UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

VICERRECTORÍA DE FORMACIÓN

DIRECCIÓN DE FORMACIÓN POSTGRADUADA

Tesis en opción al título de Máster en Informática Aplicada

ApEM – L como una nueva solución a la modelación de aplicaciones educativas multimedia en la UCI

Autor:
Ing. Febe Angel Ciudad Ricardo

Tutora:
Dra. Ailyn Febles Estrada

La Habana, Julio de 2007
"Año del 49 aniversario de la Revolución"

Así, digno y libre,
indiferente y sabio,
conocedor de los demás y
y de sí mismo, a la par
instruido e inspirado, así
ha de ser el que en
nuestros días quiera
robar una estrella más al
cielo para dejarla en la
tierra perpetuamente
atada a su nombre.

José Martí
(Publicado en "El federalista" [edición literaria]
México, febrero 11 de 1876)

DEDICATORIA

A mi abuela Ángela y mis abuelos Pancho y Alberto, sin quienes no hubiese sido posible que hoy estuviera presentando esta memoria escrita.

A mi abuelita Bena, quien allá en Banes, Holguín, no hace más que preguntar por mí cada vez que ve entrar por la puerta a mis padres y nunca ha olvidado mandarme un gran beso desde el fondo de su corazón.

A mi padre, quien me profesa un amor incondicional y a quien debo la enseñanza de los principios por los que me rijo en la vida.

A mi querida madre, quien con tanta ternura y con su paciencia infinita siempre me da fuerzas para seguir adelante y confiar en mis conocimientos.

A mi hermana que constantemente me anima a seguir adelante por el sendero de la vida tanto profesional como personal y a quien le estaré eternamente agradecido por tanta ayuda.

A Cosita, quien a pesar de las peleas, enternece mi alma con sus palabras y hace de mis amaneceres una nueva esperanza a la vida y el futuro feliz.

A Juan, quien en los últimos años de su vida me brindo tanto cariño y confió siempre en la realización de mis sueños, y en los que como él decía aún faltan por cumplir.

A mi padrino Miguelito, alguien que me hace sentir como su hijo y que no escatima esfuerzos cuando me son necesarios.

AGRADECIMIENTOS

Es difícil decir en pocas palabras el agradecimiento que uno siente hacia tantas personas en momentos como este y pecamos irremediablemente de olvidar a alguno de esos o esas que algo aportaron en este logro intermedio en mi carrera profesional. No obstante trataré de no olvidar a nadie y apelar a la sabiduría de mi memoria de poner en este papel a los que más cercanos estuvieron en estos últimos meses de arduo trabajo.

A mi cuñado Manuel Alvarez Pérez, quien hace más de 15 años junto a mi hermana, nunca ha dejado de brindarme apoyo cuando lo he necesitado y no sabe cuanto le debo en este momento. A mi hermanita Sheralin Monrose, quién aún en la distancia, es promotora de esta investigación y partícipe de ella. A mi hermana de alma, Yanet Almaguer Díaz y su madre, por sus locuras de siempre y lograr refrescar mi alma con su alegría. A mis grandes amigas Yunaisy Oliva Ríos y la Negra, por esos cafés y esos tés en los momentos de gran estrés, no saben cuanto me aliviaron. A mis compañeras de maestría y amigas de la universidad Yaneisis Pérez Heredia y Maricela Bourzac Suárez, quienes compartían las noches de insomnio y las buenas y malas noticias de estos dos años de intensos estudios, mil gracias Yano y Mari, nunca tendré como pagarles. A mi amiga Angelina y su madre Verónica, que aunque lejos no deja de entrar al chat para saber de mí y preguntar me por la maestría y su madre de llamarme por teléfono cuando me desaparezco mucho tiempo. A mis compañeros de trabajo diario, quienes siempre han aportado algún que otro criterio técnico que hoy forma parte del cuerpo de esta tesis: a Jorge, Quique, Yoenis, Alden y Gema en especial. A Elianis, quien confía en que llegaré a Doctor y me lo recuerda con el mismo saludo siempre, mil gracias mi amorcito, con respeto a Isbel claro. A una de mis madres en La Habana, doña Yanet Villanueva Armenteros, quien no ha dejado de estar ahí en los momentos más crudos para apoyarme y me dice siempre que menos mal que no me robé el título de oro como ingeniero, sino….a Dunia Zaldivar Ramírez que me alegra tanto las mañanas y me eleva tanto la autoestima con sus saludos de "hay mi madre, se alborota La Habana cuando te veo" y Olivia Ramírez, quién me ayudó mucho al llegar a la UCI y me abrió sus brazos incondicionalmente. A Lenna y Diana, quienes me soportan a diario y no se cansan, gracias por las encuestas, no saben cuanto les agradezco. A mis vecinos de edificio por nunca decir no cuando llegaba con el vasito y preguntaba ¿me pueden regalar un poquito de azúcar para un te que esta maestría me tiene fundío? A Aliosmi López Velázquez, quien además de alumna es una gran amiga y me ha brindado, aún con su juventud, grandes consejos y buenos caminos a seguir.

A Nurileidis Almeida Cintra, mi negra contenta, quién me ha demostrado que en la vida todo se

puede cuando se le pone empeño y que siempre hay alguien más chivao que uno, como me dice siempre. A los estudiantes del grupo 9505, por ser mis mejores alumnos en la UCI y permitirme compartir con ellos tantos momentos felices y tristes a la vez, gracias por confiar tanto en mí y darme tanto ánimo cuando me decían: profe, ¡como usted no hay dos! A mis amigos Luis Raciel Rodríguez Silva, Yolanda Zulaima Mejías García, Nara Lidia Pérez Solá y Dialina Emelina Luis García por tanto apoyo siempre. A mis amigos Roidel y Rodolfo, por darme en los últimos meses tanto ánimo y abrirme las puertas de su casa para sitio de reposo y descanso de mi alma y refugio de tranquilidad. A Yosnel, quien en la UCI es un gran hermanito menor, como el principito que siempre está presente y me recuerda lo correcto como mi conciencia, gracias chamo por tanto apoyo. A mis compañeros de apartamento y de tantos años de estudio y de beca, Oleysis Socarrás Sosa, Amado Espinosa Hidalgo y Rafael Rodríguez Montero, por tanto apoyo incondicional y ser mis hermanos en la Uci, les estaré eternamente agradecidos. A Jesús Mesa, que no duda en apoyarme, pero sin dejar de decirme cuando las cosas están mal hechas. A mis amistades de años y años en Holguín: Elisa, Yudiasy, Magdalenis, Miladys, y Héctor por el apoyo brindado y los excelentes ratos que pasamos juntos cada vez que visito mi ciudad. A mis profesores que desde la universidad han forjado en mí esa alma de investigador: Dr. Mauro García, Dr. Félix Rodríguez, MSc. Pedro Escalona, Dr. Carlos Cruz y muy en especial para ese pedagogo por excelencia y maestro de todos los tiempos, Dr. Manuel Eduardo Mariño Betancourt, quien supo y sabe actualmente conducirte por los caminos de la enseñanza con una sabiduría y paciencia infinitas, confiando siempre en que sabrá sacar el genio que uno lleva dentro escondido, lleguen a él mis más grandes agradecimientos. A mi amigo Henryt por los correos con los chistes. A Abelito por decirme la verdad cuando la merecía y al mismo tiempo brindarme su hombro como sostén en los momentos difíciles. A mi jefe y Decano, Dr. José Ortiz Rojas, que aún con los encontronazos que tenemos, no permite que se laceren nuestras relaciones profesionales y personales. Un gran agradecimiento a mi Tutora de maestría, la Dra. Ailyn Febles Estrada, quien ha tenido la paciencia del siglo para soportar mi insistencia en las revisiones y encuentros de trabajo y mi testarudez en entender sus criterios siempre acertados, un beso infinito y enorme para ti. A mis tesistas, Yordanys, Jacqueine, Nuri, Mirelys, Yudi, Yuniersy, Ana, Daymeris y Erick, quienes me ayudaron a obtener este resultado y si cuyo apoyo incondicional cada vez que lo necesité no hubiese sido posible. Gracias Yordanys por las revisiones bibliográficas tan buenas y el apoyo y confianza que siempre me diste. A la Revolución y al Comandante en Jefe Fidel Castro Ruz, que me han dado la posibilidad primero de formarme como Revolucionario, luego como Ingeniero, trabajar en un proyecto

tan bello como la UCI y superarme día a día para servir mejor a mi patria y hacer honor a los grandes héroes de nuestra historia. Gracias a los que por olvido no mencioné y siempre preguntaban al verme: y la tesis ¿cómo va? Gracias a todas y todos por todo.

DECLARACIÓN JURADA DE AUTORÍA

Yo Febe Ángel Ciudad Ricardo, con carné de identidad 79012518147, declaro que soy el autor principal del resultado que expongo en la presente memoria titulada *"ApEM – L como una nueva solución a la modelación de aplicaciones educativas en la UCI."* para optar por el título de Máster en Informática Aplicada.

Este trabajo fue desarrollado en el período comprendido entre el mes de enero de 2006 y junio de 2007, en colaboración con los estudiantes diplomantes: Yordanys Piñeiro Gómez, Jacqueline Gallardo Collazo, Nurileidis Almeida Cintra, Yoannia Viera Cisneros, Erick Rodríguez Figueredo, Mirelys Hidalgo Ricardo, Yudisleydis Peña Lemus, Yuniersy Hernández Díaz, Ana Bárbara Prado Baldor y Daymeri Díaz Rodríguez; quienes me reconocen la autoría principal del resultado expuesto en esta memoria.

Finalmente declaro que todo lo anteriormente expuesto se ajusta a la verdad, y asumo la responsabilidad moral y jurídica que se derive de este juramento profesional.

Y para que así conste, firmo la presente declaración jurada de autoría en Ciudad de la Habana a los 5 días del mes de Julio del año 2007.

Autor Principal: Ing. Febe Ángel Ciudad Ricardo. _____

Tutora: Dra. Ailyn Febles Estrada _____

Colaboradores:

Yordanys Piñeiro Gómez _____

Jacqueline Gallardo Collazo _____

Nurileidis Almeida Cintra _____

Yoannia Viera Cisneros _____

Erick Rodríguez Figueredo _____

Mirelys Hidalgo Ricardo _____

Yudisleydis Peña Lemus _____

Yuniersy Hernández Díaz _____

Ana Bárbara Prado Baldor _____

Daymeri Díaz Rodríguez _____

DATOS DE CONTACTO

Ing. Febe Ángel Ciudad Ricardo.

Graduado de Ingeniería en Informática en el Instituto Superior Politécnico “José Antonio Echavarría” (IPSJAE) en conjunto con la Universidad de Holguín “Oscar Lucero Moya” (UHo) en el año 2004.

Profesor de las asignaturas: Ingeniería de Software I y II, Gestión de Software, Metodología de la Investigación Científica y Seminario de Tesis.

Centro de Trabajo: Universidad de las Ciencias Informáticas (UCI). Departamento de la Especialidad de la Facultad 9

Cargo: Jefe del Departamento.

Dirección: Carretera a San Antonio, Km 5 ½, Reparto Torrens, La Lisa, Ciudad de La Habana.

Teléfono: + (53)(7) 8372557.

Correo electrónico: fciudad@uci.cu

Dra. Ailyn Febles Estrada.

Graduada de Licenciatura en Ciencias de la Computación en la Universidad de La Habana (UH).

Profesora de las asignaturas pertenecientes a las disciplinas de Programación e Ingeniería y Gestión de Software.

Centro de Trabajo: Universidad de las Ciencias Informáticas (UCI). Dirección de Calidad.

Cargo: Directora.

Dirección: Carretera a San Antonio, Km 5 ½, Reparto Torrens, La Lisa, Ciudad de La Habana.

Teléfono: + (53)(7) 8352427.

Correo electrónico: ailyn@uci.cu

RESUMEN

A lo largo de la historia, la Ingeniería de Software (ISW) ha estado determinada por varias etapas marcadas por diferentes paradigmas de programación, como el estructurado, el orientado a eventos y el orientado a objetos. Este último tiene gran aceptación por las facilidades que brinda en la conceptualización y representación del entorno software. Asimismo, este desarrollo en los lenguajes, hasta llegar a los Lenguajes de Alto Nivel (H.L.L por sus siglas en inglés correspondientes a High Level Language) y las técnicas de programación, estuvo acompañado por la concepción de metodologías de desarrollo, que en sus inicios tenían asociadas las notaciones para la representación de los respectivos productos informáticos. Con el avance en los H.L.L. se comenzaron a dividir los caminos de las notaciones y las metodologías y procesos de desarrollo, surgiendo RUP y UML, ambos con una aceptación generalizada.

El ámbito informático actual para el desarrollo de software educativo, establece grandes retos a estos estándares, entre los que se encuentran la representación de las características pedagógicas y funcionales de este tipo de aplicaciones, máxime con las características y los pilares de la pedagogía cubana. En la última década, se extendió el lenguaje notacional UML a OMMMA-L para el tratamiento de aplicaciones educativas multimedia, respetando igualmente los estándares establecidos ya por el primer lenguaje mencionado, el cual ha sido de gran aceptación. No obstante, por las propias condiciones descritas anteriormente, uno de los grandes retos para el establecimiento y utilización definitiva de este lenguaje es la no posibilidad de representación de la totalidad de las características del software educativo cubano y sus entornos de desarrollo.

En el trabajo que se presenta a continuación se abordan las notaciones que hoy existen para la representación de este tipo de aplicaciones, así como las características más fundamentales del software educativo cubano actual, para abarcar luego el tema más controvertido: un lenguaje notacional para este tipo de aplicaciones. La comparación entre las notaciones se desarrolla en cuanto a aspectos como las vistas de representación del comportamiento, definición de funciones del software, utilización de estándares y tratamiento del detalle. Como finalidad del mismo se presenta una propuesta de lenguaje para la modelación de aplicaciones educativas en el contexto productivo cubano sobre la base del estudio mencionado.

PALABRAS CLAVES

Notaciones, Software Educativo, Lenguaje de modelación, ApEM–L.

INDICE DE TABLAS Y FIGURAS

INDICE

Introducción

INTRODUCCIÓN

En el mensaje a los lectores que sirve de introducción al texto “Industria Cubana del Software. Perfil y Perspectivas” publicado por el Grupo de Promoción y publicidad de dicha industria en el año 2003, el Ing. Ignacio González Planas plantea: “Cuba ha dado importantes pasos en los años recientes para promover el uso masivo de las tecnologías de la información y las telecomunicaciones en el gobierno, la economía y los servicios” (MIC, 2003). Más adelante en el propio texto aparece: “La industria del software, constituye un sector de alta prioridad, especialmente aplicada a los sectores de Educación y Salud. (…) Se elabora una estrategia nacional para convertir a Cuba en una “Nación de Excelencia” en la producción de software y servicios de tecnologías de la información, que la convierta en referente en la región por la calidad de sus soluciones y relación costo/beneficio.” (MIC, 2003).

Como pasos para el cumplimiento de estos propósitos el gobierno cubano ha ejecutado acciones como: inserción en las enseñanzas primaria, secundaria y media superior de tecnologías (computadoras personales, televisores y reproductores de video) que potencien el uso de las Tecnologías de la Información y las Comunicaciones (TIC); aumento del número de instalaciones Joven Club de Computación al doble en todo el país y la creación en septiembre del 2002, de la Universidad de las Ciencias Informáticas (UCI); última acción esta, encaminada también a promover la Industria Cubana del Software (InCuSoft) dentro de sus objetivos.

Precisamente por ser la UCI el “(…) punto de enlace entre el Sistema de Centros de Educación Superior y de la Ciencia con la Industria propiciando la transferencia tecnológica” (MIC, 2003) es donde mejores condiciones creadas existen para unificar los esfuerzos y conocimientos existentes en el desarrollo de software educativos, impulsando la inserción continua de mejoras en el proceso de creación de este tipo de aplicaciones y en especial la modelación de las mismas; así como sus análisis y sus diseños.

“En Cuba se acumulan más de 20 años de experiencia en la elaboración de software educativo. En estos resultados han tenido una importante participación los Ministerios de Educación y Educación Superior por medio de los Centros de estudio de software y las Universidades respectivamente, así como la organización de Joven Club de conjunto con el sector empresarial”. (MIC, 2003)

En el transcurso de esos 20 años de experiencia acumulados en el desarrollo de software educativo en Cuba, el resto de la producción de software y en menos medida el desarrollo de estos software educativos, ha transitado por diferentes paradigmas en la programación como son: el estructurado, el orientado a eventos y el orientado a objetos. Las principales preocupaciones en este tiempo no han

estado en las caraterísticas informáticas (dicho de algún modo) de este tipo de aplicaciones, sino en sus posibilidades educativas y pedagógicas.

Teniendo en cuenta que hasta inicios de la última década, los profesionales dedicados al desarrollo de este tipo de software fueron los propios profesionales del área pedagógica que por las necesidades de la ensañanza comenzarona a incursionar en el desarrollo de software educativos; es de fácil conclusión que no se dedicaba el suficiente tiempo a la modelación de este tipo de producto y la documentación de estos para posteriores trabajos ingenieriles.

Durante las dos pasadas décadas, el mismo tiempo de experiencia cubana en el desarrollo de software educativo; se han desarrollado un gran número de métodos de modelado. Los investigadores han identificado los problemas del análisis y sus causas y han desarrollado varias notaciones de modelado y sus correspondientes conjuntos de heurísticas para solucionarlos. Cada método de análisis tiene su punto de vista. Sin embargo, todos los métodos de análisis se relacionan por un conjunto de principios operativos: (Pressman, 2002)

1. Debe representarse y entenderse el dominio de información de un problema.
2. Deben definirse las funciones que debe realizar el software.
3. Debe representarse el comportamiento del software (como consecuencia de acontecimientos externos).
4. Deben dividirse los modelos que representan información, función y comportamiento de manera que se descubran los detalles por capas (o jerárquicamente).
5. El proceso de análisis debería ir desde la información esencial hasta el detalle de la implementación.

A lo largo del tiempo mencionado aparecieron en la industria internacional de software metodologías como la Relationship Management Methodology (RMM) para la guía de procesos de desarrollo de aplicaciones con tecnología multimedia; la técnica más utilizada en el desarrollo de aplicaciones de corte educativo, que anexaban una notación para el modelado de este tipo de software. Cumplió su objetivo durante un buen tiempo, pero tecnológicamente no logra responder a todas nuestras necesidades actuales en la representación computacional de las soluciones informáticas educativas. De manera similar, en Cuba se trabajó en nuestras necesidades surgiendo la metodología MultiMet, que igualmente traía asociada un lenguaje de representación de este tipo de software.

Al mismo tiempo, como ya fue mencionado, evolucionaron las técnicas y lenguajes de programación, y con estas mejoraron las formas de programar el software educativo, llegando en la actualidad a

utilizar en mayor medida técnicas orientadas a eventos y en sus mejores exponentes orientados a objetos completamente.

La presentación en el año 1998 por sus autores: I. Jacobson, G. Booch y J. Rumbaugh de la primera versión del Lenguaje Unificado de Modelado (UML) para la modelación de software de propósito general orientado a objetos, fue de un gran impacto en la industria internacional y tuvo una excelente aceptación, al punto de convertirse en un estándar mundial en este sentido.

"Desafortunadamente, UML no soporta todos los aspectos de las aplicaciones multimedia de una forma adecuada e intuitiva. Especialmente, las características del lenguaje para modelar los aspectos de la interfaz de usuario, no se aplican explícitamente en los entornos multimedia. Otros conceptos de UML no son lo formalmente aplicables a la multimedia y de ser utilizados tal y como han sido planteados complicarían la modelación de este tipo de aplicaciones. Por estas razones, y gracias a las facilidades de extensión, si bien permitidas en UML, y he aquí su riqueza como lenguaje de modelado, es que sus principales conceptos y notaciones son aplicables a los entornos multimedia, más se hizo necesario el desarrollo de una extensión para este tipo de aplicaciones denominada Lenguaje Orientada a Objetos para la Modelación de Aplicaciones Multimedia (OMMMA-L), que facilita el modelado de un gran rango de aspectos de aplicaciones multimedia interactivas de una forma integrada y comprensiva." (Sauer, y otros, 2000?)

El software educativo cubano, como respuesta y representante del sistema educacional de la isla, posee un conjunto de características que lo diferencian sustancialmente del que se produce en el resto del mundo; potenciando áreas como:

- Procesamiento de volúmenes de información considerables en varios formatos de presentación (animaciones, videos, imágenes, sonidos, textos e hipertextos).
- Utilización de técnicas de Inteligencia Artificial (IA) para el tratamiento de entornos educacionales a la medida del usuario.
- Concepción de seguimientos pedagógicos o desarrollo de la traza de utilización y avance cognitivo del usuario, lo que inplica la utilización de bases de datos.
- Posibilidad de retroalimentación del tutor o profesor guía del proceso docente educativo donde se utiliza el software.
- Posibilidad del profesor de adecuar el funcionamiento de la aplicación a las caraterísticas de cada uno de los estudiantes o grupos de estos.

- Posibilidad del profesor de adicionar elementos multimedia al software para un mejoramiento y actualización del mismo.

Si señalamos además que de los diferentes tipos de software educativo existentes según la clasificación ofrecida por (Marqués, 1996?): tutoriales, bases de datos, simuladores, constructores y programas herramienta; el software educativo cubano es un producto que fusiona elementos representativos de cada uno de los tipos específicos mencionados anteriormente; aumentando considerablemente la complejidad del tipo de software y su modelación, hecho en Cuba.

"Durante la actividad de evaluación y síntesis de la solución, el analista crea modelos del sistema en un esfuerzo de entender mejor el flujo de datos y de control, el tratamiento funcional y el comportamiento operativo y el contenido de la información. El modelo sirve como fundamento para el diseño del software y como base para la creación de una especificación del software." (Pressman, 2002)

Estas funciones descritas por Pressman en el párrafo anterior, son las que corresponden a los especialistas de la UCI: convertir en modelos ingenieriles informáticos, con capacidad de diseño e implementación, las necesidades y requerimientos de los software educativos solicitados por las diferentes instituciones nacionales e internacionales. Para ellos se hace necesario la utilización de un lenguaje notacional que logre representar en modelos: la estructura lógica, el comportamiento y las funciones del futuro software a desarrollar enmarcadas en un determinado contexto pedagógico.

La Universidad de las Ciencias Informáticas, al ser hoy un centro con solo 5 años de creado, consta con un personal especializado de pocos años de experiencia en su mayoría, inmerso en el impulso de la InCuSoft, aplicando las mejores prácticas de la ingeniería informática en su totalidad e investigando en aspectos relacionados con la generalización de soluciones y adecuación de estas al contexto productivo UCI.

A raíz de las condiciones descritas hasta el momento, se identificó como ***problema científico*** de esta investigación: escases de notaciones que permitan representar los elementos estructurales, lógicos, funcionales, pedagógicos y de patrones de ingeniería del software educativo en la UCI, lo que produce una deficiente documentación y modelación de este tipo de software.

Se plantea como ***hipótesis*** de trabajo la siguiente: Contar con una notación para representar los elementos estructurales, lógicos, funcionales, pedagógicos y de patrones de ingeniería al modelar los software educativos en la UCI, permitirá una mejor documentación técnica y una mas clara representación ingenieril en este tipo de aplicaciones.

El problema descrito genera como ***objeto de estudio*** de esta investigación: las notaciones y lenguajes de modelación del software educativo, enmarcando los resultados científicos de la misma en el área de la documentación del software educativo en la UCI como ***campo de acción***.

Para darle solución al problema científico mencionado se planteó como ***objetivo general*** del presente trabajo: Diseñar una notación para la modelación del software educativo; a partir de las notaciones existentes, las mejores prácticas en la modelación de este tipo de aplicaciones y la adecuación al contexto productivo UCI.

Al mismo tiempo se plantearon los siguientes ***objetivos específicos*** para un mejor logro del objetivo general descrito anteriormente:

1. Definir los elementos estructurales y funcionales significativos del software educativo producido en la UCI.
2. Identificar las insuficiencias actuales de las notaciones utilizadas en el desarrollo de software educativo en Cuba.
3. Definir los puntos de interacción o comunes de las notaciones utilizadas actualmente en la producción de de software educativo con tecnología multimedia e hipermedia.
4. Validar la notación propuesta con su aplicación en un proyecto productivo real.

La investigación científica se guió por el conjunto de ***tareas*** que se describen a continuación para darle cumplimiento a los objetivos tanto general como específicos planteados:

1. Estudiar el estado actual de la producción del software educativo en Cuba.
2. Analizar y evaluar las formas utilizadas por las notaciones estándares para la representación de los software.
3. Identificar los elementos reutilizables en las notaciones estándares para la representación de los software.
4. Examinar y evaluar las notaciones utilizadas actualmente en el desarrollo del software educativo en el mundo.
5. Examinar y evaluar las notaciones utilizadas actualmente en el desarrollo del software educativo en Cuba.
6. Estudiar las capacidades de extensibilidad de las notaciones utilizadas actualmente en el desarrollo del software educativo en Cuba y el mundo.
7. Evaluar los flujos de producción de software educativo utilizados en Cuba.

8. Identificar y evaluar los elementos pedagógicos significativos y comunes en los software educativos cubanos.
9. Estudiar e identificar las técnicas de programación más comunes en el desarrollo del software educativo cubano.
10. Estudiar e identificar los patrones de ingeniería más comunes en el desarrollo del software educativo cubano.
11. Definir los elementos estructurales y funcionales, así como los patrones de ingeniería a utilizar en la propuesta de notación.
12. Diseñar los elementos notacionales a partir de los elementos estructurales y funcionales identificados.
13. Validar la eficacia de la solución propuesta implantando la mima en un proyecto productivo de software educativo en la Universidad de las Ciencias Informáticas (UCI).

En el desarrollo de la investigación científica se han utilizado un conjunto de ***métodos científicos*** para la obtención, procesamiento y llegada a conclusiones. Dentro de estos podemos mencionar los siguientes:

- Métodos Teóricos:
  1. Hipotético – deductivo: para la determinación de la idea a defender de esta investigación y proponer nuevas líneas de trabajo a partir de los resultados parciales.
  2. Sistémico: para lograr que los diferentes elementos que forman parte del entorno de la notación descrita funcionen de manera armónica e igualmente garantizar el acoplamiento con otros lenguajes o notaciones ya existentes.
  3. Histórico – lógico y Dialéctico: para el estudio crítico de los trabajos anteriores y para utilizar estos como punto de referencia y comparación de los resultados alcanzados.
  4. Analítico – sintético: al descomponer el problema de investigación en elementos por separado y profundizar en el estudio de cada uno de ellos, para luego sintetizarlos en la solución propuesta.
- Métodos Empíricos:
  1. Observación: para obtener el registro del avance del proyecto productivo que sirvió de muestra para el caso de estudio desarrollado, en la clasificación y consignación de los acontecimientos pertinentes para la investigación.

2. Entrevista: utilizado para obtener información verbal y real de distintos miembros internos y externos a la actividad de modelación en el proyecto, así como a los especialistas pedagógicos inmersos en la investigación.
3. Encuesta: para conformar las estadísticas de la situación actual con las notaciones existentes, así como las de impacto de la solución propuesta en la investigación.

Como ***aspectos novedosos*** de este trabajo se pueden mencionar los siguientes:

- Se obtuvo una comparación teórica de las distintas notaciones posibles a utilizar en la modelación de este tipo de aplicaciones.
- Se hizo una propuesta de una nueva notación para la modelación del software educativo tomando como base el Lenguaje Unificado de Modelado (UML).
- Se lograron incorporar a la propuesta presentada los estándares establecidos por el Lenguaje de Modelación de Objetos (OCL) en su versión 2.0 lo que brinda elementos de sustancial robustez de la propuesta.

Al mismo tiempo como ***valores prácticos*** de la investigación se enuncian los que siguen:

- La obtención de documentos científicos explicativos de la notación propuesta en el cuerpo de este propio trabajo, con valor didáctico para los investigadores y estudiosos de la ingeniería de software aplicada al ámbito educativo.
- La documentación desarrollada, al ser la relativa a un proyecto productivo de la Facultad 9 de la UCI, servirá como un valor agregado a dicho proyecto tanto en la dimensión educativa como de aplicación ingenieril.

El trabajo que se presenta a continuación está conformado por 3 capítulos:

- *Las notaciones y los lenguajes para la modelación de aplicaciones educativas*: en este capítulo se abordan los elementos relacionados con la Informática Educativa y las tecnologías multimedia e hipermedia. Se realiza a su vez una presentación del estado del arte en Cuba y el mundo de las notaciones y los lenguajes de modelación; así como una descripción breve de cada una de las metodologías y notaciones aplicables al contexto educativo. Se culmina el capítulo presentando los elementos estructurales significativos del software educativo en Cuba.
- *El Lenguaje para la modelación de Aplicaciones Educativas Multimedia: ApEM – L*; es un capítulo en el cual se describe de manera íntegra la nueva notación propuesta para la modelación de aplicaciones educativas, su estructura sistémica, su sintaxis, las vistas generales y sus posibilidades de extensión.

- *Una experiencia práctica de la aplicación de ApEM – L*: como capítulo final del trabajo acerca al lector a la aplicación práctica de esta nueva notación para la modelación del entorno del nivel I del producto “A Jugar” desarrollado en la Facultad 9 de la UCI y establecer las comparaciones necesarias con las anteriores notaciones utilizadas en este propio proyecto.

Capítulo 1

Las notaciones y los lenguajes para la modelación de aplicaciones educativas.

Introducción

"El objetivo del analista es el reconocimiento de los elementos básicos del problema tal y como los percibe el cliente/usuario." (Pressman, 2002)

La frase anterior, expresada por Roger Pressman, una de las personalidades más reconocidas en el campo de la Ingeniería de Software (ISW), da pie al contenido del presente capítulo, en el cual se abordarán los elementos que servirán de fundamento teórico a la investigación presentada. Conceptos y teorías como Software Educativo, multimedia, hipermedia, notación, estereotipos, lenguajes de modelación, y herramientas CASE serán trabajadas con detenimiento para servir como fundamentación científica de lo investigado.

1.1 La informática y el software, al servicio de la educación.

En las últimas tres (3) décadas se ha desarrollado un vertiginoso desarrollo de la incorporación de la informática al entorno educativo, dando surgimiento al área de la Informática Educativa y con ella a conceptos como Software Educativo, que se hace muy común en nuestros días. El propio desarrollo de este tipo de software ha traído como consecuencia variaciones importantes en los flujos de los procesos productivos y en los artefactos de modelación para generar su documentación en el uso de las tecnologías multimedia e hipermedia.

1.1.1 La tecnología multimedia.

Según la enciclopedia Microsoft Encarta'97 la multimedia se puede clasificar como *una forma de presentar información, en su combinación de texto, sonido, imágenes, animaciones y video*. Ejemplos de aplicaciones multimedia informáticas son: juegos interactivos, programas de aprendizaje, materiales de referencia, por ejemplo enciclopedias. En los últimos años, varios autores han intentado conceptualizar la tecnología multimedia. Pero una concepción multifocal de la multimedia es la que plantea Rodríguez Lamas:

"*Es una nueva plataforma donde se integran componentes para hacer ciertas tareas que proporcionan a los usuarios nuevas oportunidades de trabajo y acceso a nuevas tecnologías. Es un nuevo medio donde la computadora junto con los medios tradicionales dan una nueva forma de*

expresión. Es una nueva experiencia donde la interacción con los medios es radicalmente diferente y donde tenemos que aprender como usarlos. Es una nueva industria donde, con una nueva plataforma, un nuevo medio y una nueva experiencia, nos llevan a tener nuevas oportunidades de negocios." (Rodríguez, 2000)

La tecnología multimedia aparece en el año 1984, cuando la Apple Computer lanza la primera variante de Macintosh, la cual tenía amplias capacidades de reproducción de sonidos; apoyándose en su sistema operativo, propicio para el diseño gráfico y la edición. En 1987, con los videojuegos, los avances en las Tecnologías de la Información y las Comunicaciones (TIC) y el desarrollo por la Philips del Disco Compacto (CD), aumentan las posibilidades para el desarrollo de la tecnología multimedia. En 1992, se presenta en la feria "*Consumer Electronic Show*" (CES) de Las Vegas, el CD multiusos, que luego dio paso luego a la televisión interactiva. En 1993, el nuevo término de multimedia, obliga a una revisión de los conceptos que permiten el funcionamiento de la nueva tecnología, buscando el desarrollo de estándares para lograr la uniformidad en el desarrollo. Hoy se ha desarrollado una nueva técnica que surgió a partir de la multimedia: la hipermedia, que tiene su base en el hipertexto, permitiendo al usuario un recorrido o navegación por distintos archivos o partes del programa de acuerdo a sus intereses personales.

La multimedia tiene varias aplicaciones entre las cuales se pueden mencionar las siguientes: (Corrales Díaz, 1994)

- *En la diversión y el entretenimiento*: donde se sitúan los juegos de video y las aplicaciones de pasatiempos de tipo cultural.
- *Multimedia en los negocios*: sus principales exponentes están en los kioscos de información, las presentaciones, intercambio y circulación de información.
- *En publicidad y marketing*: sus ejemplares son; la presentación multimedia de negocios, de productos y servicios, la oferta y difusión de los productos y servicios a través de los kioscos de información.
- *En la difusión del saber y conocimiento*: La característica de la interactividad de multimedia, que permite navegar por el programa y buscar la información sin tener que recorrerlo todo, logra que la tecnología se aplique en los nuevos medios de modos diferentes y se use de forma alternativa.

Y por último entre los muchos beneficios que ofrecen la tecnología multimedia se puede mencionar: el impacto al *incorporar imágenes*, efectos de *sonido, video y animación* en tercera dimensión para crear presentaciones vivas y de extraordinaria calidad. *La flexibilidad*, ya que el material digital puede

ser fácil y rápidamente actualizado y presentado a través de innumerables medios. *El control por parte del emisor*, al seleccionar la cantidad y tipo de información que desea entregar así como la forma de entregarla al igual que *el control por parte del receptor*, al elegir la información que quiere recibir y en el momento en que desea recibirla. *El ahorro de recursos en* materiales impresos difíciles de actualizar y presentándola en innumerables ocasiones sin ninguna restricción.

1.1.2 La tecnología hipermedia.

Vannevar Bush, en la década de los 40, y Theodor Holme Nelson, en los 60, se consideran los artífices de la estructuración no lineal y de la interconexión de la información, asuntos que constituyen conceptos claves para el desarrollo de la interactividad informática aplicada a la comunicación. El hipertexto no es más que un conjunto de bloques de texto interconectados por nexos, que forman diferentes itinerarios para el usuario, donde estos nexos "electrónicos" unen fragmentos te texto internos o externos a la obra (soportes cerrados – off line – o abiertos – on line – respectivamente), creando un texto que el lector experimenta como no lineal o, mejor dicho, como multilineal o multisecuencial.

La hipermedia surge como resultado de la fusión de dos tecnologías, el hipertexto y la multimedia. 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 sobre los diferentes formatos de información.

Estas hipermedias y multimedias 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 (Pastor, 1997?), pudiendo soportar la autoría de documentos complejos y el procesamiento de ideas, especialmente en entornos de trabajo colaborativos. Para la aplicación de esta facilidad los alumnos necesitan la capacidad de anotar nodos de información y ser capaces de colaborar con otros estudiantes y con el profesor sobre unidades específicas de información. Aún más importante que esto es que deben ser capaces de construir su propio sistema de conocimiento a través de la investigación, abstracción, readaptación y adición de conocimientos a una base de datos existentes.

1.1.3 La Informática y las aplicaciones educativas.

A continuación analicemos a fondo una aplicación de las tecnologías multimedia e hipermedia: el Software Educativo. "*Todas las formas de software educativos han sido absorbidas por esta tecnología, lo cual no es pura casualidad, sino el resultado de un proceso histórico que ha pretendido combinar los diferentes métodos para transmitir la información, en esperanza de una mayor calidad del propio proceso de adquisición de conocimientos."* (Lee, y otros, 2000)

"*Las tecnologías de la información y la comunicación (TIC) ofrecen grandes posibilidades al mundo de la educación. Pueden facilitar el aprendizaje de conceptos y materias, ayudar a resolver problemas y contribuir a desarrollar las habilidades cognitivas. Las áreas de aplicación de todas estas técnicas, englobadas en lo que normalmente se denomina informática educativa, son tanto la enseñanza reglada, comúnmente denominada curricular, como la formación en todos los ámbitos posibles. De esta manera, se nos presenta la posibilidad de aprovechar la tecnología para crear situaciones de aprendizaje y enseñanza novedosas.* "(Diáz-Antón, y otros, 2002?)

Una de las clasificaciones más conocida fue dada por los norteamericanos Stephen M. Alessi y Stanley Trollip, cuando plantearon que el uso de las computadoras en la educación podía dividirse en: (Pérez Fernández, 1999?)

- *Uso administrativo.*
- *Enseñanza sobre computadoras*.
- *Enseñanza con computadoras,* dentro de la cual se enmarca mayoritariamente el área de la informática educativa.

Según Raúl Rodríguez Lamas, la Informática Educativa se puede definir como *la parte de la ciencia de la Informática encargada de dirigir, en el sentido más amplio, todo el proceso de selección, elaboración, diseño y explotación de los recursos informáticos dirigido a la gestión docente, entendiéndose por éste la enseñanza asistida por computadora y la administración docente.* (Rodríguez, 2000)

La verdadera revolución se ha producido a raíz de la generalización de los CD-ROMs como soporte de información para las aplicaciones multimedia y al mismo tiempo una auténtica revolución en el software, sobre todo en la interfaz de usuario. Vinculando la enseñanza por computadora con las nuevas tecnologías multimedia, surge lo que conocemos como los software multimedia educativos; herramientas poderosas dentro del contexto de la Informática Educativa.

Según Vicenta Pérez Fernández un software multimedia educativo es "*una aplicación informática, soportada sobre una bien definida estrategia pedagógica, y que apoya directamente el proceso de enseñanza-aprendizaje* constituyendo un efectivo instrumento para el desarrollo educacional del hombre del próximo siglo." (Pérez Fernández, 1999?)

Los software multimedia educativos, permiten agrupar una serie de factores presentes en otros medios, pero a la vez agregar otros hasta ahora inalcanzables: (Pérez Fernández, 1999?) (Fernández, 2000)

- Permite la interactividad con los estudiantes, retroalimentándolos y evaluando lo aprendido
- Facilita las representaciones animadas.
- Incide en el desarrollo de las habilidades a través de la ejercitación. Permite simular procesos complejos.
- Reduce el tiempo que se dispone para impartir gran cantidad de conocimientos facilitando un trabajo diferenciado, introduciendo al estudiante en el trabajo con los medios computarizados.
- Facilita el trabajo independiente y a la vez un tratamiento individual de las diferencias.
- Permite al usuario (estudiante) introducirse en las técnicas más avanzadas.
- Posibilidades de estudiar procesos que no es posible observar directamente.
- Autocontrol del ritmo de aprendizaje.

Concluyendo, los buenos software multimedia formativos dentro del marco de la Informática Educativa, son *eficaces y facilitan el logro de sus objetivos*, debido al supuesto buen uso por parte de los estudiantes y profesores, a una serie de características que atienden a diversos aspectos funcionales, técnicos y pedagógicos. A continuación se mencionan algunos: (Autores, 1998?)

- Facilidad de uso e instalación.
- Versatilidad (adaptación a diversos contextos).
- Calidad del entorno audiovisual.
- La calidad en los contenidos (bases de datos).
- Navegación e interacción.
- Originalidad y uso de tecnología avanzada.
- Capacidad de motivación.
- La documentación.

1.2 Las notaciones en la ingeniería del software educativo.

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 y dichos requerimientos pueden incluir áreas tales como: funcionalidad o comportamiento 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 que juega el modelado por una simple razón: *"El hombre hace modelos de sistemas complejos porque no puede entenderlos en su totalidad"*.

1.2.1 La representación de un pensamiento ingenieril.

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* (Object Constraints Language) 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.

Algunos metodólogos del software orientado a objetos distinguen tres grandes "generaciones" cronológicas de técnicas de modelado de objetos. (Pressman, 2002)

- 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 en la actualidad es UML.

1.2.2 Las notaciones en el contexto productivo mundial.

1.2.2.1 ¿Qué es un modelo?

Según el Diccionario Electrónico de la Real Academia Española de la Lengua (DERAE), el término modelo posee las siguientes acepciones: (Lengua, 2007?)

- Arquetipo o punto de referencia para imitarlo o reproducirlo.
- En las obras de ingenio y en las acciones morales, ejemplar que por su perfección se debe seguir e imitar.
- Representación en pequeño de alguna cosa.
- *Esquema teórico, generalmente en forma matemática, de un sistema o de una realidad compleja, como la evolución económica de un país, que se elabora para facilitar su comprensión y el estudio de su comportamiento.*

Tomando en consideración las acepciones presentadas anteriormente, la última de estas se acerca bastante a los intereses de esta investigación, dejando claridad en la idea de dos aspectos: el de esquema teórico y su elaboración para facilitar la comprensión, ideas con las cuales el autor de este trabajo está plenamente de acuerdo.

Un modelo es una representación, en cierto medio, de algo en él mismo u otro medio. El modelo capta los aspectos importantes de lo que estamos modelando, desde cierto punto de vista, y simplifica u omite el resto. La ingeniería, la arquitectura y muchos otros campos creativos usan modelos. El modelo tiene semántica y notación y puede adoptar varios formatos que incluyen texto y gráficos. Es la representación más formal para las divisiones que propicien el entendimiento y la administración de la complejidad de estos. (Booch, y otros, 2000) (Rumbaugh, 1997)

En el ámbito de la Ingeniería de Software (ISW) desarrollar modelos, implica representar los sistemas de software en cada una de sus partes, desde diferentes vistas: lógicas, estructurales, de comportamiento, de arquitectura, de ensamblaje; para de esta manera permitir el consenso en el

lenguaje, el entendimiento del problema y las soluciones y la toma de decisiones de los equipos de desarrollo y los clientes.

Los modelos, así como la modelación de los sistemas; permiten propósitos como los que plantean tres de los investigadores más autorizados en la actualidad a definir estos términos: Booch, Jacobson y Rumbaugh; en su texto *"El Lenguaje Unificado de Modelado. Manual de Referencia"*: (Booch, y otros, 2000)

1. Para capturar y enumerar exhaustivamente los requisitos y el dominio de conocimiento, de forma que todos los implicados puedan entenderlos y estar de acuerdo con ellos.
2. Para pensar en el diseño de un sistema.
3. Para capturar decisiones del diseño en una forma mutable a partir de los requisitos.
4. Para generar productos aprovechables para el trabajo.
5. Para organizar, encontrar, filtrar, recuperar, examinar y corregir la información en grandes sistemas.
6. Para explorar económicamente grandes soluciones.
7. Para domesticar los sistemas complejos.

1.2.2.2 La modelación del software.

Similar al concepto de modelo, el término modelación al ser una conjugación, debemos estudiarlo a través del verbo *"modelar"*, el cual según el DERAE presenta los siguientes significados: (Lengua, 2007?)

(De *modelo*).

- Formar de cera, barro u otra materia blanda una figura o adorno.
- Configurar o conformar algo no material.
- Presentar con exactitud el relieve de las figuras.
- *Ajustarse a un modelo.*

Como se expresa en la última acepción presentada, modelar no es más que la acción de ajustarse al modelo. En términos de informática esto significaría: *representar sistemas, ajustándose a un conjunto de normas y signos de representación determinados, según la notación correspondiente, precisando los modelos de importancia para el entendimiento de las partes y el todo del software y su entorno.*

La acción de modelar un software, implica en su hacer el tener en cuenta al construir modelos aspectos como:

- *Arquitectura del sistema que se construye:* se hace necesario la observación de la arquitectura que se utiliza o utilizará en la solución, para lograr una estructura sólida y sostenible de la aplicación en el tiempo, así como un entendimiento del o los principios de funcionamiento de la aplicación que se desarrolla.
- *Disponibilidad del hardware para la solución propuesta. (disponibilidad tecnológica de la solución):* no todas las soluciones pueden convertirse en la óptima que se desea, sino que al construir nuestros sistemas no podemos olvidar la tecnología con la que se cuenta o aquella que se pueda adquirir en el futuro inmediato para sustentar la propuesta de solución.
- *Relación Costo/Beneficio económico de la solución:* factor este de gran importancia en las soluciones que se modelen; pues desde el momento de la representación de la solución, es donde se deben valorar (sin costo añadido, salvo el costo del esfuerzo humano en la valoración) las posibles soluciones desde el punto de vista económico, para permitir seleccionar la de mejor relación en cada caso al discernir entre los componentes materiales y humanos a intervenir en las soluciones.
- *Posibilidades de cambios futuros y asimilación de nuevas tecnologías o soluciones informáticas:* los modelos de representación de un sistema, tienen que estar preparados o permitir, sin tocar el programa o código fuente del mismo, hacer trabajo de mesa para el análisis de cambios, asimilación de nuevos requerimientos o de nueva tecnología. Esto hace que los modelos tengan que ser entendibles y exactos en la representación de elementos de esta naturaleza.
- *Posibilidad de reutilización de partes o componentes de otros sistemas ya desarrollados anteriormente:* hoy día, al desarrollar sistemas cada vez más grandes e intrincados, así como existiendo problemáticas similares en muchos lugares del planeta, no es ventajoso en momento alguno repetir soluciones en distintas partes del mundo, sino todo lo contrario, trabajar en colectivo, para la generación de mejores soluciones entre todos. Eso hace que la solución sea la reutilización de partes ya existentes, pero que deben de representarse igualmente en los modelos, no importa si como cajas negras, desconocidas para los desarrolladores (pero aún existirá en el modelo y se sabrá que se utiliza en la solución) o como componentes ampliamente representados para su entendimiento en la solución.

1.2.2.3 Las notaciones y los paradigmas de programación.

Al consultar el DERAE para obtener información de los términos *"notación"* y *"lenguaje"*, aparecen como acepciones más cercanas a nuestros propósitos, las que se presentan a continuación respectivamente: (Lengua, 2007?)

Notación: (Del lat. *notatĭo, - ōnis*).

- Acción y efecto de notar (señalar).
- Escritura musical.
- *Sistema de signos convencionales que se adopta para expresar conceptos matemáticos, físicos, químicos, etc.*

Lenguaje: (Del prov. lenguatge)

- *Conjunto de señales que dan a entender algo*.
- (m. inform.) Conjunto de signos y reglas que permite la comunicación con un ordenador.
- (m. inform.) (de alto nivel) Lenguaje que facilita la comunicación con un computador mediante signos convencionales cercanos a los de un lenguaje natural.

En términos de informática, estas dos palabras tienen un significado bastante similar, pues ambas identifican la acción de: *establecer un conjunto de signos, reglas, normas y semántica para la representación de la estructura y el comportamiento de los sistemas informáticos, permitiendo de esta forma la homogenización de los términos utilizados por el equipo de desarrollo y la construcción de modelos de representación del futuro software.*

A lo largo de la historia de la computación, el desarrollar lenguajes que permitan expresar la lógica del funcionamiento humano a través de programas entendibles para la computadora, ha transitado por varios paradigmas o formas, dentro de las cuales los más significativos son:

- Paradigma estructurado.
- Paradigma Orientado a Eventos. (OE)
- Paradigma Orientado a Objetos. (OO)

En el momento de existencia de cada uno de los paradigmas anteriores, los investigadores y especialistas de la ISW, presentaron metodologías de desarrollo de software que anexaban a estas notaciones, simbologías o lenguajes para la representación y descripción de las soluciones informáticas; factor este que atentó contra el entendimiento entre desarrolladores de diferentes regiones del mundo de las soluciones ofrecidas por sus colegas, así como también.

Al mismo tiempo, por la complejidad de los lenguajes de programación de computadoras, no eran abundantes los desarrolladores de sistemas informáticos, lo que hacía poco necesario la documentación de las aplicaciones, pues los equipos de desarrollos fueron unipersonales o en parejas, llegando muy pocas veces a ser de mayor cantidad de miembros, pues se potencializaba el trabajo artesanal.

Con el desarrollo de la tecnología de la electrónica, las comunicaciones y la telefonía, el tratamiento de la información cambió vertiginosamente. Se produjo un salto tecnológico muy grande que trajo como consecuencia la necesidad de aunar el trabajo de especialistas de diferentes campos y con diferentes lenguajes formales y de representación de los sistemas. Se hizo necesario el comienzo del desarrollo de lenguajes notacionales, que permitieran representar los paradigmas de las programaciones existentes, para aunar los esfuerzos y disminuir los costos del desarrollo de sistemas.

Las notaciones han estado siempre muy ligadas a los paradigmas de programación. Tanto es así que los lenguajes notacionales, como los humanos, solo nos sirven para representar determinados conceptos de un determinado paradigma, no lo contrario. De esta manera se llegó a lo largo de cuatro décadas a notaciones para la modelación estructurada, para la modelación orientada a eventos y para la modelación orientada a objetos en la actualidad.

1.2.3 Las notaciones en la producción de software educativo.

"En Cuba cada vez es más visible un cambio trascendental; las (…) tecnologías de la información y las comunicaciones se expanden hacia el ciudadano común. La computadora, el correo electrónico, el acceso a las redes informáticas y servicios de infocomunicaciones se introducen en la vida diaria de los cubanos." (MIC, 2003)

Como consecuencia del propósito descrito arriba, el gobierno cubano ejecutó el proyecto de desarrollo de software educativo para la escuela cubana como un programa más de numerosos priorizados por el Estado, entre los meses finales del año 2001 y el año 2002, consistiendo en el desarrollo de 30 paquetes de software para la escuela primaria con temas que abarcaron todos los contenidos curriculares y extracurriculares de los escolares de 6 a 11 años.

1.2.3.1 Los guiones de las aplicaciones educativas.

"Sobre el tema de los guiones interactivos para multimedia ya se ha escrito algo, aunque todavía hay bastante que decir y, sobre todo que hacer, pero se van encontrando pautas que caracterizan este tipo de obra." (Barrera Yanes, 1998)

"(…) el guión (…) abarca desde los aspectos estructurales y funcionales hasta los formales y estéticos, con un nivel de detalle que en principio permite desarrollar la ejecución del proyecto sin ambigüedades. Se discute mucho hasta dónde deben especificarse los detalles, pero mientras más preciso sea el guión mucho más rápidamente y a un menor costo se podrá llevar a cabo la elaboración de la obra." (Barrera Yanes, 1998)

"La función primordial del guión es dar objetividad al proyecto de la obra multimedia de forma tal que pueda independizarse el proceso de ejecución del proyecto de concepción y diseño. (…) En segundo lugar, el guión es imprescindible para lograr una comunicación clara y precisa entre los integrantes del equipo de trabajo de manera que aunque realicen tareas independientes todos conozcan cómo tienen que hacer su labor y cómo encaja cada uno dentro de la obra." (Barrera Yanes, 1998)

Se pueden emplear diferentes formatos o técnicas para elaborarlo, pero en esencia todos deben tener una estructuración por escenas y cada una con un determinado nivel de detalle. Los guiones además no son más que una breve sinopsis o descripción general y esquemática de cómo estará constituida la obra, acompañada de la estructura orgánica y la funcionalidad de cada elemento. La mayoría de los guiones contemplan como partes o componentes un *Esquema de escenas* y un *Grafo de navegación.* Otros equipos desarrolladores apoyan el trabajo de creación de los software educativos con artefactos como: *listado de medias* (textos, imágenes, videos, animaciones y sonidos) a utilizar en la aplicación educativa y *Documentación de los parlamentos* si se utiliza una mascota de guía en el software.

Es de notar con claridad como en la descripción ofrecida sobre los guiones multimedia, se distingue la recarga considerable del mismo en los aspectos que desde el punto de vista visual o de interés para el usuario en una aplicación educativa existen. Sin embargo es de difícil consecuencia la generación de una documentación de interés ingenieril a partir de un guión de este tipo.

En la última década se ha avanzado en este campo, logrando en la modelación de entornos educativos, la incorporación de lenguajes notacionales de propósito general como UML, o sus extensiones como OMMMA – L, ambos descritos en este mismo capítulo. No obstante estos lenguajes no logran denotar todos los elementos, tanto ingenieriles como pedagógicos de las aplicaciones educativas cubanas en sus gráficos y semánticas utilizadas.

1.3 Los estereotipos y su clasificación

Según el Diccionario electrónico de la Real Academia de la Lengua Española (Lengua, 2007?), la acepción del término estereotipo es la siguiente: "*Imagen o idea aceptada comúnmente por un grupo o sociedad con carácter inmutable.*" Su significado en términos de informática no está muy alejado de

este; pues “Un estereotipo en un lenguaje de modelación es un mecanismo bien formado para expresar extensiones definibles por el usuario, refinamientos o redefiniciones de elementos del lenguaje sin (directamente) modificar el meta – modelo del lenguaje.” (Berner, y otros, 2000)

“Los estereotipos clasifican los objetos de acuerdo a su uso, independientemente de su primera clasificación por clases y jerarquía de clases. Esta clasificación ayuda a organizar mejor un modelo y mejora el entendimiento del modelo.” (Berner, y otros, 2000)

En 1997, fue desarrollado por la IBM, el documento de especificación del Lenguaje de Modelación de Objetos (Object Constraint Language) en su versión 1.1, el cual luego fue aprobado por el OMG como estándar y mejorado a su versión 2.0 en el año 2003. En la memoria que se presenta se ha hecho una recopilación de los mejores elementos representativos propuestos por (Berner, y otros, 2000) los cuales tienen como base científica el lenguaje OCL. En este mismo sentido utilizamos como base además para la propuesta el lenguaje UML, el cual igualmente respeta el estándar OCL, lo que hace que la propuesta presentada sostenga sólidas bases científicas y de uso de estándares mundiales.

Según la clasificación propuesta por Berner y otros autores en el año 2000, los estereotipos pertenecen a alguna de las siguientes cuatro (4) categorías: (ver Figura 1.1)

- Estereotipos *decorativos*.
- Estereotipos *descriptivos*.
- Estereotipos *restrictivos*.
- Estereotipos *redefinitorios*.

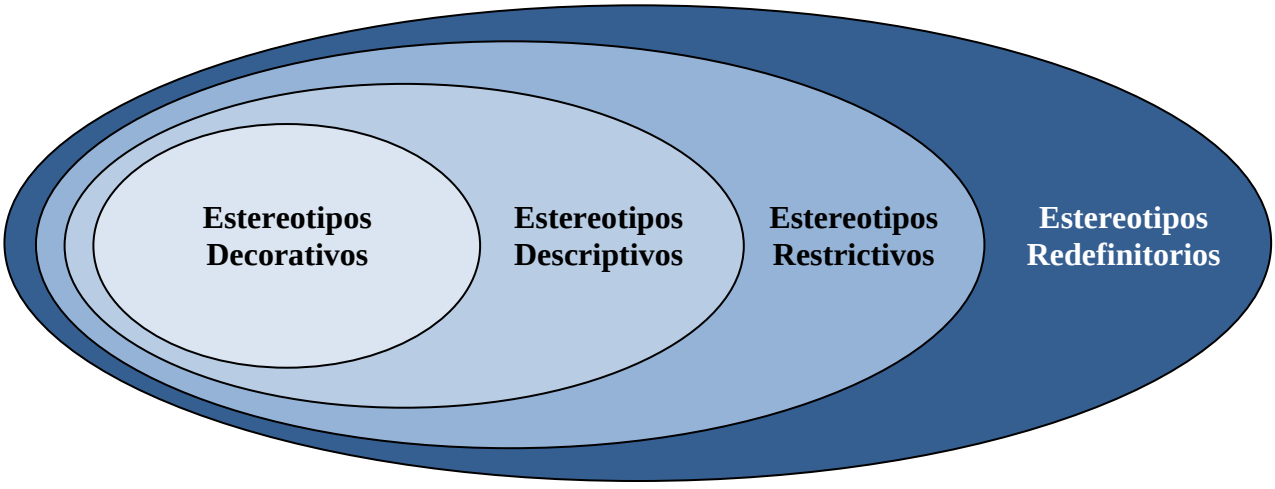


Figura 1.1: Clasificación de los estereotipos de acuerdo a su poder expresivo en cuatro categorías y su jerarquía de inclusión.

Comúnmente los cambios o extensiones que se producen en un lenguaje son cambios del tipo gráfico, lo que hace que se estén modificando aspectos decorativos de los mismos, por lo que no sufre el lenguaje base grandes cambios. Al contrario ocurre con modificaciones en estereotipos restrictivos, los cuales producen cambios del tipo semánticos en los lenguajes bases y por consiguientes en cada uno de sus subconjuntos contenidos en él. Es válido señalar por último que "Los estereotipos son rasgos o características muy potentes. Permiten un rango de modificaciones desde ligeros cambios notacionales hasta redefiniciones del lenguaje base. Sin embargo, el poder de los estereotipos supone riesgos. Estereotipos mal diseñados pueden dañar un lenguaje de modelación." (Berner, y otros, 2000)

1.4 Las notaciones en la producción de aplicaciones educativas en la Universidad de las Ciencias Informáticas.

En Septiembre del 2002 surge la Universidad de las Ciencias Informáticas (UCI), una entidad productiva y educativa, de la cual Fidel Castro planteó que debe ser una Universidad de excelencia, "…buscando lo óptimo, lo más perfecto dentro de las cosas humanas, lo más nuevo, lo más creativo, algo que no solo sirva a los intereses de nuestro país, sino sirva de ejemplo para el resto del mundo"….la cual permite disponer en la actualidad de un centro destinado a la informatización de la sociedad cubana y a otros programas de la Revolución.

Para darle cumplimiento a los objetivos de la UCI, se ha estructurado la universidad en un conjunto de 10 facultades docentes y una Infraestructura Productiva (IP) con un conjunto de direcciones, dentro de las cuales se encuentra la Dirección de Producción de Software Educativo, que en conjunto con las facultades 5, 8 y 9 desarrolla en la actualidad productos destinados a la educación en diferentes sectores de la sociedad cubana y la exportación.

Uno de los miembros de la Dirección de Producción de Software Educativo en la UCI plantea: "La dirección de software educativo de nuestra universidad tiene como uno de sus principales objetivos el desarrollo de multimedia educativas. Hasta ahora este trabajo se realiza ***manualmente***, esto implica comenzar a desarrollar cada software y realizar su implementación a partir de cero a veces sin el conocimiento necesario para ello, lo que además de consumir grandes recursos humanos y de tiempo, afecta la calidad de los productos que se realizan." (Lorente Rodríguez, 2006)

Estando de acuerdo con lo que plantea el especialista citado, cabe mencionar que hoy en la universidad se utilizan para la modelación de las aplicaciones lo siguiente:

- Guiones de contenido y técnicos.

- Árboles o mapas de navegación.
- Análisis de posibles arquitecturas para los productos software, máxime cuando estos son colecciones.

Recordando las seis características distintivas del software educativo cubano, mencionadas en la introducción de este trabajo, es válido señalar que hoy en la universidad no llega a concretarse con eficiencia la documentación de este tipo de aplicaciones que posibilite lo siguiente:

- Durante la ejecución de los proyectos productivos:
  1. *Entendimiento sin ambigüedades* (utilización de iguales términos y conceptos semánticos) de los diferentes roles (especialistas pedagógicos, diseñadores gráficos, analistas, programadores y arquitectos) participantes en la elaboración del producto.
  2. *Generación de una documentación técnica* que posibilite la programación orientada a objetos de las aplicaciones de forma directa, que incorpore las mejores técnicas y prácticas internacionales en la industria del software.
- Al finalizar los mismos y en su mantenimiento:
  1. Utilización de la *documentación del proyecto para generar pruebas de unidad y sistema* a las aplicaciones, basadas en los algoritmos y flujos de funcionamiento de los software educativos.
  2. Utilización de la *documentación para el mantenimiento de las aplicaciones y la posibilidad de eliminación de errores* a partir de los modelos que del mismo se dispongan.
- Generación de nuevas versiones o portabilidad a otros entornos:
  1. Utilización de la *documentación ingenieril del producto para el análisis de nuevas soluciones* a partir de modelos de diseños horizontales en el desarrollo, que permitan general a partir de él nuevas versiones en otros lenguajes, plataformas o arquitecturas de solución; determinando la razón costo/beneficio de un cambio de este tipo sin incurrir en grandes gastos económicos.
  2. Utilización de la *documentación para la determinación de posibilidades de migración a nuevas plataformas o sistemas operativos*, determinando la razón costo/beneficio de un cambio de este tipo sin incurrir en grandes gastos económicos o necesidad de hacerlo in situ en laboratorios de prueba, sino desde la concepción ingenieril basada en los modelos existentes.

Las dificultades mencionadas anteriormente, sustentan la necesidad en la universidad de desarrollar un lenguaje notacional que permita adecuar los mejores lenguajes existentes en este ámbito al contexto productivo cubano y de la UCI en particular y trabajar sobre la representación de los siguientes elementos:

1. Dominio de información del software.
2. Requerimientos pedagógicos asociados a los software.
3. Características de las interfaces de comunicación con el usuario, punto neurálgico de este tipo de aplicaciones.
4. Conjunto de clases y entidades que sustentan la lógica de funcionamiento de las aplicaciones y sus relaciones.
5. Navegabilidad dentro de las aplicaciones.
6. Comportamiento de los elementos componentes del software.
7. Arquitectura que sostiene el modelo de desarrollo de las soluciones.
8. Posibilidad de división de las soluciones en unidades de trabajo más pequeñas desde el punto de vista funcional y visual.
9. Composición física de la solución generada.
10. Despliegue físico de la solución construida.

1.5 El entorno tecnológico y científico en la representación del software educativo.

A lo largo de las últimas dos décadas se han desarrollado en la industria mundial y nacional un conjunto de soluciones informáticas para el tratamiento de entornos con cierto grado de similitud u objetivos similares a los entornos educativos cubanos modelados en nuestras aplicaciones. Mencionaremos en este epígrafe una descripción lo más breve posible de cada uno de estas soluciones por el interés futuro en establecer comparaciones al respecto y obtener los rasgos repetitivos o generalizadores.

“Desde alrededor de 1990, una amplia variedad de lenguajes de modelación han sido desarrollados. Estos lenguajes son utilizados para describir los requerimientos y el diseño de un sistema software. Desde 1996, varias tentativas han sido ejecutadas para unificar diferentes métodos y lenguajes. Como resultado de este esfuerzo, un lenguaje fue desarrollado: el Lenguaje Unificado de Modelado (UML). UML introduce una nueva característica distintiva: permite a los usuarios extender o incluso modificar

el lenguaje base para adaptar el lenguaje a situaciones o necesidades específicas. La construcción del lenguaje que es utilizada para implementar esta característica es llamada: estereotipo." (Berner, y otros, 2000)

1.5.1 RMM: Metodología de Administración de Relaciones (Relationship Management Methodology).

La metodología RMM, fue desarrollada por T. Isakowitz, en la Universidad de Nueva York en el año 1995. Se realiza la modelación de las aplicaciones a través de RMDM (Relationship Management Data Model), basado en el modelo Entidad – Relación y posee una herramienta CASE denominada: Relationship Management Case Tool – RMCase.

"RMM (Relationship Management Methodology) contiene el diseño y la construcción de aplicaciones hipermedia en un proceso de siete pasos. Es al mismo tiempo un enfoque "top down" y "bottom up". Durante la fase del diseño Entidad – Relación, entidades y relaciones son identificadas las cuales se convertirán en nodos y enlaces en la hipermedia resultante. El segundo paso, diseño de cortes (slices), involucra el agrupamiento de atributos de entidades para la presentación. Los cortes (slices) son "unidades de presentación" que aparecen como páginas de una aplicación hipermedia. La separación del contenido y los aspectos de la presentación no son satisfechos en este paso. RMM especifica la navegación con primitivas de acceso, como enlaces (links), agrupamiento (menus), índices (index) y recorridos guiados (guided tours). La técnica propuesta para el diseño de la interfaz de usuario es la elaboración de maquetas y proptotipos." (Baumeistier, y otros, 2001)

1.5.2 OOHDM: Metodología de Diseño Hipermedia Orientada a Objetos (Object – Oriented Hypermedia Design Methodology).

Desarrollada por Schwabe y Rossi, en la Universidad de Rio de Janeiro, Brasil y Universidad Nacional de la Plata, Buenos Aires respectivamente en el año 1996. Adopta la notación y los mecanismos de abstracción de la Programación Orientada a Objetos (POO) y consta de cuatro pasos para su ejecución: diseño conceptual, diseño navegacional, diseño de interfaz abstracta e implementación. Trabaja la representación a través de los siguientes modelos: Esquema de clases, esquema de navegación, esquema contextual de navegación y vista abstracta de datos.

"El Modelo de Diseño de Hipermedias Orientado a Objetos: OOHDM (Object – Oriented Hypermedia Design Model) comprende cuatro actividades; estas son modelo conceptual, diseño de navegación, diseño de interfaces abstractas e implementación. Estas actividades son ejecutadas en un estilo de desarrollo mixto a partir de los modelos incremental, iterativo y basado en prototipos. Este método trata a la aplicación como una vista superior al modelo conceptual. El concepto de contexto de navegación es introducido para describir la estructura de navegación. Es un concepto potente que permite diferentes agrupamientos de objetos de navegación con el propósito de navegar en ellos en diferentes contextos. Una notación especial es utilizada para la representación de la estructura de navegación. En trabajos tempranos de investigación, OMT se propuso como la notación para el esquema conceptual; un poco más tarde los trabajos ya utilizan UML. Sin embargo, los diagramas de OOHDM no obedecen los patrones UML, sino que utilizan una notación propia para la perspectiva de los atributos en los diagramas de clase y proponen otros tipos de diagramas para el diseño de la navegación y de las interfaces de usuarios abstractas." (Baumeistier, y otros, 2001)

1.5.3 WSDM: Método de Diseño de Sitios Web (Web Site Design Method).

El Método de Diseño de Sitios Web (WSDM) fue desarrollado por Vrije De Troyer, en la Universidad de Brucelas, en el WISE Research Group, en el año 1997; siendo "(…) un enfoque centrado al usuario definiendo objetos de navegación basados en información de requerimientos del usuario de una aplicación Web. WSDM consiste en tres fases principales: modelación del usuario, diseño conceptual y diseño de implementación. En la fase de modelación del usuario, el usuario potencial del Sitio Web es identificado y clasificado. Diferentes perspectivas son definidas para las clases de usuario, siendo estas diferentes formas en las cuales las clases de usuario utilizan la misma información. El modelo de navegación consiste en un número de rutas de navegación que expresan como los usuarios de perspectiva particular pueden navegar a través de la información disponible. Este método usa su propia notación gráfica para los objetos del modelo de navegación: el diseño de navegación alcanza aplicaciones Web que tengan una marcada estructura jerárquica." (Baumeistier, y otros, 2001)

1.5.4 UML: Lenguaje Unificado de Modelado (Unified Modeling Language).

El Lenguaje Unificado de Modelado (UML) fue desarrollado por Ivar Jacobson, Grady Booch y James Rumbaugh, en la Corporación "Rational Software", en el año 1996 en su primera versión. Mejorado en la actualidad a una versión superior 2.0 y desde sus inicios respetó los elementos estándares del lenguaje OCL, desarrollado inicialmente por la IBM. Consta de varias áreas conceptuales como son: estructura estática, comportamiento dinámico, construcciones de implementación, organización del modelo y los mecanismos de extensión. A su vez, consta de ocho vistas: estática, de casos de uso, de implementación, de despliegue, de máquinas de estado, de actividad, de interacción y de gestión del modelo; para la modelación de los productos, a través de un conjunto de diagramas distribuidos por cada una de estas vistas. Tiene como base teórica lo siguiente: Unified Method (Grady Booch, 1994), OMT (Grady Booch y James Rumbaugh, 1995) así como el Método OOSE (Ivar Jacobnson, 1996). Posee varias herramientas CASE de modelado como son Rational Rose Enterprise Edition y Visual Paradigm. Actualmente en un lenguaje estandarizado por la OMG a partir del año 1997 y ha servido como base para posteriores desarrollos de lenguajes a partir de sus posibilidades de extensión. (Booch, y otros, 2000)

1.5.5 OMMMA – L: Lenguaje para la modelación Orientada a Objetos de Aplicaciones Multimedia (Object – Oriented Modeling of Multimedia Applications).

OMMMA – L fue desarrollado por Stefan Sauer y Gregor Engels, en la Universidad de Paderborn, Alemania, en el año 2001, tomando como base el lenguaje UML. Consta de cuatro vistas fundamentales en su modelación: vista lógica, vista de presentación espacial, vista de comportamiento temporal predefinido y vista de control interactivo. Modifica los diagramas originales de UML de: clases, secuencia y estado. Añade como parte de la vista de presentación espacial un nuevo diagrama: el diagrama de presentación, para la representación espacial de los elementos visuales del futuro software multimedia. Basa su descripción en el patrón de arquitectura MVC_{MM}. (Sauer, y otros, 2001)

Conclusiones parciales

A lo largo de este capítulo se han ofrecido los elementos teóricos que sirven de sustento científico a la investigación, así como se ha ofrecido cada uno de los aspectos a tener en cuenta en la situación problemática que genera el problema científico razón de esta investigación. El análisis de cada uno de estos aspectos mencionados permite arribar de manera parcial a las siguientes conclusiones:

1. Hoy se trabaja en el mundo, las aplicaciones educativas a través de la utilización de las tecnologías multimedia e hipermedia, utilizando un conjunto de lenguajes de alto nivel (HLL) o sistemas de autor diseñados para estos fines.
2. La utilización de notaciones en la construcción de software educativo es muy pobre y está mayormente limitada a la documentación de los aspectos visuales o de contenido de este tipo de software pero no así de la modelación de los elementos de la ingeniería necesarios para una correcta implementación, mantenimiento, reingeniería, reutilización de código y portabilidad del sistema desarrollado.
3. Existen un conjunto de lenguajes y notaciones desarrolladas en Cuba y el mundo, para el desarrollo de aplicaciones de entornos hipermedia, pero que no permiten la representación eficiente de todos los elementos que desde el punto de vista pedagógico son abordados en el software educativo cubano.

Capítulo 2

El Lenguaje para la modelación de Aplicaciones Educativas Multimedia: ApEM – L.

Introducción

El desarrollo de software educativo es una disciplina nueva y aún envolvente. El proceso de aprender como desarrollar grandes aplicaciones educativas solo acaba de comenzar. Aplicaciones educativas para la Web o CD-ROM son justamente el resultado de una implementación ad – hoc, creciendo usualmente de pequeñas o grandes aplicaciones y convirtiéndose muy rápido en difíciles de mantener. Algunos principios y herramientas están comenzando a aparecer asistiendo a los desarrolladores de software educativos. Pero estas prácticas actuales a menudo fallan debido a técnicas, procesos o metodologías inapropiadas. (Hennicker, y otros, 2000)

"Un número de enfoques han sido propuestos para la modelación de aplicaciones multimedia. Predominantemente, estos se enfocan en la modelación de las relaciones temporales y la sincronización de las presentaciones multimedia. Algunos modelos más elaborados incluyen la interactividad. Otros se concentran en la estructura lógica y los conceptos de navegación para hipermedia." (Sauer, y otros, 2001)

"Durante los últimos 5 años, varios métodos para el diseño de hipermedia y Web han sido propuestos. La mayoría de estos no están basados en UML, como RMM (Relationshio Management Methodology) y OOHDM (Oject – Oriented Hypermedia Design Model). Estos utilizan los Diagramas de Entidad – Relación (E-R Diagrams), OMT o sus propias notaciones o técnicas. Recientemente, algunos nuevos enfoques proponen extensiones UML para el dominio hipermedia, por ejemplo el proceso de desarrollo de Conallen (1999), la extensión para aplicaciones multimedia OMMMA – L (1999) y la extensión UML propuesta por Baumeister – Koch - Mandel (1999). El primer enfoque es basado en el Proceso Unificado de Desarrollo (RUP) y se enfoca principalmente en la arquitectura de las aplicaciones Web. El segundo extiende los diagramas de secuencia de UML para modelar los procesos multimedia. El tercero provee elementos de modelación para el diseño de la navegación y la presentación de este tipo de aplicaciones." (Hennicker, y otros, 2000)

Por las condiciones descritas en los párrafos anteriores, en el presente capítulo se describirá la propuesta de solución que le da cuerpo a la presente memoria. *ApEM – L* está basado en el lenguaje de modelación UML, tomando elementos representativos de extensiones del mismo como las

desarrolladas por (Berner, y otros, 2000), (Hennicker, y otros, 2000) y (Sauer, y otros, 2001) y descansa toda su estructura sobre los elementos planteados por el estándar OCL, en su versión actualizada 2.0 del 2003.

Para obtener mejores criterios sobre el estado de producción del software educativo en la universidad y sus principales elementos ingenieriles de consideración para esta investigación, se realizó una encuesta (consultar Anexo 5) a un 11 desarrolladores, 9 estudiantes y 2 especialistas, de tres de los proyectos productivos que hoy se ejecutan en la UCI; uno de cada una de las áreas fundamentales de trabajo: Facultades 5, 9 y la propia Dirección de Software Educativo. Para la determinación de las unidades de estudio, se tuvo en consideración la heterogeneidad de los proyectos productivos de este corte en la UCI. Valorando esta situación, solo se tomó en consideración la variable *condiciones de trabajo* produciendo la selección de una unidad e estudio por cada una de las áreas, dando como resultado la selección de los proyectos: GAIM (Dirección Producción No. 2), Laboratorios Virtuales (Facultad 5) y A Jugar (Facultad 9).

Los resultados ofrecidos por la estadística obtenida a partir del procesamiento de los resultados de la encuesta reflejó los elementos siguientes:

1. Solo se conocen las notaciones UML (81,18 %), RMM (9,09 %) y OMMMA – L(9,09 %) y 2 de los encuestados pertenecientes a la misma unidad de estudio, no conocían notación alguna para la modelación.
2. Sólo en 1 unidad de estudio se utiliza una notación (UML) para la modelación, lo que denota la falta de seriedad en la documentación de este tipo de aplicaciones.
3. Todos los proyectos productivos utilizan el Guion como elemento rector en la generación de código y guía del trabajo de desarrollo del producto.
4. Más del 50 % de los encuestados (54,54 %), aunque con representación de todas las unidades de estudio, consideraban que los Guiones y Mapas de Navegación solo satisfacen *parcialmente* la modelación de software educativos.
5. Casi la totalidad de los encuestados, coincide en que los conceptos que se listan más adelante, no son abordados por los artefactos utilizados hoy día para la producción de software educativo: clases (72,72 %), eventos (18,18 %), entidades (18,18 %), relaciones de navegación (63,63 %), relaciones entre clases o entidades (27,27 %), tipos de clases o entidades (27,27 %), estructuras de las interfaces (63,63 %) y arquitectura de software (36,36 %).

6. El 90 % de los encuestados coincide en la necesidad de documentar ingenierilmente los software educativos.
7. Más del 63 % de los participantes en la encuesta coinciden en opinar que debe explotarse aún más la Programación Orientada a Objetos para la implementación de este tipo de software. El 27 % considera que debe de usarse la Orientada a Eventos y solo un 18 % cree que debe ser una mezcla de ambas.
8. El 90 % de los desarrolladores aseguran que se utiliza el patrón Modelo – Vista – Controlador (MVC) como arquitectura de sus proyectos, pero no el modificado MVC – E desarrollado en la UCI y más contextualizado a las características de la producción nacional.
9. El 72,72 % de los encuestados, con representación mayoritaria de las tres unidades de estudio, manifiestan que los estereotipos ofrecidos por UML/OMMMA – L son cómodos y fáciles de usar.
10. El 50 % considera que debe de reforzarse en soluciones de modelación para el software educativo, las siguientes vistas: Interacción (54,54 %), Presentación (27, 27 %), Casos de Uso (27, 27 %), Lógica (18,18 %) e Implementación (18,18 %).
11. El 90,90 % de los encuestados se manifiestan total o parcialmente satisfechos con la manera en la que la Descripción Textual de los Casos de Uso puede describir los flujos de eventos de los software educativos multimedia.

2.1 Elementos a tomar en consideración para la arquitectura del software educativo.

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. (Pressman, 2002)

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.

2.1.1 El patrón Modelo – Vista – Controlador MM (MVC_{MM}).

El **Model-View-Controller** (Modelo-Vista-Controlador, en adelante MVC) fue introducido inicialmente en la comunidad de desarrolladores de Smalltalk-80. Según uno de los *Sistemas de Patrones de Arquitectura* más extendido en el mundo: *Pattern Oriented Software Architecture*, publicado por Buschmann en 1996, el patrón MVC básico, se sitúa en la tercera de las cuatro categorías en las cuales clasifica a los patrones: *Del barro a la estructura* (From mud to Structure), *Sistemas Distribuidos* (Distributed Systems), ***Sistemas Interactivos*** (Interactive Systems) y *Sistemas Adaptables* (Adaptable Systems). De igual forma si utilizamos otro de los sistemas más difundidos: *Pattern of Enterprise Application Architecture*, descrito recientemente por Fowler en el pasado 2003, lo ubica en la tercera de las siete categorías que se mencionan a continuación: *Patrones de lógica del dominio* (Domain Logia Patterns), *Patrones de Mapeo a Bases de Datos Relacionales* (Mapping to Relational Database Patterns), ***Patrones de Presentación Web*** (Web Presentation Patterns), *Patrones de Distribución* (Distribution Patterns), *Patrones de Concurrencia Offline* (Offline Concurrency Patterns), *Patrones de Estado de Sesión* (Session State Patterns) y *Patrones Base* (Base Patterns).

MVC divide una aplicación interactiva en tres áreas: procesamiento, salida y entrada; trabajando con tres tipos fundamentales de clases: clases vista, clases controladoras y clases modelo. En la variante para modelar aplicaciones multimedia (MVC_{MM}) se diversifica la clase modelo incorporando al esquema dos nuevos tipos de clases: la *modelo dinámica* y la *modelo estática*, la cual a su vez se subdivide en las clases *media* y *lógica de la aplicación* (consultar Anexo 2).

Aunque se pueden encontrar diferentes implementaciones de MVC, el flujo que sigue el control generalmente es el siguiente: (Ciudad Ricardo, 2006)

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. 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.
5. La interfaz de usuario espera nuevas interacciones del usuario, comenzando el ciclo nuevamente.

El flujo recién descrito en el párrafo anterior deja claramente las siguientes responsabilidades a las clases conformantes de la arquitectura:

- Clase *Vista*: Recibe las peticiones del usuario al sistema. Muestra la información de salida al usuario. Pueden existir múltiples vistas del modelo. Cada vista tiene asociado un componente controlador.
- Clase *Modelo*: Encapsula los datos y las funcionalidades. Gestiona el procesamiento y almacenamiento de la información. El modelo es independiente de cualquier representación de salida y/o comportamiento de entrada.
- Clase *Controladora*: Reciben las entradas para la gestión de las mismas, a partir de las clases vistas, usualmente como eventos que codifican los movimientos o pulsación de botones del ratón, pulsaciones de teclas, etc. Los eventos son traducidos a solicitudes de servicio (“service requests”) para el modelo o la vista. Gestionan el modelo que realizará el procesamiento así como la vista que generará la respuesta al usuario.

2.1.2 El patrón Modelo – Vista – Controlador – Entidad (MVC-E) como propuesta arquitectónica del software educativo.

Durante las dos pasadas décadas, se han desarrollado un gran número de métodos de modelado. Los investigadores han identificado los problemas del análisis y sus causas y han desarrollado varias notaciones de modelado y sus correspondientes conjuntos de heurísticas para solucionarlos (…) Se emplean modelos para poder comunicar de forma compacta las características de la función y su comportamiento. Se aplica la partición para reducir la complejidad. Son necesarias las visiones

esenciales y de implementación del software para acomodar las restricciones lógicas impuestas por los requisitos del procesamiento y las restricciones físicas impuestas por otros elementos del sistema. (Pressman, 2002)

Este enfoque unido a las características distintivas del software educativo producido en Cuba y descritas en la introducción de este trabajo, ha traído como consecuencia el análisis de una variante de solución al MVC_{MM} para las aplicaciones educativas cubanas, donde se descarguen las responsabilidades de la clase modelo concernientes al procesamiento y almacenamiento de la información persistente de las aplicaciones, incorporando una nueva clase al modelo denominada ***Modelo Entidad***, con dos tipos fundamentales, la clase ***Modelo-Entidad-Media*** y ***Modelo-Entidad-Persistente***. La primera de estas con la responsabilidad de agrupar las clases que identifican las medias y su árbol de jerarquía en la aplicación y la segunda tiene como responsabilidad la gestión de la información persistente, que antes sobrecargaba a la clase *Modelo* del patrón MVC original. Esta variación a su vez sustenta las características actuales de los sistemas multimedia como son: comunicación con bases de datos, archivos XML, o sistemas externos. (consultar Anexo 3).

2.2 Perspectiva general de ApEM-L.

"UML está diseñado para ser utilizable con múltiples propósitos. Aunque UML es un lenguaje universal que puede expresar un elevado número de propiedades fundamentales de modelado, existen ocasiones en que resulta deseable una jerga hecha a la medida de un determinado dominio de conocimiento. En los lenguajes naturales, la jerga, las germanías y los vocabularios especializados suelen facilitar la comunicación dentro de un arte o disciplina especializado, pero impiden la comunicación con los extraños. UML se puede personalizar de forma similar, empleando distintos mecanismos tales como convenciones de denominación, reglas de estilo, clases predefinidas y correspondencias implícitas con el software. En particular, se puede definir una extensión de UML empleando estereotipos, restricciones y valores etiquetados predefinidos." (Booch, y otros, 2000)

UML utiliza para su extensión diferentes mecanismos, los cuales son: estereotipos, valores etiquetados y restricciones. Un ***estereotipo*** es una nueva clase de elemento de modelado con la misma estructura que un elemento existente pero con restricciones adicionales, una interpretación diferente de un icono, y un tratamiento diferente por los generadores de código y otras herramientas de bajo nivel. Un ***valor etiquetado*** es un par arbitrario de cadenas etiqueta-valor, que pueden enlazarse a cualquier tipo de elemento de modelado, para almacenar información arbitraria, como información de gestión de proyecto, guías para los generadores de código, y valores requeridos por los estereotipos. La etiqueta y

el valor son representadas como cadenas. Una ***restricción*** es una condición "bien formada" expresada en una cadena de texto en algún lenguaje restringido, tal como un lenguaje de programación, un lenguaje especial limitado, o lenguaje natural.

"Desafortunadamente, UML no soporta todos los aspectos de las aplicaciones multimedia de una manera adecuada e intuitiva. Especialmente, características del lenguaje para el modelado de aspectos de la interfaz de usuario no están explícitamente proporcionadas. Otros conceptos de UML, no son lo suficientemente maduros o son muy poco gráficos y esto agrava la modelación de multimedia innecesariamente." (Sauer, y otros, 2001)

ApEM – L se presenta como una extensión de UML, tomando como bases teóricas principales OMMMA – L (2001) y OCL – 2.0 (2003), lo que produce las siguientes ventajas:

- Puede utilizar para su representación todas las herramientas CASE que existen actualmente para la modelación de UML.
- Es un lenguaje que utiliza el estándar internacional OCL, para la modelación de la programación Orientada a Objetos.
- No modifica la semántica del lenguaje base UML, sino que trabaja en estereotipos restrictivos, por lo que a su vez produce modificaciones descriptivas y decorativas en la representación de los componentes del lenguaje base.

2.2.1 Objetivos de ApEM-L.

La concepción del *Lenguaje para la Modelación de Aplicaciones Educativas* (ApEM – L) tuvo los siguientes objetivos:

1. Desarrollar una extensión del lenguaje de modelado UML, tomándolo como base e incorporando a este, a través de sus mecanismos de extensión, los elementos fundamentales del proceso productivo UCI. De esta forma se produjo un lenguaje de propósito particular para la modelación de aplicaciones educativas.
2. Incorporar los elementos más significativos de extensiones anteriores como OMMMA – L (2001) y a su vez respetar lo establecido por el estándar OCL (2003), para de esta forma lograr una extensión consistente y escalable en el tiempo.
3. No complicar ApEM – L con elementos que lo convirtieran o abarcaran un método de desarrollo de aplicaciones educativas, sino solo el área de la representación y la documentación de este tipo de aplicaciones.

4. No circunscribir ApEM – L a un proceso de desarrollo en específico, sino expresarlo de manera tal que pueda ser utilizado con cualquiera de los existentes, aunque se sugiere la utilización de procesos de desarrollo iterativos, incrementales y basados en prototipos, que permitan la modelación de sistemas orientados a objetos.

2.2.2 Áreas conceptuales de ApEM-L.

Los conceptos y modelos de ApEM – L pueden agruparse en las siguientes áreas conceptuales:

- ***Estructura lógica*:** está compuesta por la vista estática y la vista de arquitectura. La primera de ellas está compuesta por el diagrama de clases y el diagrama de casos de uso. Cualquiera de los modelos presentados por ApEM – L define los conceptos claves de la aplicación que modela, las propiedades internas de estos y sus relaciones. Estos conceptos son modelados como clases, describiendo cada una un conjunto de objetos que almacenan información y se comunican para implementar su comportamiento. La información almacenada se representa como atributos de estas clases y las operaciones a través de los métodos de dichas clases. A su vez la vista de arquitectura la componen el diagrama de componentes y el diagrama de despliegue. Se tomaron como base las ideas expresadas por (Sauer, y otros, 2001) y mejoradas para el contexto productivo UCI con la incorporación del modelo arquitectónico propuesto por el patrón MVC-E. De los diagramas mencionados solo han sido modificados los siguientes: diagrama de clases y el diagrama de componentes, el resto mantuvo lo establecido por UML. También, aún manteniendo lo establecido originalmente para los diagramas de casos de uso, se adicionaron un conjunto de elementos a la descripción textual de los casos de uso propuesta por UML, para una mejor descripción del contexto productivo de los software educativos.
- ***Comportamiento dinámico*:** realmente esta área no ha sido grandemente modificada, pues solo se ha hecho una pequeña adición al diagrama de secuencia, salvo que ciertamente se ha enriquecido la semántica original de UML para estos diagramas. El comportamiento de la aplicación está descrito por la vista de comportamiento, la cual está compuesta por los diagramas: de actividad, de secuencia, de colaboración y de estados, donde solo ha sido modificado el segundo de los listados anteriormente; adicionando una variable de tiempo donde quiera que se necesario su especificación para un mejor entendimiento.
- ***Gestión del modelo*:** esta área es la que ha sufrido grandes cambios tanto en su carácter semántico como sintáctico, con la incorporación de estereotipos restrictivos en todos los diagramas a partir de nuevos conceptos incorporados a los diagramas de clases originales o

básicos de UML. Se crean dos nuevos diagramas: el de estructura de la presentación y el de estructura de la navegación.

2.3 El sistema de vistas de ApEM-L.

Tanto ApEM – L como UML; no establecen ninguna línea entre los diferentes conceptos y construcciones del lenguaje, pero por conveniencia del primero, este se ha dividido en varias ***vistas***, modelando cada una de estas construcciones que representan un aspecto del sistema. La división ha sido sobre la base de las áreas conceptuales ya presentadas: ***estructura lógica***, ***comportamiento dinámico*** y ***gestión del modelo***, como se puede apreciar en la Tabla 2.1. A continuación se describirán cada una de las vistas mencionadas y sus diagramas, deteniéndonos en aquellos en los cuales se produzcan cambios con respecto a lo establecido por el lenguaje base UML.

2.3.1 Vista estática.

La ***Vista estática*** de ApEM – L está compuesta por dos diagramas, el ***Diagrama de clases*** y el ***Diagrama de Casos de Uso***. El último de estos no sufre modificación alguna de lo planteado semánticamente por UML, aunque si sufren modificaciones las descripciones textuales de estos, agregando elementos que a consideración de los especialistas consultados son significativos en la producción de software educativo y que han sido reflejados en anexo 4 de esta memoria. Solo describiremos lo correspondiente al Diagrama de clases. Este diagrama está dividido en dos grandes zonas, la de la izquierda dedicada al árbol jerárquico de las ***clases modelo entidad medias*** que representan los recursos mediáticos de la aplicación y en la zona de la derecha del diagrama las clases que controlan la *lógica del negocio* de la aplicación propiamente dicha. Como se muestra en la ***Figura 2.1***; esta zona de la derecha vuelve a subdividirse en cuatro (4) zonas (consultar Figuras 2.2 y 2.3). La primera dedicada a las clases ***vista***, la contigua a esta y en el extremo superior derecho dedicada a las clases ***controladoras***, inmediatamente debajo de esta sección, la destinada a las clases ***modelo***, quedando una banda inferior derecha dedicada en su extremo derecho a las clases ***modelo entidad persistentes*** para el tratamiento de la información persistente de la aplicación; y en el extremo izquierdo las clases correspondientes al Lenguaje de Alto Nivel (HLL en sus siglas en inglés correspondientes a High Level Languege) con el que se programe (Lingo, ActionScript, C#, C++, Object Pascal, PHP, etc.).

Tabla 2.1 Vistas y diagramas de ApEM – L.[Modificado de (Booch, et al., 2000)]

Área	***Vista***	***Diagramas***	***Conceptos Principales***
Estructura lógica	Vista Estática	Diagrama de clases	Clase, asociación, generalización, dependencia, realización, interfaz.
		Diagrama de casos de uso	Caso de uso, actor, asociación, extensión, inclusión, generalización.
	Vista de arquitectura	Diagrama de componentes	Componente, interfaz, dependencia, realización.
		Diagrama de despliegue	Nodo, componente, dependencia, localización.
Comportamiento dinámico	Vista de comportamiento	Diagrama de actividad	Estado, actividad, transición de terminación, división, unión.
		Diagrama de secuencia	Interacción, objeto, mensaje, activación.
		Diagrama de colaboración	Colaboración, interacción, rol de colaboración, mensaje.
		Máquina de estados	Estado, evento, transición, acción.
Gestión del modelo	Vista de Presentación	Diagrama de Estructura de navegación	Clases de navegación, elementos de navegación, composición, asociación, dependencia, uso.
		Diagrama de Presentación	Clases de presentación, clases Frameset, asociaciones, componentes medias, paquetes.

<table>
<tr><td rowspan="3">Clases Modelo Entidad correspondientes a la representación del árbol jerárquico de medias.</td><td rowspan="2">Clases Vista que recibirán las peticiones del usuario al sistema y mostraran los mensajes de salida o respuestas.</td><td>Clases Controladoras que gestionarán las peticiones y la muestra de las respuestas.</td></tr>
<tr><td>Clases Modelo que contendrán la lógica de negocio para el procesamiento de la información.</td></tr>
<tr><td>Clases correspondientes al HLL seleccionado para la programación del software.</td><td>Clases Modelo Entidad para el procesamiento de la información persistente.</td></tr>
</table>

Figura 2.1 Distribución por sesiones del Diagrama de Clases de ApEM – L, tomando como base la arquitectura propuesta por el patrón MVC-E.

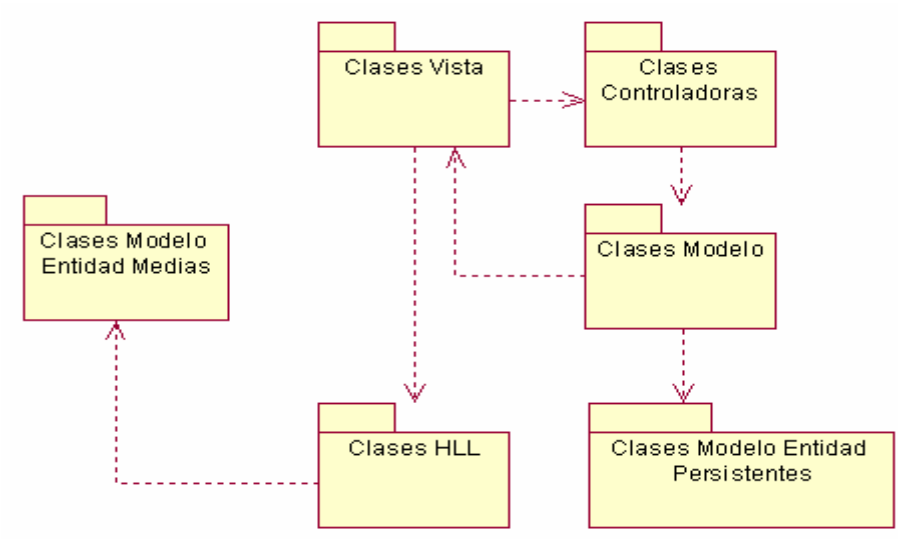


Figura 2.2 Vista de Gestión de un diagrama de clases de ApEM – L.

2.3.2 Vista de arquitectura.

La vista de arquitectura está compuesta por el ***diagrama de componentes*** y el ***diagrama de despliegue***. El último de los mencionados no sufre cambios en ApEM – L, no así el de componentes donde se incorporan restricciones en los tipos de componentes. Al seguir la arquitectura propuesta por el patrón MVC-E, normalmente los componentes podrán ser organizados por paquetes, que identificarían unidades físicas de encapsulamiento del código, como se representa el paquete *Presentación* en la figura 2.4. En su interior se encontrarán las clases vista, modelos, controladoras, HLL y modelo – entidad, organizadas en componentes que estarían internamente en el paquete Presentación. Eso mismo sucedería con cada uno de los módulos o subsistemas conformantes de las aplicaciones educativas en el futuro. Solo sería necesario luego un diagrama de componentes a un nivel de abstracción superior que represente como se comunican los distintos tipos de paquetes componentes del sistema.

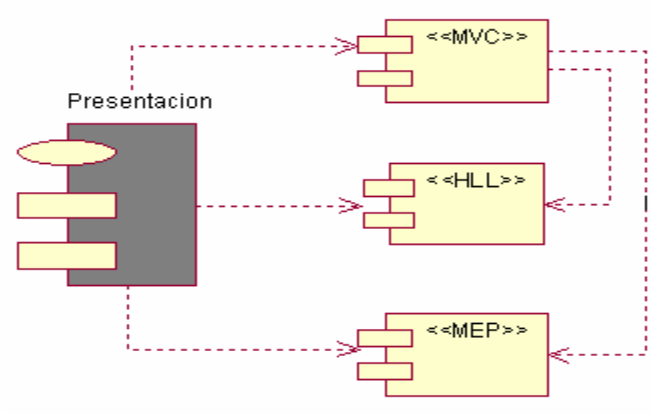


Figura 2.4 Diagrama de componentes con la incorporación de los estereotipos restrictivos de ApEM – L, para modelar el componente Presentación de la aplicación muestra.

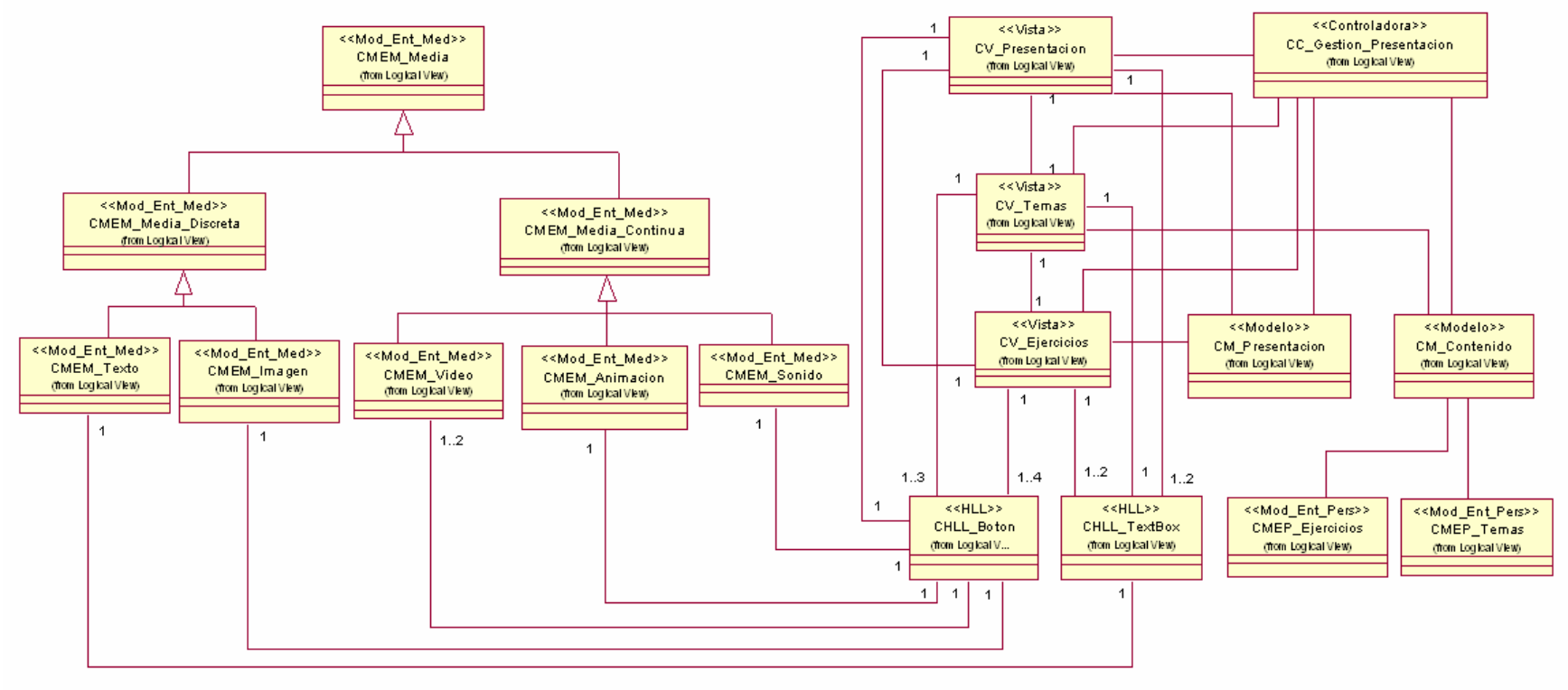


Figura 2.3 Ejemplo de un Diagrama de clases de ApEM – L con la estructura proporcionada con el patrón Modelo – Vista – Controlador – Entidad (MVC – E).

Tabla 2.2 Comparación de UML con ApEM – L en sus vistas estática y de arquitectura.

Aspecto o Elemento	Lenguaje Unificado de Modelado (UML)	Lenguaje para la Modelación de Aplicaciones Educativas (ApEM - L)
Diagrama de clases	• No establece un patrón arquitectónico que rija la concepción del diseño. • No contiene la semántica de tratamiento de las clases asociadas a las tecnologías multimedia e hipermedia.	• Establece el patrón arquitectónico MVC-E para la concepción del diseño de las aplicaciones educativas. • Plantea la semántica y los estereotipos restrictivos y descriptivos para las clases asociadas a las tecnologías multimedia e hipermedia. • Organiza la estructura del diagrama en secciones para la representación lógica de los distintos tipos de clases, incorporando las clases abstractas del modelo conceptual.
Diagrama de casos de uso	• No se establecen modificaciones a lo planteado por el lenguaje base UML.	
Diagrama de componentes	• No se modifica la semántica del lenguaje base para este tipo de diagrama, sino que se extiende esta al incorporar los elementos de organización en paquetes asociados al patrón arquitectónico MVC-E y sus relaciones de funcionamiento.	
Diagrama de despliegue	• No se establecen modificaciones a lo planteado por el lenguaje base UML.	

2.3.3 Vista de comportamiento.

La vista de comportamiento está compuesta por cuatro diagramas: ***de actividades***, ***de estado***, ***de secuencia*** y ***de colaboración***; siendo estos dos últimos generalizados en diagramas de interacción, pues como plantea la semántica de UML, representan la manera en la que los objetos de la aplicación intercambian mensajes para darle cumplimiento a sus responsabilidades. En ApEM – L solo ha sido modificado el diagrama de interacción de secuencia, con un estereotipo descriptivo y por ende decorativo para denotar el tiempo como variable de sumo interés en aplicaciones de este tipo, como se muestra en la Figura 2.5 de este capítulo.

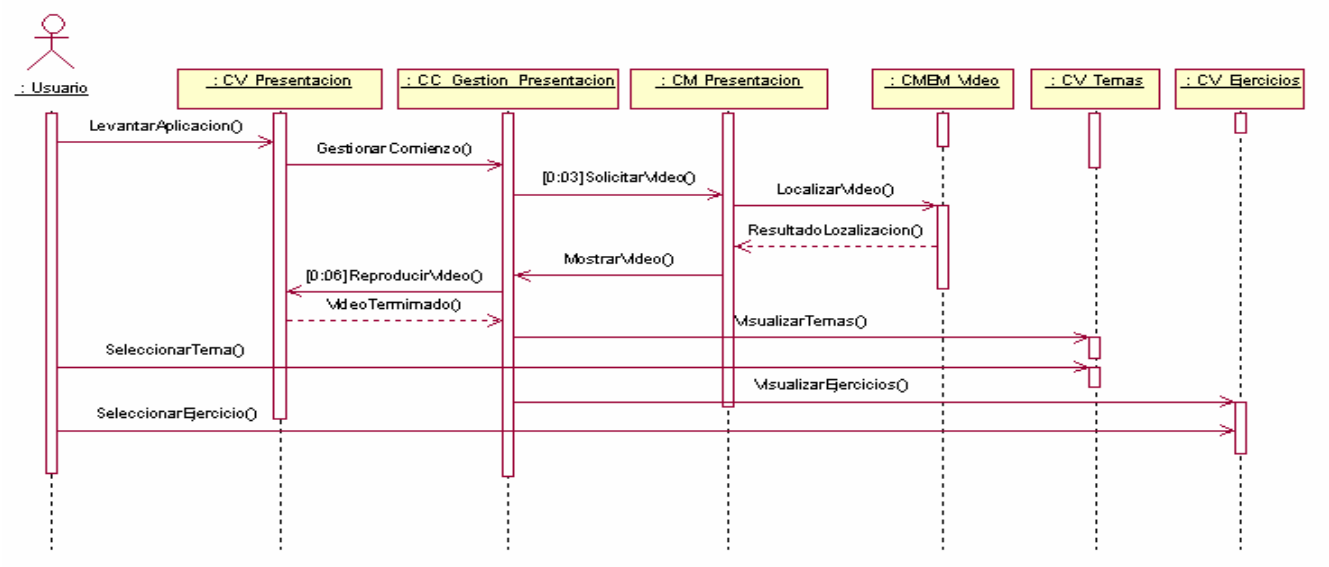


Figura 2.5 Diagrama de secuencia de ApEM – L.

Tabla 2.3 Comparación de UML con ApEM – L en su vista de comportamiento.

Aspecto o Elemento	Lenguaje Unificado de Modelado (UML)	Lenguaje para la Modelación de Aplicaciones Educativas (ApEM - L)
Diagrama de actividad	• No se modifica la semántica del lenguaje base para este tipo de diagrama, sino que se extiende esta al incorporar el estereotipo restrictivo asociado al tiempo y su representación siempre que sea necesario enriqueciendo la descripción funcional de este tipo de aplicaciones.	
Diagrama de secuencia	• No se establecen modificaciones a lo planteado por el lenguaje base UML.	
Diagrama de colaboración	• No se establecen modificaciones a lo planteado por el lenguaje base UML.	
Diagrama de estado	• No se establecen modificaciones a lo planteado por el lenguaje base UML.	

2.3.4 Vista de presentación.

"Una característica fundamental de las aplicaciones multimedia, es la composición de diferentes aspectos: en adición a las características estáticas como contenido de consideración, los tipos de media utilizados, el comportamiento temporal predefinido y el comportamiento controlado de interacción, así como también el diseño de la interfaz de usuario, incluyendo la planificación espacial de la presentación, tienen que ser tomadas en consideración." (Sauer, y otros, 2001)

Ciertamente el punto débil de la modelación en las aplicaciones educativas es la relacionada con los elementos de presentación y navegación, los cuales son de sumo interés para los desarrolladores de este tipo de aplicaciones. La vista que se describe en este epígrafe ha sido incorporada completamente al

lenguaje base UML, para permitir utilizar la semántica original de dicho lenguaje en la construcción de estructuras lógicas de presentación y navegación, incorporando un conjunto de estereotipos restrictivos y descriptivos para una mejor modelación, construyendo el ***diagrama de estructura de navegación*** y el ***diagrama de estructura de presentación***.

2.3.4.1 Diagrama de estructura de navegación

"Los diagramas clásicos provistos por UML, como el diagrama de clases o el de estado no son suficientes para modelar aspectos de los sistemas multimedia, por ejemplo, para modelar el espacio de navegación y para representar este modelo gráficamente." (Baumeistier, y otros, 2001)

Como consecuencia de la ineficiencia del lenguaje base UML para modelar este tipo de contexto en los software educativos, se ha definido el ***diagrama de estructura de navegación***. Para el desarrollo del diagrama de estructura de navegación se han definido, sobre el concepto de clase, una segunda clasificación de las mismas, por lo que se enriquece la semántica del concepto original de clases, quedando las siguientes: ***clase menú***, ***clase índice***, ***clase consulta*** y ***clase botón***, además de utilizar las ya definidas ***clases modelo-entidad-media texto*** y ***modelo-entidad-media imagen*** pues estas pueden ser elementos importantes de navegación en el modelo, así como comportarse como clases que visualizan información. La ***clase menú*** (<<Menu>>), es el elemento de composición de una clase vista, desde donde se puede llegar a otras diferentes clases vistas con las cuales se conecta este menú, pues contiene una lista de las opciones de movimiento siguiente, pero no los valores de esas opciones, los cuales serán ofrecidos por el resto de los tipos de clases. Dicho menú no tiene que necesariamente llevar a una clase vista directamente, sino que puede ser a través de otro tipo de clase de navegación (guía, consulta o índice). La ***clase consulta*** (<<Consulta>>) es aquel tipo de clase que permite el enlace con una clase vista a y través de un valor directo (variable de consulta) asignado a la búsqueda para la visualización del elemento a mostrar, que se sitúa como identificador en la relación entre las clases. Por último la ***clase índice*** (<<Indice>>), es aquella clase que denota el valor de direccionamiento hacia una clase vista a partir de una opción determinada. La ***clase botón*** (<<Boton>>) es aquella clase que denota la existencia de un elemento interactivo del tipo botón para producir un camino en la navegación hacia una clase vista, y utilizando como intermediarias a clases de tipo consulta o índice. (Consultar Figura 2.6)

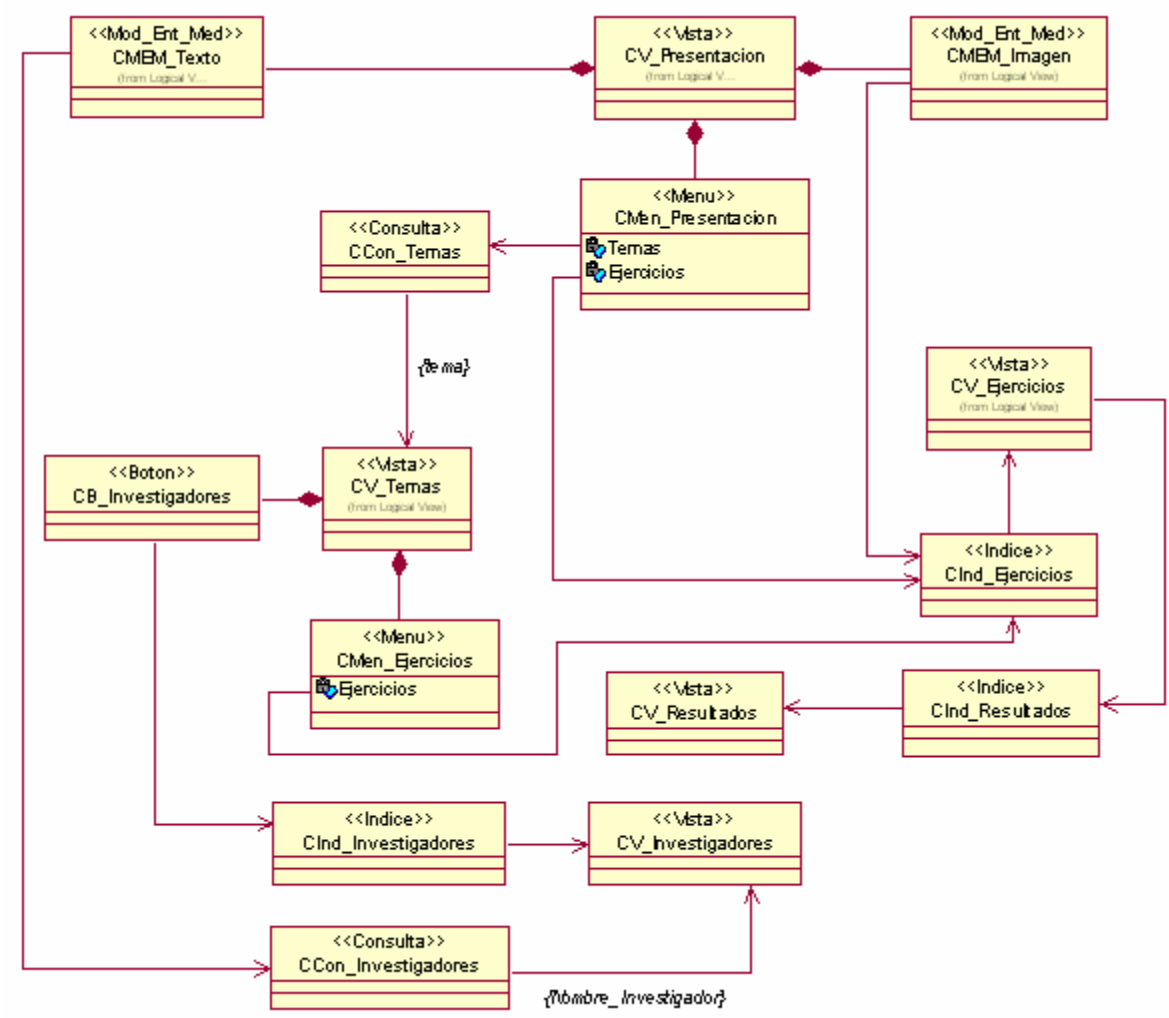


Figura 2.6 Diagrama de estructura de navegación de ApEM – L donde se representan las clases que posibilitan la navegación entre las distintas clases vistas que componen la aplicación.

2.3.4.2 Diagrama de estructura de presentación

"Una modelación explícita de la planificación espacial es necesaria para ser capaces de especificar presentaciones uniformes de diferentes objetos como parte de un modelo integral de la aplicación." (Sauer, y otros, 2001)

Un aspecto esencial en las aplicaciones educativas con tecnología multimedia e hipermedia es la definición de la estructura que tendrán las futuras interfaces de comunicación con el usuario en cuanto a sus componentes y distribución espacial. En el contexto de ApEM – L no se trabajará la distribución espacial, pues las herramientas CASE actuales que soportan el lenguaje no lo permiten tal y como pretendemos dibujarlo hoy día. Por el contrario trabajaremos la estructura que tendrán esas interfaces

de comunicación incorporando un conjunto de estereotipos restrictivos y descriptivos que permitirán una organización lógica de los elementos conformantes de dichas interfaces y dejándole a los diseñadores gráficos la función de decidir donde y como serán en términos visuales dichos elementos.

Para la mejor estructuración del modelo, se realiza una segunda clasificación sobre el concepto original de clase, definiendo dos nuevos tipos de estas: la ***clase Estáticos*** y la ***clase Interacción***, las cuales seccionarán los elementos que cumplan con cada una de las características que denotan los propios nombres. La clase *estáticos* agrupará los componentes que solo tiene como función visualizar información, pero que no permiten interacción con el usuario. Todo lo contrario con los agrupados bajo la clase *interacción,* los cuales serán los elementos de la vista que permiten interacción del usuario con el sistema informático modelado. Un ejemplo se muestra en la Figura 2.7.

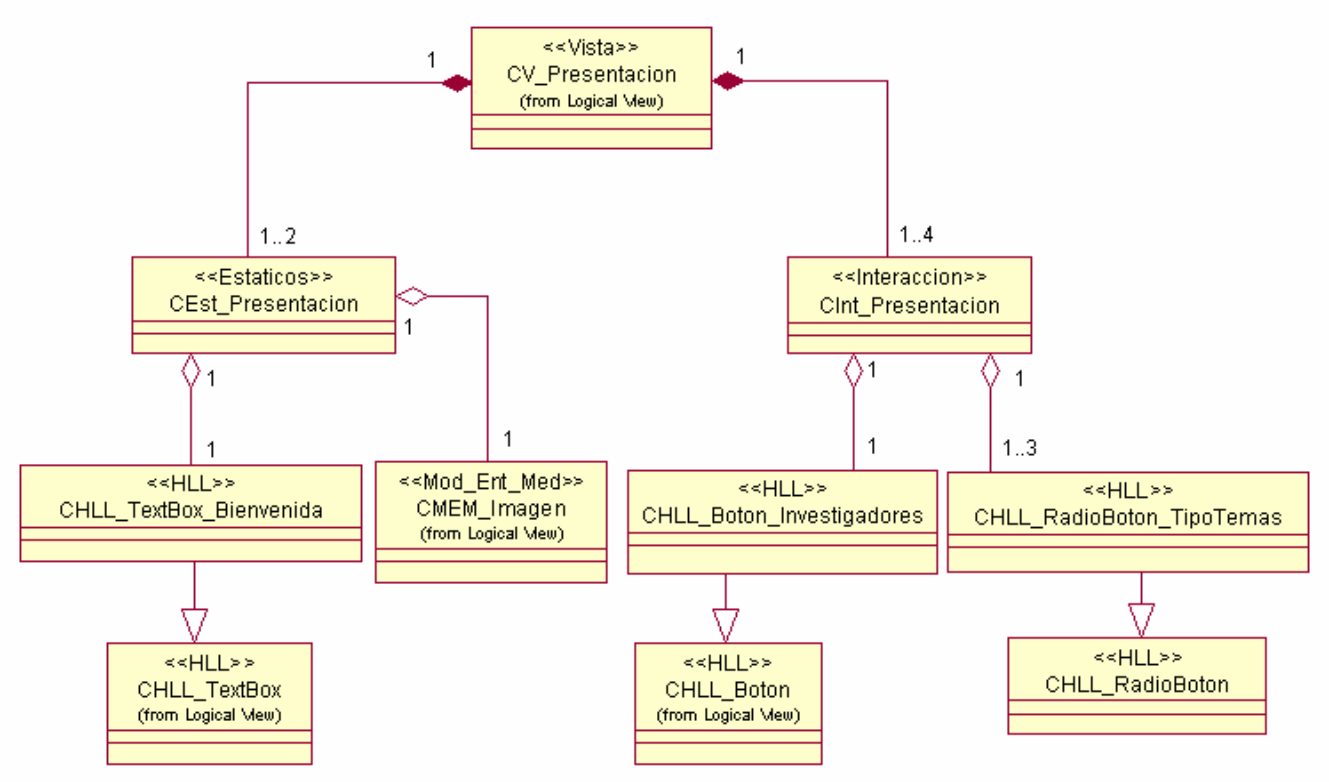


Figura 2.7 Diagrama de estructura de presentación de ApEM – L.

Tabla 2.4 Comparación de UML con ApEM – L en su vista de presentación.

Aspecto o Elemento	Lenguaje Unificado de Modelado (UML)	Lenguaje para la Modelación de Aplicaciones Educativas (ApEM - L)
Diagrama de estructura de navegación	• No existe este diagrama, lo que dificulta la representación de la estructura o mapa de navegación dentro de la aplicación.	• Extiende la semántica de los diagramas de clases para poder utilizarlos en este tipo de representación, incorporando los estereotipos restrictivos de clases: menú, índice, consulta y botón, además de utilizar las ya definidas: clases modelo entidad media texto y modelo entidad media imagen.
Diagrama de estructura de presentación	• No existe este diagrama, lo que dificulta la representación de la estructura o elementos de composición visual de las interfaces de comunicación con el usuario, o lo que es lo mismo las clases vista.	• Extiende la semántica de los diagramas de clases para poder utilizarlos en la representación de la estructura de las presentaciones, incorporando los estereotipos restrictivos de clases: estáticos e interacción y un árbol jerárquico a partir de estos estereotipos que agrupa los componentes visuales de acuerdo a su función en la interfaz a la que pertenece.

Conclusiones parciales.

En el capítulo que recién se ha presentado se han utilizado las posibilidades de extensión ofrecidas por el Lenguaje Unificado de Modelado (UML), para desarrollar una extensión que permita modelar las aplicaciones educativas que hoy se producen en la universidad. Al mismo tiempo se ha podido arribar a las siguientes conclusiones:

1. El Leguaje Unificado de Modelado no soporta con sus vistas y diagramas todos los aspectos que son necesarios modelar en el software educativo cubano.
2. Se hace necesario introducir un conjunto de estereotipos descriptivos y restrictivos para la extensión de la semántica de los establecido por el lenguaje base y de esta forma permitir incorporar los conceptos manejados en el contexto del software educativo.
3. Se ha respetado el lenguaje base UML, así como lo establecido por el lenguaje OCL como estándar en la modelación orientada a objetos.

4. ApEM – L incorpora la vista de presentación como aporte fundamental a la modelación de este tipo de aplicaciones con dos diagramas que permiten definir la estructura para la navegación y la estructura para la presentación.

Capítulo 3

Una experiencia práctica de la aplicación de ApEM-L.

Introducción

"Diferentes lenguajes de modelación pueden ser utilizados, pero la importancia de usar un estándar es clara: provee un lenguaje común el cual facilita la comunicación entre los miembros del proyecto así como con el mundo externo y futuros lectores de la documentación del sistema." (Baumeistier, y otros, 2001)

Necesariamente se hace inminente probar ApEM – L en un entorno productivo real de la universidad, que permita de alguna manera probar si los conceptos manejados y extendidos sobre el lenaguaje base permiten representar al menos una buena parte de los sistemas educativos y mejoran el entendimiento de los desarrolladores y la calidad de la documentación ingenieril que desarrollamos.

En el presente capítulo se dedicará especial atención a la presentación de una aplicación del nuevo lenguaje de modelación a un proyecto productivo de la Facultad 9: A Jugar.

3.1 El entorno productivo del proyecto "A Jugar".

A Jugar es un proyecto productivo que se desarrolla en la Facultad 9 de la Universidad de las Ciencias Informáticas (UCI) desde hace cuatro años atrás, que ha transcurrido por diversas etapas, matizadas por la falta de documentación del producto, además de una descuidada gestión de configuración, lo que ha traído como consecuencia una excesiva demora en el cumplimiento de los cronogramas. El objetivo fundamental del proyecto, es el desarrollo de un producto educativo interactivo para desarrollar el aprendizaje de los niños en edades preescolares. El proyecto tiene como principales características principales las siguientes:

1. Tiene hoy un equipo de desarrollo formado por 10 estudiantes de la carrera de Ingeniería en Ciencias Informáticas y 1 profesor líder del proyecto.
2. Cuenta con un equipo de trabajo de diseño que es el responsable de la construcción de las medias solicitadas por los guionistas.

3. Cuenta con un equipo numeroso de especialistas en educación preescolar, que fungen como guionistas del producto, los cuales son los encargados de desarrollar los guiones de contenido de la aplicación.
4. Solo se trabaja con los guiones para el desarrollo del producto, lo que produce ambigüedades en el entendimiento de lo expresado en estos y la no disponibilidad de información o documentación ingenieril para futuras versiones o el mantenimiento del producto.

Se ha seleccionado este proyecto tomando en consideración las características mencionadas, para la descripción tanto de la ***Presentación*** del producto, como de la ***Tarea No. 1*** del ***Nivel 4*** del producto según el guión de contenido del mismo, a través de las vistas y diagramas presentados en el capítulo 2 de esta memoria.

3.2 Diagrama de Casos de Uso.

Teniendo en cuenta el dominio del problema dentro del contexto del proyecto presentado en el epígrafe anterior, y sabiendo que a este diagrama no se le realizó modificación alguna por ApEM – L se presenta a continuación (Ver Figura 3.1) el diagrama de casos de uso correspondiente al dominio seleccionado.

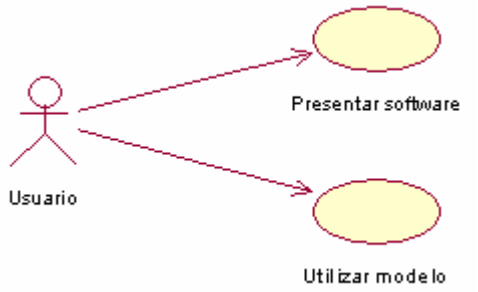


Figura 3.1 Diagrama de Casos de Uso del dominio seleccionado del Proyecto Productivo A Jugar.

La Descripción Textual de los Casos de Uso quedaría como sigue a continuación, utilizando la modificación presentada en el capítulo 2 de esta memoria, para incorporar los elementos significativos del software educativo.

Descripción Textual del Caso de Uso *Presentar Software*			
Actores del caso de uso	Usuario (inicia)		
Propósito	El caso de uso tiene como propósito permitir al usuario ejecutar la aplicación y visualizar la presentación de la misma.		
Resumen	El caso de uso se inicia cuando el usuario pulsa sobre el ejecutable del programa, se verifica que no esté ejecutándose otra instancia del mismo, se verifican los requerimientos de hardware necesarios, se visualiza el video de presentación y culmina el caso de uso cuando se presenta el menú principal.		
Casos de uso asociados			
Referencias	R1, R2, R3.		
Precondiciones	• Otra instancia del caso de uso no puede estarse ejecutando.		
Poscondiciones	• Menú principal de la aplicación mostrado.		
Curso Normal de los Eventos			
Acciones del Actor		**Respuesta del Sistema**	
1. El usuario pulsa el fichero ejecutable del software educativo.		1.1 El sistema verifica que no se esté ejecutando otra instancia del caso del caso de uso. 1.2 Si la respuesta es negativa, se verifican los requerimientos no funcionales de video, memoria y velocidad de procesamiento. 1.3 Si la verificación es positiva se muestra el video de presentación de la aplicación. 1.4 Al concluir el video de presentación, se muestra el menú principal de la aplicación concluyendo el caso de uso.	
Cursos Alternos de los Eventos			
Acción	**Curso Alterno**		
1.2	Si se está ejecutando otra instancia del caso de uso, se le muestra un mensaje de error al usuario con la información y se culmina el caso de uso.		
1.3	Si la verificación no es satisfactoria se le presenta un mensaje de error al usuario, informando que la estación no tiene los requerimientos mínimos para la ejecución del software y se culmina el caso de uso.		
Prioridad	Crítica.		
Mejoras			
Medias a utilizar	**Tipo de Media**	**Descripción**	**Estado**
	Imagen	Fondo del menú principal	Existente
		Icono representativo de cada opción del menú.	En construcción
		Iconos representativos de las opciones estándares en la aplicación	En construcción
	Video o Animación	Video de presentación del software	En construcción

	Sonido	Sonido de pequeña duración para cuando se pase por encima de las opciones del menú principal.	En localización
		Sonido para cuando se seleccione una opción del menú principal	En localización
	Texto	Textos de las opciones	En construcción
		Testo de bienvenida al softwate	En construcción
Elementos pedagógicos	No puede ser posible que el usuario de la aplicación tenga la posibilidad de abandonar el video de presentación, debido a que en él se explican las características principales de la mismas y su forma de funcionamiento.		

Descripción Textual del Caso de Uso *Utilizar modelo*	
Actores del caso de uso	Usuario (inicia)
Propósito	El caso de uso tiene como propósito permitir al usuario desarrollar la tarea número 1 del nivel 4 del software donde el objetivo fundamental es hacer click y arrastrar el mouse.
Resumen	El caso de uso se inicia cuando el usuario selecciona el nivel 4, y dentro de este la tarea No 1 y dentro de esta el Ejercicio 1 , luego el usuario se enfrentará a un conjunto de indicaciones donde las acciones principales son selección con el click y arrastrar los objetos hasta un punto determinado, donde si están correctos es satisfactorio y culmina el caso de uso.
Casos de uso asociados	
Referencias	R5, R6, R7, R8.
Precondiciones	• Niveles previos vencidos. • El usuario ha seleccionado Nivel 4, Tarea 1, Ejercicio 1.
Poscondiciones	• El usuario ha avanzado hacia la tarea siguiente la cual es mostrada en pantalla.

Curso Normal de los Eventos	
Acciones del Actor	**Respuesta del Sistema**
1. El usuario selecciona la opción Nivel 4.	1.1 El sistema muestra un listado con las tareas de este nivel.
2. El usuario selecciona la Tarea 1.	2.1 El sistema muestra el listado de los ejercicios disponibles.
3. El usuario selecciona el Ejercicio 1.	3.1 Aparece en el borde superior derecho dela pantalla la mascota. 3.2 Se muestra en la parte inferior derecha la carta de orientación. 3.3 Se muestra el árbol en la parte inferior izquierda, la fuente en el centro y el banco en la parte superior derecha.
4. El usuario hace click sobre uno de los elementos según la carta de orientación.	4.1 Se verifica que el objeto seleccionado corresponda con el establecido por la carta de orientación. 4.2 Si corresponde, el objeto se ilumina.

5. El usuario arrastra el objeto hasta el ómnibus y suelta el botón del ratón.	5.1 Se verifica que el objeto haya llegado hasta el ómnibus. 5.2 Si llegó el objeto al ómnibus, este se ilumina. 5.3 Se verifica que queden objetos a seleccionar por el usuario. 5.4 Si quedan objetos por seleccionar el usuario pasa a la acción 6.
6. El usuario vuelve a la acción 4 para seleccionar nuevamente un objeto según la carta de orientación.	6.1 Se repiten las acciones del sistema desde la 4.1 hasta la 5.3

Cursos Alternos de los Eventos

Acción	**Curso Alterno**
4.2	Si no corresponde el objeto con el establecido en la carta de orientación, los elementos palidecen y la mascota vuelve a atrás nuevamente, retornando el caso de uso a la acción 3.3
5.2	Si el objeto no ha llegado al ómnibus, quiere decir que el usuario no completó la tarea, por lo que este vuelve a su posición original y el caso de uso a la tarea 3.3
5.4	Si no quedan objetos por seleccionar, se visualiza la animación del ómnibus hacia el campismo, y la opción en el borde inferior derecho de seguir al próximo ejercicio; culminando de esta forma el caso de uso.
Prioridad	Secundaria
Mejoras	

Medias a utilizar	**Tipo de Media**	**Descripción**	**Estado**
	Imagen	Carta de orientación (contará con 3 elementos: árbol, fuente y banco)	En construcción.
		Ómnibus	Existente
		Árbol	Existente
		Banco	Existente
		Fuente	Existente
		Campismo	Existente
	Video o Animación	Video del ómnibus hacia el campismo	En construcción
	Sonido	Parlamento de la mascota con la explicación inicial del ejercicio	En construcción
		Sonido grave que identifique error en la acción cometida	En construcción
	Texto	Orientaciones para la carta de guía del ejercicio	En construcción
Elementos pedagógicos	Tener en cuenta que el ómnibus correcto es el más cercano al banco.		

3.3 Diagrama de actividad.

Es sumamente ventajoso, tal y como sucede en la modelación del software convencional representar los flujos de eventos a través de diagramas de actividades. Estos diagramas permitir visualizar mejor los flujos y ayudan al entendimiento del equipo de desarrollo. Corresponden a los casos de uso descritos en el epígrafe anterior los diagramas de actividad siguientes, mostrados en las figuras 3.2 y 3.3 respectivamente.

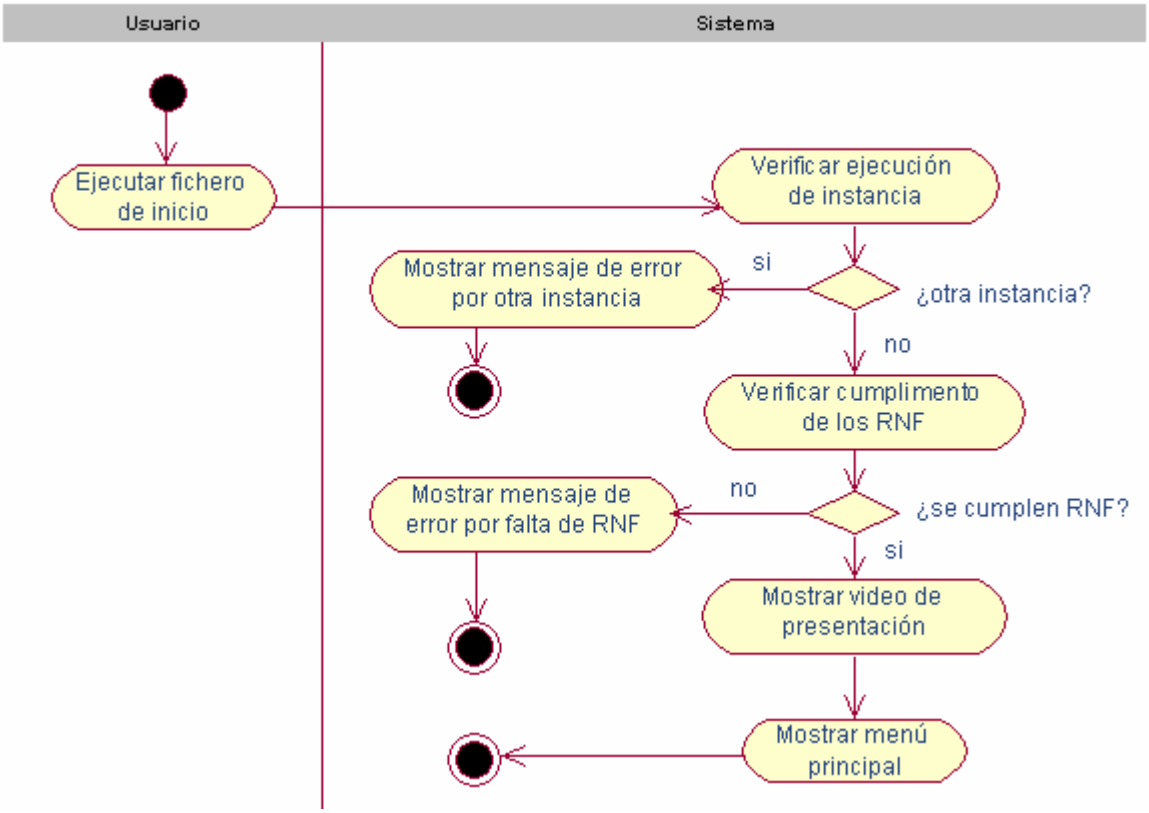


Figura 3.2 Diagrama de actividades del Caso de Uso Presentar Software.

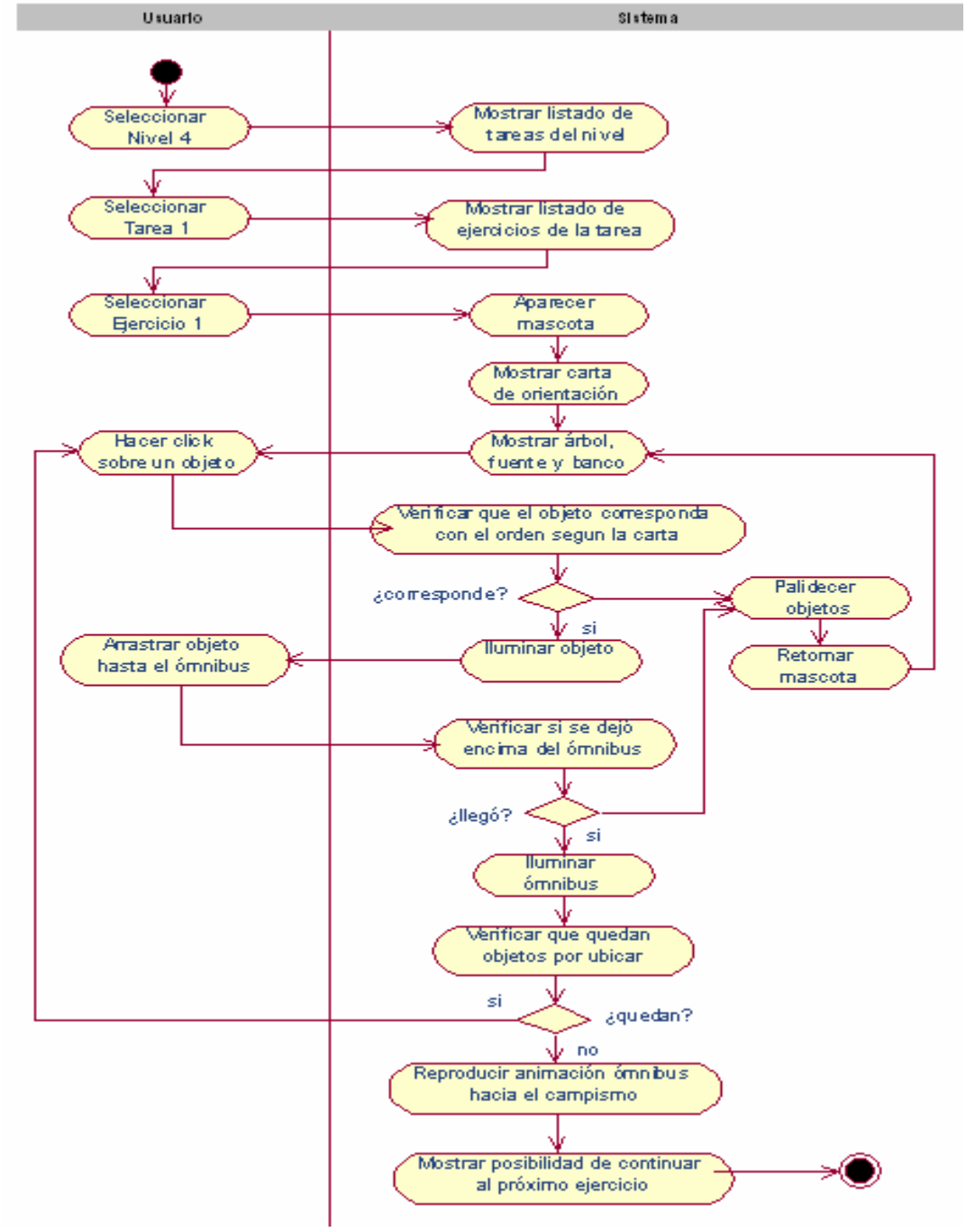


Figura 3.3 Diagrama de actividades del Caso de Uso Utilizar Modelo

3.4 Diagrama de clases (Modelo Conceptual).

"La estructura lógica de una aplicación multimedia es un aspecto independiente y tiene que ser especificado explícitamente. Como en los procesos convencionales de la Ingeniería de Software, este es el punto de comienzo para la modelación de aplicaciones." (Sauer, y otros, 2001) Es muy importante y de una gran relevancia posterior el desarrollo de un exhaustivo modelo conceptual de la aplicación que nos permita conocer y relacionar los conceptos que sustentan el funcionamiento de la aplicación educativa.

"El resultado de la fase de análisis es el modelo conceptual del dominio del problema definido por clases relevantes para el dominio y las asociaciones entre estas clases. La meta es la captura de la semántica del dominio y tanto como sea posible de los aspectos de navegación y presentación." (Baumeistier, y otros, 2001)

Se presenta en la Figura 3.4, el diagrama de clases correspondiente al ***Caso de Uso Presentar Software***, en el cual se utilizan todos los elementos descritos en el capítulo 2 para las modificaciones que ApEM – L introduce al lenguaje base UML, en pos de representar todos los conceptos lógicos y estructurales necesarios en este tipo de aplicaciones.

3.5 Diagrama de estructura de navegación.

"La estructura lógica de una aplicación, así como también su control interactivo, acompañado por una jerarquía de señales de diálogo particularizadas, y partes del comportamiento temporal, pueden ser adecuadamente modeladas por UML. Pero construcciones del lenguaje especializadas y más avanzadas son necesarias para describir el ensamblaje temporal entre diferentes objetos media." (Sauer, y otros, 2001)

Ese ensamblaje temporar entre los objetos a los que se refieren (Sauer, y otros, 2001) no son más que las posibilidades de navegación en una aplicación educativa, que se hace necesario representar, como ha quedado descrito en el capítulo anterior y para lo que en el caso concreto del producto en análisis se presenta en la Figura 3.5.

3.6 Diagrama de estructura de presentación.

"El último paso en la tarea de diseño es el diseño de la presentación, donde es producida una versión ruda de la interfaz de usuario." (Baumeistier, y otros, 2001) Se dice ruda, por solo contener los elementos que tendrá visualmente la aplicación y su distribución espacial. Lamentablemente, todas las herramientas CASE que hoy soportan la modelación en UML, no disponen de los estereotipos ni tienen incorporada la semántica que nos permita realizar agrupamientos visuales de elementos para representar interfaces, tal y como han intentado autores anteriores en los últimos 7 años. Por este motivo, en esta memoria solo se ha preferido representar una estructura de la presentación y no su distribución espacil, para de esta forma ayudar a los desarrolladores a entender los componentes visuales necesarios para una interfaz, sin importar su distribución espacial, que sigue estando muy condicionada por las decisiones de los diseñadores gráficos de las aplicaciones. En la Figiura 3.6 se muestra el Diagrama de estructura de presentación para el Caso de Uso Presentar Software.

3.7 Diagrama de Secuencia.

Se ha seleccionado en este caso el flujo de eventos descrito en el Caso de Uso Presentar Software, y apoyándonos en el Diagrama de clases presentado para la estructura lógica del mismo. En la Figura 3.7, se refleja la modificación introducida para la representación del tiempo cuando su incorporación sea necesaria, por la importancia del control de esta variable en las aplicaciones educativas y en especial en el trabajo con las medias continuas como sonido, video y animaciones.

3.8 Elementos a considerar con el uso de ApEM – L.

Al concluir el trabajo en la Tarea 1, Ejercicio 1, Nivel 4 del proyecto productivo A Jugar, se realizó una encuesta (Consultar Anexo 6) a diez (10) de los 16 miembros del equipo de desarrollo para conocer sus consideraciones del uso y la eficiencia de la nueva propuesta. La composición del grupo de trabajo en este proyecto tiene la siguiente composición:

Rol	**Cantidad**	**Nivel Escolaridad Técnica**
Líder del proyecto	1	Ingeniero en Telecomunicaciones.
Diseñadores gráficos	3	Ingenieros en Diseño Industrial
Guionistas (Especialistas Pedagógicos)	4	Licenciados en Educación Pre-escolar (1 MSc. y 3 Dr.)
Programadores	8	Estudiantes del 4to año de la UCI

Los miembros del equipo entrevistados conformaban la siguiente muestra:

- 1 Lider del proyecto. (100 %)
- 2 Diseñadores gráficos. (66,66 %)
- 2 Guionistas (50 %)
- 5 Programadores. (62,5 %)
- **10 entrevistados en Total (62,5 %)**

El análisis cualitativo de la estadística que arrojó la encuesta realizada a los desarrolladores es el siguiente:

1. El 70 % considera que es totalmente ventajosa y el 30 % parcialmente ventajosa, la utilización de la extensión ApEM – L para la modelación de las aplicaciones educativas.
2. Consideran que se sienten totalmente cómodos con la utilización de la nueva notación y los estereotipos restrictivos utilizados para la notación el 70 % de los encuestados, 10 % parcialmente y solo el 20 % considera no sentirse cómodo con la nueva propuesta.
3. El 80 % de los desarrolladores creen que es sencilla y fácil de entender y aplicar si se tienen conocimientos básicos de programación.
4. El 70 % cree que es muy ventajosa la manera en la que se aborda la vista de presentación, el 50% aprueba la manera en que se trabaja la vista arquitectónica y el 60 % coincide con la propuesta de modelación de la vista de presentación.
5. Aprueban la utilización de las descripciones textuales de los casos de uso para la descripción de los flujos de eventos en el software de manera total el 60 %, parcial el 30 % y solo el 10 % no coincide con lo planteado por la propuesta.
6. Los miembros del proyecto encuestados plantean que deben de trabajarse aún más los elementos del Diagrama de estado y el de actividades.
7. La totalidad de los participantes en la encuesta considera necesario el desarrollo de una herramienta CASE que posibilite un trabajo mejor y más humanizado al documentar las aplicaciones informáticas utilizando ApEM – L.
8. El 70 % de los desarrolladores participantes de la investigación posterior a la utilización de ApEM – L consideran obsoletos con el uso de la extensión de UML, artefactos como el Mapa de Navegación y el Guión de contenido en flujos posteriores al de Requerimientos en procesos iterativos e incrementales como RUP.

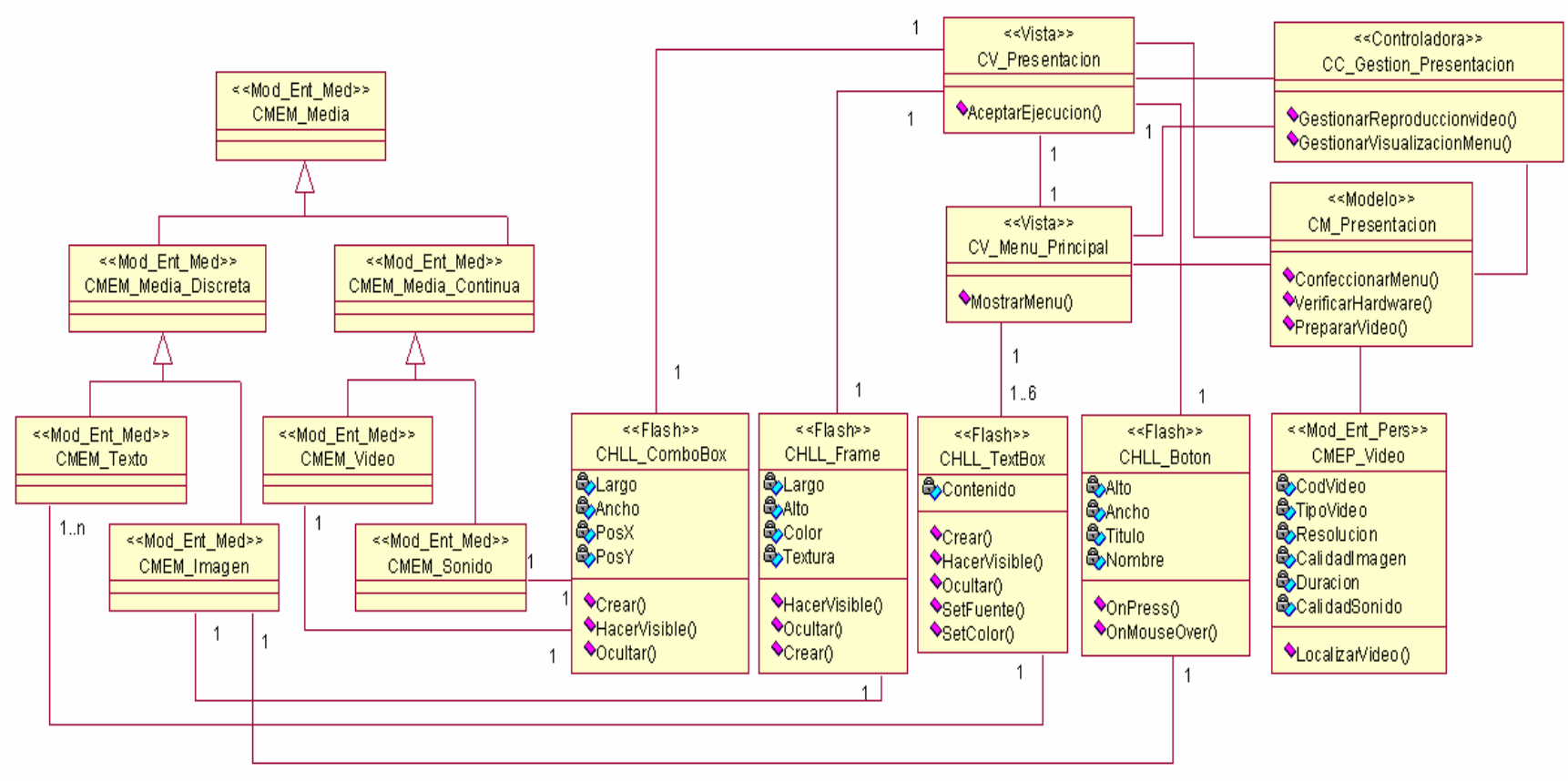
Si se valora con detenimiento la información ofrecida en los ocho puntos anteriores se denota claramente que la utilización de la nueva extensión propuesta para RUP, es viable en todo sentido, mejora considerablemente la calidad de la documentación de los software educativos y permite una representación ingenieril más clara para los desarrolladores, lo que directamente se revierte en mejora de la calidad del producto y aumento del nivel de comunicación entre los diferentes miembros del equipo de trabajo.

Conclusiones parciales.

El trabajo en un proyecto productivo para el desarrollo de un producto convencional se vuelve difícil cuando la documentación no soporta las diferentes fases del mismo, ni apoya la comunicación y el entendimiento tecnológico entre los diferentes miembros del equipo. En el desarrollo de una aplicación educativa sucede algo similar e incluso mucho más agudizado si se tiene en cuenta lo heterogéneo del equipo y la necesidad en aumento del entendimiento, así como la urgencia de reflejar con exactitud los distintos caminos que desde el punto de vista pedagógico puede tomar el flujo de eventos. Teniendo en cuenta lo recién abordado, al finalizar lo expresado en este capítulo se puede arribar a las siguientes conclusiones:

1. Se determinó la no disponibilidad de documentación para el desarrollo del proyecto productivo A Jugar y en especial de la carencia de modelos del software que facilitaran su desarrollo, implementación y mantenimiento.
2. Se comprobó la veracidad de la propuesta de modificaciones para la modelación de los software educativos.
3. Se generó la documentación del Ejercicio 1, de la Tarea 1, del Nivel 4 del producto mencionado.

Figura 3.4 Diagrama de Clases (Modelo Conceptual) del Caso de Uso Presentar Software.



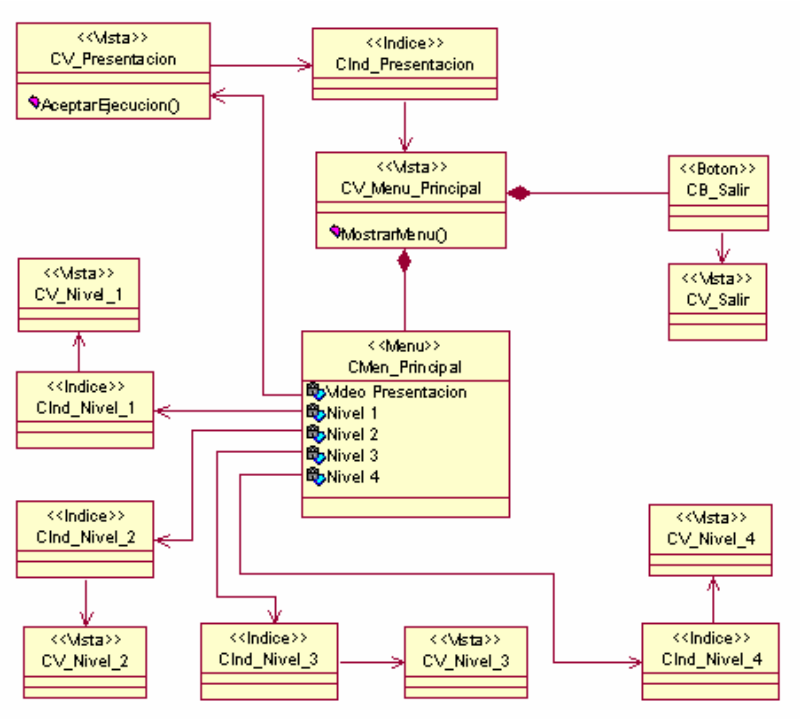


Figura 3.5 Diagrama de estructura de navegación del Caso de Uso Presentar Software.

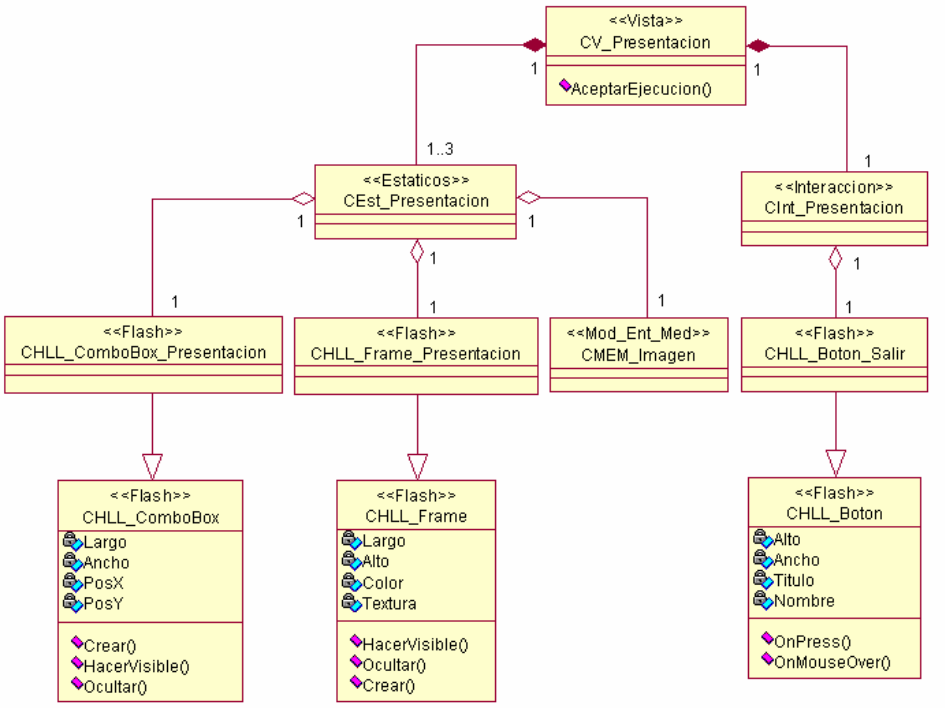


Figura 3.6 Diagrama de Estructura de Presentación de la Interfaz de usuario Presentación.

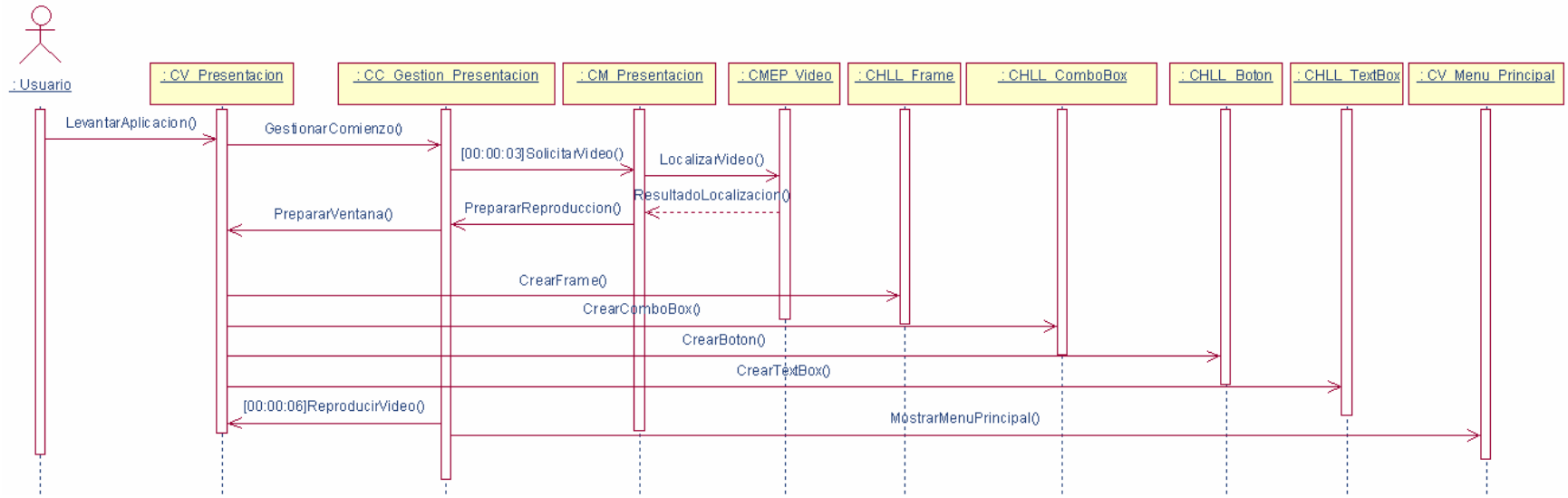


Figura 3.7 Diagrama de secuencia del Caso de Uso Presentar Software.

Conclusiones

CONCLUSIONES GENERALES

"Diferentes lenguajes de modelación pueden ser utilizados, pero la importancia de usar un estándar es clara: provee un lenguaje común el cual facilita la comunicación entre los miembros del proyecto así como con el mundo externo y futuros lectores de la documentación del sistema. Adicionalmente, las herramientas basadas en estos estándares pueden ser usadas para el desarrollo, prueba y validación de los modelos. UML ha sido aceptado como el estándar industrial de facto para la modelación orientada a objetos. Puede ser extendido a través del uso de mecanismos de estereotipos, valores etiquetados y el lenguaje OCL (Object Constraint Language)." (Baumeistier, y otros, 2001)

La meta de esta memoria, fue reflejar en un grado reducido pero claro, los elementos fundamentales de una nueva extensión al lenguaje base UML, para la modelación de aplicaciones educativas en el entorno productivo de la Universidad de las Ciencias Informáticas (UCI), logrando incidir en los elementos mencionados por (Baumeistier, y otros, 2001) en el párrafo anterior.

Al finalizar el presente trabajo, se consideran cumplido el objetivo trazado y suficientemente fundamentada la idea a defender. A este planteamiento se puede arribar a partir de las siguientes conclusiones:

1. Se cumplieron correctamente los objetivos tanto general como cada uno de los específicos planteados para la investigación.
2. Se demostró la hipótesis expuesta en esta memoria como guía de la investigación científica.
3. Se estudiaron las notaciones, modelos y métodos actuales, desarrollados en los últimos 10 años, utilizados para la modelación de aplicaciones con tecnología multimedia e hipermedia y se determinaron los elementos reutilizables y genéricos que pudieron extenderse al contexto productivo nacional.
4. Se revisaron exhaustivamente el Lenguaje de Modelación de Objetos (OCL) y la Clasificación de Estereotipos para lenguajes de modelación orientada a objetos, en aras de utilizar los elementos que rigen los estándares en ambos y dar cuerpo a la solución presentada.
5. Se identificaron los elementos del contexto productivo UCI, que debían incluirse en la propuesta sobre la base de las encuestas y entrevistas realizadas, para garantizar una mejor inserción en los procesos de desarrollo de software educativos actuales de la universidad.

6. Se redefinió la variación al patrón Modelo – Vista – Controlador (MVC) original y sobre este a la variante de Modelo – Vista – Controlador – Entidad (MVC – E) desarrollado en el 2006, para una mejor identificación de las clases media en media – entidad y media – persistencia.
7. Se validó la propuesta en un área del proyecto productivo “A Jugar” de la Facultad 9 logrando entregar a dicho equipo de desarrollo una variante ingenieril sólida en la modelación de este tipo de aplicaciones.
8. Se generó una extensión al lenguaje de modelado orientado a objetos UML, sobre la base de sus elementos para la extensión y apoyados en el trabajo con estereotipos descriptivos y restrictivos, adicionando una nueva vista para el trabajo con la navegación y la presentación de las interfaces de comunicación con el usuario.

Recomendaciones

RECOMENDACIONES

Al concluir la presente memoria escrita, se hace necesario realizar el siguiente conjunto de recomendaciones para perfilar de esta forma investigaciones futuras en el propio ámbito de la actual y mejorar los resultados científicos que se han obtenido:

1. Enriquecer los elementos conceptuales de la nueva extensión propuesta, sobre la base de conceptos de la Programación Orientada a Objetos como polimorfismo y encapsulamiento, para brindarle a ApEM – L una mayor solidez en el tratamiento de la información.
2. Valorar el diseño de estereotipos decorativos, sobre la base de una mejor representación visual de los elementos componentes del software educativo, para aumentar el entendimiento de especialistas pedagógicos y de diseño con el resto de los miembros de los equipos de desarrollo.
3. Aplicar la nueva notación propuesta a los proyectos productivos de software educativo en la UCI, en una fase de prueba, que permita obtener elementos para la mejora de la solución presentada y lograr una fácil incorporación de soluciones a las características productivas de la universidad.
4. Conformar con especialistas de la producción de software educativo en la UCI, investigadores del área del conocimiento y desarrolladores de otras instituciones a fines, un grupo de estudio y desarrollo de la notación propuesta para lograr el desarrollo de nuevas versiones superiores y escalables rápidamente.

Referencias

RFERENCIAS BILBIOGRÁFICAS

Aizenbud-Reshef, Netto, y otros. 2006. Model traceability. [En línea] 2006. [Citado el: 30 de octubre de 2006.] http://find.galegroup.com/itx/infomark.do?&contentSet=IAC-Documents&type=retrieve&tabID=T002&prodId=CDB&docId=A9472469&source=gale&srcprod=CDB&userGroupName=ucinf&version=1.0.

Almeida Cintra, Nurileidis y Viera Cisnero, Yoannia. 2007. *Principios para la evaluación y certificación de la calidad de los productos de software educativo en la UCI. (tesis para optar por el título de Ingeniero en Ciencias Informáticas).* La Habana : UCI, 2007.

Autores, Colectivo de. 1998?. Software Educativo. [En línea] 1998? [Citado el: 26 de abril de 2006.] http://cavalletto.freeservers.com/pag%20educativa%20de%20amortiza.htm.

Balasubramanian, Venkatraman, Bieber, Michael y Isakowitz, Tomás. 2001. A case study in systematic hypermedia design. [En línea] 2001. [Citado el: 13 de diciembre de 2006.] http://citeseer.ist.psu.edu/cache/papers/cs/29502/http:zSzzSzweb.njit.eduzSz~bieberzSzpubzSzisj01.pdf/balasubramanian01case.pdf.

Barrera Yanes, Rafael. 1998. *Del objetivo al guión interactivo.* La Habana : GIGA, Vol 1, 1998.

Baumeistier, Hubert, Koch, Nora y Mandel, Luis. 2001. Towars a UML Extension for Hypermedia Design. [En línea] 2001. [Citado el: 18 de diciembre de 2006.] http://citeseer.ist.psu.edu/cache/papers/cs/31048/http:zSzzSzwww.fast.dezSzProjektezSzforsoftzSzuml99zSzuml99.pdf/baumeister99towards.pdf.

Berner, Stefan, Glinz, Martin y Joos, Stefan. 2000. A Clasification of stereotypes for Object - Oriented Modeling Languages. [En línea] 2000. [Citado el: 14 de enero de 2007.] http://citeseer.ist.psu.edu/cache/papers/cs/21217/http:zSzzSzwww.ifi.unizh.chzSzgroupszSzreqzSzstaffzSzglinzzSz..zSz..zSzftpzSzpaperszSzUML99.pdf/berner99classification.pdf.

Berner, Stefan, y otros. 2001. A visualization concept for hierarchical objects models. [En línea] 2001. [Citado el: 14 de mayo de 2007.] http://citeseer.ist.psu.edu/cache/papers/cs/21217/http:zSzzSzwww.ifi.unizh.chzSzgroupszSzreqzSzstaffzSzglinzzSz..zSz..zSzftpzSzpaperszSzASE98.pdf/berner98visualization.pdf.

Booch, Grady, Jacobson, Ivar y Rumbaugh, James. 2000. *El Lenguaje Unificado de Modelado. Manual de Referencia (traducción del original en inglés: The Unified Modeling Language. Reference Manual, 1999).* Madrid : Addison-Wesley, 2000. 1ra edición.

—. 2000. *El Proceso Unificado de Desarrollo de Software (traducido del original en inglés: The Unified Software Development Process, 1999).* Madrid : Addison-Wesley, 2000. 1ra edición.

Botturi, Luca y Switzerland, Lugano. 2005?. Desing models as emergent features: An empirical study in communication and shared mental models in instructional. [En línea] 2005? [Citado el: 3 de junio de 2007.] http://www.cjlt.ca/content/vol32.2/botturi.html.

Cannataro, Mario, y otros. 2002. Modeling Adaptive Hypermedia with and Object - Oriented aproach and XML. [En línea] 2002. [Citado el: 19 de mayo de 2007.] http://citeseer.ist.psu.edu/cache/papers/cs/26938/http:zSzzSzwww.webdyn.orgzSz.zSzproceedingszSzcannataro_adaptive_hypermedia.pdf/cannataro02modeling.pdf.

Carzotto, Franca, Mainetti, Luca y Paolini, Paolo. 1995. Hypermedia Design, analysis and evaluation issues. [En línea] 1995. [Citado el: 28 de junio de 2006.] http://citeseer.ist.psu.edu/cache/papers/cs/15808/http:zSzzSzwww.dsi.unive.itzSz~smmzSzdocszSzgarzotto95.pdf/garzotto98hypermedia.pdf.

Ciudad Ricardo, Febe Angel. 2004. *EMBRIOCIM - Enciclopedia Médica - Colección de Ciencias Morfológicas (tesis en opción al título de Ingeniero en Informática).* La Habana : ISPJAE, 2004.

—. 2006. *Utilización del patrón Modelo - Vista - Controlador (MVC) en el diseño de software educativos.* La Habana : Memorias Uciencia 2006 (CD-ROM), 2006.

Codina, Luis. 1998. H de hipertext, la teoría de los hipertextos revisada. [En línea] 1998. [Citado el: 2 de febrero de 2006.] http://www.ucm.es/info/multidoc/multidoc/revista/cuad6-7/codina.htm.

Corrales Díaz, Carlos. 1994. LA TECNOLOGIA MULTIMEDIA: Una Nueva Tecnología de Comunicación e Información. Características, concepciones y aplicaciones. [En línea] 1994. [Citado el: 10 de octubre de 2006.] http://iteso.mx/~carlosc/pagina/documentos/multidef.htm#inicio.

Davis, A. 1993. *Software requirements: Objects, functions and states.* New York : Prentice Hall, 1993. 1ra edición.

De Bra, Paul, Houben, Geert-Jan y Wu, Hongjing. 1999. AHAM: A Dexter-based Reference Model for Adaptive Hypermedia. [En línea] 1999. [Citado el: 13 de noviembre de 2006.] http://coblitz.codeen.org:3125/citeseer.ist.psu.edu/cache/papers/cs/15284/http:zSzzSzwwwis.win.tue.nlzSz~debrazSzht99zSzht99.pdf/debra99aham.pdf.

DEI, Grupo. 2002?. Diseño y evaluación de sistemas hipermedia. [En línea] 2002? [Citado el: 17 de febrero de 2007.] http://dei.inf.uc3m.es/docencia/p_s_ciclo/dsh/practicas/metodos.pdf.

Diáz-Antón, María Gabriela, y otros. 2002?. Propuesta de una metodología de desarrollo de software educativo bajo un enfoque de calidad sistémica. [En línea] 2002? [Citado el: 14 de octubre de 2005.] http://www.infedu.coord.usb.ve/proyectos/proyecto3.html.

Dolog, Peter. 2004. Model - driven navigation design for semantic web applications with the UML - Guide. [En línea] 2004. [Citado el: 28 de enero de 2007.] http://citeseer.ist.psu.edu/cache/papers/cs2/605/http:zSzzSzwww.l3s.dezSzzCz7EdologzSzpubzSznavigationengforsemanticweb.pdf/model-driven-navigation-design.pdf.

Dolog, Peter y Bieliková, Mária. 2002. Hypermedia Modeling using UML. [En línea] 2002. [Citado el: 11 de enero de 2007.] http://citeseer.ist.psu.edu/cache/papers/cs/27032/http:zSzzSzwww.learninglab.dezSz~dologzSzpubzSzism2002.pdf/dolog02hypermedia.pdf.

Engels, Gregor y Sauer, Stefan. 2000?. UML-based Behavior. Specification of Interactive Multimedia Applications. [En línea] 2000? [Citado el: 06 de abril de 2004.] http://wwwcs.upb.de/cs/ag-engels/Papers/2001/SauerHCC01.pdf.

Engels, Gregor, Sauer, Stefan y Neu, Bettina. 2003. Integrating Software Engineering and User-centred Design for Multimedia Software Developments. [En línea] 2003. [Citado el: 06 de abril de 2004.] http://wwwcs.upb.de/cs/ag-engels/Papers/2003/EngelsSauerNeu-HCC03.pdf.

Fernández, A. 2000. *El formador de Formación Profesional y Ocupacional.* Barcelona : Octaedro, 2000.

Fernández, Carmen y Montes de Oca, Martha. 2006. Aspectos a garantizar en la confección de cursos virtuales. *Revista Cubana de Ciencias informáticas.* MIC-UCI, 2006, Vol. 1, 1.

Fraternal, Piero y Paolini, Paolo. 2000. Model-Driven Development of Web Aplications: The Autoweb System. [En línea] 2000. [Citado el: 19 de enero de 2007.] http://find.galegroup.com/itx/infomark.do?&contentSet=IAC-Documents&type=retrieve&tabID=T002&prodId=CDB&docId=A71186583&source=gale&srcprod=CDB&userGroupName=ucinf&version=1.0.

Hennicker, Rolf y Koch, Nora. 2000. A UML-based Methodology for Hypermedia Design. [En línea] 2000. [Citado el: 06 de abril de 2004.] http://www.pst.informatik.uni-muenchen.de/personen/kochn/Uml2000.pdf.

—. 2000?. Systematic Design of Web Applications with UML. [En línea] 2000? [Citado el: 8 de diciembre de 2006.] http://www.pst.informatik.uni-muenchen.de/personen/kochn/book.pdf.

Isakowitz, Tomás, Kamis, Arnold y Koufaris, Marios. 1997?. Reconciling Top-Down and Bottom-Up Design Approaches in RMM. [En línea] 1997? [Citado el: 19 de octubre de 2006.] http://jmis.bentley.edu/rmm/papers/RWITS97.pdf.

Kruchten, Philippe. 1999?. A Rational Development Process. [En línea] 1999? [Citado el: 06 de abril de 2004.] http://www.rational.com/media/whitepapers/xtalk.pdf.

Larman, Craig. 2000. *UML y patrones. (Traducción del original en inglés: Applying UML and Patterns. An Introduction to Object – Oriented Análisis and Design, 1998).* Madrid : Addison-Wesley, 2000. 2da edición.

Lee, W. W. y Owens, D. L. 2000. *Multimedia based instruction.* Massachussets : Jossey - Bass, 2000.

Lengua, Real Academia Española de la. 2007?. Diccionario Electrónico en línea de la Lengua Española (Vigésimo segunda edición). [En línea] 2007? [Citado el: 12 de marzo de 2007.] http://buscon.rae.es/draeI/.

López Hernández, Yadira Mislay. 2007. *Aplicación de la notación OMMMA - L para el software educativo. (tesis para optar por el título de Ingeniero en Ciencias Informáticas).* La Habana : UCI, 2007.

Lorente Rodríguez, Abel Ernesto. 2006. *Plataformas para el desarrollo y gestión de Cursos Educativos Multimedia.* La Habana, Cuba : CD - ROM Memorias Uciencia, Universidad de las Ciencias Informáticas (UCI), 2006.

Lowe, David y Webby, Richard. 1996?. Hypermedia Process Modelling. [En línea] 1996? [Citado el: 26 de febrero de 2007.] http://citeseer.ist.psu.edu/cache/papers/cs/7291/http:zSzzSzwww.eng.uts.edu.auzSz~dblzSzpaperszSz98ht01.pdf/hypermedia-process-modelling.pdf.

Lyardet, Fernando Daniel, Rossi, Gustavo H. y Schwabe, Daniel. 1998. Using Design Patterns in Educational Multimedia Aplications. [En línea] 1998. [Citado el: 29 de junio de 2006.] http://citeseer.ist.psu.edu/cache/papers/cs/3492/http:zSzzSzwww.inf.puc-rio.brzSz~schwabezSzpaperszSzedmedia98.pdf/lyardet98methodologies.pdf.

Marqués, Pere. 1996?. El software educativo. [En línea] 1996? [Citado el: 13 de abril de 2006.] http://www.lmi.ub.es/te/any96/marques_software/.

Martínez, José Manuel y Hilera, José Ramón. 1998. Modelado de documentación multimedia e hipermedia. [En línea] 1998. [Citado el: 2 de febrero de 2006.] http://www.ucm.es/info/multidoc/multidoc/revista/cuad6-7/artmulti.htm .

MIC, Colectivo Autores. 2003. *Industria Cubana del Software. Perfil y perspectivas.* La Habana : Grupo de Promoción y Publicidad de la InCuSoft, 2003.

OMG. 2003. UML 2.0 OCL Specification. [En línea] 14 de Octubre de 2003. [Citado el: 21 de febrero de 2007.] http://www.omg.org/docs/ptc/03-10-14.pdf.

Pages-Jones, Meilier, y otros. 1990. Modeling object-oriented system: the Uniform Object Notation (a complete and consistent notation for object desing). [En línea] 1990. [Citado el: 21 de febrero de 2007.] http://find.galegroup.com/itx/infomark.do?&contentSet=IAC-Documents&type=retrieve&tabID=T002&prodId=CDB&docId=A9472467&source=gale&srcprod=CDB&userGroupName=ucinf&version=1.0.

Pastor, Juan Antonio. 1997?. La escritura hipermedia. [En línea] 1997? [Citado el: 27 de febrero de 2006.] http://www.ucm.es/info/multidoc/multidoc/revista/cuad6-7/saorin.htm.

Peña Lemus, Yudisleydis y Hernández Díaz, Yuniersy. 2007. *SIMETSE: SIstema de METricas para evaluar el Software Educativo. (tesis para optar por el título de Ingeniero en Ciencias Informáticas).* La Habana : UCI, 2007.

Pérez Fernández, Vicenta. 1999?. *Curso de Informática Educativa (Folleto).* La Habana : IPSJAE, 1999?

Pérez Rodríguez, Yurelixis y Sánchez Torres, Jorquiel. 2007. *La Metodología RMM aplicable al desarrollo de software educativo cubano "Multimedia Topografía". (tesis para optar por el título de Ingeniero en Ciencias Informáticas).* La Habana : UCI, 2007.

Petrinjak, Anita. 2004?. Creating Learning Objects from Pre-Authored Course Materials: Semantic Structure of Learning Objects - Design and Technology. [En línea] 2004? [Citado el: 19 de marzo de 2007.] http://www.cjlt.ca/content/vol30.3/petrinjak.html.

Pigueiras Otero, Danae y otros. 1997. MultiMet: metodología para el desarrollo de aplicaciones que utilizan técnicas de hipermedia. (Tesis para optar por el título de Máster en Informática Aplicada). *CEIS, CUJAE.* 1997.

Piñeiro Gómez, Yordanis y Gallardo Collazo, Jacqueline. 2007. *NEPSE: Principios estratégicos para la guía del proceso de desarrollo de software educativo en la UCI. (tesis para optar por el título de Ingeniero en Ciencias Informáticas).* La Habana : UCI, 2007.

Prado Baldor, Ana Bárbara y Díaz Rodríguez, Daymeris. 2007. *MULTI-CONT: Software Educativo para la asignatura de Contabilidad y Finanzas en la UCI. Módulo de Ejercitación Práctica. (tesis para optar por el título de Ingeniero en Ciencias Informáticas).* La Habana : UCI, 2007.

Pressman, Roger S. 2002. *Ingeniería de Software. Un enfoque práctico (Traducción del original en inglés: Software Engeering. A practical Aproach).* New York : Mac Graw Hill, 2002. 5ta edición.

Reina, A. M. y Torres, J. 2002. Analysing the navigational aspect. [En línea] 2002. [Citado el: 28 de febrero de 2007.] http://citeseer.ist.psu.edu/cache/papers/cs/25868/http:zSzzSzi44w3.info.uni-karlsruhe.dezSz~pulvermuzSzworkshopszSzaosd2002zSzsubmissionszSzquintero.pdf/reina02analysing.pdf.

Rodríguez Figueredo, Erick y Hidalgo Ricardo, Mirelys. 2007. *Una guía de tratamiento de riesgos para el software educativo en la Universidad de las Ciencias Informáticas. (tesis para optar por el título de Ingeniero en Ciencias Informáticas).* La Habana : UCI, 2007.

Rodríguez, Raúl. 2000. Introducción a la Informática Educativa (conferencia). ISPJAE, 2000.

—. 2001. Uso de la multimedia en Delphi (II Parte). GIGA, 2001, 1.

Romero Tena, Rosalía. 2004?. Reflexiones sobre el software educativo. [En línea] 2004? [Citado el: 2 de junio de 2006.] http://tecnologiaedu.us.es/rromero/reflexiones.htm.

Rumbaugh, James. 1997. Models through the development process. *Journal of Object Oriented Programming, SIGS Publications.* Mayo, 1997.

Sauer, Stefan y Engels, Gregor. 2001. Extending UML for Modeling of Multimedia Applications. [En línea] 2001. [Citado el: 06 de abril de 2004.] http://www.itec.uni-klu.ac.at/~harald/proseminar02/sauer1.pdf.

—. 2001. MVC-Based Modeling Support for Embedded REal-Time Systems. [En línea] 2001. [Citado el: 06 de abril de 2004.] http://www.itec.uni-klu.ac.at/~harald/proseminar02/sauer1.pdf.

Schwabe, Daniel y Rossi, Gustavo. 1998?. An Object Oriented Approach to Web-Based Applications design. [En línea] 1998? [Citado el: 6 de marzo de 2007.] http://citeseer.ist.psu.edu/cache/papers/cs/12458/http:zSzzSzwww.inf.puc-rio.brzSz~schwabezSzpaperszSzOOWebAplDesign.pdf/schwabe98object.pdf.

Schwabe, Daniel, Rossi, Gustavo y Barbosa, Simone D. J. 1996?. Systematic Hypermedia Application Design with OOHDM. [En línea] 1996? [Citado el: 3 de marzo de 2007.] http://citeseer.ist.psu.edu/cachedpage/95040/1.

Shaddock, Philip. 1992?. *Multimedia Creations. Hands-on workshop for exploring animation and sound.* Corte Madera, CA. : Waite Group Press, 1992?

Song, Hae-Deok, Koszalka, Tiffany A. y Grabowski, Barbara. 2001?. Exploring Instructional Design Factors Prompting Reflective Thinking in Young Adolescents. [En línea] 2001? [Citado el: 28 de enero de 2007.] http://www.cjlt.ca/content/vol31.2/song.html.

Sottolichio Leighton, Bruno y Farrán Leyva, Yussef. 2002. Una propuesta metodológica para el diseño de interfaces y mapas de navegación en aplicaciones hipermediales. [En línea] 2002. [Citado el: 2 de abril de 2007.] http://www.inf.udec.cl/%7Erevista/ediciones/edicion4/paper_metodologia.PDF.

Taylor, David K. y Hecht, Alan. 1990. Using CASE for object-oriented design with C++. (tutorial). [En línea] 1990. [Citado el: 13 de abril de 2007.] http://find.galegroup.com/itx/infomark.do?&contentSet=IAC-Documents&type=retrieve&tabID=T002&prodId=CDB&docId=A9537537&source=gale&srcprod=CDB&userGroupName=ucinf&version=1.0.

Turk, Dan. 1999. The Unified Modeling Languege User Guide (Review). [En línea] 1999. [Citado el: 17 de enero de 2007.] http://find.galegroup.com/itx/infomark.do?&contentSet=IAC-Documents&type=retrieve&tabID=T002&prodId=CDB&docId=A65805440&source=gale&srcprod=CDB&userGroupName=ucinf&version=1.0.

Ulizama García, José Luis. 1996?. Tecnologías multimedia en el ámbito educativo. [En línea] 1996? [Citado el: 25 de septiembre de 2006.] http://www.sav.us.es/pixelbit/articulos/n10/n10art/art104.htm.

Valdés Pardo, Víctor Giraldo. 2003. *Nuevas tecnologías de la información y la comunicación.* Santa Clara : Feijoo, 2003.

Warren, Keufell. 1991. Transformation strategies. (include a related article on structure chart notation)(strategies for producing a system design that meets user´s requeriments)(tutorial). [En línea] 1991. [Citado el: 11 de noviembre de 2006.] http://find.galegroup.com/itx/infomark.do?&contentSet=IAC-Documents&type=retrieve&tabID=T002&prodId=CDB&docId=A11407113&source=gale&srcprod=CDB&userGroupName=ucinf&version=1.0.

Wasserman, Anthony y Pircher, Peter. 1991. Object-oriented structured design and C++. (the best way to desing complex system)(tutorial). [En línea] 1991. [Citado el: 14 de mayo de 2007.] http://find.galegroup.com/itx/infomark.do?&contentSet=IAC-Documents&type=retrieve&tabID=T002&prodId=CDB&docId=A9826167&source=gale&srcprod=CDB&userGroupName=ucinf&version=1.0.

Waters, Richard C. y Chikofsky, Elliot. 1994. Reverse engineering. (brief summary of some of the papers presented at the May 1993 Association for Computing Machinery/IEEE Computer Society´s Working Conference on Reverse Engineering). [En línea] 1994. [Citado el: 11 de noviembre de 2006.] http://find.galegroup.com/itx/infomark.do?&contentSet=IAC-Documents&type=retrieve&tabID=T002&prodId=CDB&docId=A15352988&source=gale&srcprod=CDB&userGroupName=ucinf&version=1.0.

Bibliografía

BIBLIOGRAFÍA

Aizenbud-Reshef, Netto, y otros. 2006. Model traceability. [En línea] 2006. [Citado el: 30 de octubre de 2006.] http://find.galegroup.com/itx/infomark.do?&contentSet=IAC-Documents&type=retrieve&tabID=T002&prodId=CDB&docId=A9472469&source=gale&srcprod=CDB&userGroupName=ucinf&version=1.0.

Almeida Cintra, Nurileidis y Viera Cisnero, Yoannia. 2007. *Principios para la evaluación y certificación de la calidad de los productos de software educativo en la UCI. (tesis para optar por el título de Ingeniero en Ciencias Informáticas).* La Habana : UCI, 2007.

Autores, Colectivo de. 1998?. Software Educativo. [En línea] 1998? [Citado el: 26 de abril de 2006.] http://cavalletto.freeservers.com/pag%20educativa%20de%20amortiza.htm.

Balasubramanian, Venkatraman, Bieber, Michael y Isakowitz, Tomás. 2001. A case study in systematic hypermedia design. [En línea] 2001. [Citado el: 13 de diciembre de 2006.] http://citeseer.ist.psu.edu/cache/papers/cs/29502/http:zSzzSzweb.njit.eduzSz~bieberzSzpubzSzisj01.pdf/balasubramanian01case.pdf.

Barrera Yanes, Rafael. 1998. *Del objetivo al guión interactivo.* La Habana : GIGA, Vol 1, 1998.

Baumeistier, Hubert, Koch, Nora y Mandel, Luis. 2001. Towars a UML Extension for Hypermedia Design. [En línea] 2001. [Citado el: 18 de diciembre de 2006.] http://citeseer.ist.psu.edu/cache/papers/cs/31048/http:zSzzSzwww.fast.dezSzProjektezSzforsoftzSzuml99zSzuml99.pdf/baumeister99towards.pdf.

Berner, Stefan, Glinz, Martin y Joos, Stefan. 2000. A Clasification of stereotypes for Object - Oriented Modeling Languages. [En línea] 2000. [Citado el: 14 de enero de 2007.] http://citeseer.ist.psu.edu/cache/papers/cs/21217/http:zSzzSzwww.ifi.unizh.chzSzgroupszSzreqzSzstaffzSzglinzzSz..zSz..zSzftpzSzpaperszSzUML99.pdf/berner99classification.pdf.

Berner, Stefan, y otros. 2001. A visualization concept for hierarchical objects models. [En línea] 2001. [Citado el: 14 de mayo de 2007.] http://citeseer.ist.psu.edu/cache/papers/cs/21217/http:zSzzSzwww.ifi.unizh.chzSzgroupszSzreqzSzstaffzSzglinzzSz..zSz..zSzftpzSzpaperszSzASE98.pdf/berner98visualization.pdf.

Booch, Grady, Jacobson, Ivar y Rumbaugh, James. 2000. *El Lenguaje Unificado de Modelado. Manual de Referencia (traducción del original en inglés: The Unified Modeling Language. Reference Manual, 1999).* Madrid : Addison-Wesley, 2000. 1ra edición.

—. 2000. *El Proceso Unificado de Desarrollo de Software (traducido del original en inglés: The Unified Software Development Process, 1999).* Madrid : Addison-Wesley, 2000. 1ra edición.

Botturi, Luca y Switzerland, Lugano. 2005?. Desing models as emergent features: An empirical study in communication and shared mental models in instructional. [En línea] 2005? [Citado el: 3 de junio de 2007.] http://www.cjlt.ca/content/vol32.2/botturi.html.

Cannataro, Mario, y otros. 2002. Modeling Adaptive Hypermedia with and Object - Oriented aproach and XML. [En línea] 2002. [Citado el: 19 de mayo de 2007.]

http://citeseer.ist.psu.edu/cache/papers/cs/26938/http:zSzzSzwww.webdyn.orgzSz.zSzproceedingszSzcannataro_adaptive_hypermedia.pdf/cannataro02modeling.pdf.

Carzotto, Franca, Mainetti, Luca y Paolini, Paolo. 1995. Hypermedia Design, analysis and evaluation issues. [En línea] 1995. [Citado el: 28 de junio de 2006.] http://citeseer.ist.psu.edu/cache/papers/cs/15808/http:zSzzSzwww.dsi.unive.itzSz~smmzSzdocszSzgarzotto95.pdf/garzotto98hypermedia.pdf.

Ciudad Ricardo, Febe Angel. 2004. *EMBRIOCIM - Enciclopedia Médica - Colección de Ciencias Morfológicas (tesis en opción al título de Ingeniero en Informática).* La Habana : ISPJAE, 2004.

—. 2006. *Utilización del patrón Modelo - Vista - Controlador (MVC) en el diseño de software educativos.* La Habana : Memorias Uciencia 2006 (CD-ROM), 2006.

Codina, Luis. 1998. H de hipertext, la teoría de los hipertextos revisada. [En línea] 1998. [Citado el: 2 de febrero de 2006.] http://www.ucm.es/info/multidoc/multidoc/revista/cuad6-7/codina.htm.

Corrales Díaz, Carlos. 1994. LA TECNOLOGIA MULTIMEDIA: Una Nueva Tecnología de Comunicación e Información. Características, concepciones y aplicaciones. [En línea] 1994. [Citado el: 10 de octubre de 2006.] http://iteso.mx/~carlosc/pagina/documentos/multidef.htm#inicio.

Davis, A. 1993. *Software requirements: Objects, functions and states.* New York : Prentice Hall, 1993. 1ra edición.

De Bra, Paul, Houben, Geert-Jan y Wu, Hongjing. 1999. AHAM: A Dexter-based Reference Model for Adaptive Hypermedia. [En línea] 1999. [Citado el: 13 de noviembre de 2006.] http://coblitz.codeen.org:3125/citeseer.ist.psu.edu/cache/papers/cs/15284/http:zSzzSzwwwis.win.tue.nlzSz~debrazSzht99zSzht99.pdf/debra99aham.pdf.

DEI, Grupo. 2002?. Diseño y evaluación de sistemas hipermedia. [En línea] 2002? [Citado el: 17 de febrero de 2007.] http://dei.inf.uc3m.es/docencia/p_s_ciclo/dsh/practicas/metodos.pdf.

Diáz-Antón, María Gabriela, y otros. 2002?. Propuesta de una metodología de desarrollo de software educativo bajo un enfoque de calidad sistémica. [En línea] 2002? [Citado el: 14 de octubre de 2005.] http://www.infedu.coord.usb.ve/proyectos/proyecto3.html.

Dolog, Peter. 2004. Model - driven navigation design for semantic web applications with the UML - Guide. [En línea] 2004. [Citado el: 28 de enero de 2007.] http://citeseer.ist.psu.edu/cache/papers/cs2/605/http:zSzzSzwww.l3s.dezSzzCz7EdologzSzpubzSznavigationengforsemanticweb.pdf/model-driven-navigation-design.pdf.

Dolog, Peter y Bieliková, Mária. 2002. Hypermedia Modeling using UML. [En línea] 2002. [Citado el: 11 de enero de 2007.] http://citeseer.ist.psu.edu/cache/papers/cs/27032/http:zSzzSzwww.learninglab.dezSz~dologzSzpubzSzism2002.pdf/dolog02hypermedia.pdf.

Engels, Gregor y Sauer, Stefan. 2000?. UML-based Behavior. Specification of Interactive Multimedia Applications. [En línea] 2000? [Citado el: 06 de abril de 2004.] http://wwwcs.upb.de/cs/ag-engels/Papers/2001/SauerHCC01.pdf.

Engels, Gregor, Sauer, Stefan y Neu, Bettina. 2003. Integrating Software Engineering and User-centred Design for Multimedia Software Developments. [En línea] 2003. [Citado el: 06 de abril de 2004.] http://wwwcs.upb.de/cs/ag-engels/Papers/2003/EngelsSauerNeu-HCC03.pdf.

Fernández, A. 2000. *El formador de Formación Profesional y Ocupacional.* Barcelona : Octaedro, 2000.

Fernández, Carmen y Montes de Oca, Martha. 2006. Aspectos a garantizar en la confección de cursos virtuales. *Revista Cubana de Ciencias informáticas.* MIC-UCI, 2006, Vol. 1, 1.

Fraternal, Piero y Paolini, Paolo. 2000. Model-Driven Development of Web Aplications: The Autoweb System. [En línea] 2000. [Citado el: 19 de enero de 2007.] http://find.galegroup.com/itx/infomark.do?&contentSet=IAC-Documents&type=retrieve&tabID=T002&prodId=CDB&docId=A71186583&source=gale&srcprod=CDB&userGroupName=ucinf&version=1.0.

Hennicker, Rolf y Koch, Nora. 2000. A UML-based Methodology for Hypermedia Design. [En línea] 2000. [Citado el: 06 de abril de 2004.] http://www.pst.informatik.uni-muenchen.de/personen/kochn/Uml2000.pdf.

—. 2000?. Systematic Design of Web Applications with UML. [En línea] 2000? [Citado el: 8 de diciembre de 2006.] http://www.pst.informatik.uni-muenchen.de/personen/kochn/book.pdf.

Isakowitz, Tomás, Kamis, Arnold y Koufaris, Marios. 1997?. Reconciling Top-Down and Bottom-Up Design Approaches in RMM. [En línea] 1997? [Citado el: 19 de octubre de 2006.] http://jmis.bentley.edu/rmm/papers/RWITS97.pdf.

Kruchten, Philippe. 1999?. A Rational Development Process. [En línea] 1999? [Citado el: 06 de abril de 2004.] http://www.rational.com/media/whitepapers/xtalk.pdf.

Larman, Craig. 2000. *UML y patrones. (Traducción del original en inglés: Applying UML and Patterns. An Introduction to Object – Oriented Análisis and Design, 1998).* Madrid : Addison-Wesley, 2000. 2da edición.

Lee, W. W. y Owens, D. L. 2000. *Multimedia based instruction.* Massachussets : Jossey - Bass, 2000.

Lengua, Real Academia Española de la. 2007?. Diccionario Electrónico en línea de la Lengua Española (Vigésimo segunda edición). [En línea] 2007? [Citado el: 12 de marzo de 2007.] http://buscon.rae.es/draeI/.

López Hernández, Yadira Mislay. 2007. *Aplicación de la notación OMMMA - L para el software educativo. (tesis para optar por el título de Ingeniero en Ciencias Informáticas).* La Habana : UCI, 2007.

Lorente Rodríguez, Abel Ernesto. 2006. *Plataformas para el desarrollo y gestión de Cursos Educativos Multimedia.* La Habana, Cuba : CD - ROM Memorias Uciencia, Universidad de las Ciencias Informáticas (UCI), 2006.

Lowe, David y Webby, Richard. 1996?. Hypermedia Process Modelling. [En línea] 1996? [Citado el: 26 de febrero de 2007.] http://citeseer.ist.psu.edu/cache/papers/cs/7291/http:zSzzSzwww.eng.uts.edu.auzSz~dblzSzpaperszSz98ht01.pdf/hypermedia-process-modelling.pdf.

Lyardet, Fernando Daniel, Rossi, Gustavo H. y Schwabe, Daniel. 1998. Using Design Patterns in Educational Multimedia Aplications. [En línea] 1998. [Citado el: 29 de junio de 2006.] http://citeseer.ist.psu.edu/cache/papers/cs/3492/http:zSzzSzwww.inf.puc-rio.brzSz~schwabezSzpaperszSzedmedia98.pdf/lyardet98methodologies.pdf.

Marqués, Pere. 1996?. El software educativo. [En línea] 1996? [Citado el: 13 de abril de 2006.] http://www.lmi.ub.es/te/any96/marques_software/.

Martínez, José Manuel y Hilera, José Ramón. 1998. Modelado de documentación multimedia e hipermedia. [En línea] 1998. [Citado el: 2 de febrero de 2006.] http://www.ucm.es/info/multidoc/multidoc/revista/cuad6-7/artmulti.htm .

MIC, Colectivo Autores. 2003. *Industria Cubana del Software. Perfil y perspectivas.* La Habana : Grupo de Promoción y Publicidad de la InCuSoft, 2003.

OMG. 2003. UML 2.0 OCL Specification. [En línea] 14 de Octubre de 2003. [Citado el: 21 de febrero de 2007.] http://www.omg.org/docs/ptc/03-10-14.pdf.

Pages-Jones, Meilier, y otros. 1990. Modeling object-oriented system: the Uniform Object Notation (a complete and consistent notation for object desing). [En línea] 1990. [Citado el: 21 de febrero de 2007.] http://find.galegroup.com/itx/infomark.do?&contentSet=IAC-Documents&type=retrieve&tabID=T002&prodId=CDB&docId=A9472467&source=gale&srcprod=CDB&userGroupName=ucinf&version=1.0.

Pastor, Juan Antonio. 1997?. La escritura hipermedia. [En línea] 1997? [Citado el: 27 de febrero de 2006.] http://www.ucm.es/info/multidoc/multidoc/revista/cuad6-7/saorin.htm.

Peña Lemus, Yudisleydis y Hernández Díaz, Yuniersy. 2007. *SIMETSE: SIstema de METricas para evaluar el Software Educativo. (tesis para optar por el título de Ingeniero en Ciencias Informáticas).* La Habana : UCI, 2007.

Pérez Fernández, Vicenta. 1999?. *Curso de Informática Educativa (Folleto).* La Habana : IPSJAE, 1999?

Pérez Rodríguez, Yurelixis y Sánchez Torres, Jorquiel. 2007. *La Metodología RMM aplicable al desarrollo de software educativo cubano "Multimedia Topografía". (tesis para optar por el título de Ingeniero en Ciencias Informáticas).* La Habana : UCI, 2007.

Petrinjak, Anita. 2004?. Creating Learning Objects from Pre-Authored Course Materials: Semantic Structure of Learning Objects - Design and Technology. [En línea] 2004? [Citado el: 19 de marzo de 2007.] http://www.cjlt.ca/content/vol30.3/petrinjak.html.

Pigueiras Otero, Danae y otros. 1997. MultiMet: metodología para el desarrollo de aplicaciones que utilizan técnicas de hipermedia. (Tesis para optar por el título de Máster en Informática Aplicada). *CEIS, CUJAE.* 1997.

Piñeiro Gómez, Yordanis y Gallardo Collazo, Jacqueline. 2007. *NEPSE: Principios estratégicos para la guía del proceso de desarrollo de software educativo en la UCI. (tesis para optar por el título de Ingeniero en Ciencias Informáticas).* La Habana : UCI, 2007.

Prado Baldor, Ana Bárbara y Díaz Rodríguez, Daymeris. 2007. *MULTI-CONT: Software Educativo para la asignatura de Contabilidad y Finanzas en la UCI. Módulo de Ejercitación Práctica. (tesis para optar por el título de Ingeniero en Ciencias Informáticas).* La Habana : UCI, 2007.

Pressman, Roger S. 2002. *Ingeniería de Software. Un enfoque práctico (Traducción del original en inglés: Software Engeering. A practical Aproach).* New York : Mac Graw Hill, 2002. 5ta edición.

Reina, A. M. y Torres, J. 2002. Analysing the navigational aspect. [En línea] 2002. [Citado el: 28 de febrero de 2007.] http://citeseer.ist.psu.edu/cache/papers/cs/25868/http:zSzzSzi44w3.info.uni-

karlsruhe.dezSz~pulvermuzSzworkshopszSzaosd2002zSzsubmissionszSzquintero.pdf/reina02analysing.pdf.

Rodríguez Figueredo, Erick y Hidalgo Ricardo, Mirelys. 2007. *Una guía de tratamiento de riesgos para el software educativo en la Universidad de las Ciencias Informáticas. (tesis para optar por el título de Ingeniero en Ciencias Informáticas).* La Habana : UCI, 2007.

Rodríguez, Raúl. 2000. Introducción a la Informática Educativa (conferencia). ISPJAE, 2000.

—. 2001. Uso de la multimedia en Delphi (II Parte). GIGA, 2001, 1.

Romero Tena, Rosalía. 2004?. Reflexiones sobre el software educativo. [En línea] 2004? [Citado el: 2 de junio de 2006.] http://tecnologiaedu.us.es/rromero/reflexiones.htm.

Rumbaugh, James. 1997. Models through the development process. *Journal of Object Oriented Programming, SIGS Publications.* Mayo, 1997.

Sauer, Stefan y Engels, Gregor. 2001. Extending UML for Modeling of Multimedia Applications. [En línea] 2001. [Citado el: 06 de abril de 2004.] http://www.itec.uni-klu.ac.at/~harald/proseminar02/sauer1.pdf.

—. 2001. MVC-Based Modeling Support for Embedded REal-Time Systems. [En línea] 2001. [Citado el: 06 de abril de 2004.] http://www.itec.uni-klu.ac.at/~harald/proseminar02/sauer1.pdf.

Schwabe, Daniel y Rossi, Gustavo. 1998?. An Object Oriented Approach to Web-Based Applications design. [En línea] 1998? [Citado el: 6 de marzo de 2007.] http://citeseer.ist.psu.edu/cache/papers/cs/12458/http:zSzzSzwww.inf.puc-rio.brzSz~schwabezSzpaperszSzOOWebAplDesign.pdf/schwabe98object.pdf.

Schwabe, Daniel, Rossi, Gustavo y Barbosa, Simone D. J. 1996?. Systematic Hypermedia Application Design with OOHDM. [En línea] 1996? [Citado el: 3 de marzo de 2007.] http://citeseer.ist.psu.edu/cachedpage/95040/1.

Shaddock, Philip. 1992?. *Multimedia Creations. Hands-on workshop for exploring animation and sound.* Corte Madera, CA. : Waite Group Press, 1992?

Song, Hae-Deok, Koszalka, Tiffany A. y Grabowski, Barbara. 2001?. Exploring Instructional Design Factors Prompting Reflective Thinking in Young Adolescents. [En línea] 2001? [Citado el: 28 de enero de 2007.] http://www.cjlt.ca/content/vol31.2/song.html.

Sottolichio Leighton, Bruno y Farrán Leyva, Yussef. 2002. Una propuesta metodológica para el diseño de interfaces y mapas de navegación en aplicaciones hipermediales. [En línea] 2002. [Citado el: 2 de abril de 2007.] http://www.inf.udec.cl/%7Erevista/ediciones/edicion4/paper_metodologia.PDF.

Taylor, David K. y Hecht, Alan. 1990. Using CASE for object-oriented design with C++. (tutorial). [En línea] 1990. [Citado el: 13 de abril de 2007.] http://find.galegroup.com/itx/infomark.do?&contentSet=IAC-Documents&type=retrieve&tabID=T002&prodId=CDB&docId=A9537537&source=gale&srcprod=CDB&userGroupName=ucinf&version=1.0.

Turk, Dan. 1999. The Unified Modeling Languege User Guide (Review). [En línea] 1999. [Citado el: 17 de enero de 2007.] http://find.galegroup.com/itx/infomark.do?&contentSet=IAC-Documents&type=retrieve&tabID=T002&prodId=CDB&docId=A65805440&source=gale&srcprod=CDB&userGroupName=ucinf&version=1.0.

Ulizama García, José Luis. 1996?. Tecnologías multimedia en el ámbito educativo. [En línea] 1996? [Citado el: 25 de septiembre de 2006.] http://www.sav.us.es/pixelbit/articulos/n10/n10art/art104.htm.

Valdés Pardo, Víctor Giraldo. 2003. *Nuevas tecnologías de la información y la comunicación.* Santa Clara : Feijoo, 2003.

Warren, Keufell. 1991. Transformation strategies. (include a related article on structure chart notation)(strategies for producing a system design that meets user´s requeriments)(tutorial). [En línea] 1991. [Citado el: 11 de noviembre de 2006.] http://find.galegroup.com/itx/infomark.do?&contentSet=IAC-Documents&type=retrieve&tabID=T002&prodId=CDB&docId=A11407113&source=gale&srcprod=CDB&userGroupName=ucinf&version=1.0.

Wasserman, Anthony y Pircher, Peter. 1991. Object-oriented structured design and C++. (the best way to desing complex system)(tutorial). [En línea] 1991. [Citado el: 14 de mayo de 2007.] http://find.galegroup.com/itx/infomark.do?&contentSet=IAC-Documents&type=retrieve&tabID=T002&prodId=CDB&docId=A9826167&source=gale&srcprod=CDB&userGroupName=ucinf&version=1.0.

Waters, Richard C. y Chikofsky, Elliot. 1994. Reverse engineering. (brief summary of some of the papers presented at the May 1993 Association for Computing Machinery/IEEE Computer Society´s Working Conference on Reverse Engineering). [En línea] 1994. [Citado el: 11 de noviembre de 2006.] http://find.galegroup.com/itx/infomark.do?&contentSet=IAC-Documents&type=retrieve&tabID=T002&prodId=CDB&docId=A15352988&source=gale&srcprod=CDB&userGroupName=ucinf&version=1.0.

Anexos

ANEXOS

ANEXO 1: Entrevista realizada a la Dtra. de la Dirección de Producción No. 2 (Dirección de Producción de Software Educativo) de la Infraestructura Productiva de la UCI, M.Sc. Yadenis Piñero Pérez.

- **¿Cuáles son las notaciones que se utilizan hoy para el modelado de las aplicaciones multimedia en la UCI?**

  No hemos implantado una notación en particular o no hemos utilizado ninguna notación en particular para la modelación de las aplicaciones, sino que hemos dado la indicación de que aquellos proyectos que lo consideren necesario utilicen la extensión de UML, OMMMA – L. No utilizamos ninguna notación en particular dado que nuestros clientes no son en su mayoría especialistas en el tema de la Ingeniería de Software (ISW), lo que hace que no se interesen por temas de este tipo, además de que sus intereses principales están dados hacia la escritura de las pantallas y no del resto de los elementos funcionales, además de que en muchas ocasiones son productos tan sencillos que no se hace necesario un modelación exhaustiva. En esto interviene que no existe un rol dentro del equipo de desarrollo que se dedique a esta tarea. No entiendo ¿porqué no existe?, ¿por qué no vemos el desarrollo de SW educativo como cualquier otro tipo de desarrollo software aunque con sus particularidades? Porque si hubiese alguien con ese rol, al menos quedaría documentada la aplicación.

- **¿Qué artefactos se generan o utilizan hoy en la producción?**

  Hoy utilizamos esencialmente dos artefactos: un árbol (mapa) de navegación que exigimos tenga toda aplicación que se produce, y de alguna manera todos los proyectos trabajan los guiones, sean de contenido o técnicos, siendo los primeros aquellos que solo detallan elementos pedagógicos o relacionados con la presentación del contenido del software y los últimos aquellos que tienden a especificar con más nivel de detalle elementos de la programación propiamente dicha. Estos guiones se trata de que sean aprobados por todas las partes conformantes del equipo de desarrollo, aunque en la práctica todas las partes no lo hagan.

- **¿Logran los artefactos utilizados representar todos los elementos del Software Educativo producido en la UCI?**

Los artefactos que utilizamos hoy día para el desarrollo de nuestros productos no logran representar las características del Software Educativo que producimos. Ejemplo de ello, es el último trabajo que realizamos con Sys-Copextel, en el cual supuestamente un software muy sencillo con solo 3 pantallas de presentación de contenidos, se podría producir sin documentación alguna. Exigimos la realización del árbol de navegación y de un pequeño guión y quedaron demasiados elementos sin definirse, lo que produjo pérdida de tiempo y errores en el desarrollo del producto final. Esto influye en elementos de garantía de la calidad, pues al no poder contar con documentaciones del software nos imposibilita un chequeo más detallado de nuestras aplicaciones.

- **¿Cuáles son los elementos distintivos del Software Educativo que se produce en la universidad?**

  No creo que la diferencia esté en el tipo de software que desarrollamos, pues este es muy variante de acuerdo al cliente y a las características específicas de cada uno, lo que hace muy difícil establecer patrones de solicitudes de los clientes y por consiguiente de las aplicaciones. La diferencia fundamental con otras instituciones, universidades o casas desarrolladoras, está no en el tipo de software sino en que necesitamos y caminamos hacia una producción *industrial*, lo que implica aplicar a dicha producción elementos ingenieriles por completos, dentro de los que se encuentra la documentación de las aplicaciones y por ende del uso de una notación y de artefactos que logren representar ese proceso y ayuden a que realmente sea industrial.

- **¿Cuáles de estos elementos de los software educativos desarrollados no son representados por los artefactos que se utilizan actualmente?**

  Lo que normalmente sucede en nuestros productos es que no quedan representados, al menos explícitamente, los servicios generales en los módulos, o lo que es lo mismo, las opciones comunes a un conjunto de interfaces del software que no es más que la base del modelo pedagógico que se sigue en la producción. Esta falta nos dificulta luego una reutilización de los códigos desarrollados, pues por ejemplo a lo mejor un cliente nos pide en un nuevo producto una búsqueda en varias de sus pantallas, si tuviésemos una documentación, podríamos buscar rutinas de búsqueda que cumplan los requisitos y nos permitan reutilizar el código anterior. Esto sucede porque la tendencia al documentar software educativo es a como

describir un libro por pantallas y además considero que el desarrollo de la ISW no ha logrado todavía la documentación de este tipo de aplicaciones.

- **¿Qué tipo de paradigma de programación se utiliza hoy para la implementación de las aplicaciones educativas en la UCI?**

  Mayoritariamente se trabaja con una programación Orientada a Eventos, pero que se acerca bastante al paradigma Orientado a Objetos. La utilización de programación estructurada es muy poca en la actualidad de nuestras soluciones. Todo esto condicionado principalmente por el tipo de tecnologías y herramientas que utilizamos en nuestras soluciones.

- **¿Qué lenguajes de programación se utilizan para la implementación?**

  Para la implementación utilizamos en más del 80 % de nuestras producciones el ActioScript, también condicionado por el tipo de tecnologías que utilizamos en la actualidad.

- **¿Qué Sistemas de Autor o editores se utilizan para el desarrollo de las aplicaciones?**

  Al igual que sucede con el lenguaje de programación, en la mayoría de nuestros productos, en realidad en casi todos, utilizamos el ambiente de desarrollo de Macromedia Flash, con todos sus componentes.

- **¿Se utiliza algún tipo de arquitectura en particular para el desarrollo de los Software Educativos en la UCI?**

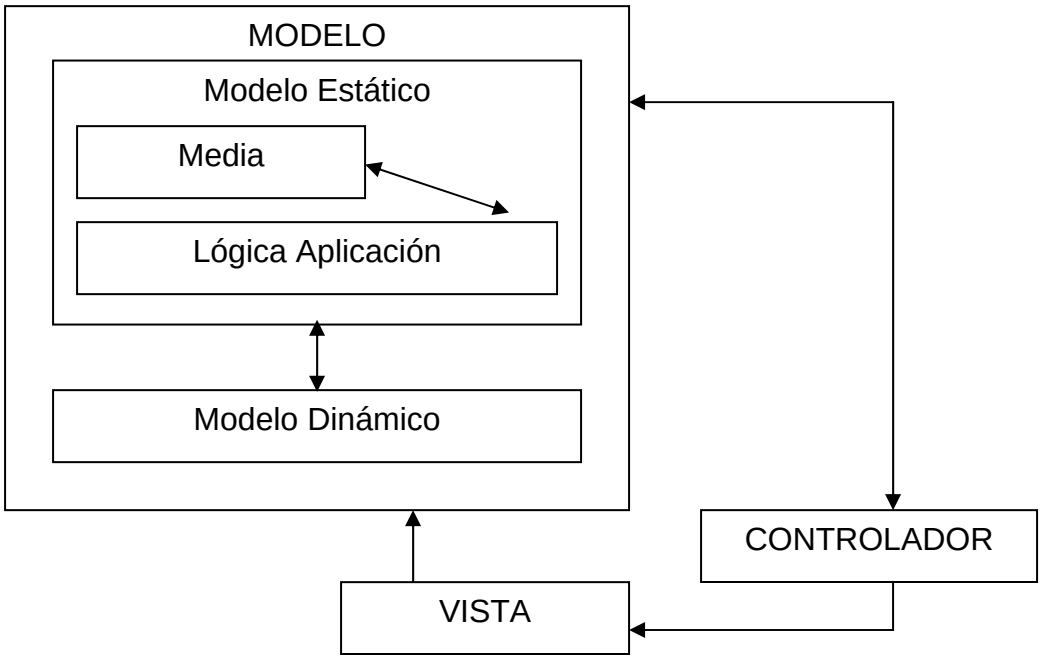
  Siempre se ha tratado de que en todos los productos se haga algún tipo de análisis básico de arquitectura. Ejemplos de esto tenemos a los productos del MINED (Ministerio de Educación), el proyecto MSD – Brasil, los cuales tuvieron Frameworks muy básicos y trabajos con plantillas y librerías de Scripts, pero al menos se hicieron análisis de arquitectura. La producción actual se mueve a eso y exigimos como parte del flujo del proceso un análisis de una arquitectura. La tendencia actual de las solicitudes que recibimos es el trabajo en colecciones completas, lo que obliga a pensar en términos de arquitectura y ya tenemos probado que cuando no lo hemos hecho de esa manera, las cosas nos salen mal.

- **¿Qué artefactos o elementos de estos deben de mantenerse en futuras soluciones?**

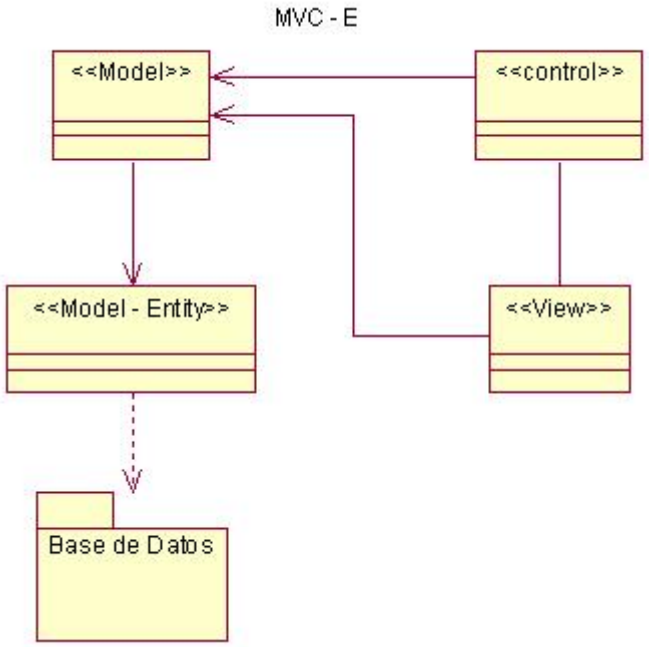
  El uso del guión, sea en cualquiera de sus versiones, debe de mantenerse para que sirva como el artefacto donde los especialistas en la parte del contenido o pedagógica del producto reflejen sus solicitudes del producto final. Esto no entra en contradicción con que se hace necesario convertir este guió a requisitos funcionales y no funcionales, tal y como se trabaje en la producción de los software tradicionales o de gestión. ¿por qué no podemos trabajar con

requisitos en vez de solicitudes o deseos de nuestros clientes que no han sido escritos como capacidades del futuro sistema a desarrollar? Soy partidaria de la utilización de los Casos de Uso, para describir nuestros escenarios de desarrollo, ¿qué es un caso de uso sino la descripción de un escenario del un guión? Claro mucho más elaborado. Creo que adicionando algunos elementos (medias, palabras calientes, etc.) que contextualicen las descripciones de los casos de uso actuales, podríamos utilizarlos sin problema alguno, misión o responsabilidad que pudiera ser asumida por el documentador o un analista.

ANEXO 2: Una extensión del patrón original Modelo – Vista – Controlador (MVC) para aplicaciones multimedia (MVCMM) [tomado de (Sauer, y otros, 2001)]



ANEXO 3: Variación del MVCMM para la arquitectura del software educativo cubano. [tomado de (Ciudad Ricardo, 2006)]



ANEXO 4: Modificaciones a la plantilla de la descripción textual de los Casos de Uso propuesta por UML en su formato estándar para la descripción del software educativo.

<table>
<tr><td colspan="5">Descripción Textual del Caso de Uso <Nombre del caso de uso></td></tr>
<tr><td>Actores del caso de uso</td><td colspan="4"><se listan los actores que intervienen en el caso de uso y se señala entre paréntesis el actor que inicia el caso de uso></td></tr>
<tr><td>Propósito</td><td colspan="4"><se menciona de forma breve el propósito del caso de uso para dejar claridad de la función del mismo y del beneficio que tendrá para el actor del sistema que lo inicia></td></tr>
<tr><td>Resumen</td><td colspan="4"><se redacta de forma breve el resumen del caso de uso, dejando claridad de la actividad que da inicio al mismo así como de lo más importante del flujo y de las actividades que concluyen el msimo></td></tr>
<tr><td>Casos de uso asociados</td><td colspan="4"><se listan los casos de uso relacionados con este a través de extensiones, inclusiones o generalizaciones></td></tr>
<tr><td>Referencias</td><td colspan="4"><se establecen los requerimientos funcionales y no funcionales a los cuales satisface el caso de uso></td></tr>
<tr><td>Precondiciones</td><td colspan="4"><se numeran las precondiciones que deben de cumplirse para que el caso de uso se ejecute></td></tr>
<tr><td>Poscondiciones</td><td colspan="4"><se numeran las poscondiciones que existirán luego que el caso de uso culmine su ejecución></td></tr>
<tr><td colspan="5">Curso Normal de los Eventos</td></tr>
<tr><td colspan="2">Acciones del Actor</td><td colspan="3">Respuesta del Sistema</td></tr>
<tr><td colspan="2"><se numeran las acciones del actor que generan respuestas por el sistema, en orden de ocurrencia></td><td colspan="3"><se numeran en correspondencia con las acciones del actor, las acciones que dan respuesta a las acciones del actor, para de esta forma completar transacciones del flujo de sucesos en el caso de uso></td></tr>
<tr><td colspan="5">Cursos Alternos de los Eventos</td></tr>
<tr><td>Acción</td><td colspan="4">Curso Alterno</td></tr>
<tr><td><se refleja la acción del sistema que tiene una acción alternativa></td><td colspan="4"><se expresa que ocurre en términos del acciones del sistema como respuesta al flujo alterno que tomaría el sistema ante la situación determinada></td></tr>
<tr><td>Prioridad</td><td colspan="4"><se expresa la prioridad de implementación del caso de uso, siendo: crítica, secundaria u opcional></td></tr>
<tr><td>Mejoras</td><td colspan="4"><se numeran las mejoras que implementa el caso de uso sobre el curso normal de los eventos descritos></td></tr>
<tr><td rowspan="9">Medias a utilizar</td><td>Tipo de Media</td><td colspan="2">Descripción</td><td>Estado</td></tr>
<tr><td rowspan="2">Imagen</td><td colspan="2" rowspan="2"><se listan las medias o su descripción de estas no existir que den claridad de las características o condiciones que tienen que cumplir las mismas para ser utilizadas></td><td rowspan="2"><se refleja el estado de la media, sea existente, en localización, en construcción></td></tr>
<tr></tr>
<tr><td rowspan="2">Video o Animación</td><td colspan="2"></td><td></td></tr>
<tr><td colspan="2"></td><td></td></tr>
<tr><td rowspan="2">Sonido</td><td colspan="2"></td><td></td></tr>
<tr><td colspan="2"></td><td></td></tr>
<tr><td rowspan="2">Texto</td><td colspan="2"></td><td></td></tr>
<tr><td colspan="2"></td><td></td></tr>
<tr><td>Elementos pedagógicos</td><td colspan="4"><se reflejan los elementos que no pueden ser expresados en el cuerpo del caso de uso (reglas pedagógicas o instructivas) a tomar en consideración para la implementación del caso de uso></td></tr>
</table>

ANEXO 5: Encuesta realizada a especialistas y estudiantes miembros de los equipos de desarrollo de software educativo en la UCI.

ENCUESTA

Cro.(a): Actualmente existe una investigación sobre las notaciones y lenguajes de modelación para el desarrollo de Software Educativo (SWE). La encuesta que se le presenta a continuación es sencilla y agradeceríamos fidelidad en la información para que sirva eficientemente al cumplimiento del objetivo de esta investigación.

Muchas Gracias,
Colectivo de Autores.

Nombre:_____ Proyecto:_____
Rol: _____ Estudiante:____ Especialista:_____

SECCIÓN I – Notaciones y lenguajes

1. Marque con una X las notaciones o lenguajes que conoce para la modelación de SWE:
   _____ UML _____ WSDL
   _____ RMM _____ MultiMet
   _____ OOHDM _____ OMMMA – L
   _____ Otro ¿Cuál?_____
2. Marque con una X la notación o lenguaje que se utiliza en su proyecto para la modelación del producto:
   _____ UML _____ WSDL
   _____ RMM _____ MultiMet
   _____ OOHDM _____ OMMMA – L
   _____ Ninguna _____ Otro ¿Cuál?_____

SECCIÓN II – Guiones y Mapas de nevagación

3. ¿Se utilizan en su proyecto Guiones para la documentación del producto?
   Si ____ No ____
4. ¿Se trabajan Mapas de Navegación de la aplicación para conocer los posibles caminos de ejecución?
   Si ____ No ____
5. ¿Considera usted que con los Guiones y los Mapas de Navegación es suficiente para modelar y documentar las aplicaciones educativas?
   _____ Totalmente _____ Parcialmente
   _____ Pobremente _____ No suficiente
6. Marque con una X los elementos que a su consideración no son abordados por los guiones o los mapas de navegación utilizados:
   _____ Clases _____ Tipos de clases o entidades
   _____ Eventos _____ Estructuras de las interfaces.
   _____ Entidades _____ Arquitectura del SWE.
   _____ Relaciones de navegación
   _____ Relaciones e/ clases o entidades
7. ¿Considera ventajoso modelar y documentar ingenierilmente los SWE?
   Si _____ No_____

SECCIÓN III – Técnicas de programación y Arquitectura

8. ¿Qué técnicas de programación considera usted deben utilizarse en la programación del SWE? Marque con una X
   _____ Estructurada.
   _____ Orientada a Eventos.
   _____ Orientada a Objetos.
   _____ Mezcla ¿Cual? _____
9. ¿Qué patrones arquitectónicos se utilizan en su proyecto para la solución propuesta?
   _____ Capas (Layer)
   _____ MVC (Modelo-Vista-Controlador)
   _____ MVC-E (Modelo-Vista-Controlador-Entidad)
   _____ Otro ¿Cuál?_____

SECCIÓN IV – Vistas y estereotipos (Completar sólo si su proyecto usa UML/OMMMA-L)

10. ¿Considera que los estereotipos visuales brindados por UML/OMMMA–L son cómodos y fáciles de usar?
    _____ Totalmente
    _____ Parcialmente
    _____ Nulo
11. ¿Cuáles vistas cree usted deben de reforzarse al modelar un SWE?
    _____ Lógica _____ Implementación
    _____ Interacción _____ Presentación
    _____ Casos de Uso _____ Otra ¿Cual?_____
12. ¿Considera eficiente los casos de uso para la descripción de los flujos de la solución?
    _____ Totalmente
    _____ Parcialmente
    _____ Nulo

ANEXO 6: Encuesta realizada a los miembros del equipo de desarrollo del proyecto productivo A Jugar luego de la utilización de ApEM – L.

ENCUESTA

Cro.(a): Actualmente existe una investigación sobre las notaciones y lenguajes de modelación para el desarrollo de Software Educativo (SWE). La encuesta que se le presenta a continuación es sencilla y agradeceríamos fidelidad en la información para que sirva eficientemente al cumplimiento del objetivo de esta investigación.

Muchas Gracias,

Colectivo de Autores.

Nombre:_____ Rol: _____

Estudiante:____Especialista:_____

1. ¿Considera usted ventajoso la aplicación de una extensión como ApEM – L para la modelación de aplicaciones educativas?
   Totalmente_____ Parcialmente_____ No_____
2. ¿Se siente usted cómodo al trabajar con la notación ApEM – L en sus distintas vistas?
   Totalmente_____ Parcialmente _____ Muy poco_____ Nulo_____
3. ¿Considera usted que es sencilla, fácil de entender y aplicar la notación con unos pocos conocimientos de programación?
   Si____ No_____
4. Marque con una X las vistas y los diagramas que considera mejor trabajados en ApEM – L para la modelación de aplicaciones educativas:
   Vista Estática_____ Vista de Arquitectura_____
   Vista de Comportamiento _____ Vista de Presentación_____
5. ¿Cree eficiente la utilización de las descripciones textuales de los casos de uso al describir los flujos de eventos de un software educativo?
   Totalmente____ Parcialmente____ Nulo_____
6. Marque con una X los elementos que a su consideración deban de reforzarse, mejorarse o ampliarse futuras versiones de la propuesta:
   Diagrama de casos de uso_____ Diagrama de actividades_____
   Diagrama de clases____ Diagrama de componentes_____
   Diagrama de despliegue_____ Diagrama de secuencia_____
   Diagrama de colaboración_____ Diagrama de estados_____
   Diagrama de estructura de navegación____ Diagrama de estructura de presentación_____
7. ¿Cree usted que debe desarrollarse un programa que permita la generación computacional de la documentación resultante de la modelación?
   Si____ No_____
8. ¿Considera que el uso de ApEM – L sustituye la necesidad en la implementación del uso del Guión y el Mapa de Navegación?
   Totalmente_____ Parcialmente_____ Muy poco_____ Nulo_____