpresas cuando el modelo ha sido convertido en un software funcional.

Algunos metodólogos del software orientado a objetos distinguen tres grandes "generaciones" cronológicas de técnicas de modelado de objetos. [11]

- En la primera generación, tecnólogos aislados y grupos pequeños desarrollaban técnicas que resolvían problemas que se encontraban de primera mano en los proyectos de desarrollo orientado a objetos. En esta generación se incluye a autores y técnicas como Rumbaugh, Jacobson, Booch, los métodos formales, Shlaer-Mellor y Yourdon-Coad.
- En la segunda generación se reconoció que muchas de las mejores prácticas pertenecían a diferentes métodos del fragmentado terreno de la metodología orientada a objetos. Se realizaron múltiples intentos para integrar dichas técnicas en marcos coherentes tales como FUSION. En cualquier caso, la comunidad del software orientado a objetos empezaba a reconocer los beneficios que la standarización de las técnicas conllevaría: abandono de las "buenas" formas de hacer las cosas en favor de "la" manera adecuada, que permitiría un lenguaje y unas prácticas comunes entre los diferentes desarrolladores.
- La tercera generación consiste en intentos creíbles de crear dicho lenguaje unificado por la industria, cuyo mejor ejemplo es UML

1.5 Lenguaje Unificado de Modelado (UML) y RUP

El Proceso Unificado ha adoptado un enfoque que se caracteriza por:

- Interacción con el usuario continua desde un inicio
- Mitigación de riesgos antes de que ocurran
- Liberaciones frecuentes
- Aseguramiento de la calidad
- Involucramiento del equipo en todas las decisiones del proyecto
- Anticiparse al cambio de requerimientos

El Proceso Unificado y MSF se enfocan en la arquitectura como el centro del desarrollo para asegurar que el desarrollo basado en componentes sea clave para un alto nivel de reuso. MSF considera que hay cuatro **perspectivas de arquitectura** que cumplen los requerimientos de una empresa [M&R 1998]:

- **Arquitectura de Negocios** - Describe como opera un negocio. Desarrolla una imagen clara de los procesos de flujo de trabajo de la organización y de cómo son apoyados por una infraestructura tecnológica basada en servicios.
- **Arquitectura de Aplicación –** Adopta un modelo de aplicación de toda la empresa para diseñar y desarrollar sistemas de negocios que puedan compartir un conjunto de componentes back-end de alto valor.
- **Arquitectura de Información –** Define qué información es necesaria para apoyar el proceso de negocios y como poner esa información eficientemente en manos de quienes que la necesitan sin crear islas de datos inaccesibles ni sistemas redundantes.
- **Arquitectura Tecnológica –** Define los estándares y guías para la adquisición y despliegue de herramientas, bloques de construcción de aplicaciones, servicios de infraestructura, componentes de conectividad de red y plataformas cliente servidor.

El Modelo de Equipo de MSF muestra como estructurar equipos de alto desempeño para crear soluciones más rápido, mejor y más baratas. El Modelo de Equipo de MSF se basa en las formas de organizar equipos para crear los propios productos de Microsoft. Hace énfasis en las comunicaciones claras y en un equipo de iguales con papeles y responsabilidades claras en todo el proyecto. La calidad del producto se asegura por cada miembro del equipo. El Proceso Unificado proporciona más detalle y guía para algunos de los roles en el proyecto.

Una **vista arquitectónica** es "una descripción simplificada (una abstracción) de un sistema desde una perspectiva particular o punto de vista, que cubre particularidades y omite entidades que no son relevantes a esta perspectiva" [Booch 1998].

Según el mismo autor, las **características primordiales del Proceso Unificado** son:

- Iterativo e incremental
- Centrado en la arquitectura

- Guiado por casos de uso
- Confrontación de riesgos

El Proceso Unificado es un proceso porque "define quién está haciendo qué, cuándo lo hacer y cómo alcanzar cierto objetivo, en este caso el desarrollo de software" [Jacobson 1998]. Según [Booch 1998], los **conceptos clave del Proceso Unificado** son:

Fases e Iteraciones	¿Cuándo se hace?
Flujos de trabajo de procesos (actividades y pasos)	¿Qué se está haciendo?
Artefactos (modelos, reportes, documentos)	¿Qué se produjo?
Trabajador: un arquitecto	¿Quién lo hace?

UML es una técnica para la especificación sistemas en todas sus fases. Nació en 1994 cubriendo los aspectos principales de todos los métodos de diseño antecesores y, precisamente, los padres de UML son Grady Booch, autor del método Booch; James Rumbaugh, autor del método OMT e Ivar Jacobson, autor de los métodos OOSE y Objectory. La versión 1.0 de UML fue liberada en Enero de 1997 y ha sido utilizado con éxito en sistemas construidos para toda clase de industrias alrededor del mundo: hospitales, bancos, comunicaciones, aeronáutica, finanzas, etc.[12]

1.5.1 Principales beneficios de UML

- Mejores tiempos totales de desarrollo (de 50 % o más).
- Modelar sistemas (y no sólo de software) utilizando conceptos orientados a objetos.
- Establecer conceptos y artefactos ejecutables.
- Encaminar el desarrollo del escalamiento en sistemas complejos de misión crítica.
- Crear un lenguaje de modelado utilizado tanto por humanos como por máquinas.
- Mejor soporte a la planeación y al control de proyectos.
- Alta reutilización y minimización de costos.

1.5.2 UML ¿Método o Lenguaje de Modelado?

UML es un lenguaje para hacer modelos y es independiente de los métodos de análisis y diseño. Existen diferencias importantes entre un método y un lenguaje de modelado. Un método es una manera explícita de estructurar el pensamiento y las acciones de cada individuo. Además, el método le dice al usuario qué hacer, cómo hacerlo, cuándo hacerlo y por qué hacerlo; mientras que el lenguaje de modelado carece de estas instrucciones. Los métodos contienen modelos y esos modelos son utilizados para describir algo y comunicar los resultados del uso del método.

Un modelo es expresado en un lenguaje de modelado. Un lenguaje de modelado consiste de vistas, diagramas, elementos de modelo ¾ los símbolos utilizados en los modelos ¾ y un conjunto de mecanismos generales o reglas que indican cómo utilizar los elementos. Las reglas son sintácticas, semánticas y pragmáticas. (Ver Fig. A)

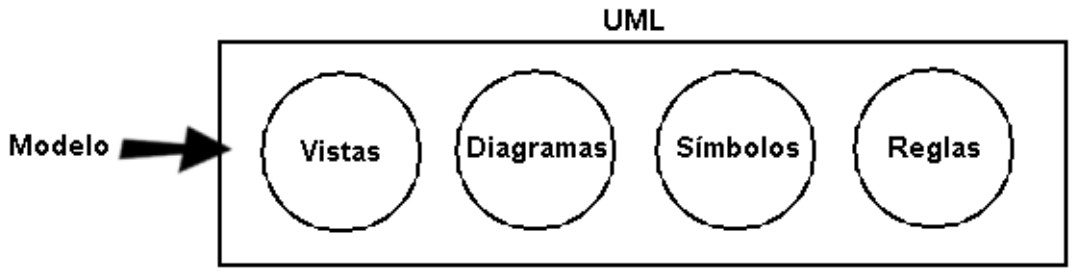


Fig. A

Vistas: Las vistas muestran diferentes aspectos del sistema modelado. Una vista no es una gráfica, pero sí una abstracción que consiste en un número de diagramas y todos esos diagramas juntos muestran una "fotografía" completa del sistema. Las vistas también ligan el lenguaje de modelado a los métodos o procesos elegidos para el desarrollo. Las diferentes vistas que UML tiene son:

- Vista Use-Case: Una vista que muestra la funcionalidad del sistema como la perciben los actores externos.
- Vista Lógica: Muestra cómo se diseña la funcionalidad dentro del sistema, en términos de la estructura estática y la conducta dinámica del sistema.
- Vista de Componentes: Muestra la organización de los componentes de código.

- Vista Concurrente: Muestra la concurrencia en el sistema, direccionando los problemas con la comunicación y sincronización que están presentes en un sistema concurrente.
- Vista de Distribución: muestra la distribución del sistema en la arquitectura física con computadoras y dispositivos llamados nodos.

Diagramas: Los diagramas son las gráficas que describen el contenido de una vista. UML tiene nueve tipos de diagramas que son utilizados en combinación para proveer todas las vistas de un sistema: diagramas de caso de uso, de clases, de objetos, de estados, de secuencia, de colaboración, de actividad, de componentes y de distribución.

Símbolos o Elementos de modelo: Los conceptos utilizados en los diagramas son los elementos de modelo que representan conceptos comunes orientados a objetos, tales como clases, objetos y mensajes, y las relaciones entre estos conceptos incluyendo la asociación, dependencia y generalización. Un elemento de modelo es utilizado en varios diagramas diferentes, pero siempre tiene el mismo significado y simbología.

Reglas o Mecanismos generales: Proveen comentarios extras, información o semántica acerca del elemento de modelo; además proveen mecanismos de extensión para adaptar o extender UML a un método o proceso específico, organización o usuario.

1.6 Lenguaje de modelado i*

i* permite expresar de forma clara y sencilla los objetivos de los actores que aparecen en los modelos y las dependencias entre ellos. Además, i* cuenta con una notación gráfica que permite tener una visión intuitiva y unificada del entorno modelado mostrando tales actores y dependencias. Esta notación fue propuesta por Eric Yu en la primera mitad de la década de los 90. [4]

En i* se propone el uso de dos modelos, cada uno correspondiente a un nivel de abstracción: el nivel intencional representado por el Strategic Dependency Model (SD) y el nivel racional representado por el Strategic Rationale Model (SR). El modelo SD consiste en un conjunto de nodos que representan actores (actors) y un conjunto de relaciones entre estos actores. Las relaciones representan dependencias (dependencies) que permiten expresar que un actor (depender) depende de otro actor (dependee) para conseguir un determinado propósito (dependum). El dependum es un elemento

intencional que puede ser de tipo: recurso (resource), tarea a realizar (task), objetivo (goal) o requisito no funcional (softgoal). El modelo permite definir la importancia (strength) de la dependencia para cada uno de los actores que intervienen en ella. Existen tres grados de importancia: abierta (open), comprometida (commited) o crítica (critical). [4]

El modelo SR permite visualizar los elementos intencionales dentro de los límites (boundary) de un actor para refinar el modelo SD y añadirle capacidad de razonamiento. Para ello, se asignan las relaciones de dependencia del SD a elementos intencionales internos al actor y se añaden relaciones medio-fin (means-end) y descomposición de tarea (task-decomposition):

- Una relación means-end se establece entre dos elementos intencionales de un mismo actor. Uno de los elementos intencionales de la relación es el medio (generalmente una tarea) que contribuye a la realización de un fin. Cuando existen varios medios para un determinado fin existe una relación OR, la cual indica las diferentes maneras de lograr un fin. Las relaciones posibles son: Goal-Task, Resource-Task, Task-Task, Softgoal-Task, Softgoal-Softgoal y Goal-Goal. En las relaciones means-end en las que aparece un softgoal como fin se indica si la contribución del medio a la consecución del softgoal es positiva o negativa.
- Una relación task-decomposition descompone una tarea en elementos intencionales de cualquier tipo. Cuando son varios los elementos intencionales que descomponen una tarea, existe una relación AND entre ellos, pudiéndose completar con restricciones (constraints) para refinar esta relación. Finalmente, se puede marcar la importancia del elemento intencional en la realización de la tarea, de la misma forma que en las dependencias del modelo SD.

Los modelos SR constan de elementos de razonamiento adicionales como son las rutinas (routines), reglas (rules) y creencias (beliefs). Una rutina es una de las opciones que aparecen en una relación means-end e indica un determinado curso de acción a seguir. Se deben considerar las reglas y creencias como información complementaria de condiciones de aplicabilidad de dichas rutinas.

El análisis de [14] ha mostrado diversas situaciones anómalas debidas principalmente a que en varios puntos de la documentación se hace referencia a una especificación en TELOS [13], que en realidad está incompleta. Esta falta de formalización ha hecho

necesario un estudio intenso de las descripciones textuales y de los gráficos para completar ciertos aspectos y ha conducido a las siguientes observaciones:

- **Ruido**. El concepto de rutina se define en la formalización y descripción del modelo SD. Este hecho induce a confusión, ya que en realidad su uso se hace visible como elemento de razonamiento únicamente en el modelo SR. Por otro lado, la relación IS-A (generalización/especialización) no forma parte del lenguaje pero se utiliza profusamente en los gráficos como una simplificación de la notación.
- **Silencio**. Se han detectado las siguientes situaciones: no se indica si se permite más de una raíz en la descomposición interna de un actor; no está explícito si cualquier tipo de elemento intencional puede ser raíz de una descomposición; no se especifica si un actor puede ser parte-de varios actores; no se detalla si un dependum puede estar relacionado con más de un dependee y/o más de un depender; la formalización y descripción de constraints de las relaciones taskdecomposition es incompleta; finalmente, a pesar de que se dan definiciones de los distintos tipos de nodos (actores y elementos intencionales), únicamente se pueden deducir sus criterios de utilización de los ejemplos incluidos en el texto.
- **Ambigüedad**. La importancia (strengh) de la dependencia se interpreta de forma diferente según se encuentre en el extremo del depender o del dependee, dando a entender que una misma dependencia puede tener diferente importancia para actor participante. Sin embargo, no existe ningún ejemplo que lo clarifique.

1.7 GRL (Goal-oriented Requirement Language)

GRL (Goal-oriented Requirement Language) es un lenguaje para apoyar el

modelado orientado a objetivos y de razonamientos con requisitos no funcionales. Este lenguaje tiene una fuerte influencia de i* así como también del NFR framework para la especificación de los requisitos no funcionales [14]. GRL forma parte de la notación URN (User Requirements Notation) [16], que ha sido propuesta como estándar de la ITU-T (International Telecommunication Union –Telecommunication Standardization Sector) [15]. GRL distingue, al igual que i*, tres principales categorías de conceptos: elementos intencionales, relaciones intencionales y actores (que no admiten especializaciones). Las principales diferencias respecto a i* son que GRL ofrece constructores para establecer

relaciones con elementos externos al modelo (elementos no-intencionales y atributos de conexión), y que GRL posee elementos adicionales de justificación y/o contextualización como creencias (beliefs), correlaciones, tipos de contribuciones y etiquetas de evaluación para especificar los estados de satisfacción, ampliando así los tipos y rango de calificaciones de las relaciones intencionales existentes en i*.

Las observaciones derivadas del resultado del análisis de GRL según los criterios establecidos, son las siguientes:

- **Ruido**. La existencia de una triple especificación sintáctica (BNF gráfica, BNF textual y especificación XML), impide una certeza inmediata de que las fórmulas estén bien formadas para GRL e incrementa innecesariamente el esfuerzo de comprensión de esta variante.
- **Silencio**. La especificación sintáctica permite una variedad de fórmulas que no son cubiertas totalmente en las explicaciones realizadas en lenguaje natural.
- **Ambigüedad**. La especificación sintáctica formal, acompañada de explicaciones concisas y ejemplos sencillos, evita ambigüedades en la construcción de expresiones GRL. Sin embargo, existen ambigüedades semánticas respecto a las contribuciones, dándose el caso de que existen contribuciones que tienen una calificación referida a operadores binarios (AND, OR), pero que permiten su construcción con un solo operando.
- **Contradicción**. Se ha detectado una contradicción entre las especificaciones sintácticas de GRL: la BNF textual determina un conjunto delimitado de valores para tipos de contribuciones con 11 elementos terminales; sin embargo, la especificación XML correspondiente, especifica como elemento terminal un tipo de dato básico (CDATA), lo cual no necesariamente correspondería a un valor dentro del conjunto delimitado de tipos. Otra contradicción aparece al estudiar la herramienta OME [16] para la edición de modelos GRL, donde se observa que se permite usar la especialización de roles, agentes y posiciones, a pesar de que la especificación formal de GRL no hace esta consideración.

1.8 Conclusiones Parciales

Este capitulo tuvo como objetivo fundamental dar una explicación a cerca de los principales fundamentos teóricos tratados a lo largo del desarrollo del presente trabajo. Para ello se ha llevado a cabo una investigación donde se abordaron diferentes temas a cerca de los procesos de Ingeniería de Software, además se trataron conceptos tales como el de software e ingeniería de software, así como las principales características de este proceso en la rama educativa, se realizo un profundo análisis a cerca de las diferentes tipologías y clasificaciones de los software educativos, así como los beneficios y las ventajas que brindan estos a los diferentes usuarios de los productos de SWE. Pudimos conocer algunas de las características de las aplicaciones multimedia así como los beneficios e inconvenientes que estas ofrecen.

Se trato el tema de los lenguajes de modelado ya que es la cuestión que nos ocupa. Abordamos diferentes aspectos de la notación UML, ya que fue a partir de esta que surge OMMMA-L, como extensión de la misma para incorporar aspectos a la hora de modelar aplicaciones Multimedia.

CAPÍTULO II

Fundamentación Tecnológica

A veces se trabaja toda la vida
para avanzar un metro
Pero así crecieron las grandes montañas,
Avanzando siglo a siglo,
Metro a metro

2.1 OMMMA-L

Han sido varios los lenguajes de modelado que se han propuestos para la especificación del proceso de desarrollo de aplicaciones multimedia, aunque todavía no existe un estándar que cubra todos los aspectos relacionados con el comportamiento dinámico e interactivo asociado a las interfaces gráficas para una generalización de herramientas, productos y procesos.

Siguiendo esta búsqueda de una modelación adecuada, el Lenguaje de Modelado Orientado a objetos de Aplicaciones Multimedia (OMMMA - L) se lanza como una propuesta de extensión de UML para la integración de especificaciones de sistemas multimedia basados en el paradigma orientado a objetos, y MVC (Modelo Vista Controlador) para la interfaz de usuario.

OMMMA-L (Object-oriented Modeling of MultiMedia Applications – the Language) captura las características de la aplicación representadas en las diversas vistas y deriva la pragmática de cómo desarrollar aplicaciones multimedia con un lenguaje orientado a objetos basado en UML. [17]

Las extensiones de OMMMA-L se pueden integrar con UML por los mecanismos incorporados de extensión de UML que permite que los elementos del modelo existentes sean especializados por estereotipos, restricciones, y valores marcados con etiqueta. Estas extensiones ligeras no influencian la sintaxis y la semántica del UML en sí mismo, pero la semántica se puede especializar para las extensiones específicas del dominio. Las extensiones se pueden entonces utilizar para construir los perfiles para los dominios de aplicación o las clases de aplicaciones específicas. OMMMA-L es presentado en las subdivisiones siguientes, comenzando por introducir un caso de estudio ejemplo que se modelará.

UML está diseñado a través de un lenguaje de diagramas y artefactos fácilmente ajustables para especificar aspectos distintivos de un sistema a modelar. Se agrupan en cuatro categorías, diagramas de caso de uso, estructurales, de comportamiento e implementación, siendo el segundo y el tercero quienes interactúan directamente con las descripciones de los modelos estáticos estructurales y de comportamiento dinámicos identificados anteriormente. Para OMMMA – L podemos 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. Por último, la distribución espacial de media contemplada en el modelo vista, puede ser descrita a través de un nuevo artefacto propuesto para el lenguaje, el diagrama de presentación. La semántica asociada a dichos diagramas, conservan en muchos casos su significado, en otras se adaptan a la interpretación de los conceptos propios de multimedia. [17]

UML ofrece varios diagramas para modelar el comportamiento de una aplicación, dado el énfasis que muestran en modelar restricciones de tiempo los diagramas de secuencia se destacan en OMMMA – L para modelar el comportamiento temporal predefinido de una aplicación multimedia. Antes, es necesario extenderlos para reflejar características tales como:

- El perfeccionamiento del eje de tiempo mediante la introducción de marcas de tiempo así como diferentes formas de medirlo, interpretarlo y adaptarlo.
- La parametrización de diagramas de secuencia, para diferenciar su funcionamiento entre los establecimientos de sincronización temporal y el tradicional paso de mensajes.
- Esperas de activación y desactivación para el manejo de la sincronización entre medias.
- Activación compuesta de objetos media para modelar concurrencia de objetos activos.

En esencia estos diagramas modelarán una secuencia de presentación predefinida dentro de una escena, permitiendo la modelación de concurrencias de varias medias, mensajes sincronizados y asíncronos, restricciones de tiempo y duración de la ejecución de una multimedia.

OMMMA-L no presenta cambios con respecto a UML en el flujo de requisitos y casos de uso. Pero en el modelo de clases de objetos que es con lo que se crea el modelo del dominio si. [17]

En el Diagrama de clases del modelo de objeto se agregan 3 conceptos. Un objeto puede ser del tipo:

- escenario: cuando representa un conjunto de pantallas que muestran una información a través de objetos con similar funcionalidad.
- aplicación: cuando agrupa elementos de media y aúna sus funcionalidades como una entidad.
- media: cuando se hace referencia a sonido, texto, imágenes, animaciones, video, botones, etc.

OMMMA-L contiene variaciones con respecto a UML solo en las fases de **Análisis** y **Diseño** manteniéndose normal en los otros flujos. Por tanto no presenta variaciones en el siguiente flujo de trabajo de **Implementación**, ni en el de **Pruebas**.

La notación OMMMA-L, permite el desarrollo de sistemas multimedia de manera ágil y eficiente. La misma cuenta con los artefactos necesarios para desarrollar un proceso completo y con todos los elementos necesarios para que el mismo cumpla con los elementos componentes de un sistema multimedia y satisfaga las necesidades del cliente. Esta notación es apropiada para sistemas multimedia de pequeña y mediana magnitud.

Actualmente, OMMMA – L se evalúa en diferentes escenarios, como proyectos industriales para la especificación de servicios de información multimedia, y se investigan características adicionales de sincronía para su especificación en el lenguaje y la formalización de un modelo para la composición dentro y entre los diferentes diagramas de comportamiento.

2.2 Vistas que sustentan OMMMA-L

OMMMA-L está sustentado en cuatro vistas fundamentales, donde cada una se asocia a un tipo de diagrama en particular.

- **Vista Lógica**: modelada a través del Diagrama de Clases de OMMMA-L, extendido del Diagrama de Clases de UML, utilizando las mismas notaciones, pero incorporando las clases correspondientes a las medias: media continua y media discreta, generalizadas en una clase medias.
- **Vista de Presentación espacial**: modelada a través de los Diagramas de Presentación de OMMMA-L, los cuales son de nueva aparición en la extensión de UML, dado que este último no contiene un diagrama apropiado para esta tarea.
- **Vista de Comportamiento temporal predefinido**: modelada por el Diagrama de Secuencia de OMMMA-L, extendido a partir del diagrama de secuencia de UML.

- **Vista de Control Interactivo**: modelado a través del Diagrama de Estado, extendido a partir del diagrama de estado de UML.

2.3 Diagramas que conforman OMMMA-L

2.3.1 Diagramas de Clases (Fase de Análisis)

EL Diagrama de Clases del Análisis: utiliza las mismas notaciones que el Diagrama de Clases de UML, pero incorporando las clases correspondientes a las medias. Este debe reflejar la correspondencia con las medias.

Los diagramas de clases en OMMMA-L consisten en clases y definición de asociaciones que describen la estructura de los objetos y sus posibles relaciones estructurales. Como en UML los rasgos para definir los diagramas de clases son suficientemente expresivos se han incorporado de forma inalterada al lenguaje OMMMA-L. Cada diagrama de clases consiste en (al menos) dos partes estrechamente interrelacionadas:

- una jerarquía de definiciones de tipo de medios de comunicación que comprenden las clases para todos (la representación) los tipos de los medios de comunicación; y
- el modelo lógico de una aplicación que comprende clases y asociaciones que describe el dominio de la aplicación objeta y sus interrelaciones.

Los dos aspectos se unen por asociaciones que interrelacionan los objetos de la aplicación con los objetos de los medios de comunicación correspondientes. Para la especificación de interacción, estas jerarquías de la clase deben acompañarse por una jerarquía señalada como una base para la interacción evento-base. Pueden desplegarse las clases de la presentación (posiblemente en paquetes diferentes) además del modelo la posible composición de interfaz del usuario como una base para los diagramas de la presentación.

El sistema de navegación contiene mapas múltiples que pueden asociarse con un número sin restricción de rutas. A su vez, pueden relacionarse las rutas al múltiplo de mapas. Para cada ruta, hay una salida y una situación del destino. Una ruta relaciona a un juego de direcciones que están calificados de la perspectiva de una ruta por un número de la parte que define su posición en la sucesión de direcciones. Para algunos de éstos las clases de la aplicación, las asociaciones a los elementos de los medios de comunicación clasifican la jerarquía se muestra.

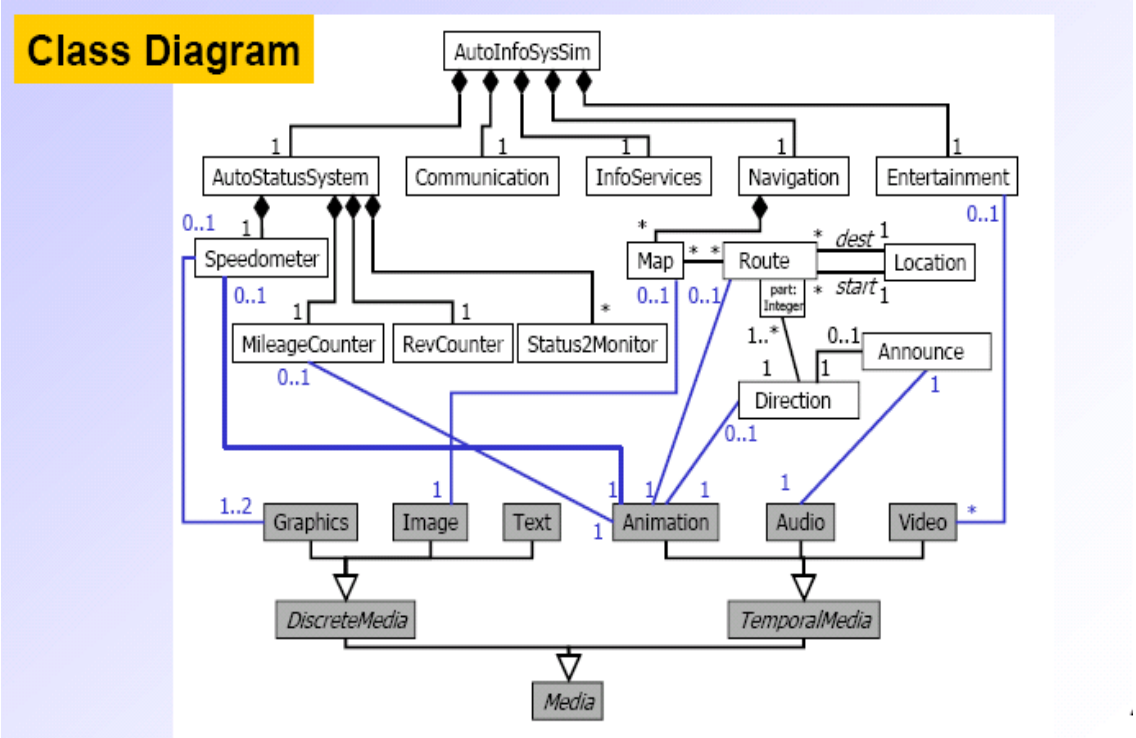


Diagrama de Clases

2.3.2 Diagramas de Presentación (Fase de Análisis)

OMMMA-L presenta un nuevo diagrama. El Diagrama de Presentación, sirve 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. . Estos diagrama tienen el propósito de declarar las interfaces de usuario con un conjunto de estructuras delimitadas en tamaño y área, dividiéndose en objetos de visualización (texto, gráfico, video, animación) e interacción (scrolls, barras de menú, botones, campos de entrada y salida, hipertextos con hipervínculos). Estos diagramas de presentación pueden ser divididos en capas virtuales de presentación donde en cada uno de ellas sólo se haga referencia a una clase específica de componentes (por ejemplo, una vista para los objetos de visualización y otra para los de interacción, u otro tipo de división para la representación de los intereses de los desarrolladores).

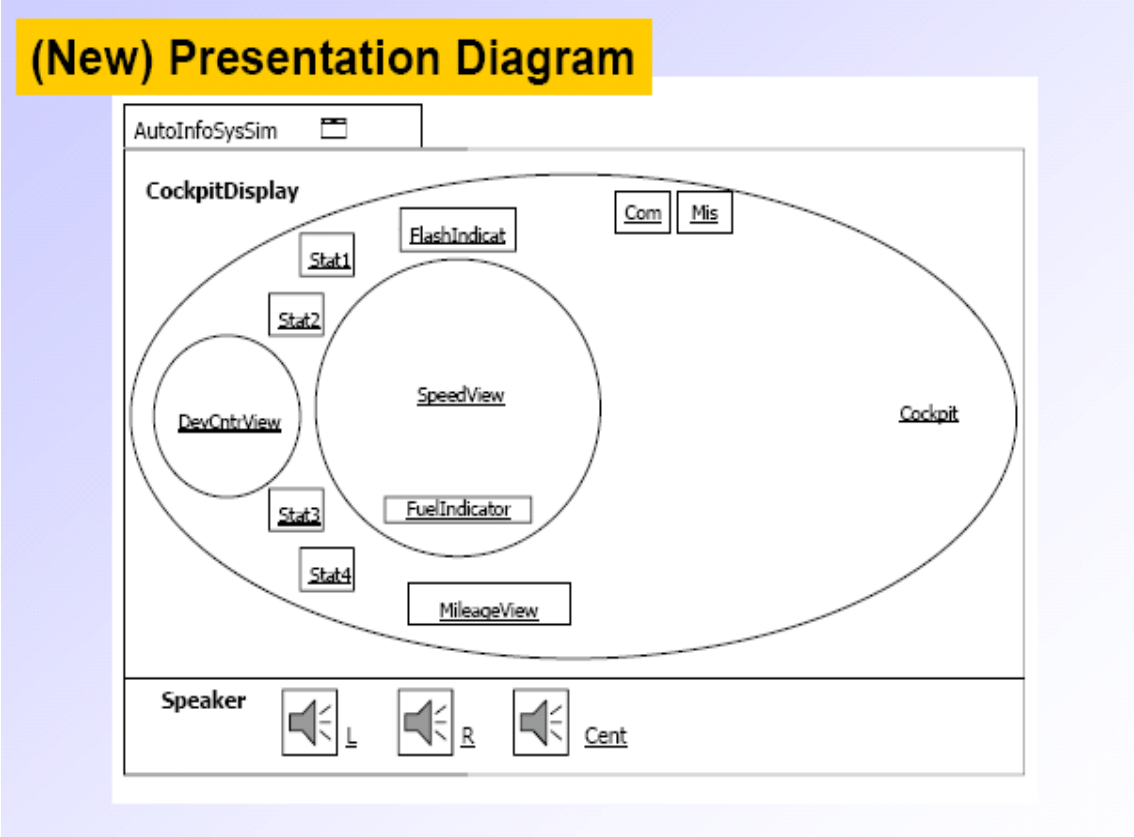


Diagrama de Presentación.

2.3.3 Diagramas de Sucesión Extendidos (Fase de Diseño)

UML ofrece varios tipos del diagrama para modelar aspectos conductuales de una aplicación.

Debido a su énfasis en modelar las sucesiones temporales (de mensajes), sucesión, se realizan los diagramas de despliegue en OMMMA-L para planear (predefinidamente) la conducta temporal de una aplicación multimedia. Pero para poder planear específicamente las características de una aplicación multimedia más directamente y así más intuitivamente, los diagramas de sucesión UML normales están extendidos por una serie de rasgos, considerando especialmente el tiempo de constreñimiento y cronometración.

Éstos son por ejemplo:

- El Refinamiento de la dimensión de tiempo definiendo las hachas de tiempo locales para los objetos apoyando una noción de tiempo local. El tiempo local puede relacionarse al tiempo global (real, representado por el horario del actor) para especificar el intra-objeto sincronización o al tiempo de otros objetos para especificar la sincronización del inter-objeto. Las

duraciones y puntos en el tiempo pueden especificarse por diferentes formularios de fijo, limites o intervalos de tiempo ilimitados que restringen las posiciones temporales. Los intervalos de Tiempo son representados por su salida y los puntos del fin. (Sintácticamente, estos requisitos cronometrando pueden escribirse como constreñimientos usando igualdades o en una anotación del intervalo.)

- Barras de Sincronización (las líneas intrépidas) en lugar de las flechas del mensaje entre objetos las activaciones para especificar la sincronización del inter-objeto continúa entre presentaciones de los medios de comunicación temporales que pueden resumir de una dirección del mensaje.
- La activación y la desactivación de objetos multimedia ordenados para modelar toleradas variaciones de sincronización relacionadas por objetos multimedia (comparando los tiempos máximos de inicio y fin)
- Una noción de Historia poco profunda y profunda en un diagrama de sucesión es que UML sólo mantiene los diagramas de estado. Le permite al diseñador especificar si un guión especificado puede interrumpirse y después puede volverse al mismo punto de tiempo virtual dónde había sido interrumpido. La historia profunda (H *) denota que esta reconstrucción temporal incluso es posible adelante. La invocación nivela con la semántica de pausa y reasume la presentación compleja, aunque la historia poco profunda (H) restringe volviendo al la situación en el nivel de la cima con la semántica permite que diagrama de sucesión se reinicie en el tiempo de la interrupción.
- Paralelamente la activación compuesta de objetos multimedia para planear la simultánea presentación de un objeto de la aplicación por la presentación diferente los cauces (o medias) y/o por los objetos de los medios de comunicación diferentes activaciones secuencialmente compuestas que producen una activación automático de las subsecuentes (los segmentos), por ejemplo para presentar un objeto animado que se presenta secuencialmente los cauces diferentes y/o por los diferentes objetos multimedia.
- Pueden anotarse activaciones de objetos de la aplicación con los objetos de la presentación, cualquier abstracciones de los dispositivos del hardware (media de presentación) o los software usuario interfaz objetos - dependiendo del nivel de uso - como canales de audio o los objetos gráficos en una pantalla, y/o objetos de los medios de comunicación

designados para representar un objeto de la aplicación durante una activación o segmento de activación. Asociado a los objetos multimedia, que deben conformar a los tipos especificado en el diagrama de clases, que están adjuntos a el. Los identificadores de objetos de presentación aparecen como los puros cordones (como ellos son usados en diagramas de presentación).

- Las activaciones de objetos pueden estar encima del layed por filtros multimedia que describen las funciones de tiempo, por ejemplo el aumento incremental de un nivel de audio con el tiempo. Cada diagrama de sucesión de OMMMA-L modela la conducta temporal de un guión predefinido de la aplicación multimedia.

El guión especificado por el diagrama de la sucesión se representa por el (inicial) el mensaje envió de un actor simbólico (o algún usuario une el componente) a un objeto dentro de la sucesión diagrama que actúa como el director del guión. El mensaje puede ser los parametrizado por ejemplo por las estampas de tiempo para la salida y fin de ejecución de un diagrama de la sucesión, para apoyar su re-uso, o por parámetros que pueden usarse en guardia expresiones o anidó las llamadas del mensaje.

2.3.4 Diagramas de Estado (Fase de Diseño)

Diagrama de Estado extendido a partir del diagrama de estado de UML, sin tácticamente igual a este último, mas con la diferencia semántica de que en el orden de unir los controles interactivos y predefinidos, no interrumpidos de los objetos, las acciones internas de estados simples tienen que llevar nombres de diagrama de secuencia en vez de diagramas de estado empotrados; queriendo esto decir que el comportamiento especificado por el diagrama de secuencia se provoca automáticamente cuando se entra al estado correspondiente donde se hace referencia.

Cada objeto escenario será representado a través de un estado con su mismo nombre, un mensaje de cambio de estado representará una interacción del usuario o de objetos que alteren el comportamiento del sistema. Los estados compuestos son detallados en nuevas máquinas de estados o empotrados en sí mismos y se representan con un color más oscuro. Una acción interna de un estado simple enruta hacia un diagrama de secuencia de comportamiento temporal. Un estado atómico, siempre que lo amerite, es descrito también por este diagrama.

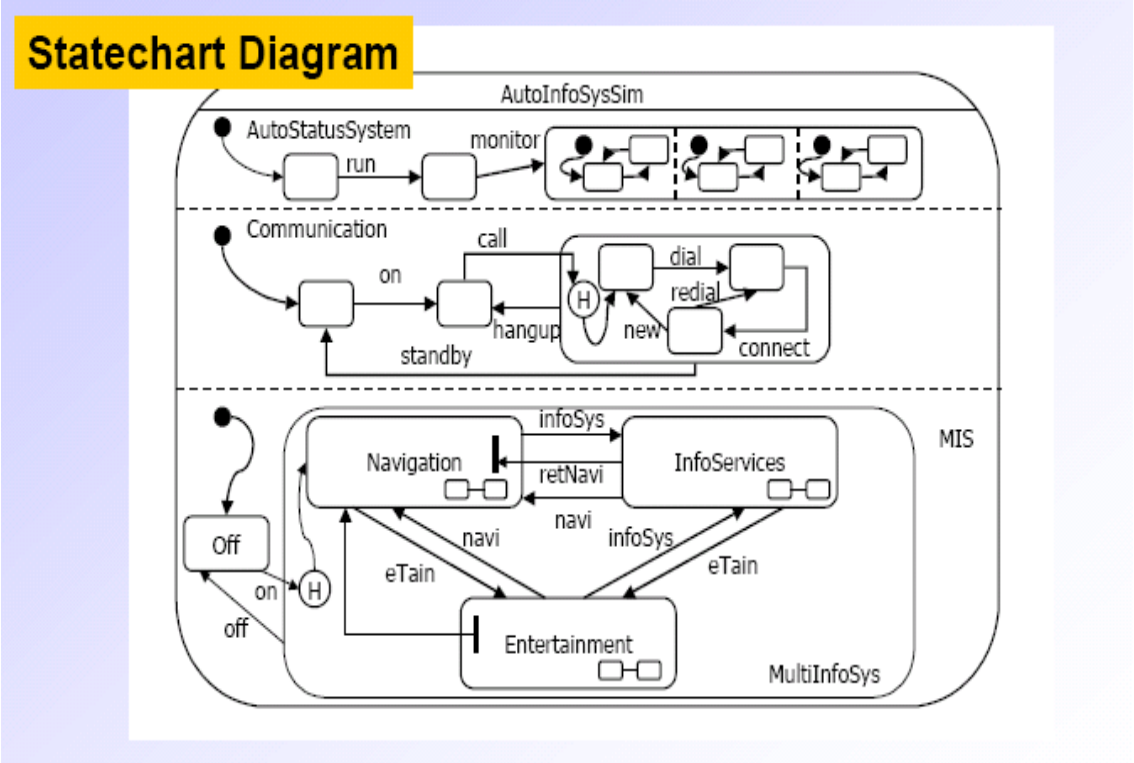


Diagrama de Estado.

2.3.5 Diagrama de Secuencia Extendido

Diagrama de secuencia: extendido a partir del diagrama de secuencia de UML. El Diagrama de secuencia modela una secuencia de una presentación predefinida dentro de una escena, donde todos los objetos dentro de un diagrama se relacionan al mismo eje del tiempo. En este diagrama se hace un refinamiento del eje del tiempo con la introducción de marcas de tiempo a través de diferentes tipos de intervalos; marcas de inicio y fin de ejecución que permite soportar su reusabilidad; marcas de activación y desactivación de demoras en objetos de tipo media, posibilitando la modelación de las tolerancias de la variación de las restricciones de sincronización para los objetos media; activación compuesta de objetos media para la agrupación de objetos concurrentemente activos.

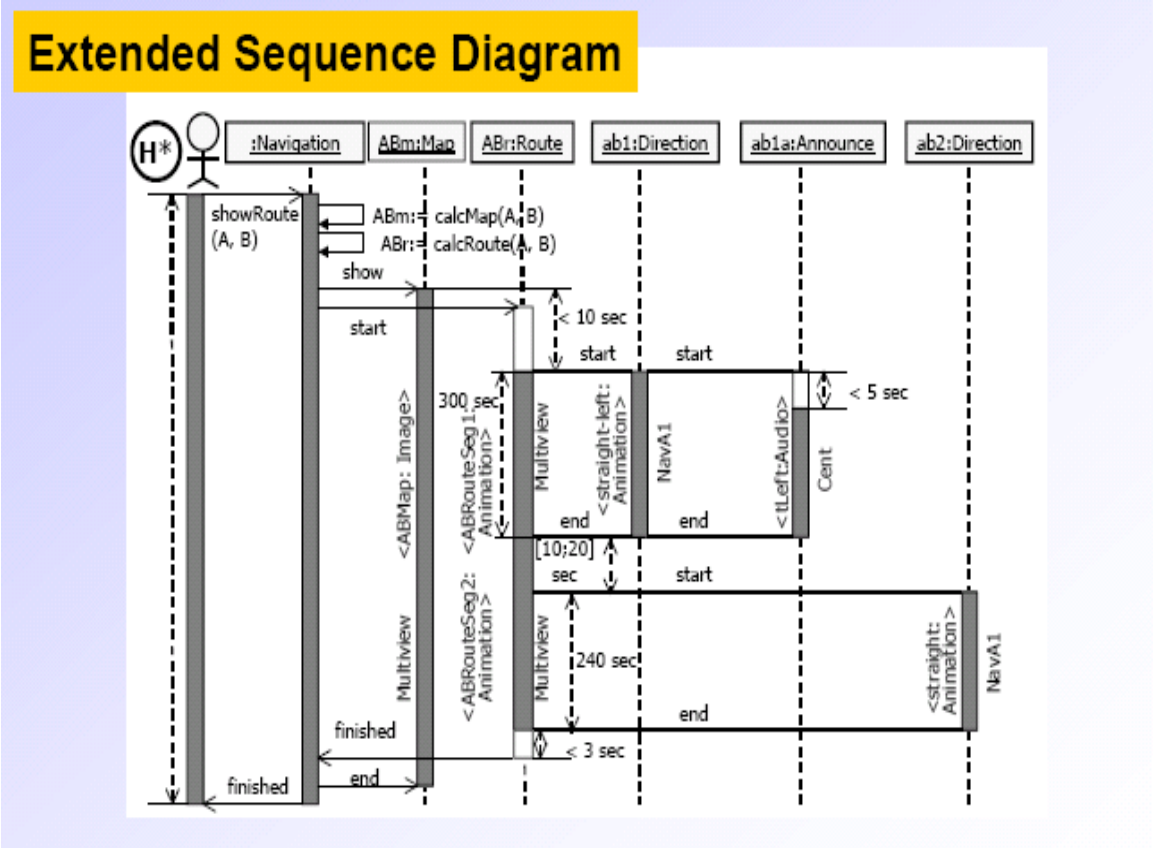


Diagrama de Secuencia Extendido

2.4 El Modelo Vista Controlador (MVC)

La arquitectura del software alude a "la estructura global del software y a las formas en que la estructura proporciona la integridad conceptual de un sistema". En su forma más simple, la arquitectura es la estructura jerárquica de los componentes del programa (módulos), la manera en que los componentes interactúan y la estructura de datos que van a utilizar los componentes. Sin embargo, en un sentido más amplio, los "componentes" se pueden generalizar para presentar los elementos principales del sistema y sus interacciones.

El diseño arquitectónico define la relación entre los elementos estructurales principales del software, los patrones de diseño que se pueden utilizar para lograr los requisitos que se han definido para el sistema, y las restricciones que afectan a la manera en que se pueden aplicar los patrones de diseño arquitectónicos.

MVC divide una aplicación interactiva en 3 áreas: procesamiento, salida y entrada. Para esto, utiliza las siguientes abstracciones:

- **Modelo** (Model): Encapsula los datos y las funcionalidades. El modelo es independiente de cualquier representación de salida y/o comportamiento de entrada.

- **Vista** (View): Muestra la información al usuario. Pueden existir múltiples vistas del modelo. Cada vista tiene asociado un componente controlador.
- **Controlador** (Controller): Reciben las entradas, usualmente como eventos que codifican los movimientos o pulsación de botones del ratón, pulsaciones de teclas, etc.

Aunque se pueden encontrar diferentes implementaciones de MVC, el flujo que sigue el control generalmente es el siguiente:

1. El usuario interactúa con la interfaz de usuario de alguna forma (por ejemplo, el usuario pulsa un botón, enlace)
2. El controlador recibe (por parte de los objetos de la interfaz-vista) la notificación de la acción solicitada por el usuario. El controlador gestiona el evento que llega, frecuentemente a través de un gestor de eventos (handler) o callback.
3. El controlador accede al modelo, actualizándolo, posiblemente modificándolo de forma adecuada a la acción solicitada por el usuario. Los controladores complejos están a menudo estructurados usando un patrón de comando que encapsula las acciones y simplifica su extensión.
4. El controlador delega a los objetos de la vista la tarea de desplegar la interfaz de usuario. La vista obtiene sus datos del modelo para generar la interfaz apropiada para el usuario donde se refleja los cambios en el modelo (si se utiliza la variante 2 descrita anteriormente, de lo contrario lo obtiene a través del Controlador). El modelo no debe tener conocimiento directo sobre la vista. Sin embargo, el patrón de observador puede ser utilizado para proveer cierta indirección entre el modelo y la vista, permitiendo al modelo notificar a los interesados de cualquier cambio. Un objeto vista puede registrarse con el modelo y esperar a los cambios, pero aun así el modelo en sí mismo sigue sin saber nada de la vista. El controlador no pasa objetos de dominio (el modelo) a la vista aunque puede dar la orden a la vista para que se actualice. Nota: *En algunas implementaciones la vista no tiene acceso directo al modelo, dejando que el controlador envíe los datos del modelo a la vista, según lo descrito en la segunda variante.*
5. La interfaz de usuario espera nuevas interacciones del usuario, comenzando el ciclo nuevamente.

2.4.1 El MVC extendido

El MVC es un patrón de diseño de software que distingue un componente modelo sosteniendo la funcionalidad del núcleo y los datos, un componente vista para mostrar la información al usuario y un componente controlador para manipular los eventos de interacción. Separando así los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos de forma que las modificaciones al componente de la vista pueden ser hechas con un mínimo impacto en el componente del modelo de datos. Un mecanismo de propagación de cambios asegura la consistencia entre el modelo y la interfaz visual. Extendiendo el paradigma MVC para multimedia a las peculiaridades de comportamiento estático y dinámico identificadas anteriormente, obtenemos MVCmm, sobre el que se basa las especificaciones de OMMMA – L (Ver Fig. 4).

2.5 Conclusiones parciales

Una vez vista las características de OMMMA – L, se argumenta su aplicación exitosa partiendo de la idea de que no es un lenguaje nuevo, sino una extensión del UML por lo que solo es necesario interpretar las características extendidas, centrados a la lógica de funcionamiento de una multimedia, que es por lo general, sencilla. No se especializa en una clasificación de producto, sino que generaliza a través del uso de la semántica original de UML. Es robusto y altamente descriptivo, refleja el proceso en todas sus etapas y hereda de RUP el ciclo de vida basado en iteraciones y el flujo de trabajo iterativo e incremental, centrado en casos de uso y en la arquitectura.

Además se pudieron abordar los aspectos fundamentales del Modelo Vista Controlador y su extensión para aplicaciones multimedia, sus componentes y tipos de implementaciones.

CAPITULO III

Aplicación de la notación OMMMA-L al caso de estudio.

Solo alcanza la grandeza
Quien cuida de los pequeños detalles

José M. Gironella

En este capítulo se aplica la notación OMMMA-L a un caso de estudio, que corresponde a una Hipermedia que tendrá carácter educativo y se utilizará como herramienta de apoyo en la enseñanza de la asignatura de Topografía de la Carrera de Ingeniería Civil en la Facultad de Ingeniería Civil del Centro Universitario José Antonio Echeverría (CUJAE). Para dar solución a la aplicación de la metodología al caso de estudio planteado se procede a realizar la captura de los requisitos funcionales y no funcionales del sistema, se obtienen y describen los casos de uso que guiarán la solución propuesta, centrándose en el Proceso Unificado de Desarrollo de Software (RUP), y utilizando el Lenguaje Unificado de Modelado (UML) extendido con el Lenguaje de Modelado Orientado a Objetos de Aplicaciones Multimedia (OMMMA-L), para modelar todos los artefactos del sistema. Para la realización de este trabajo ha sido imprescindible la utilización de la herramienta Case Rational Rose, que contribuye al desarrollo de software para una mayor calidad del mismo.

3.1 Caso de estudio.

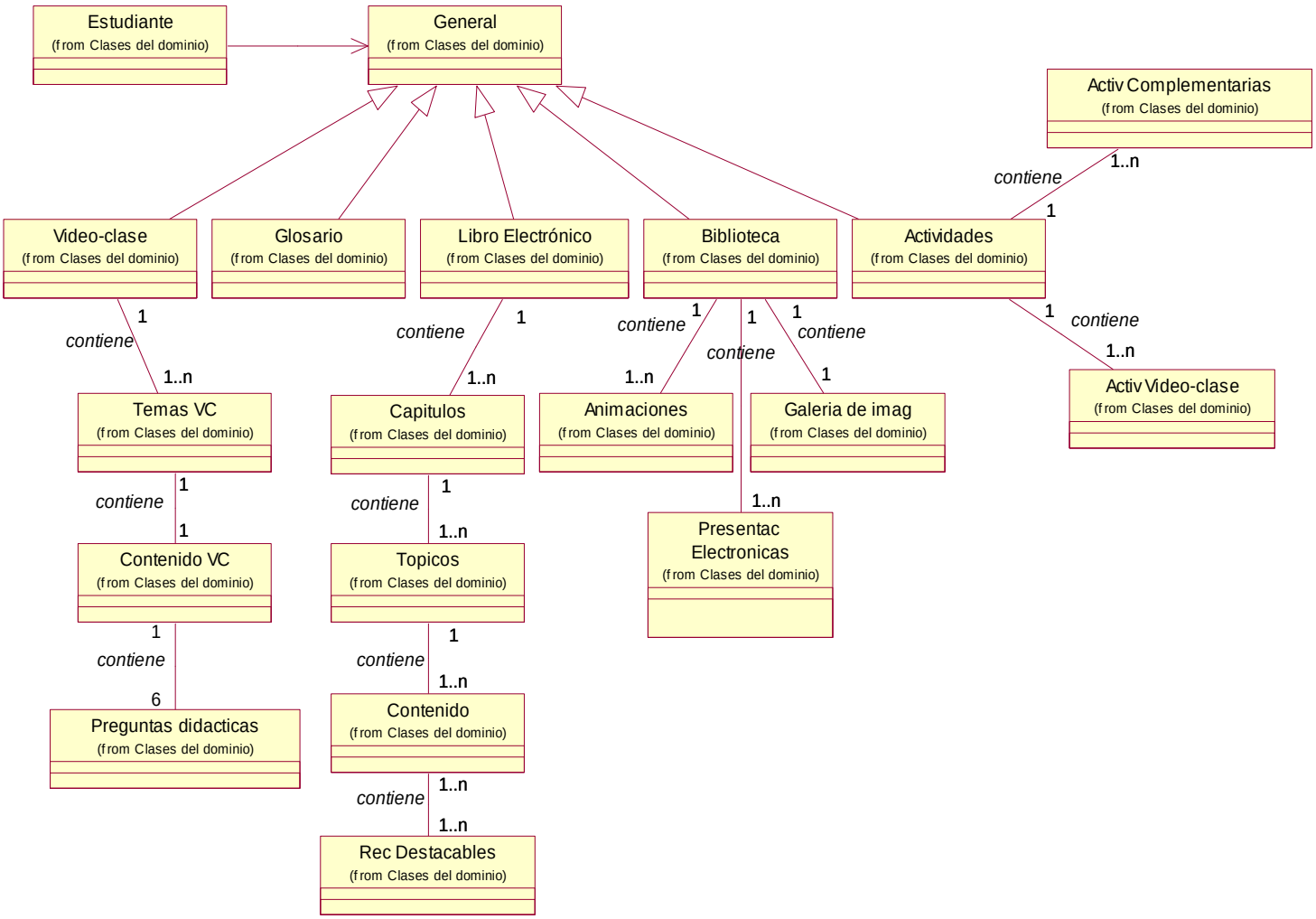
En la Facultad de Ingeniería Civil del Centro Universitario José Antonio Echeverría (CUJAE) tiene entre sus asignaturas fundamentales la Topografía. Esta ha presentado problemas con el nivel de asimilación del estudiante en sus diferentes ramas así como en el uso de los disímiles equipos de medición y nivelación del terreno, por lo que se ha propuesto la tarea de realizar un software educativo para el apoyo en la enseñanza de esta asignatura.

Se pretende realizar una Multimedia donde se abarca la información referente a la carrera de Ingeniería Civil, el Programa de disciplina de Topografía que se imparte, los contenidos referentes a la asignatura de Topografía tales como las video-clases y las actividades, así como los métodos de comprobación, ejercicios, etc. También se

tendrá en cuenta toda la bibliografía referente a los contenidos de dicha materia y a los documentos que complementen el estudia de la misma.

3.2 Modelo de Dominio

El modelo de dominio se presenta como alternativa al modelo de negocio en productos altamente basados en tecnologías informáticas, debido a la poca estructuración de los procesos que describen el negocio. Su objetivo es el de contribuir a la comprensión del contexto del sistema, y por lo tanto de los requisitos funcionales que se desprenden de este contexto.



3.2.1 Descripción del Modelo de Dominio

A causa del bajo nivel de estructuración que presenta el negocio que se estudia y por estar altamente centrado en tecnologías informáticas, se propone un modelo de

dominio, que posibilite mostrar visualmente al usuario los principales conceptos que se manejan en el dominio del sistema en desarrollo. Esto ayuda a los usuarios, clientes y desarrolladores, a utilizar un lenguaje común para poder entender el contexto en que se enmarca el mismo. El capturar correctamente los requisitos y poder construir un sistema correcto exige tener un firme conocimiento del funcionamiento del objeto de estudio. Este modelo va a contribuir posteriormente a identificar algunas clases que se utilizarán en el sistema.

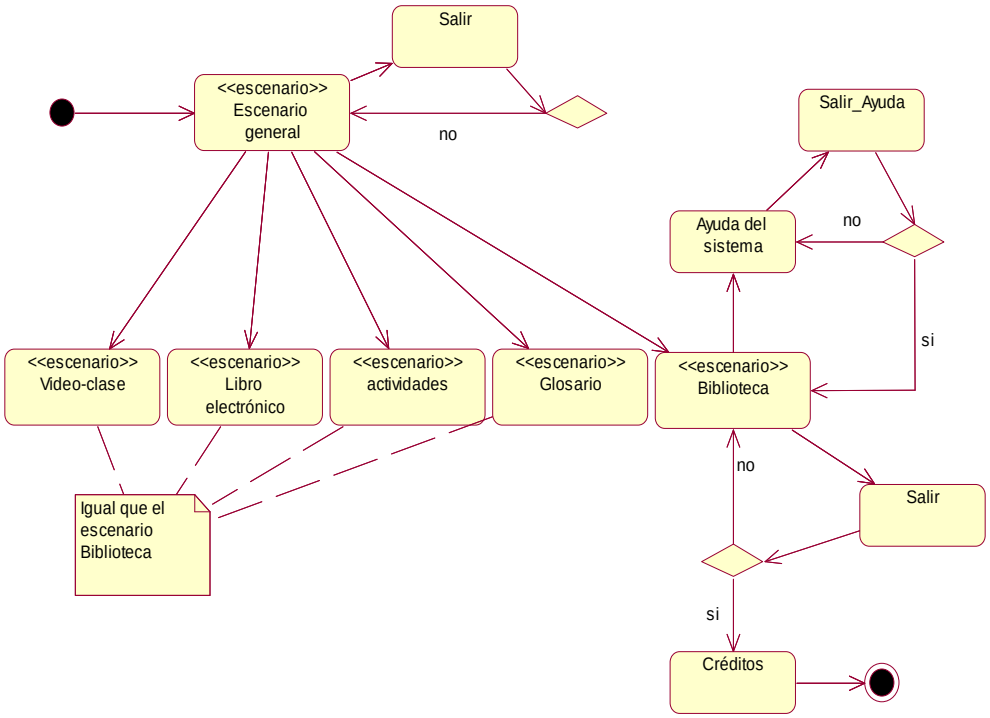
El modelo de dominio se describe mediante diagramas de UML, especialmente diagramas de clases, donde se muestran y especifican las principales clases conceptúales (clases del dominio) que pueden intervenir en el sistema, y cómo se relacionan unas con otras mediante asociaciones. Estos diagramas representarán los objetos que existen o eventos que suceden en el entorno en el que trabajará el sistema. La designación del modelo conceptual o dominio, ofrece la ventaja de subrayar una concentración en los conceptos del dominio del problema, no en las entidades del software, salvo que el dominio a modelar se refiera a conceptos del software (ejemplo: modelo interfaces gráficas para el usuario), que es el caso en que está enmarcado nuestro sistema.

3.3 Modelo de Navegación

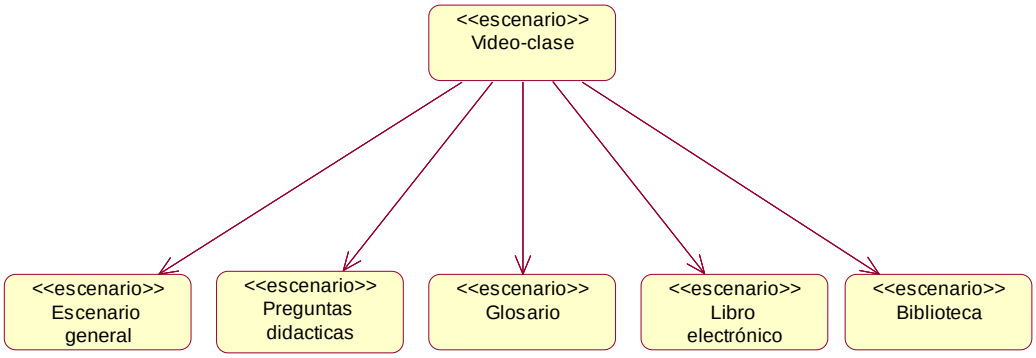
En aplicaciones multimedia el Diagrama de Actividades viene siendo un Diagrama de Navegación a través de la multimedia.

3.3.1 Diagramas de Navegación

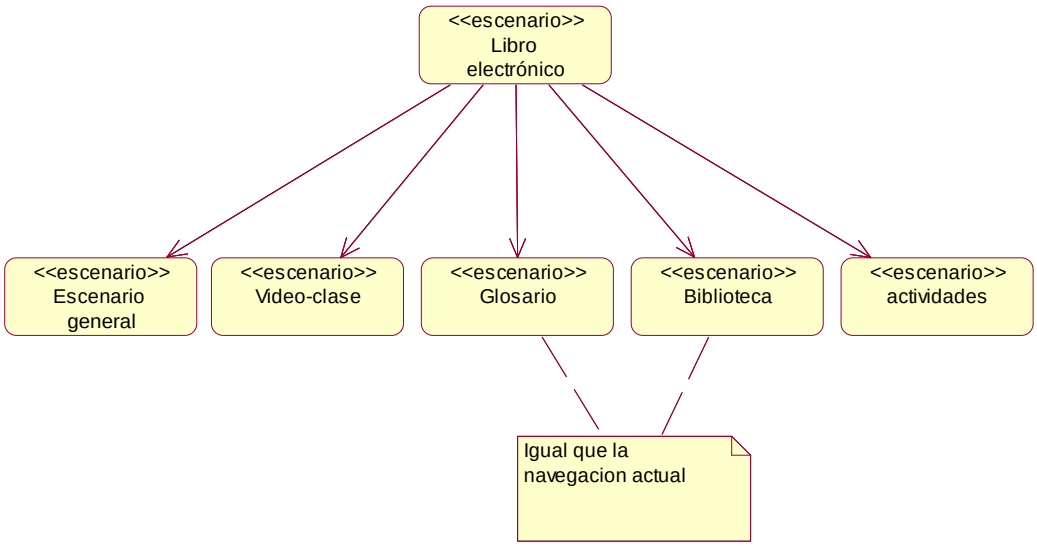
- Diagrama de navegación desde Escenario General



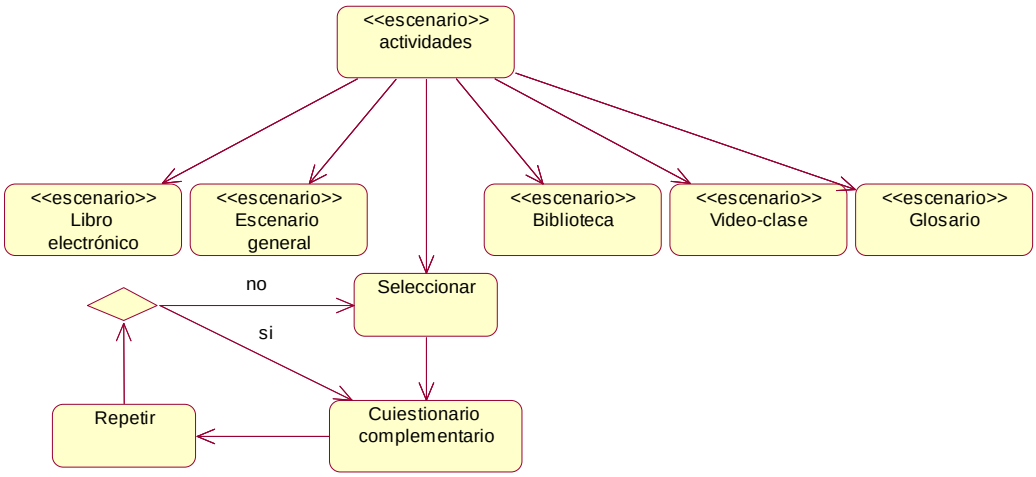
- Diagrama de navegación desde Escenario Video-clase



- Diagrama de Navegación desde Libro Electrónico



- Diagrama de Navegación desde el escenario Actividades



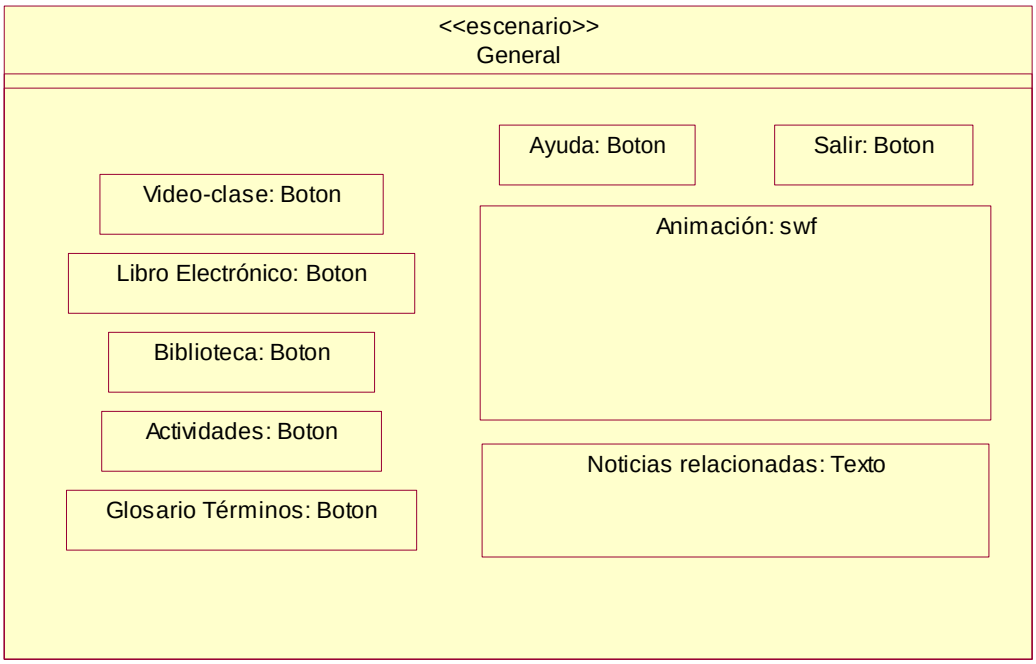
3.3 Modelo de Análisis

3.3.1 Diagramas de Presentación

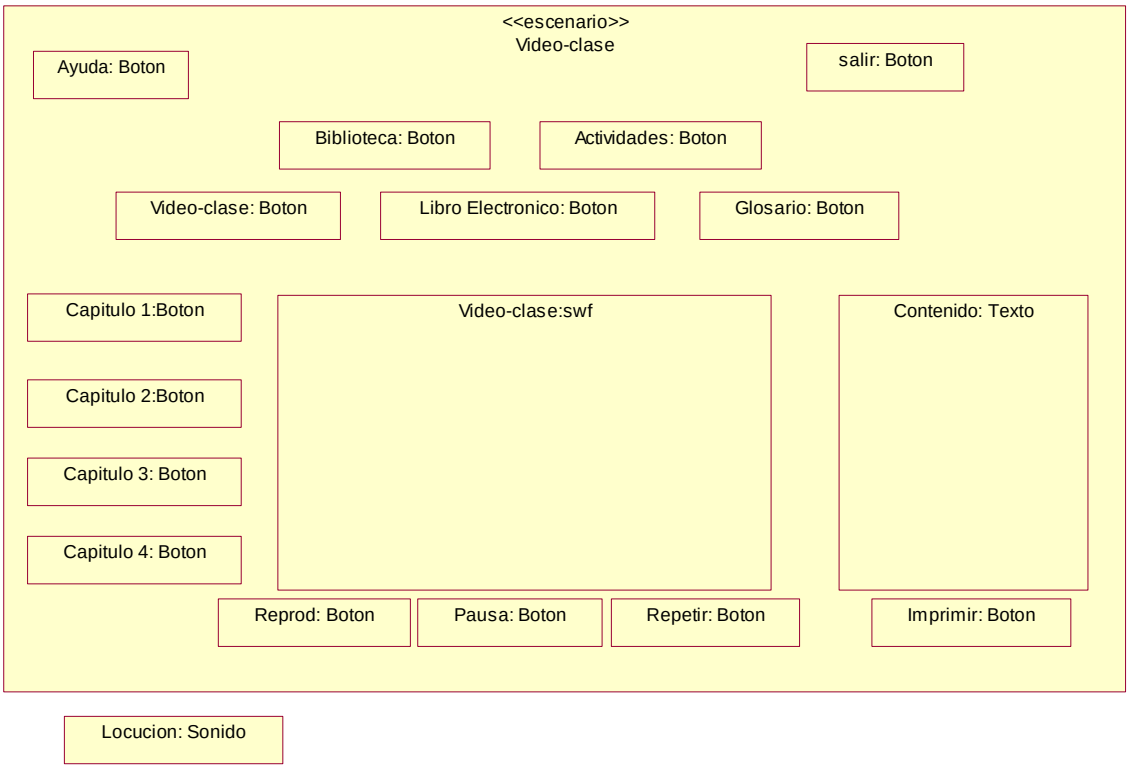
Este es un nuevo artefacto dentro del lenguaje UML, es específico de OMMMA-L. Es usado para describir la parte estática del modelo de la apreciación MVCMM [18], a través de una descripción intuitiva de la distribución espacial de los objetos visuales de la interfaz de usuario.

- Diagrama de Presentación de escenario Presentación Inicial

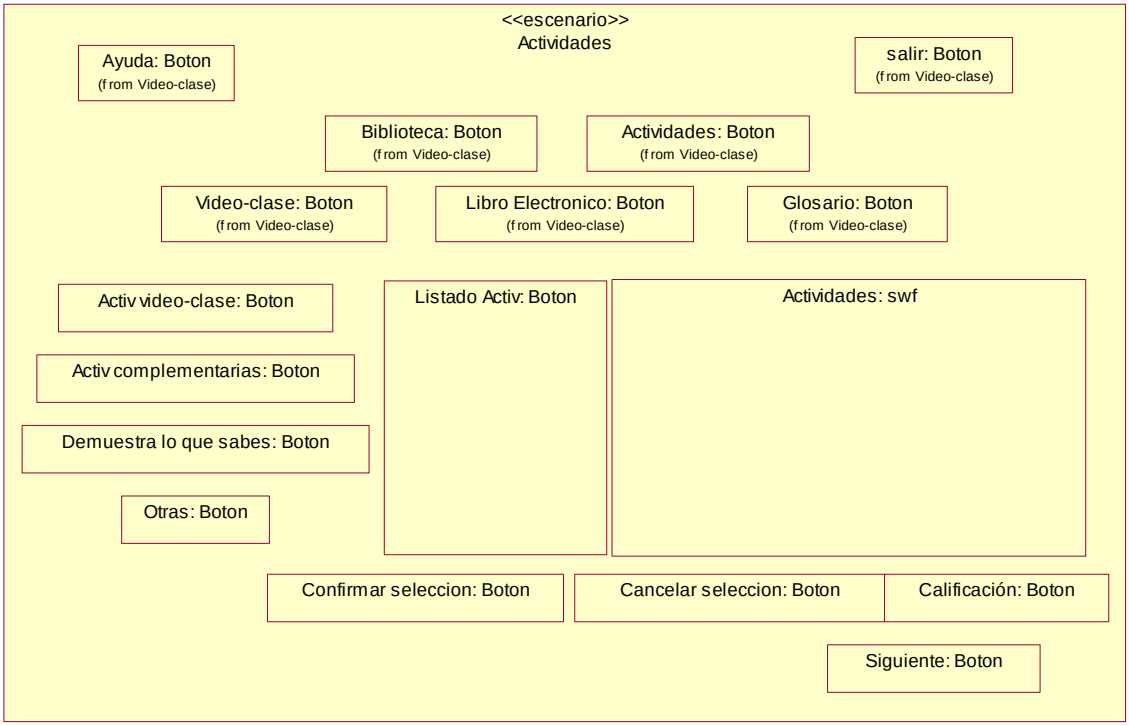
- Diagrama de Presentación de escenario General



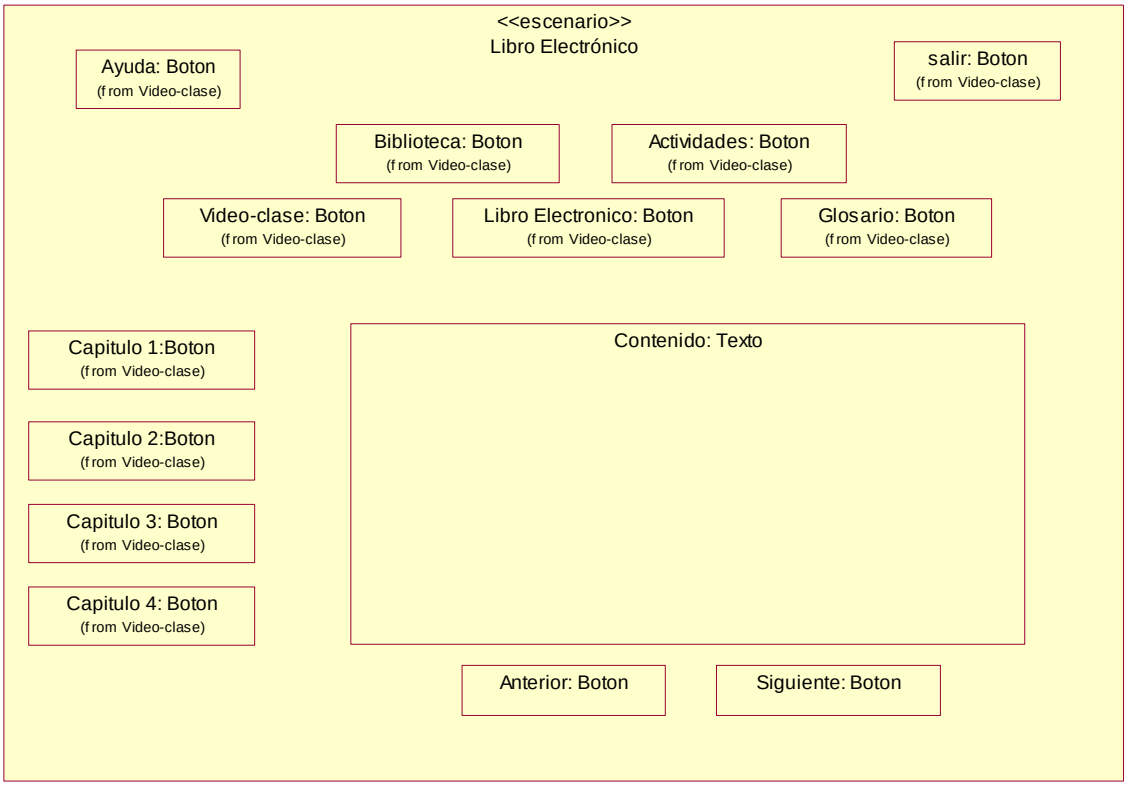
- Diagrama de Presentación de escenario Video-clase



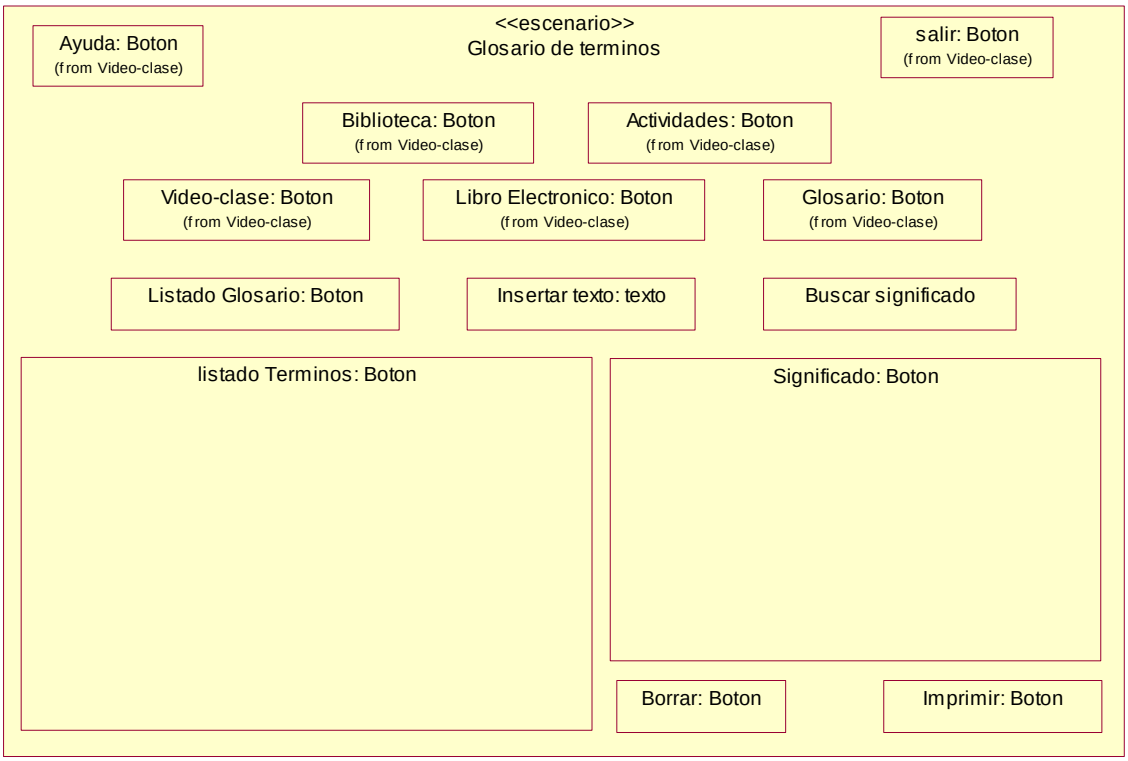
- Diagrama de Presentación de escenario Actividades



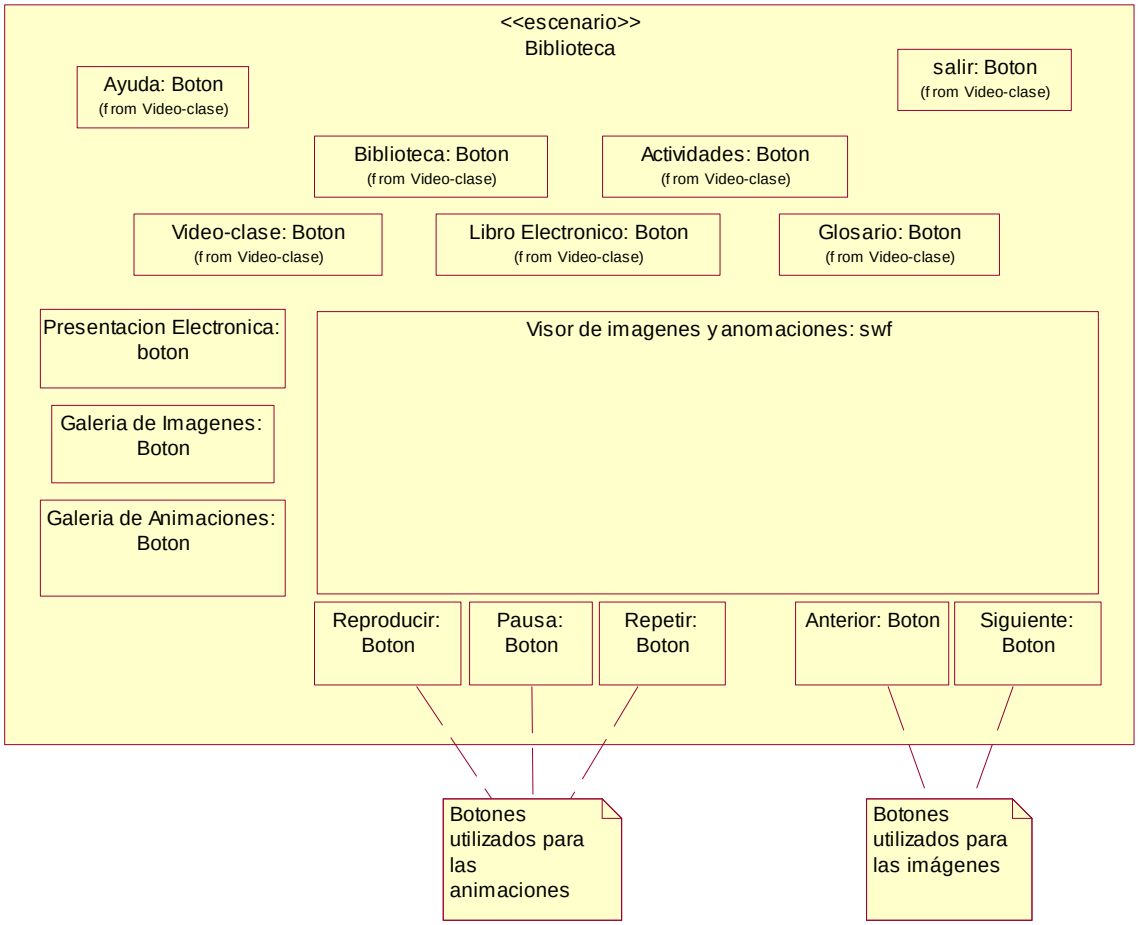
- Diagrama de Presentación de escenario Libro Electrónico



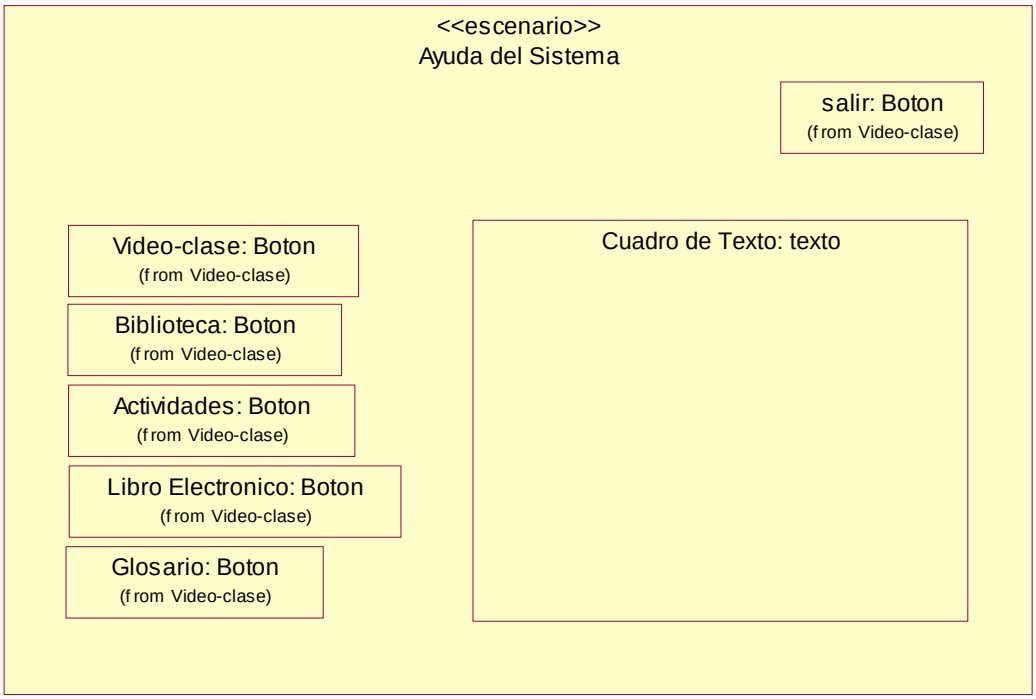
- Diagrama de Presentación de escenario Glosario de Términos



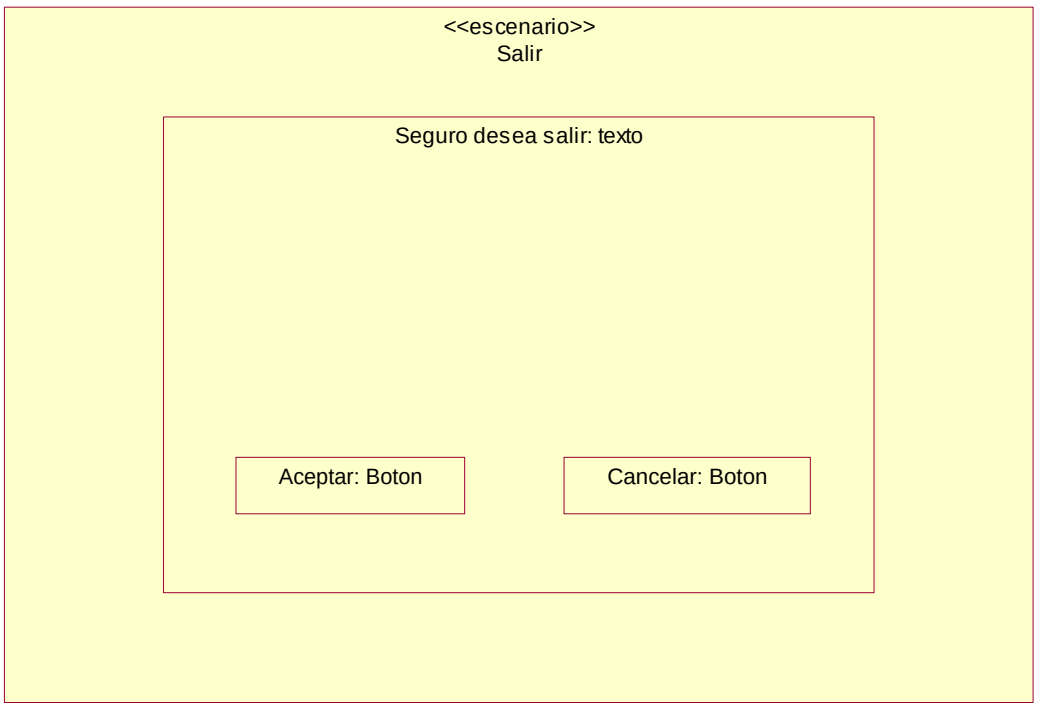
- Diagrama de Presentación de escenario Biblioteca



- Diagrama de Presentación de escenario Ayuda del sistema

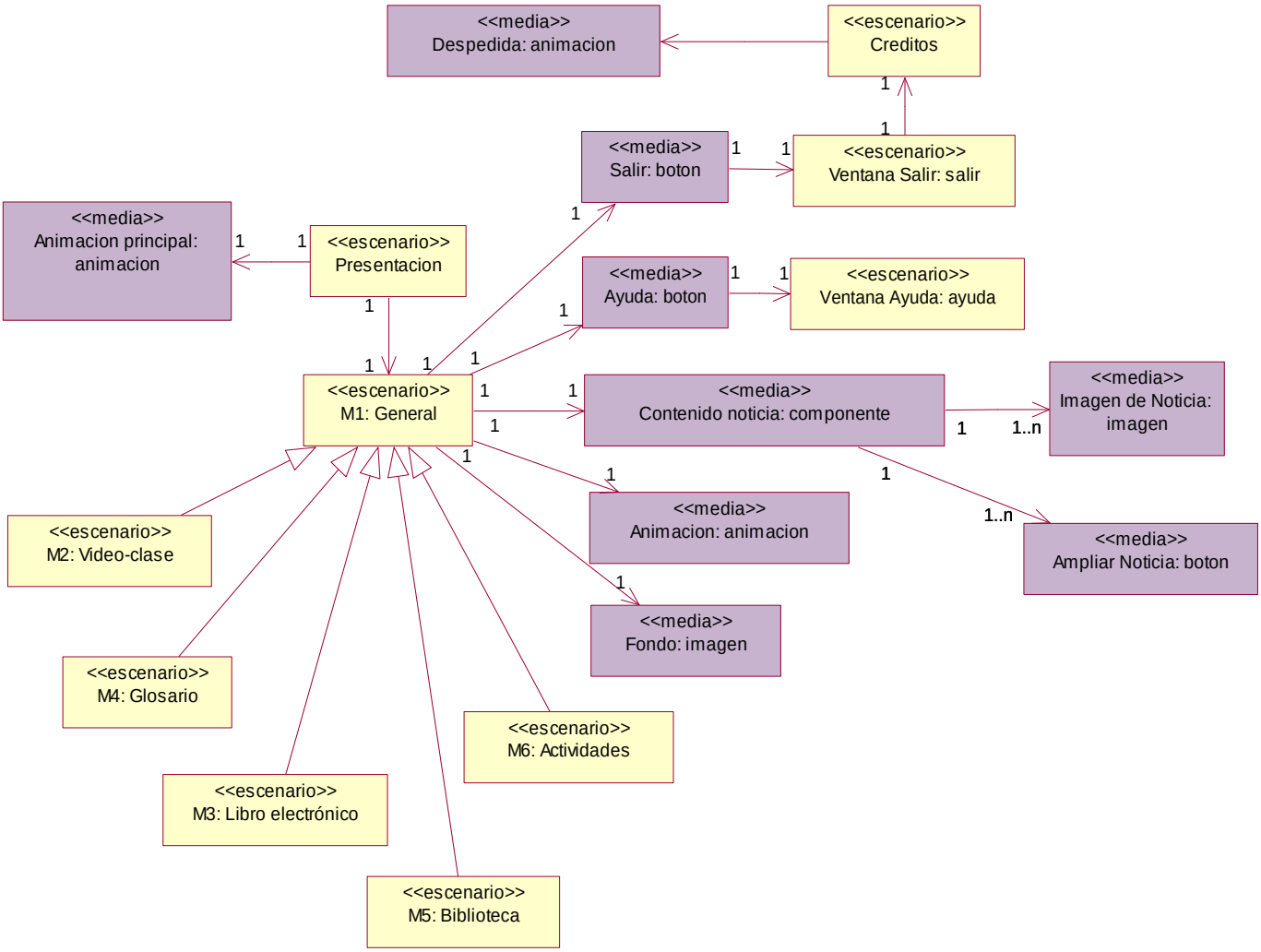


- Diagrama de Presentación de escenario Salir

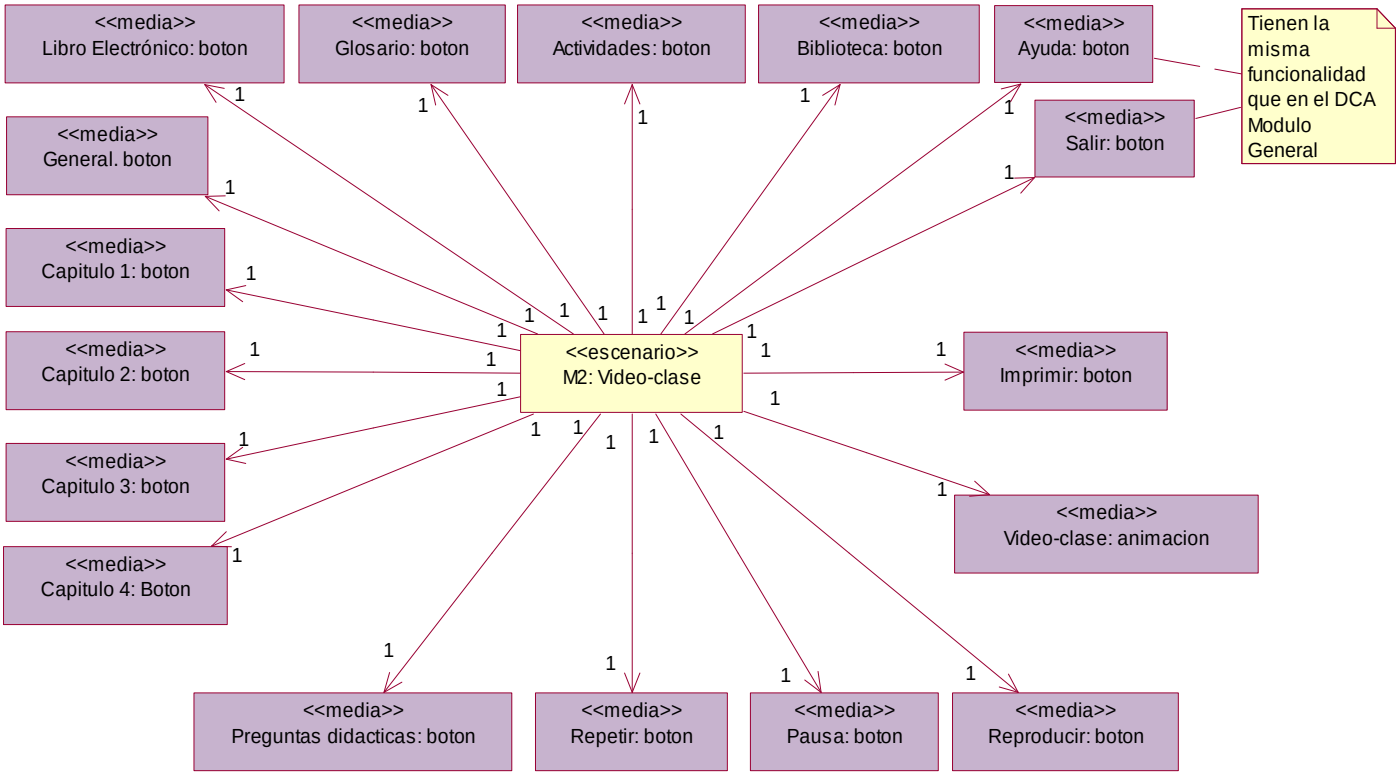


3.3.2 Diagramas de clases del Análisis

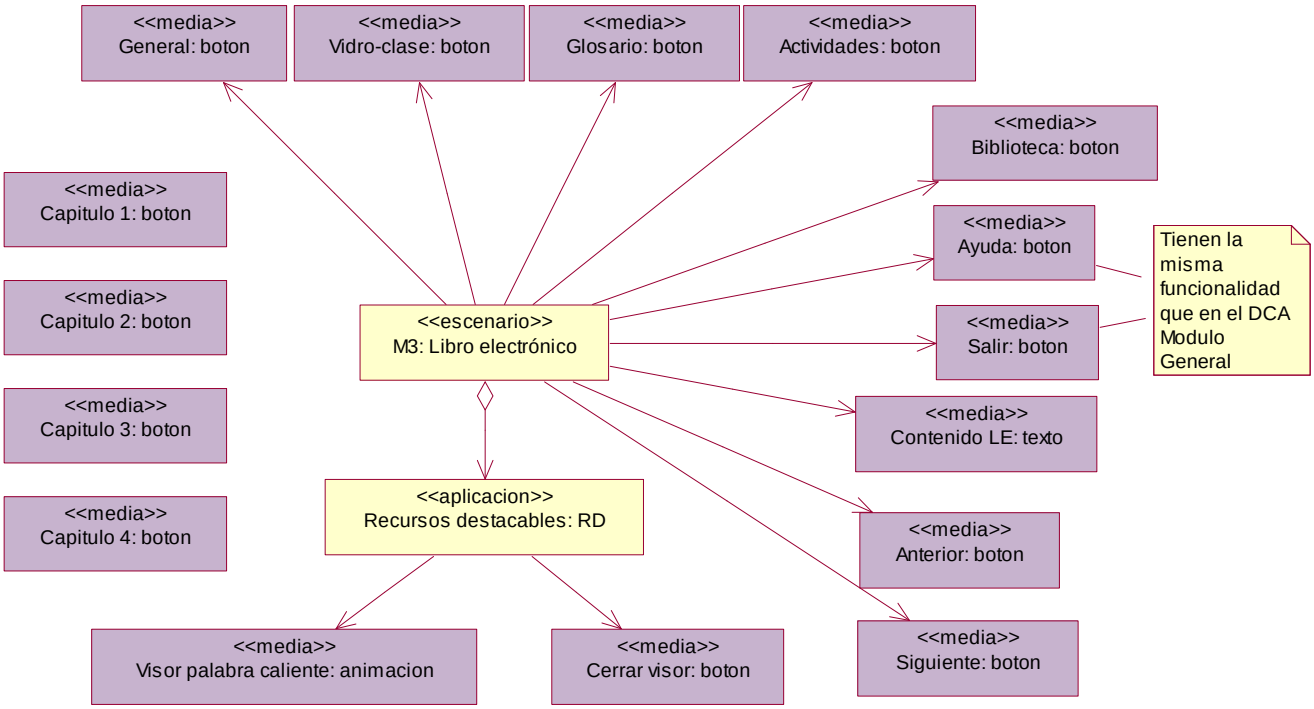
- Diagrama de Clases de Análisis del Modulo General



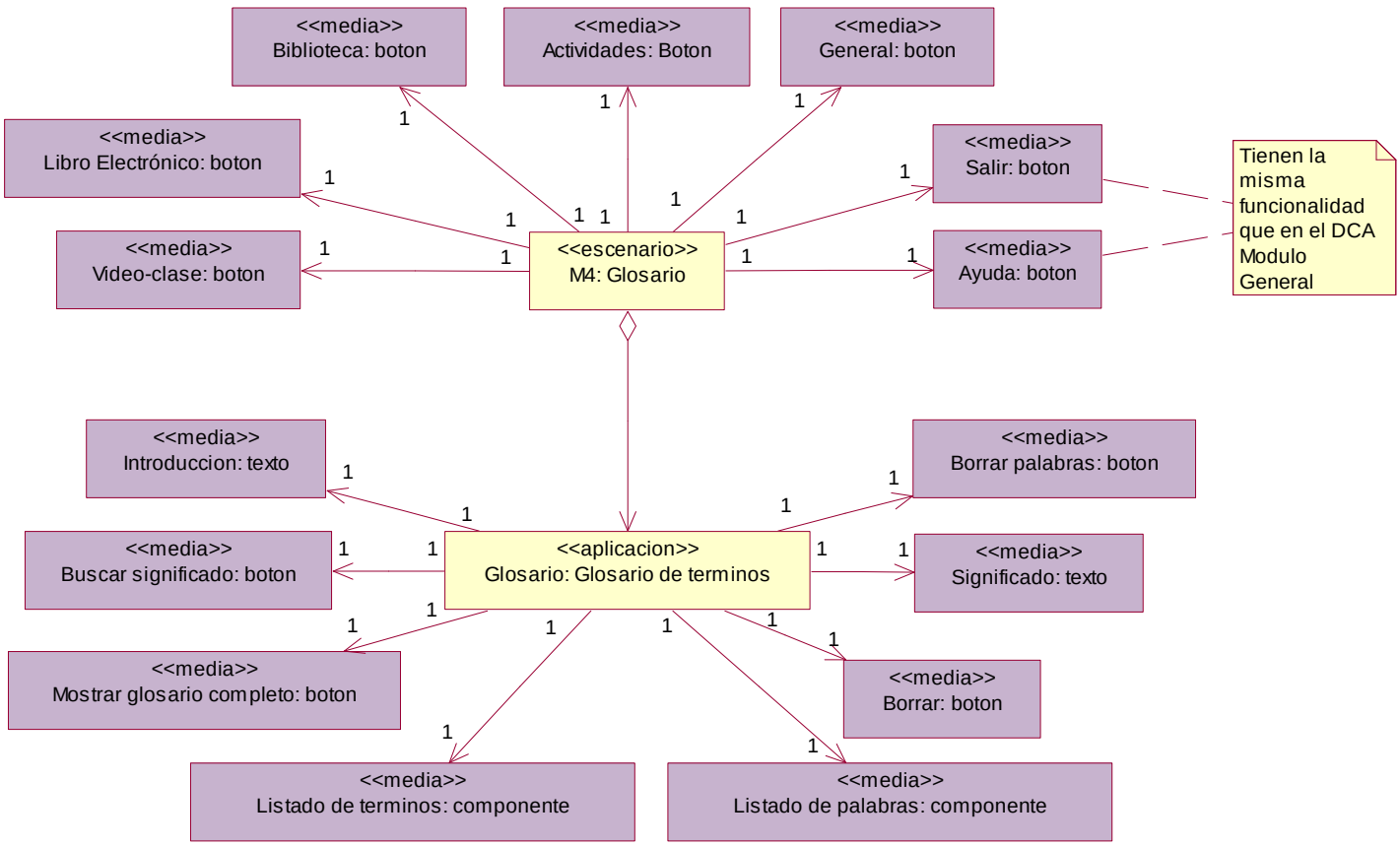
- Diagrama de Clases de Análisis del Módulo Video-clase



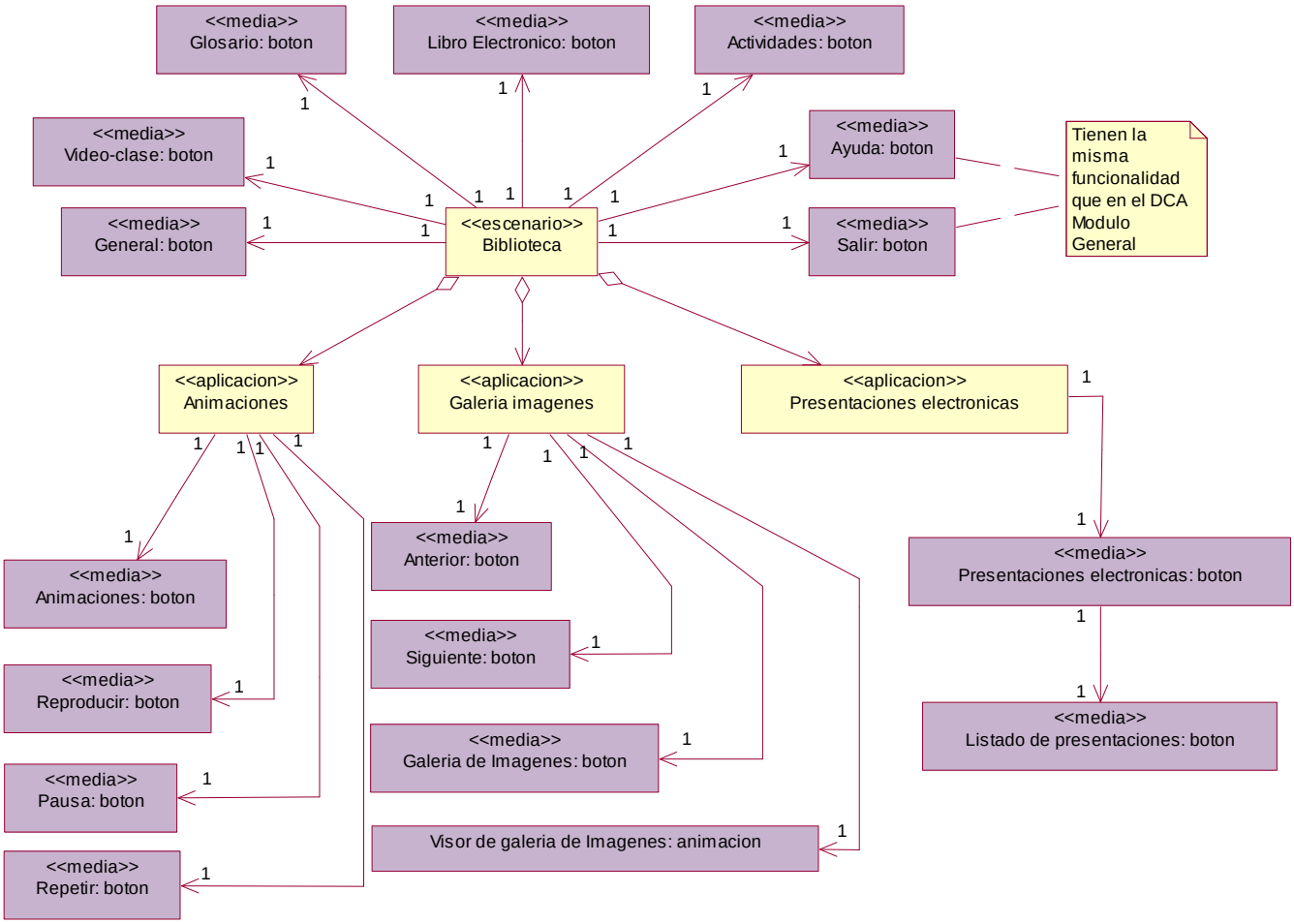
- Diagrama de Clases de Análisis del Módulo Libro Electrónico



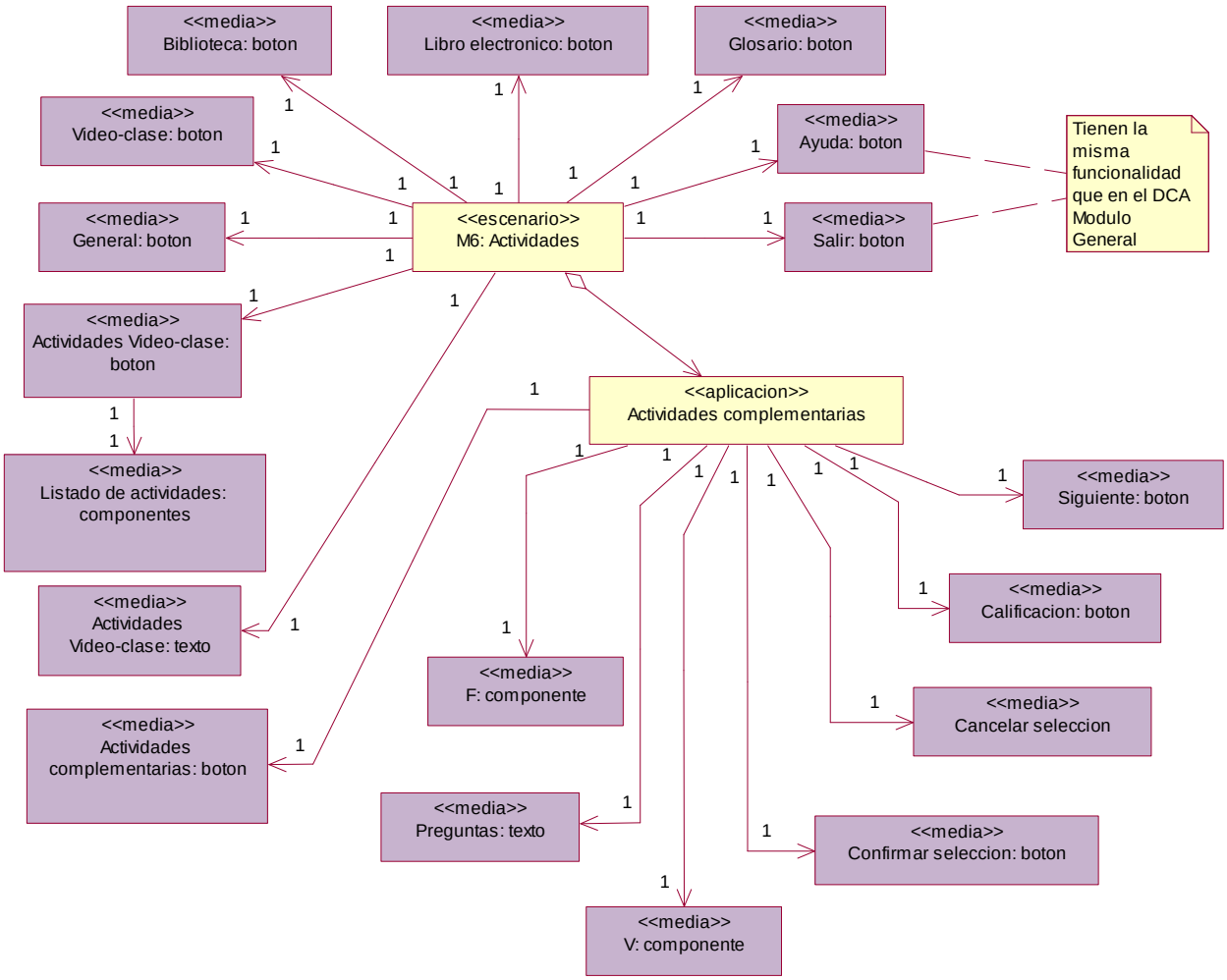
- Diagrama de Clases de Análisis del Módulo Glosario de Términos



- Diagrama de Clases de Análisis del Módulo Biblioteca



- Diagrama de Clases de Análisis del Módulo Actividades

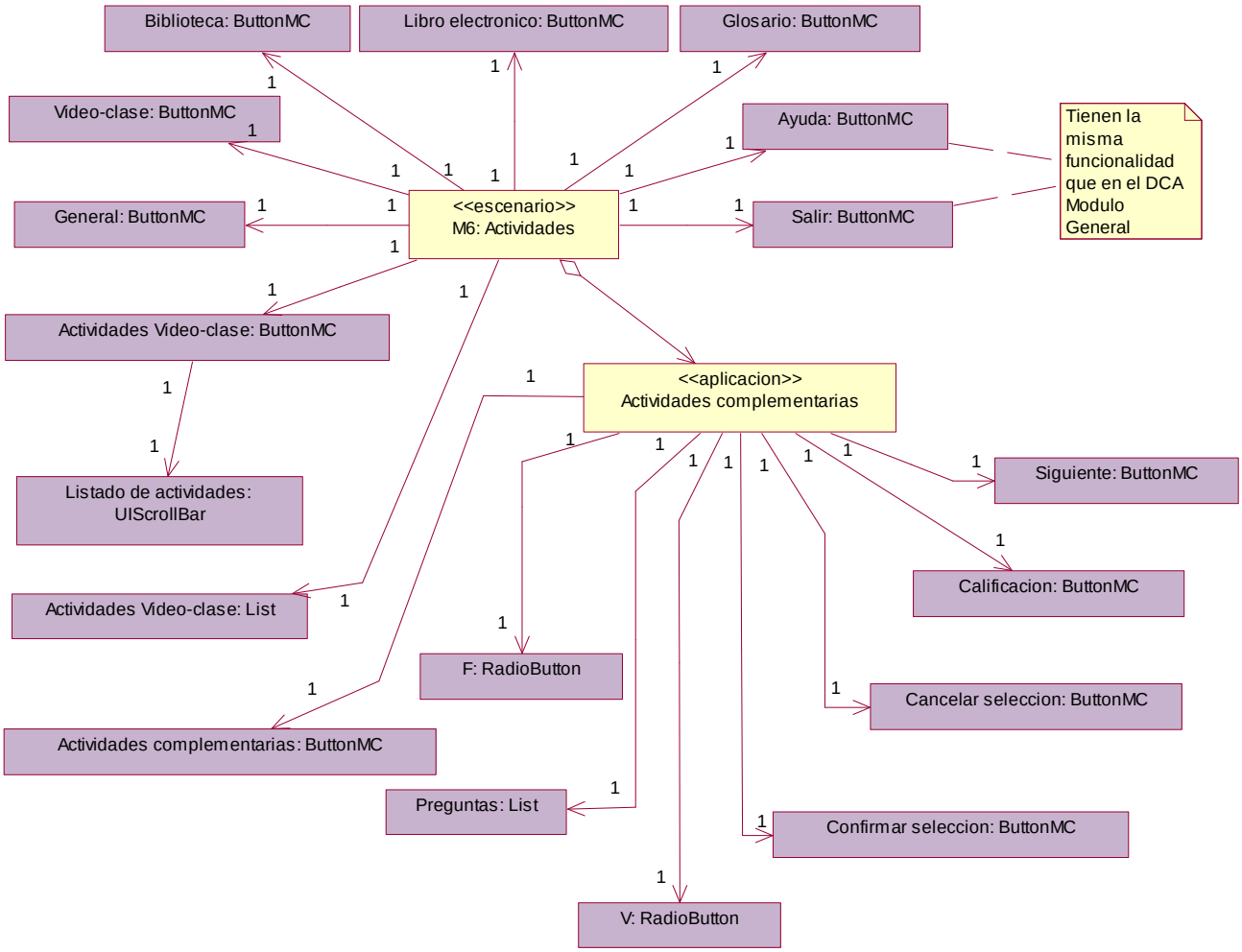


3.4 Modelo de Diseño

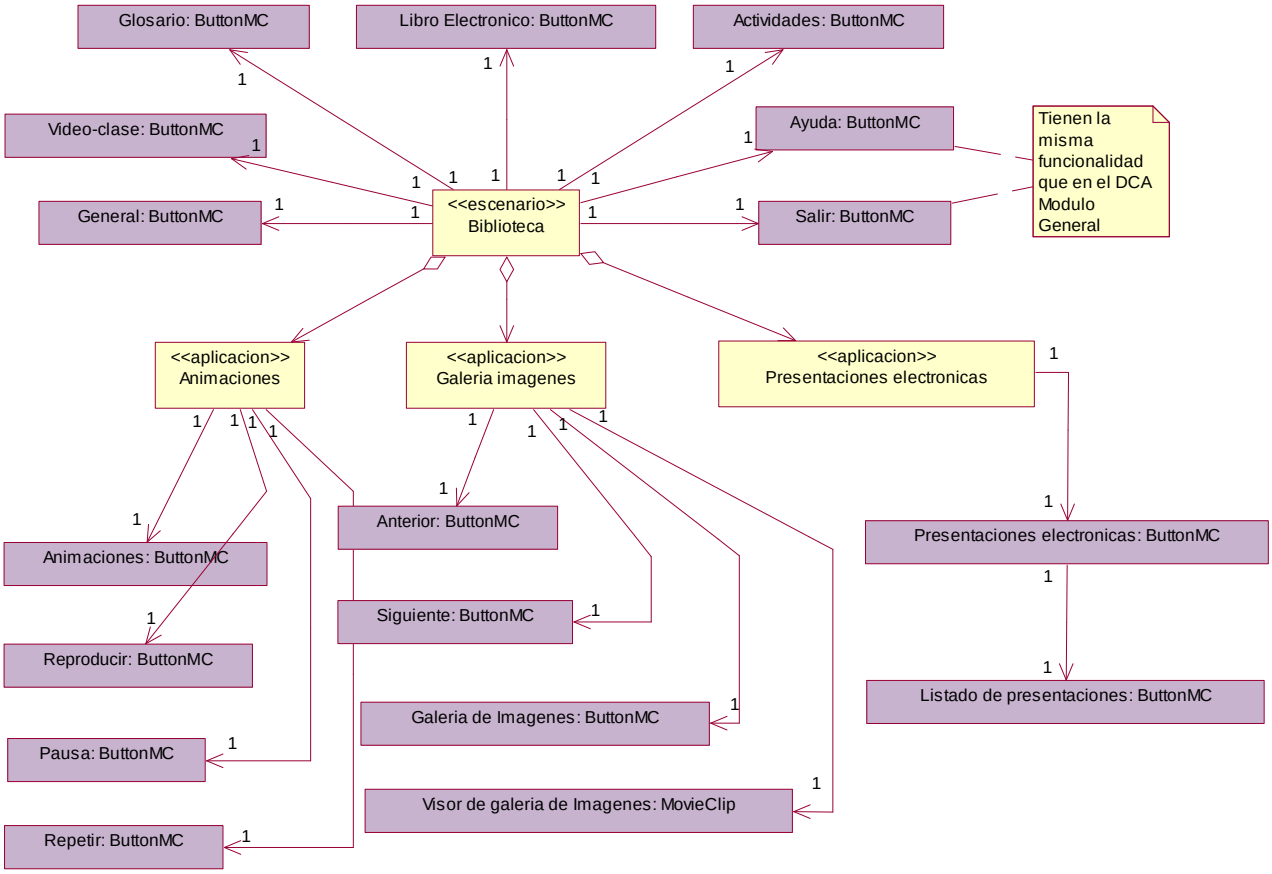
3.4.1 Diagramas de Clases del Diseño

OMMMA–L propone en cada diagrama de clases elaborado, adicionar la jerarquía de media de la herramienta y enlazar a través de relaciones las clases del tipo.

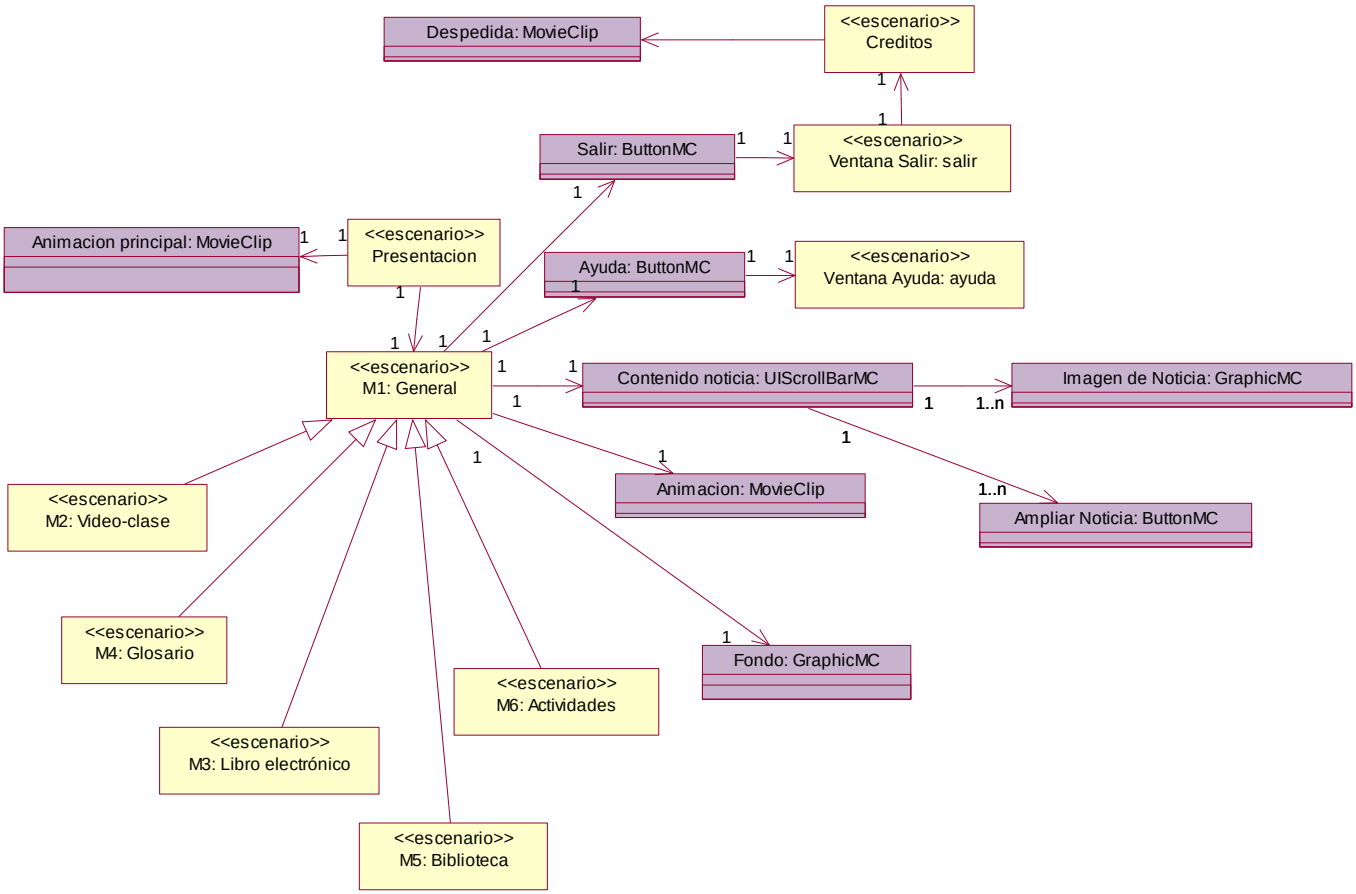
- Diagrama de clases del diseño Módulo Actividades



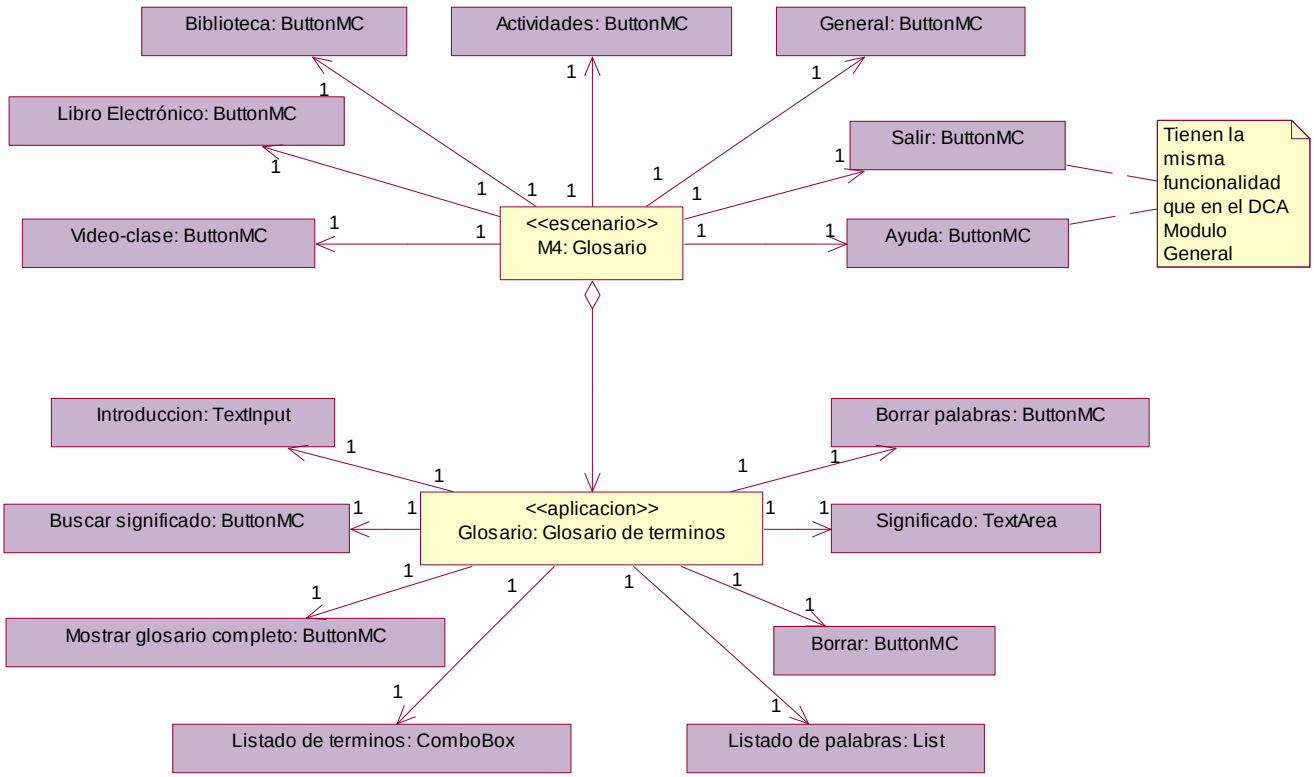
- Diagramas de Clases del Diseño del Módulo Biblioteca



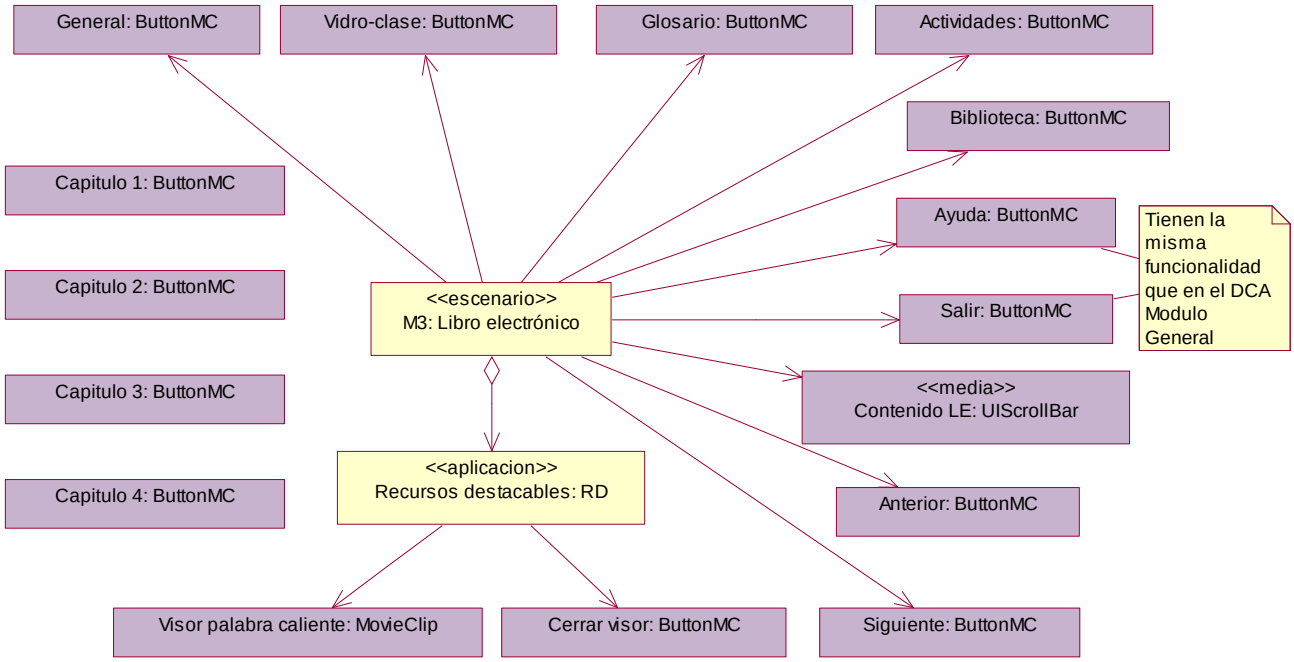
- Diagrama de Clases del Diseño del Módulo General



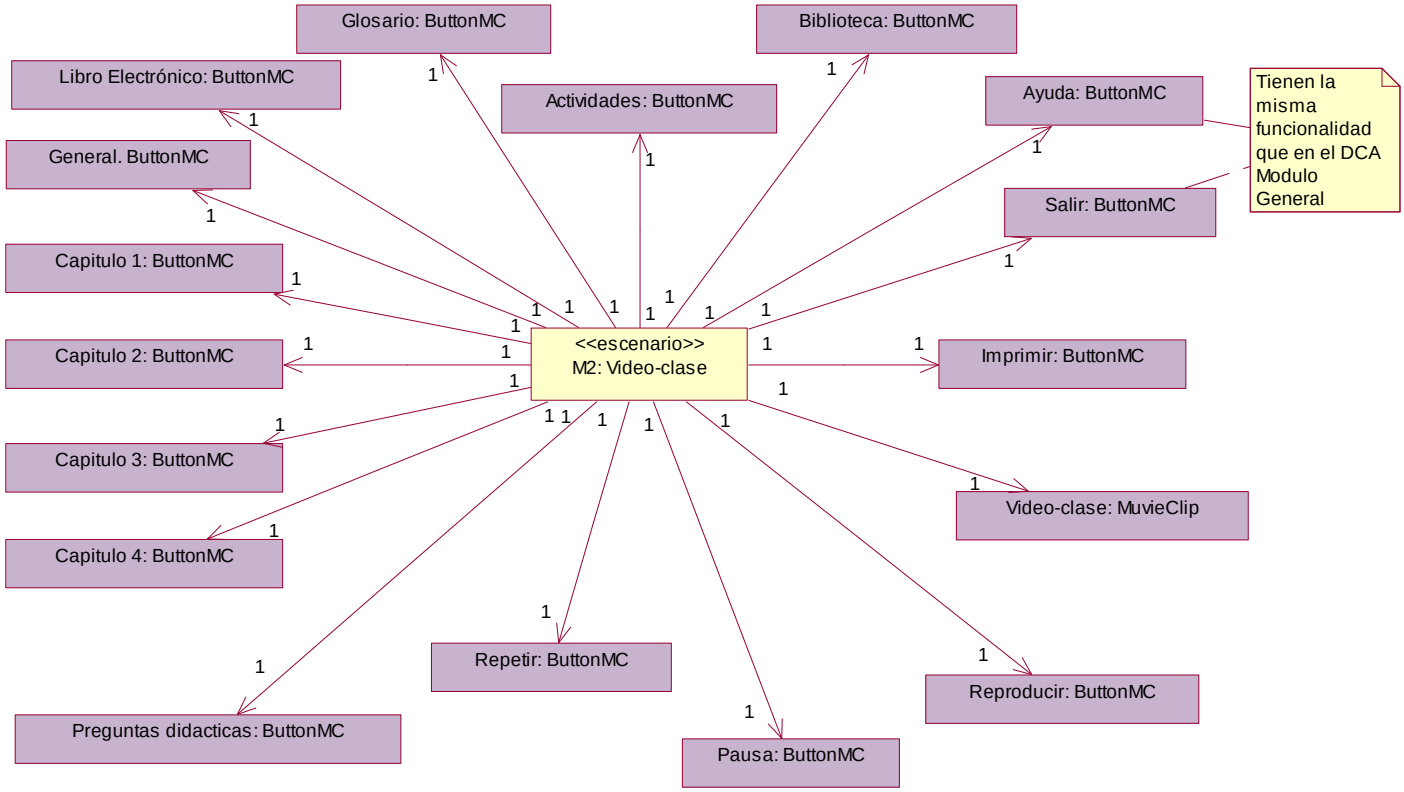
- Diagrama de Clases del Diseño del Módulo Glosario



- Diagrama de Clases del Diseño del Módulo Libro Electrónico



- Diagrama de Clases del Diseño del Modulo Video-clase.



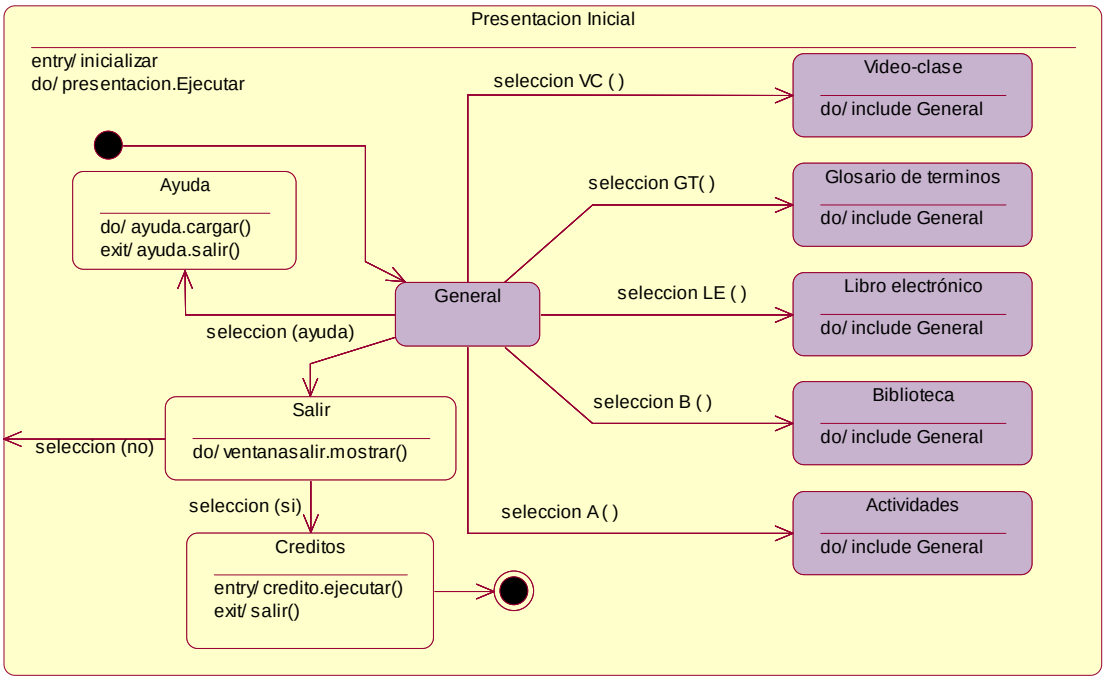
3.4.2 Diagramas de Comportamiento Interactivo

En UML, los diagramas de estado son utilizados para especificar el modelo controlador de MVCMM a través de los estados de la aplicación, así como de las interacciones activadas por la intervención del usuario u otros eventos del sistema (comportamiento espontáneo) [18].

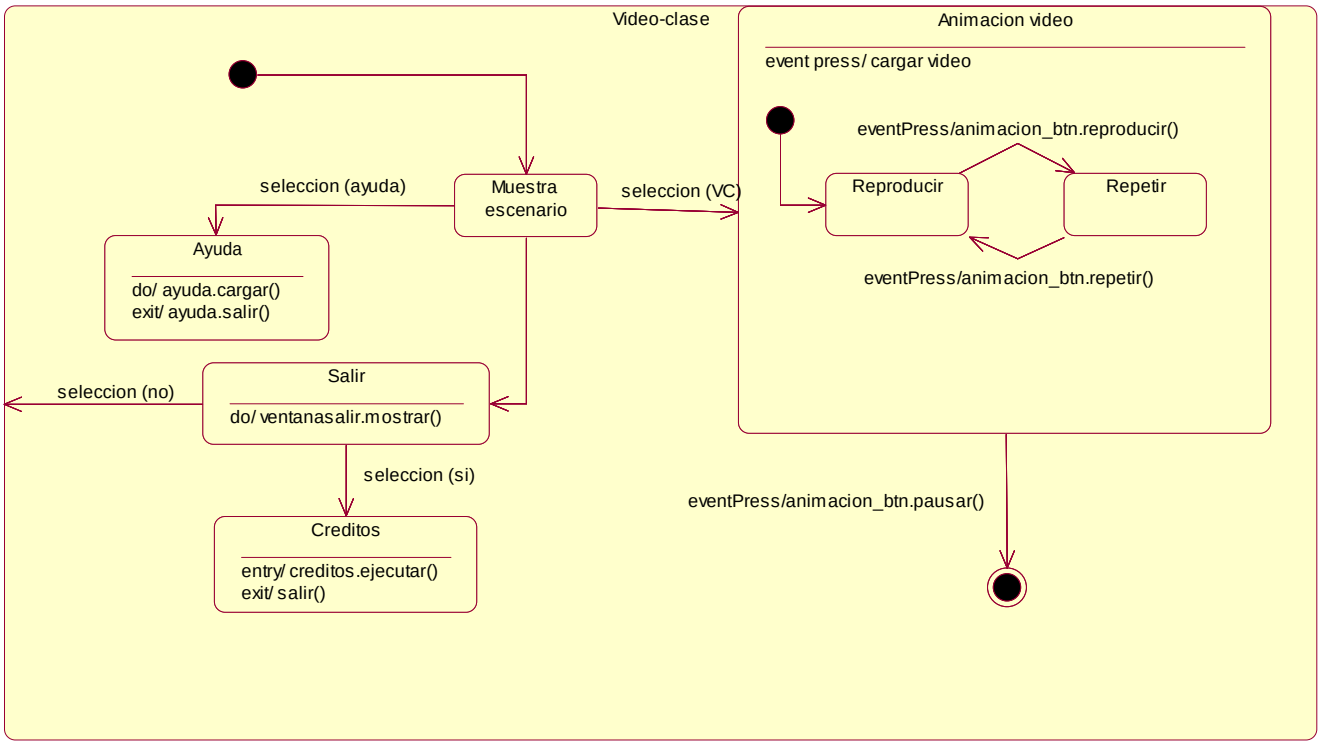
La interacción del usuario es una activación asincrónica que afecta la composición espacial y temporal de los objetos en el tiempo de presentación, donde un evento del usuario activa alguna acción dentro del sistema, posiblemente mediada por una cascada de eventos enviada entre los objetos dentro del sistema. En el diagrama de comportamiento interactivo, se muestra el ciclo de vida de la aplicación a partir de su inicio y la navegación durante el tiempo de vida de la multimedia.

Tres aplicaciones básicas se pueden modelar para multimedia a través de máquinas de estado: el comportamiento general del sistema desde un alto grado de abstracción, lo que equivaldría a una descripción de la navegación interactiva; el comportamiento de objetos activos controlando escenarios de presentaciones de media, o sea, el control de la ejecución de la multimedia por objetos sin intervenir el usuario generando procedimientos temporales; y el cambio de estado de objetos de media, aplicaciones y escenarios durante el ciclo de vida de presentación [19].

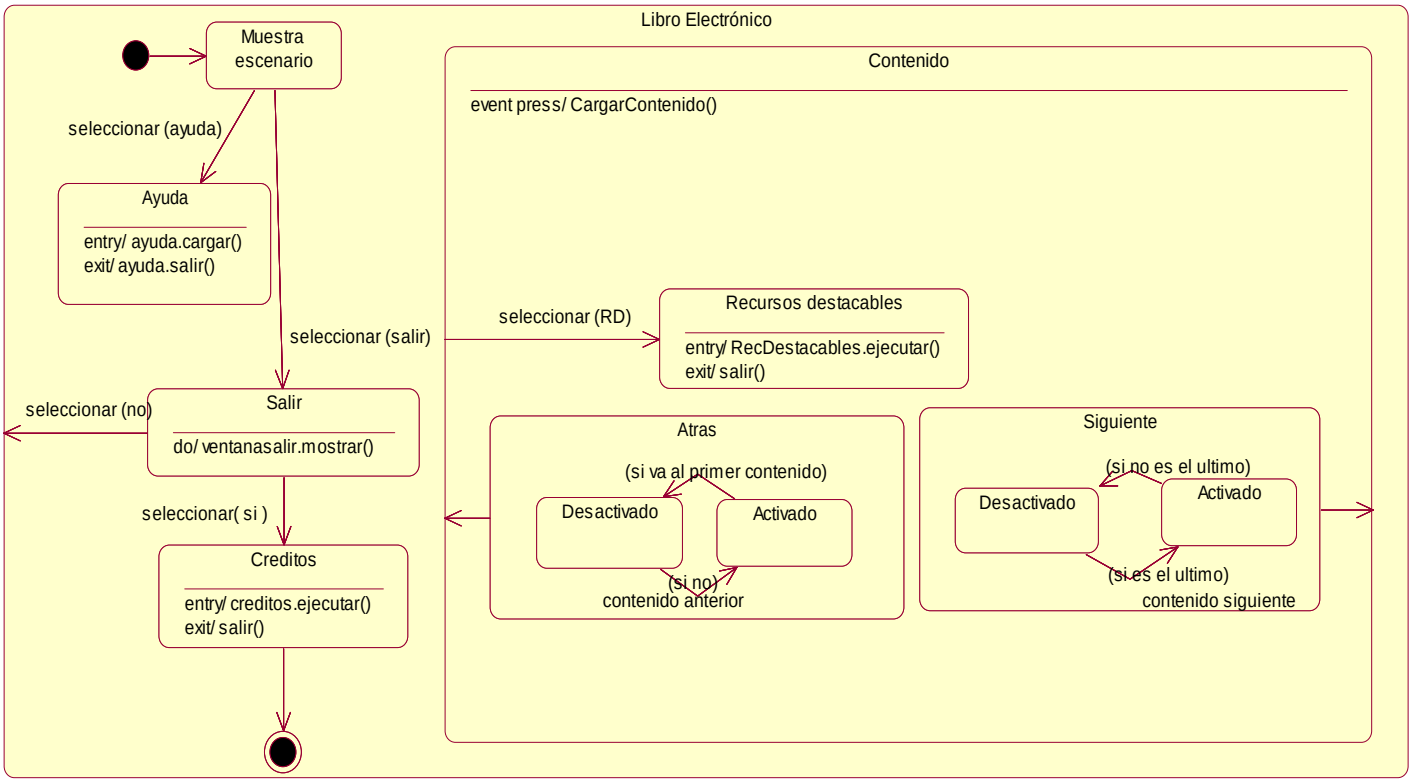
- Diagrama de Comportamiento Interactivo del Módulo Presentación Inicial



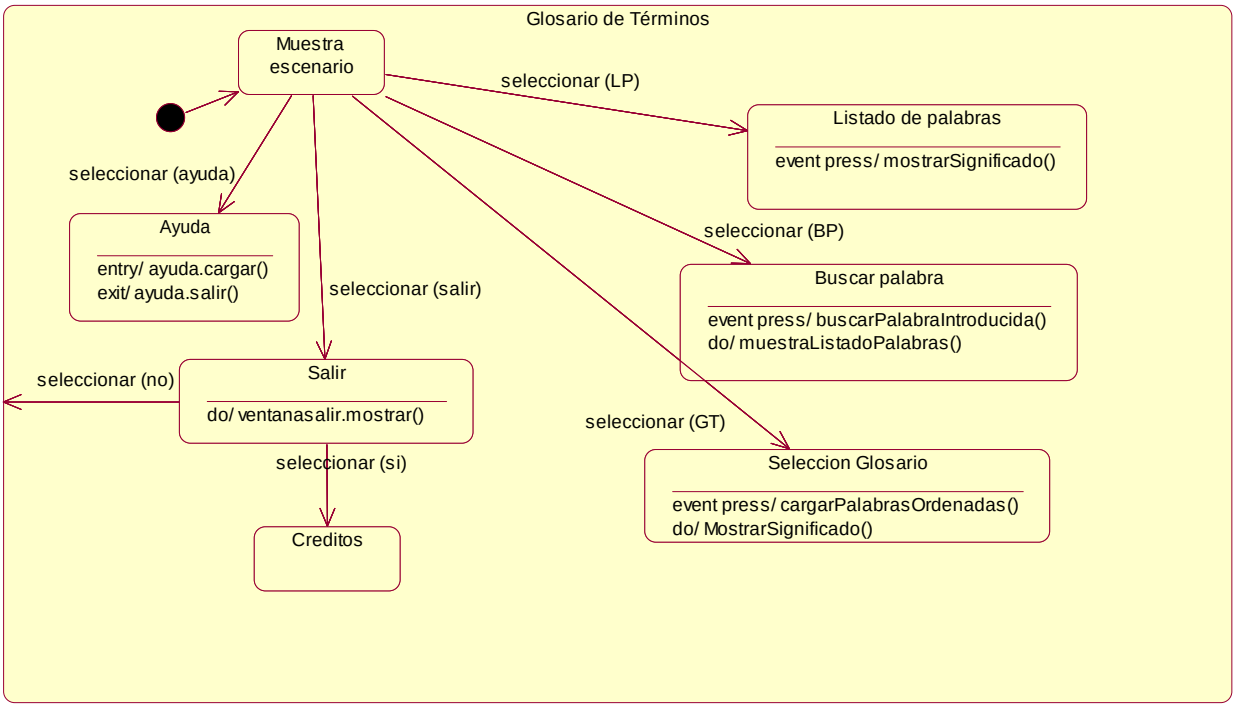
- Diagrama de Comportamiento Interactivo del Módulo Video-clase



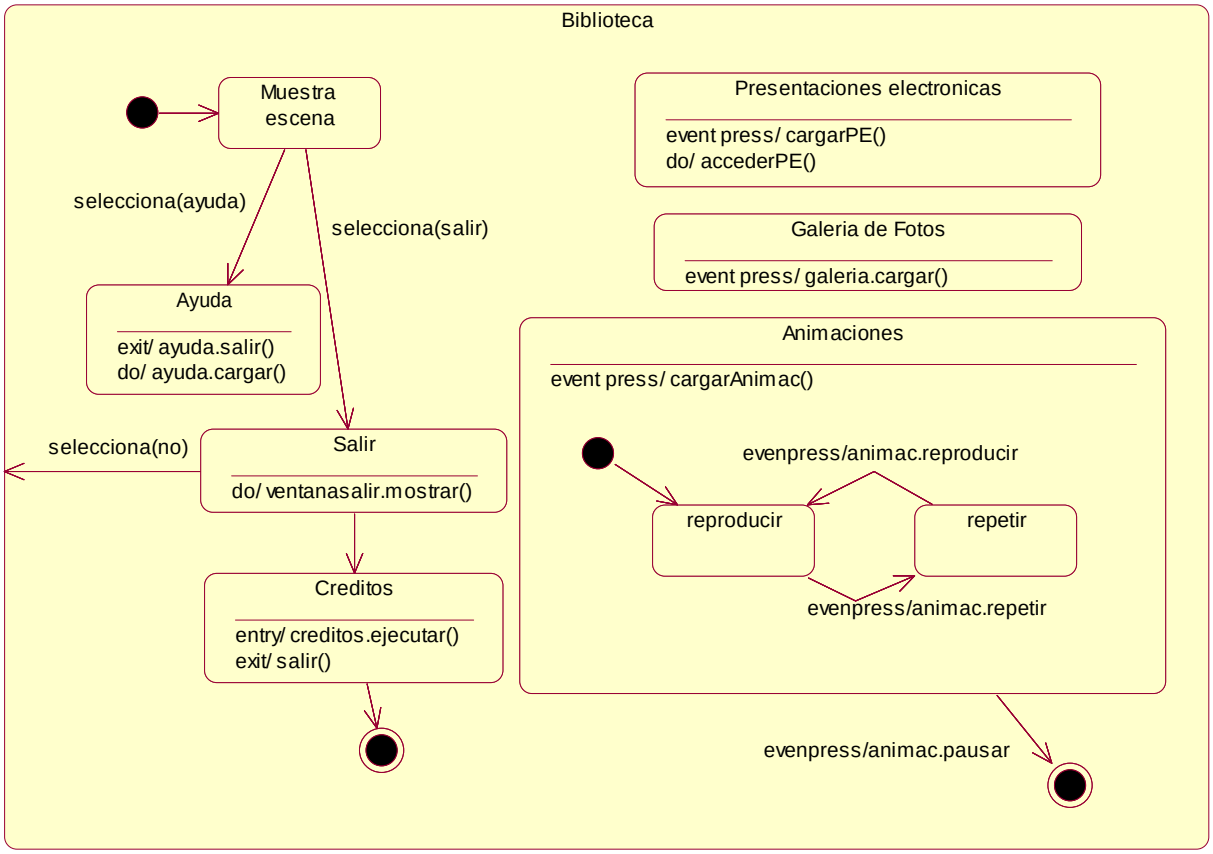
- Diagrama de Comportamiento Interactivo del Módulo Libro Electrónico



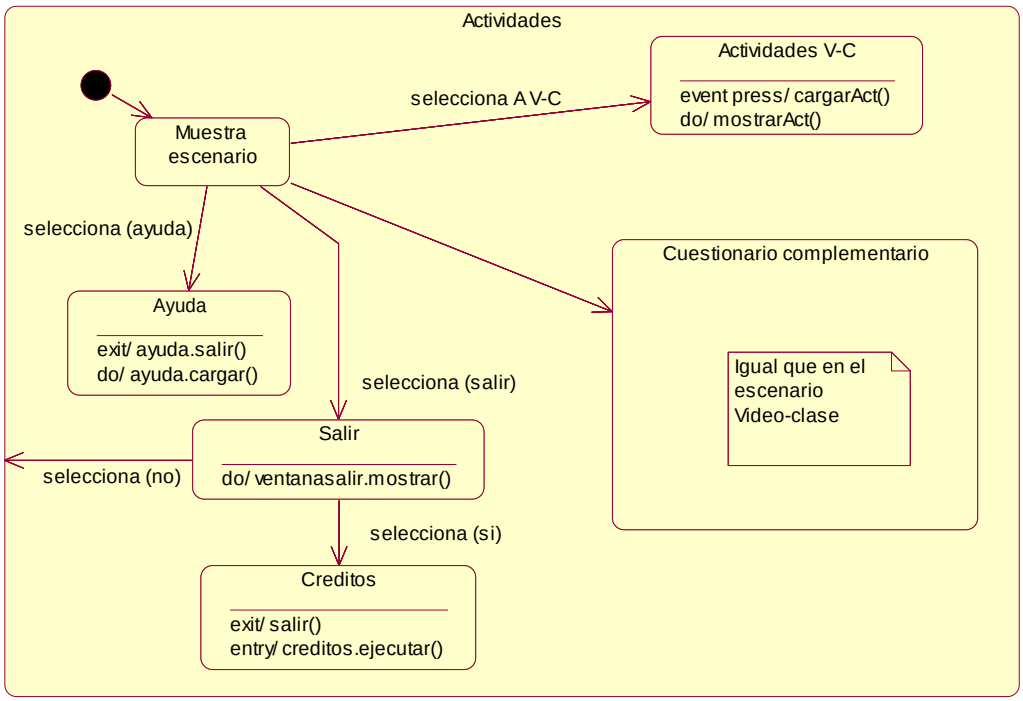
- Diagrama de Comportamiento Interactivo del Módulo Glosario de Términos



- Diagrama de Comportamiento Interactivo del Módulo Biblioteca



- Diagrama de Comportamiento Interactivo del Módulo Actividades



3.4.3 Diagrama de Comportamiento Temporal

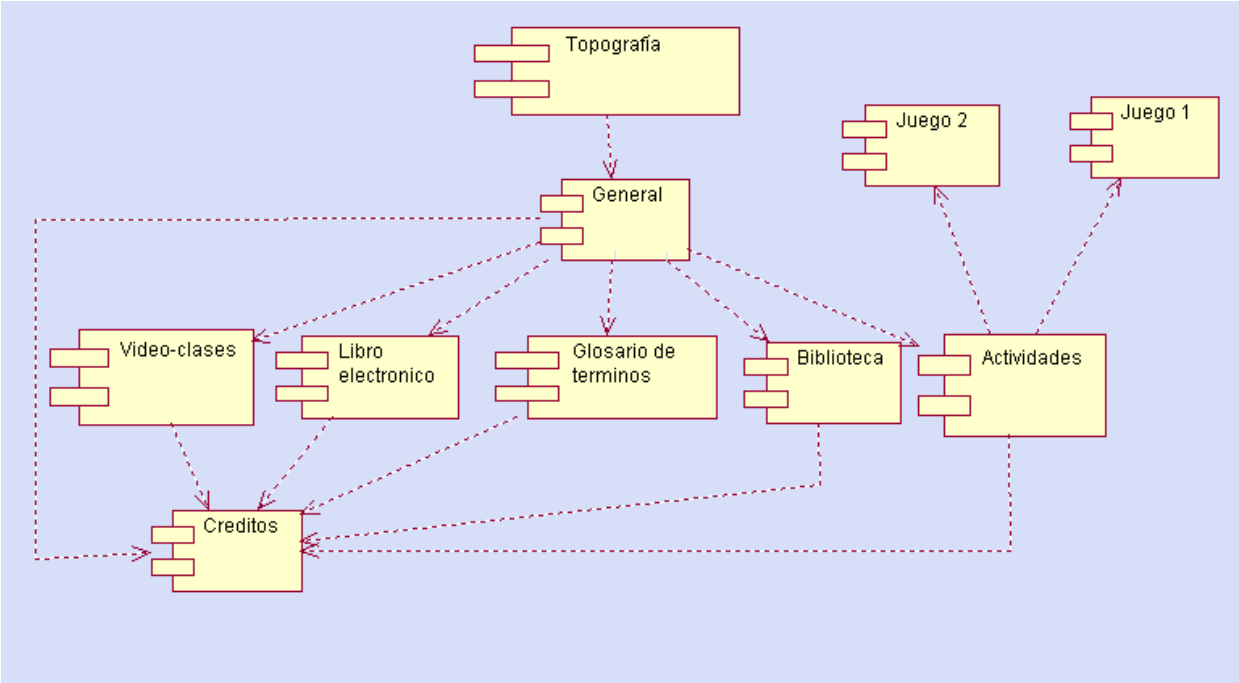
Para modelar el espacio de tiempo en que debe ocurrir una acción se utiliza un diagrama de secuencia, que mantiene su semántica, y la extiende a través de restricciones y señalizaciones de tiempo, permitiendo controlar la sincronización entre ejes de tiempo [19], en el caso de UML.

En OMMMA-L, el diagrama de comportamiento temporal es de nueva aparición, y muy útil para representar sincronizaciones de animaciones, sonidos y videos, en general para el establecimiento de tiempo de objetos continuos.

(Ver anexos)

3.4.4 Diagrama de Componentes

La declaración de cada uno de los módulos de ejecución se presentó en este diagrama, en analogía con la estructura que se fue modelando desde el diagrama de navegación.



3.5 Resultados Obtenidos

Luego de haber concluido el desarrollo del caso de estudio, este arrojo los siguientes resultados.

Beneficios:

- La notación OMMMA-L, permite el desarrollo de sistemas multimedia de manera ágil y eficiente
- OMMMA-L cuenta con los artefactos necesarios para desarrollar un proceso completo y con todos los elementos necesarios para que el mismo cumpla con los elementos componentes de un sistema multimedia y satisfaga las necesidades del cliente
- Permite a los desarroladores de los proyectos multimedia una mayor comprensión del prototipo físico de la misma, esto esta dado por los diagramas de Presentación.
- Brinda facilidades a la hora de desarrollar los diferentes escenarios de la multimedia ya que están detalladamente especificados en los Diagramas de Comportamiento Interactivo.
- La notación con que son desarrollados los diagramas permite que estos sean comprendidos por los miembros no técnicos del equipo de desarrollo.
- La clasificación orientada a objetos de los estereotipos de los diferentes objetos ofrece una adecuada abstracción de su arquitectura.
- El lenguaje es fácilmente extensible a UML
- El proceso de modelado puede ser muy flexible

Inconvenientes:

- Esta notación es apropiada para sistemas multimedia de pequeña y mediana magnitud.
- Es un lenguaje de modelado relativamente nuevo, por lo tanto poco explotado y debido a esto son varias las aplicaciones multimedia que no son modeladas con el.

CONCLUSIONES GENERALES

A lo largo del presente trabajo se ha podido llegar a la conclusión de que con la utilización del lenguaje de modelado OMMMA-L es mucho más sencilla y comprensible la realización de determinados sistemas multimedia ya que esta notación es específica para la creación de este tipo de proceso de software y nos brinda todas las facilidades para una mayor comprensión de dicho proceso. Se obtuvo una noción más amplia de cómo modelar proyectos multimedia. Esta notación permite además que los miembros del equipo de realización de una multimedia que no sean especialistas en la materia puedan comprender e una manera mas sencilla todo lo que ocurre en cada fase o iteración dentro de su creación, ya que se ve de una manera detallada a través de la aplicación de este lenguaje de modelado.

Se puede plantear que de manera eficiente se ha dado cumplimiento al objetivo propuesto ya que a la multimedia Topografía de la asignatura de Ingeniería Civil que se había propuesto como caso de estudio se le aplicó este lenguaje de modelado obteniendo buenos resultados y teniendo una idea más amplia de cómo se lleva a cabo este proceso.

Se tuvo una idea más amplia de los diferentes lenguajes de modelado que existen y sus características fundamentales así como las principales aplicaciones que pueden tener los mismos. Se realizo una investigación a cerca de las principales características de los software educativos ya que para la realización del caso de estudio, ya que este tiene fines educativos. Conjuntamente se hizo alusión a los diferentes tipos de software existentes y sus principales particularidades, así como las tipologías fundamentales existentes y sus principales características.

RCOMENDACIONES

Sugerimos a las personas dedicadas a la realización de Aplicaciones Multimedia:

- La utilización de la notación OMMMA-L en la realización de sus proyectos, ya que este lenguaje de modelado es de gran eficiencia a la hora de modelarlos.
- Recomendamos un estudio más exhaustivo de OMMMA-L para ampliar el espectro de conocimientos de dicho lenguaje de modelado.

GLOSARIO DE TÉRMINOS

A:

APRENDIZAJE: El aprendizaje es uno de los procesos más importantes para la psicología científica actual. El aprendizaje es un cambio casi permanente en el comportamiento organismo, mediante el aprendizaje es posible modificar lo que se ha aprendido anteriormente. A diferencia de los animales que nacen con instrucciones genéticas para la supervivencia los humanos tenemos la capacidad de aprendizaje la cual nos da más flexibilidad para adaptarnos al medio ambiente, podemos aprender a resguardarnos de cambios climáticos y adaptarnos a cualquier ambiente, nuestra capacidad de aprendizaje nos permite afrontar cambios.

APRENDIZAJE COGNOSCITIVO: Trata de explicar como los animales y el hombre pueden aprender conductas nuevas sin experiencia previa, o como se pueden recordar respuestas de gran complejidad durante un periodo largo de tiempo y sin reforzamiento, o como se pueden realizar aprendizaje de gran complejidad. Se considera al organismo un ser activo capaz de elaborar la información y de generar conductas por motivaciones internas. Este aprendizaje subraya los aspectos cognitivos. Se basa en representaciones cognitivas de la conducta, en vez de la asociación de estímulos y respuestas. Sólo se da en especies animales superiores y en el hombre. El aprendizaje se puede realizar no solo por condicionamiento, sino que podemos aprender imitando a otros sujetos o simplemente al recibir la información de algo. Se llama aprendizaje vicario, observacional o por modelos.

APRENDIZAJE SIGNIFICATIVO: El aprendizaje significativo es un aprendizaje relacional. El sentido lo da la relación del nuevo conocimiento con: conocimientos anteriores, con situaciones cotidianas, con la propia experiencia, con situaciones reales

C:

COGNITIVO: Proceso exclusivamente intelectual que precede al aprendizaje, las capacidades cognitivas solo se aprecian en la acción, es decir primero se procesa información y después se analiza, se argumenta, se comprende y se produce nuevos enfoques. El desarrollo de lo cognitivo en el alumno debe ser el centro del proceso de

enseñanza por parte del docente. Este término es utilizado por la psicología moderna, concediendo mayor importancia a los aspectos intelectuales que a los afectivos y emocionales, en este sentido se tiene un doble significado: primero, se refiere a una representación conceptual de los objetos. La segunda, es la comprensión o explicación de los objetos.

COGNOSCITIVISMO: Rama de la psicología que se encarga del estudio del proceso de adquisición del aprendizaje.

CONSTRUCTIVISMO: Corriente que afirma que el conocimiento de todas las cosas es un proceso mental del individuo, que se desarrolla de manera interna conforme el individuo obtiene información e interactúa con su entorno.

H:

HIPERMEDIA: Surge como resultado de la fusión de dos tecnologías, el hipertexto y la multimedia.

HIPERTEXTO: es la organización de una determinada información en diferentes nodos, conectados entre sí a través de enlaces.

I:

IEEE: Corresponde a las siglas del Instituto de Ingenieros Eléctricos y Electrónicos,. Es una asociación estadounidense dedicada a la estandarización internacional sin fines de lucro formada por profesionales de las nuevas tecnologías.

L:

LENGUAJE DE MODELADO DE OBJETOS: Es un conjunto estandarizado de símbolos y de modos de disponerlos para modelar (parte de) un diseño de software orientado a objetos.

M:

METACOGNITIVO: Aprender a aprender estrategias que requieran una conciencia individual y Proceso exclusivamente intelectual que precede al aprendizaje, la regulación de los procesos cognitivos utilizados.

S:

SOFTWARE: Es la suma total de los programas de cómputo, procedimientos, reglas documentación y datos asociados que forman parte de las operaciones de un sistema de cómputo

INGENIERÍA DE SOFTWARE: Es la rama de la ingeniería que aplica los principios de la ciencia de la computación y las matemáticas para lograr soluciones costo-efectivas (eficaces en costo o económicas) a los problemas de desarrollo de software

SOFTWARE EDUCATIVO (SWE.): Se define como un programa automatizado diseñado con una intencionalidad educativa para ser utilizado en el proceso de aprendizaje, utiliza procedimientos para que el estudiante aprenda, se fomenta el análisis de problemas, facilita el trabajo en grupo, provee soporte en actividades docentes y en el sentido más amplio, mejora las habilidades del pensamiento y la resolución de problemas.

T:

TIC: Tecnologías de la Información y las Comunicaciones

REFERENCIAS BIBLIOGRÁFICAS

[Booch, 1986] Booch, G. 1988. Object Oriented Development. Trans. of Soft. Eng. Vol. SE-12. Num. 2. Feb. 1986. pp. 211-221.

[Booch 1998] Booch G. 1998. Software Architecture and the UML. Presentación disponible en: http://www.rational.com/uml como arch.zip.

[Cota, 1994] Cota A. 1994 "Ingeniería de Software". Soluciones Avanzadas. Julio de 1994. pp. 5-13.

[Greiff, 1994] Greiff W. R. Paradigma vs Metodología; El Caso de la POO (Parte II). *Soluciones Avanzadas*. Ene-Feb 1994. pp. 31-39.

[Jacobson, 1998] Jacobson, I. 1998. "Applying UML in The Unified Process" Presentación. Rational Software. Presentación disponible en http://www.rational.com/uml como UMLconf.zip

[Lewis G, 1994] "What is Software Engineering?" DataPro (4015). Feb 1994. pp. 1-10.

[M&R 1998] Microsoft y Rational 1998. *A White Paper on the Benefits of Integrating Microsoft Solutions Framework and The Rational Process.* Rational Software Corporation y Microsoft Corporation. Documento msfratprocs.doc Disponible en http://www.rational.com/uml/papers.

BIBLIOGRAFÍA

[1] IEEE Std, IEEE Software Engineering Standard: Glossary of Software Engineering Terminology. IEEE Computer Society Press, 1993.

[2] Cota A. 1994 "Ingeniería de Software". Soluciones Avanzadas. Julio de 1994. pp. 5-13.

[3] La Ingeniería del Software, término utilizado por primera vez por Fritz Bauer en la primera conferencia sobre desarrollo de software patrocinada por el Comité de Ciencia de la OTAN celebrada en Garmisch, Alemania, en octubre de 1968, según Alan Davis.

[4] E. Yu. Modelling Strategic Relationships for Process Reengineering. PhD. thesis, University of Toronto, 1995.

[5] J. Mylopoulos, A. Borgida, M. Jarke, M. Koubarakis. “Telos: Representing Knowledge about Information Systems”. ACM Transactions Information Systems, 8 (4). October, 1990.

[6] L. Chung, B.A. Nixon, E. Yu, J. Mylopoulos Non-Functional Requirements in Software Engineering. Kluwer Academic Publishers, 2000.

[7] D. Amyot, G. Mussbacher. “URN: Towards a New Standard for the Visual Description of Requirements”. Telecommunications and beyond: The Broader Applicability of SDL and MSC. Third International Workshop (SAM 2002). June 24-26, 2002, Aberystwyth, UK.

[8] Z.151 (GRL). International Telecommunication Union (ITU). September 2003.

[9] ADARRAGA, Pablo (1985). Criterios educacionales en la selección de software. En PFEIFFER, Amalia; GALVÁN, Jesus. *Informática y Escuela*. Madrid: Fundesco

[10] BARTOLOMÉ, Antonio. (1994). Sistemas Multimedia. En SANCHO, Joana Mª y otros. (1994). Para una Tecnología Educativa. Madrid: Horsori.

[11] Cota A. 1994 "Ingeniería de Software". Soluciones Avanzadas. Julio de 1994. pp. 5-13.

[12] Jacobson, I. 1998. "Applying UML in The Unified Process" Presentación. Rational Software

[13] J. Mylopoulos, A. Borgida, M. Jarke, M. Koubarakis. “Telos: Representing Knowledge about Information Systems”. ACM Transactions Information Systems, 8 (4). October, 1990.

[14] E. Yu. Modelling Strategic Relationships for Process Reengineering. PhD. thesis, University of Toronto, 1995.

[15] Z.151 (GRL). International Telecommunication Union (ITU). September 2003.

[16] Página web de la herramienta OME3: http://www.cs.toronto.edu/km/ome. Fecha de consulta: Julio 2004.

[17] St. Sauer and G. Engels. Extending UML for modelling of multimedia applications. In M. Hirakawa and P. Mussio, editors, Proc. IEEE Symposium on Visual Languages (VL’99), pages 80–87, September 13-16, 1999.

[18] Sauer, Stefan-Engels Gregor; UML-Based behavior Specification of Interactive multimedia Applications (Febrero 2007).

[19] Sauer, Stefan-Engels Gregor, MVC-Based Modeling Support for Embedded Real-Time Systems (Marzo-2007).

ANEXOS

Capítulo 1

Fig. 1

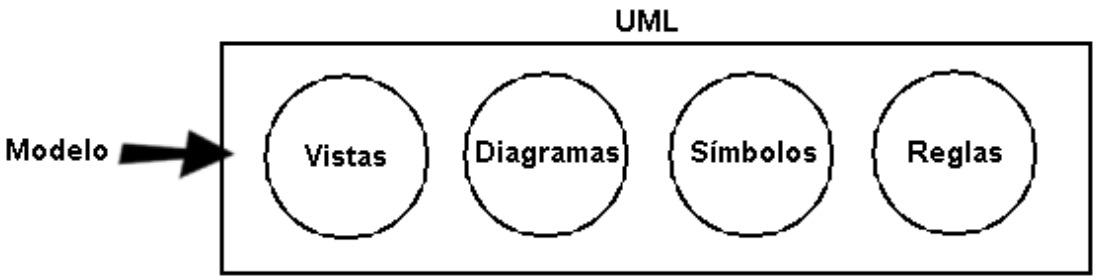


Fig. 2 “Modelo UML”

Capítulo 2

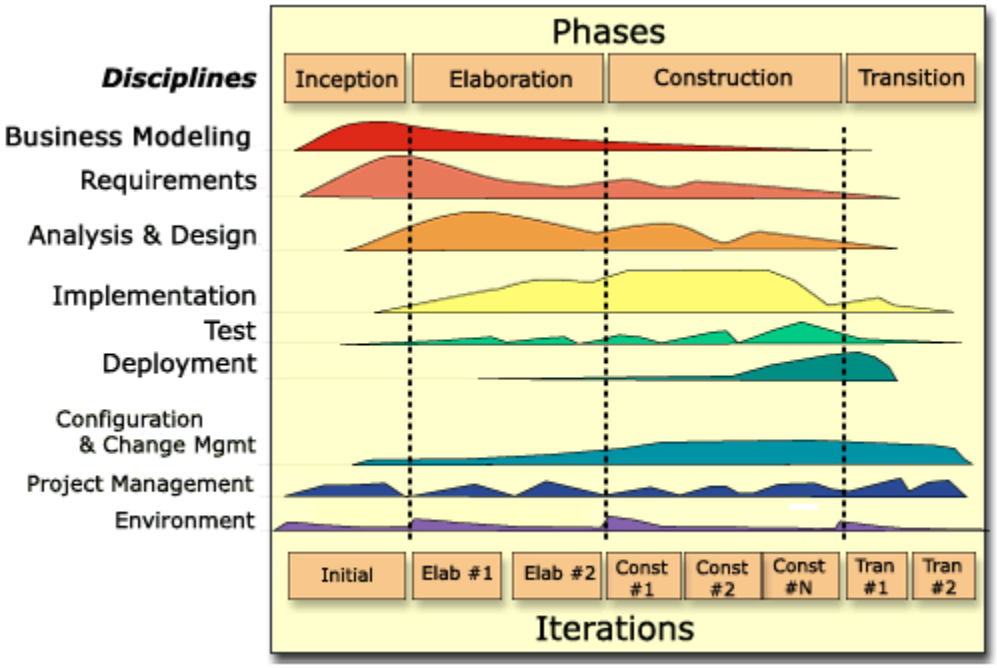


Fig. 3 “Fases de RUP”

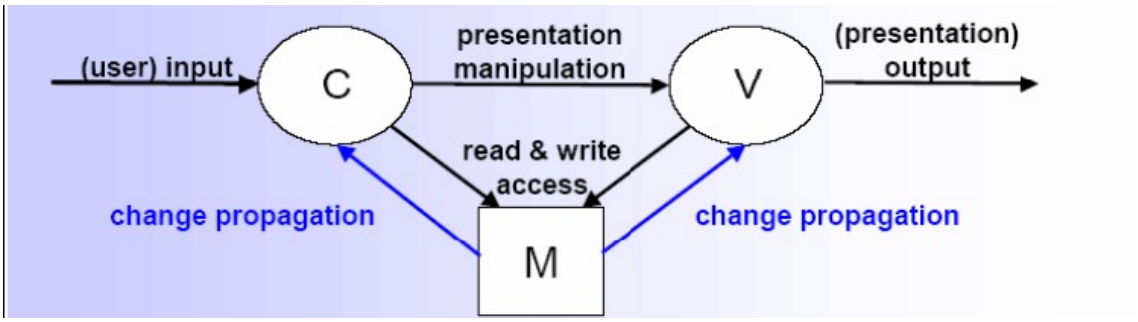


Fig. 4 “Patrón Modelo Vista Controlador”

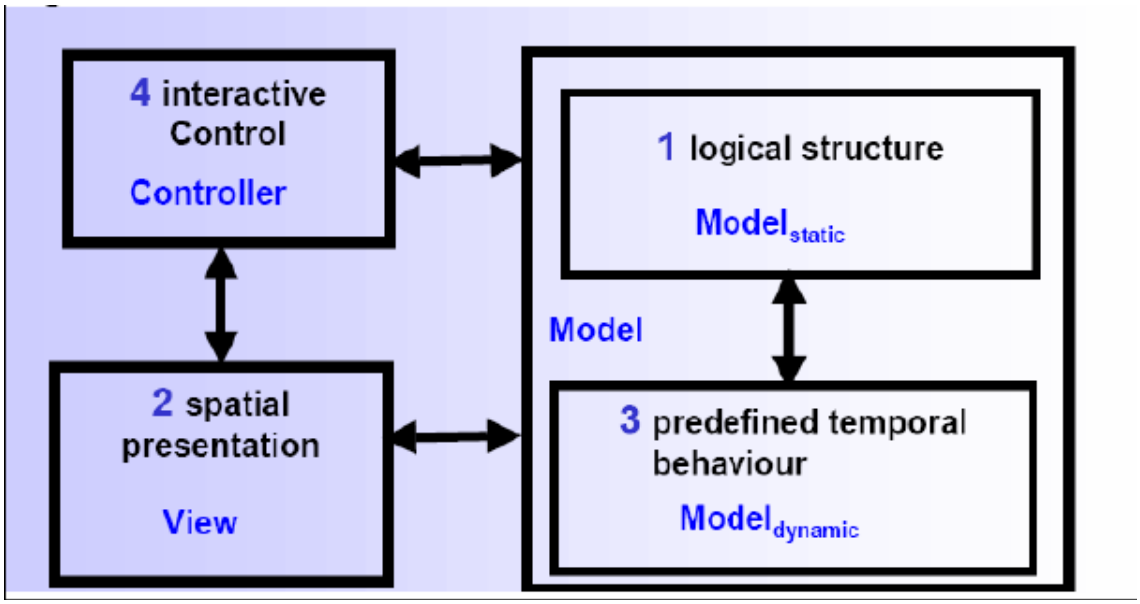


Fig. 3 “Patrón Modelo Vista Controlador para Multimedia”

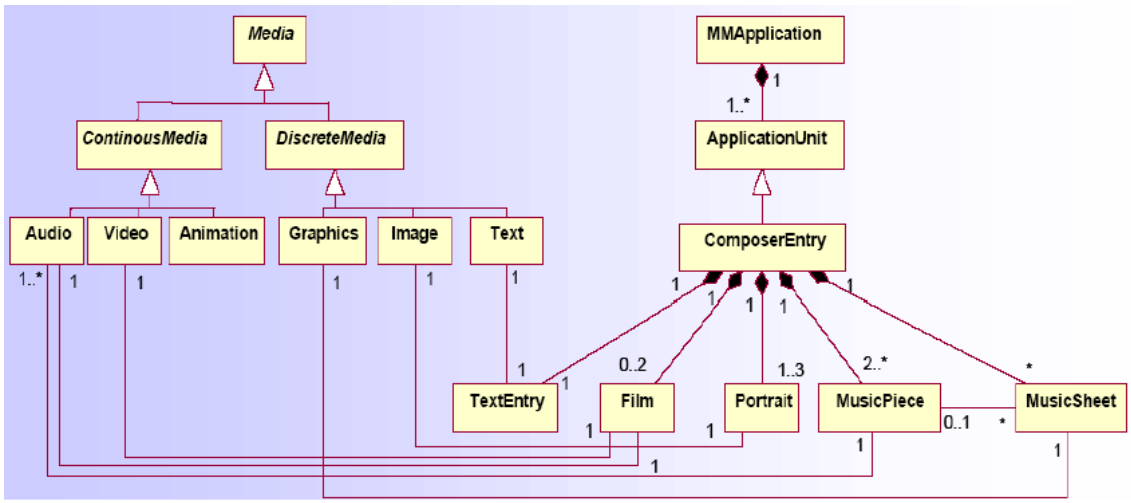


Fig. 4 “Mapeo de clases en OMMMA-L”

Capítulo 3

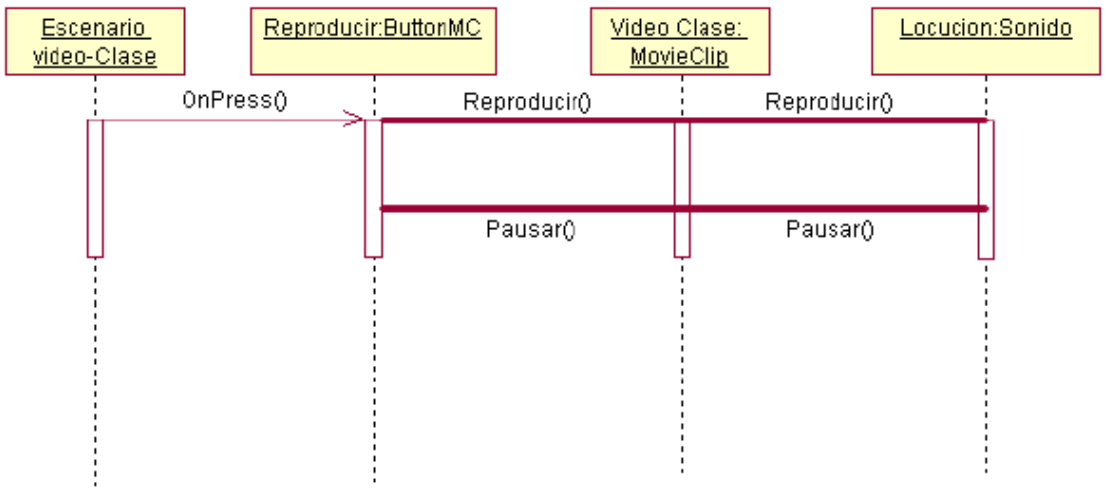


Fig. 5 “Diagrama de Comportamiento Temporal”