Saber, Universidad de Oriente, Venezuela.Vol. 26 Nº 3: 297-304. (2014)
ISSN: 2343-6468 Digital / ISSN: 1315-0162 / Depósito Legal pp 198702U187

ANÁLISIS DE ALTERNATIVAS METODOLÓGICAS PARA EL DESARROLLO DE SOFTWARE EDUCATIVO

ANALYSIS OF METHODOLOGIC ALTERNATIVES FOR THE DEVELOPMENT OF EDUCATIONAL SOFTWARE

Ingrith Marcano, Gladys Benigni

Universidad de Oriente, Núcleo de Nueva Esparta, Programa de Licenciatura en Informática, Guatamare, Isla de Margarita, Venezuela. E-mail: ingrith.marcano@ne.udo.edu.ve

RESUMEN

De forma notoria la tecnología se ha expandido en amplitud y profundidad a lo largo de todas las áreas del saber. La Educación es una de las áreas sociales en la que se aplican mejoras a través de las tecnologías de información y comunicación, traduciéndose esto en la producción de software educativo (SE), así como otros tipos de innovaciones. Este trabajo documental presenta un estudio acerca de cuatro alternativas metodológicas que acompañan el desarrollo de SE, a propósito de hacer un análisis crítico de éstas, con relación a las actividades fundamentales para el desarrollo de software y a las buenas prácticas que subyacen en la ingeniería del software (IS). Como resultado, las alternativas estudiadas convergen en varios factores: a) proceso del software; b) la interdisciplinariedad del equipo de desarrollo; c) la calidad del material y estrategias educativas (enseñanza y evaluación) empleadas en la producción; d) la prueba y validación del producto. Por otra parte, cada una presenta su esquema de funcionamiento en donde precisamente se destacan en calidad, la ISE-OO de Galvis (1998) y el Modelo MOOMH de Benigni (2004).

Palabras clave: Ingeniería del software, ISE-OO, MOOMH, ADITE.

ABSTRACT

Noticeably the technology has expanded in breadth and depth across all areas of knowledge. Education is one of the social areas in which improvements have been applied through information and communication technologies, resulting in the production of educational software (ES) and other innovations. This documentary research presents a study of four alternative methodologies that support the development of ES, with the purpose to make a critical analysis of them, in relation to the fundamental activities of software development and good practices underlying software engineering (SE). As a result, the studied alternatives converge on several factors: a) software process, b) the need of an interdisciplinary development team, c) the quality of materials and instructional strategies (teaching and assessment) used in the production, d) testing and product validation. In the other hand, each methodology has its operation scheme and the ones that stand out for their precision and quality are the ISE-OO of Galvis (1998) and Model MOOMH of Benigni (2004).

Key words: Software engineering, ISE-OO, MOOMH, ADITE.

INTRODUCCIÓN

El software es definido como un conjunto de programas de computadora que desarrollados en atención a necesidades de clientes, también pueden estar dirigidos a un mercado general; la definición incluye a su vez, toda la documentación asociada al mismo (Sommerville 2007). El software ha asistido a la educación de forma paralela a su evolución y a la del hardware; en principio mediante la digitalización de lecciones que los docentes impartían a estudiantes, valiéndose de los recursos multimediales que ofrece el computador, y que promueven, incentivan el placer y la curiosidad por el estudio de temas presentados en soportes multimedia. El desarrollo de programas informáticos, constituye un proceso, siendo la ingeniería del software la disciplina que comprende todos los aspectos de su producción. A través de la ingeniería del software se establecen prácticas efectivas para desarrollar y entregar un software adaptado al propósito que impulsa su desarrollo; esta utilidad se verá reflejada a través de atributos como la funcionalidad y el rendimiento que ofrezca al usuario, por otra parte, el software también debe ser confiable, fácil de mantener y de utilizar.

El proceso de producción de software incluye un conjunto de actividades que conllevan al producto final, según Sommerville (2007), se pueden definir cuatro actividades fundamentales: la especificación del software, para identificar el alcance del producto a producir; el desarrollo, donde se diseña el programa; la validación, que consiste en revisar el software para asegurar su funcionamiento en correspondencia con las exigencias de su desarrollo; y la evolución, en donde se modifica el software para actualizarlo, adaptándolo a los posibles cambios que se requieran para garantizar su

Recibido: enero 2014. Aprobado: abril 2014.
Versión final: mayo 2014.

funcionamiento. Sin embargo, según sea la naturaleza del software, se deben incorporar particulares procesos de desarrollo, que se ajusten a determinadas especificaciones que deben plantearse en medio de la tarea de diseño, de tal forma que se pueda crear una arquitectura de información adecuada; tal es el caso del software educativo, cuyo contenido es netamente didáctico, dinámico, dirigido docentes y estudiantes en medio del proceso de enseñanza aprendizaje. Este tipo de software amerita un proceso de desarrollo que apuntale hacia lo evolutivo, de forma tal que las funciones de especificación de requisitos, el desarrollo, así como la validación, se entrelacen; bajo este concepto, al emprender el proceso de producción de software, se desarrolla un sistema inicial y se refina para adoptar en su totalidad las directrices de sus usuarios, en este caso en particular se refiere a docentes y estudiantes.

Un software educativo es una aplicación informática, que construido sobre una base pedagógica bien definida, puede constituirse en un apoyo docente durante el proceso de enseñanza aprendizaje; de lo contrario, podría llegar a considerarse un elemento distractor en el proceso. Galvis (2000), lo define como programas que permiten cumplir y apoyar funciones educativas, como por ejemplo: los que dan soporte al proceso de enseñanza y aprendizaje (un sistema para enseñar matemáticas, contenidos o ciertas habilidades cognitivas). Fernández y Delavaut (2008) lo refieren como programas de computadora para la educación. Rodríguez (2000) señala que es una aplicación informática, que soportada sobre una estrategia pedagógica bien definida, apoya directamente el proceso de enseñanza aprendizaje constituyendo un efectivo instrumento para el desarrollo educacional del hombre del nuevo siglo. Las definiciones citadas anteriormente están en sintonía con el propósito de lo que persigue la producción de software educativo, el cual amerita un tratamiento especial a la hora de emprender el proceso de su desarrollo. En este sentido, se otorga relevancia a la elección de la metodología para su producción, es así como en esta investigación se hace una revisión documental y se estudian cuatro metodologías de desarrollo de software educativo, enfocando el análisis en las cuatro actividades fundamentales presentes en el desarrollo de software (especificación, desarrollo, validación y evolución); con el propósito de dar a conocer la pertinencia que tienen en su campo de aplicación, sirviendo éste, como ejercicio de revisión de alternativas metodológicas que se suscitan en la ingeniería del software, cuyo abordaje es ampliamente necesario desde los espacios académicos donde se construyen estos sistemas en apego a lineamientos metodológicos.

METODOLOGÍAS PARA EL DESARROLLO DE SOFTWARE EDUCATIVO

Se seleccionaron cuatro metodologías para ser estudiadas en esta revisión; para la selección se consideraron los siguientes parámetros: claridad, documentación y fiabilidad a través de ejemplos de aplicación de cada una. Es así como en este apartado se esbozaron las metodologías consultadas, obteniendo de este modo, el insumo necesario para la elaboración de un esquema comparativo de opciones metodológicas implementadas para el desarrollo de este tipo de software.

A. Metodología Diseño y Desarrollo Multimedia, propuesta por Marquès (2005), consta de once pasos agrupados en tres fases; en la primera, *el análisis instructivo*, se define el problema, se concreta la génesis de la idea, construyendo el diseño instructivo y el estudio de la viabilidad; en la segunda, *el desarrollo*, se concreta el guion multimedia, la creación de los contenidos, construcción del prototipo Alfa-Test, el cual se somete a evaluación interna, para proceder a la construcción de una versión Beta-Test, la cual se someterá a una evaluación externa que posteriormente dará paso a la construcción de la versión final; la tercera, *post-producción*, se refiere a la edición, distribución, mantenimiento y post-venta o colocación en el mercado de interés.

1. En la definición del problema se precisa lo que se pretende alcanzar, estableciendo objetivos. También se identifica a quién va dirigido, el tipo de contenido que contendrá, el uso que tendrá el software, el contexto y el tiempo en que estará disponible.

2. La génesis de la idea, parte de la definición previa del problema, enfocándose precisamente en favorecer los procesos de enseñanza y aprendizaje de una situación en concreto. Se consideran los aspectos: objetivos educativos; contenidos; actividades que se ofrecerán, detallando el tipo de ejercicios e interacción del usuario-sistema; entorno audiovisual y navegación; documentación que acompañará al material; y el sistema de teleformación (el cual podría ofrecerse como apoyo al usuario a través de Internet).

3. El diseño instructivo (guion educativo o diseño funcional), incorporará los fundamentos pedagógicos. Es el primer papel de trabajo, elaborado por un especialista con apoyo de un equipo multidisciplinario constituido por: profesores con experiencia didáctica en el tema

a desarrollar, la población a quien irá dirigido y las actividades de aprendizaje, pedagogos o psicopedagogos que aporten instrumentos relacionados con el análisis y diseños en su área, especialistas en tecnología educativa que permita unificar y coordinar cada uno de los integrantes del equipo. Podrá generarse un primer borrador de las pantallas, se obtendrá la literatura de apoyo. También plantea la elaboración de un proyecto que contenga la presentación, objetivos, plataforma operativa, contenidos, tipos de actividades interactivas, estrategias de enseñanza y aprendizaje, entorno audiovisual a emplear y el sistema de navegación conveniente; por último, el diseño funcional incluirá también un esquema con una primera aproximación al formato y al contenido de la documentación que acompañará al programa.

4. La viabilidad determinará si el proyecto es factible, considerando tanto los aspectos pedagógicos, funcionales, aspectos técnicos, aspectos económicos y los aspectos comerciales. Si el estudio de viabilidad resulta positivo, se concretará el marco de desarrollo del proyecto.

5. En el guion multimedia, se detalla el contenido de la aplicación, de forma minuciosa, lo cual incluye: justificación, temática, objetivos, contenidos que se tratan, destinatarios, breve descripción, tipología y usos posibles, rasgos más característicos (enfoque pedagógico, estrategias de enseñanza y aprendizaje, esfuerzo cognitivo), integración curricular (contextos de utilización), plataforma de usuario; el mapa de navegación, que contempla el diagrama general del programa, y la descripción de sus módulos, incorporando, información, actividades interactivas, ayuda evaluación, parámetros ajustables, diagrama de los principales itinerarios pedagógicos previstos (implícitos del programa, explícitos del alumno); sistema de navegación; entorno audiovisual, vinculado con el diseño gráfico de las pantallas, en éste se consideran los elementos básicos tales como títulos, menús, ventanas, iconos, botones, espacios de texto-imagen, formularios, barras de navegación y de estado, elementos hipertextuales, fondo, entre otros.

6. Creación de los contenidos, a cargo de los expertos temáticos y profesores especialistas en la materia de que trate el programa, y también de los técnicos en diseño y desarrollo multimedia. Ésta se divide en diseño de contenidos y documentación. Por su parte, la documentación, estará a cargo de profesores especialistas en las temáticas del programa y especialistas en diseño instructivo y materiales didácticos, los cuales deben incluir: una ficha resumen, un manual instructivo, una guía didáctica y/u otros materiales complementarios que sea necesarios.

7. Elaboración del prototipo Alfa-Test, donde el grupo de especialistas (informáticos, programadores y especialistas en multimedia) desarrollan el primer prototipo interactivo del material. La evaluación interna: ésta la realizan los integrantes del equipo de diseño y desarrollo del material. Se desarrollará aplicando la metodología definida para estos tipos de materiales considerando los criterios de calidad establecidos.

8. Elaboración de la versión Beta-Test, luego de las conclusiones internas y realizados los oportunos ajustes en el diseño, base de datos y programa interactivo, el material se somete a un severo testeo técnico para depurar los posibles problemas de funcionamiento debidos a errores de programación.

9. Evaluación externa, la evaluación externa de la versión Beta-Test del programa la realizarán personas ajenas al equipo que ha participado en su diseño y desarrollo. Ésta se realiza mediante unas plantillas, y en ella participan: personal técnico, profesores, estudiantes, usuarios finales.

10. Versión final 1.0, a partir de los resultados de la evaluación externa, se hacen los últimos ajustes al material y se obtiene la versión 1.0 del programa (ob. cit.).

B. La segunda metodología considerada es la propuesta por Galvis (1998) denominada Ingeniería de Software Educativo-Orientada a Objetos; esta constituye un modelo que contempla una serie de fases sistemáticas: análisis, diseño, desarrollo, prueba piloto y pruebas a lo largo del desarrollo.

1. El Análisis, tiene como propósito determinar el contexto y los requerimientos que deberá atender la solución interactiva, y se establece como mínimo la siguiente información: características de la población objetivo, conducta de entrada y campo vital, el problema o necesidad a atender, principios pedagógicos y didácticos aplicables, justificación de uso de los medios interactivos. Obteniéndose la documentación contentiva de los requerimientos detectados, la misma debe incluir: la descripción de la aplicación, las restricciones, así como los diagramas de interacción; es decir, proveerá información precisa acerca de lo que hará la aplicación; las restricciones que tendrá y una descripción de los posibles

escenarios de interacción que tendrá el usuario. Estas restricciones están relacionadas con aspectos tales como: población objetivo y sus características (información recopilada en la fase de análisis), las áreas de contenido y sus características, los principios pedagógicos aplicables, modos de uso de la aplicación: individual, grupal, con apoyo de instructor, la conducta de entrada. Todo aquello con lo que el usuario cuenta antes de usar la aplicación, ejemplo: experiencia, conocimiento y habilidades.

2. El Diseño, se construye en función directa de los resultados de la etapa de análisis, es importante hacer explícitos los datos que caracterizan el entorno del software: destinatarios, área del contenido, necesidad educativa, limitaciones y recursos para los usuarios, equipo y soporte lógico. Atiende tres tipos de diseño: Educativo (este debe resolver las interrogantes que se refieren al alcance, contenido y tratamiento que debe ser capaz de apoyar el software educativo), comunicacional (es donde se maneja la interacción entre usuario y maquina se denomina interfaz), y computacional (con base en las necesidades se estable qué funciones son deseables que cumpla el software educativo en apoyo de sus usuarios, el docente y los estudiantes). Este enfoque hacia el diseño computacional en complemento del diseño educativo, permitirá que los resultados y formulaciones realizadas sean fácilmente identificadas en la implementación de la aplicación, garantizando por consiguiente, un diseño computacional y posterior implementación con una alta calidad.

3. En el Desarrollo se implementa toda la aplicación usando la información recabada hasta el momento. Se codifica con el lenguaje escogido tomando en consideración los diagramas de interacción mencionados anteriormente. Es preciso establecer la herramienta de desarrollo sobre el cual se va a efectuar el programa, atendiendo a recursos humanos necesarios, costo, disponibilidad en el mercado, portabilidad, facilidades al desarrollar, cumpliendo las metas en términos de tiempo y calidad de software educativo.

4. Prueba Piloto y prueba a lo largo y al final del desarrollo, aquí se pretende ayudar a la depuración del software a partir de su utilización por una muestra representativa de los tipos de destinatarios para los que se hizo y la consiguiente evaluación formativa. Es imprescindible realizar ciertas validaciones (efectuadas por expertos) de los prototipos durante las etapas de diseño y prueba en cada uno de los módulos desarrollados, a medida que estos están funcionales. Superada la depuración y ajuste, se pone a disposición una versión beta del software. El autor resalta que esto conviene hacerlo con una muestra de la población; se pretende a través de dicha prueba piloto verificar que efectivamente la aplicación satisface las necesidades y cumple con la funcionalidad requerida (ob. cit.).

C. Como tercera opción metodológica, se seleccionó el modelo ADITE, propuesto por Polo (2003), el cual se sustenta en una concepción constructivista del aprendizaje, y se caracteriza además, por la no linealidad en cuanto a sus componentes: Análisis, Diseño Instruccional, Diseño Tecnológico y Evaluación (Fig. 1).

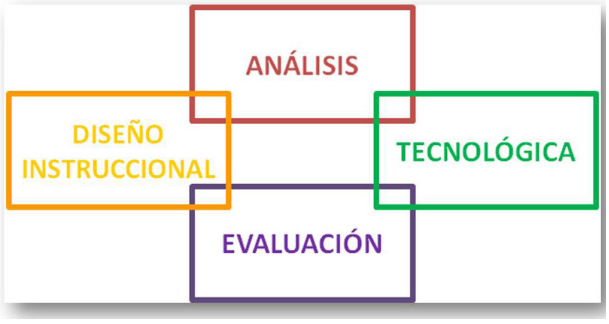

Figura 1. Modelo ADITE, tomado de Polo (2003).

1. El Análisis, tiene por objeto el estudio de los resultados esperados y condiciones de utilización y administración del medio (Polo 2003). Este componente incluye los subcomponentes siguientes: análisis del problema instruccional a resolver, análisis de la población a la cual se dirige el medio instruccional, análisis del contenido según tipos de conocimientos, análisis sobre la fundamentación teórica que se asumirá en el diseño instruccional del medio, análisis de las estrategias cognoscitivas que se activarán en el estudiante, y el análisis de la administración tecnológica. La autora del método señala que cuando se diseña un medio instruccional basado en tecnologías, hay que saber qué se espera de él, qué tipo de necesidad educativa va a solucionar, cuáles son las características de la población a la cual va dirigido, qué tipos de conocimientos se mediarán. Por lo tanto, las teorías del aprendizaje y de la instrucción deben ser analizadas en esta fase, en este sentido, se plantean con claridad los fundamentos teóricos que suscriben la propuesta. Con base en lo anterior, la ADITE considera la implementación de: a) la teoría pedagógica de la comprensión, de Perkins, que contempla una forma específica de plantear la enseñanza; b) la teoría de Gagné, basada en el procesamiento de la información y la planificación de los eventos instruccionales; c) la teoría de Merrill, fundamentada en las transacciones como sistemas de interactividad; d) la teoría de la flexibilidad cognitiva de Spiro y Jehng, que

establece hipertextos libres en el diseño.

2. El segundo componente es el de Diseño, en el que se desarrollan y formulan las especificaciones de las metas y objetivos que se quieren lograr; se explican los procesos, estructuras y estrategias que se requieren para aprender el conocimiento o asimilar y desarrollar cualquier habilidad. Comprende los siguientes subcomponentes: formulación de metas y objetivos de aprendizaje, selección de contenidos y estructuración de la secuencia de los mismos, selección de estrategias y actividades instruccionales, y diseño de estrategias e instrumentos de evaluación de los aprendizajes. En este orden, se conforma una instrucción planificada a partir de formas adecuadas para la construcción de los conocimientos.

3. El tercer componente es el Tecnológico, importante por cuanto implica la interdisciplinariedad del recurso humano que se necesita para el diseño de situaciones instruccionales mediadas por la tecnología. Implica los siguientes subcomponentes: definición del proceso de interacción, definición de la aplicación de programación, definición del ambiente de aprendizaje, definición del sistema de control, y la definición de la implementación. Aquí se establecen los medios de representación, con base en los procesos interactivos de las transacciones pedagógicas, es decir, contempla su fundamento teórico, la interacción es por ende, centrada en el alumno.

4. Finalmente, el componente Evaluación, que está presente en los demás componentes, en tanto que la revisión del trabajo que se va realizando es inherente a todo el proceso de diseño. Comprende los siguientes subcomponentes: diseño de estrategias de evaluación de los aprendizajes, especificación de la evaluación formativa de los componentes del sistema, revisión de los ambientes de aprendizaje, definición del sistema de control, y la implementación de la evaluación sumativa del sistema. En sí, la evaluación constará de una serie de procesos que permitirán constatar los avances y logros del estudiante. Además la autora de ADITE plantea que este componente requiere de la elaboración de un conjunto de instrumentos de evaluación para observar cuánta retroalimentación se le dio al estudiante a través de sus consultas al docente.

5. Luego, cabe destacar que resulta imprescindible proceder a la evaluación formativa de todos los componentes. El modelo de evaluación formativa propuesto se realiza durante las fases de desarrollo, para recoger información sobre el diseño que se va produciendo. Se buscan respuestas a las preguntas siguientes: ¿Qué se evalúa? ¿Quiénes evalúan? ¿Cuáles son los procedimientos e instrumentos para realizar la evaluación? ¿Cómo se analizan los resultados? ¿Qué decisiones pueden tomarse con los datos obtenidos? A través de la evaluación, se podrá determinar la validez y efectividad del diseño y del medio producido. Se evaluará si realmente el alumno podrá tener un aprendizaje significativo; si las estrategias de aprendizaje planificadas permitirán el logro de los objetivos y la tecnología es la adecuada (ob. cit.).

D. La cuarta opción metodológica consultada se denomina MOOMH, metodología orientada a objetos para desarrollar software multimedia e hipermedia, desarrollada por Benigni (2004) Está subdividida en cuatro modelos interrelacionados: requerimientos, análisis, diseño e implementación (Fig. 2).

Figura 2. Modelo MOOMH, tomado de Benigni (2004).

1. En el modelo de requerimientos, se establece hacia quién va dirigido el software, a modo de incorporar todo aquello necesario para el funcionamiento del mismo. Para ello, comprende las siguientes etapas: a) estudio de la factibilidad, que determina las necesidades del sistema y su factibilidad, la cual viene dada por la disponibilidad real en cuanto a los recursos necesarios para el desarrollo del prototipo. De esta manera se debe analizar la problemática para determinar si puede ser o no resuelto efectivamente tomando en cuenta todos aquellos aspectos que influyen sobre él; y b) determinación de los requerimientos del problema, que significa hacer el levantamiento de información, usando para ello los casos de uso, que permiten capturar y modelar el comportamiento actual del usuario, facilitando tanto a los

diseñadores de software como a los clientes, llegar a un acuerdo sobre las condiciones que debe cumplir el sistema. Una vez determinadas las necesidades y la factibilidad (disponibilidad técnica, operativa y económica), y hacia quién va dirigido el software, se procede a definir las lecciones (en caso de ser un software educativo), unidades de información (si se trata de soporte didáctico) o bloques administrativos (aplicación educativa en la web) correspondientes al tópico seleccionado. Este tópico debe escogerse con personal especializado en el área (en este caso, maestros(as) o profesores(as), analizando aquellos que demuestren mayor complejidad e importancia.

2. El modelo de análisis está compuesto de las fases siguientes: a) identificación de los objetos, donde se definen los mismos y la relación existente entre ellos, destacando que se está a un nivel macro del problema y que los objetos se definen en ese mismo nivel; es importante acotar que para llevar a un adecuado término el tópico seleccionado, es imprescindible definir los objetos claramente así como las relaciones entre ellos y seguidamente los objetivos del mismo; b) elaboración del mapa de navegación del sistema, éste representará el prototipo a desarrollar, a través de nodos (objetos) y las asociaciones o enlaces se indicarán según lo asocia el desarrollador del software; y c) diseño de los objetos, donde se representan de manera sencilla las lecciones, unidades de información o bloques administrativos, diseñando tablas en las que se muestran el registro de los elementos multimedia que se propone y también deben incluirse los bocetos.

3. El modelo de diseño, pretende mostrar un prototipo de las pantallas del sistema a los posibles usuarios finales, evaluando así la usabilidad de la aplicación y la interacción entre ellos. Se realiza en tres pasos, los cuáles podrían ser excluyentes, dependiendo de la aplicación que se esté desarrollando: a) prototipo de la interfaz, aquí el diseñador muestra a los posibles usuarios los prototipos de la interfaz (diseño de interfaz), y luego de haber hecho la evaluación o análisis de las mismas, procede a indicar los medios (sonido, texto, imágenes, entre otras) presentes en cada una de ellas, utilizando para esto lo que se denominará bibliotecas; b) diseño de la base de datos, donde puede emplearse el modelo de datos entidad relación o el diagrama de clases; y c) modelado en la Web, el objetivo de este paso es el de modelar los componentes web, lo cual consiste en diagramar solo la "lógica de negocio" y no la lógica de presentación; se utiliza la extensión de la notación de Lenguaje de Modelado Unificado (Unified Modeling Lenguaje - UML) destinado para tal fin.

4. En el modelo de implementación, se seleccionan los recursos computacionales necesarios para programar el sistema, se debe hacer una evaluación exhaustiva con el personal que colaboró en los modelos anteriores y se debe elaborar el manual del programador o manual del sistema respectivo. Consta de las fases: escritura del código fuente; b) arquitectura o capas OSI, donde se verifica bajo qué arquitectura o capas OSI será implementada la aplicación, luego, al finalizar la programación del sistema, el mismo debe ser evaluado por especialistas no solo del área de informática, sino de aquellas personas que participaron como expertos en el área de análisis, diseño y desarrollo para efectuar los cambios respectivos (de haberlos); y c) pruebas del sistema, deben efectuarse las pruebas (generalmente pruebas alfa y beta) para asegurar que la aplicación cumple con el propósito planteado y que el mismo pueda ser lanzado en su primera versión, cumpliendo con los estándares de calidad y usabilidad. Estas pruebas deberían aplicarse en cada modelo del método, para garantizar el éxito en la primera versión del prototipo desarrollado.

COMPARATIVO ENTRE METODOLOGÍAS

1. La propuesta de Marquès (2005), implícitamente obedece a un proceso de desarrollo en cascada, y contempla la descripción precisa de las consideraciones que cada miembro del grupo desarrollador debe ejecutar, lo que da minuciosidad al proceso y rol de cada quien; promueve además la generación de un proyecto que sintetiza en máximo quince hojas, el contenido del producto; sin embargo, no muestra evidencia de instrumentos para la representación de los avances obtenidos tras el seguimiento de cada fase, de tal forma que carece de medios para mantener convenientemente organizados los resultados de las fases; lo que por consiguiente, representa una debilidad observable en el método, que no está en correspondencia con la ingeniería del software en lo que respecta a la documentación de sistemas. Básicamente ha sido esfuerzo de la ingeniería del software, velar por la producción de modelos de desarrollo de software, pero también ha establecido adelantos en cuanto a la unificación de lenguajes, que puedan describir externa e internamente la naturaleza de los productos software, por ejemplo UML (Unified Modeling Language), ampliamente utilizado hoy en día.

2. Galvis (1998), en su metodología Ingeniería de Software Educativo-Orientada a Objetos, plantea el uso del lenguaje de modelado Unificado (UML),

como notación para la diagramación de artefactos que representan el software en construcción, por ejemplo, diagramas de clase, diagramas de interacción, casos de uso, entre otros. De hecho, en cada diseño anunciado, el autor establece los criterios que se deben seguir para alcanzar el objetivo de cada uno de ellos, contando así con una especificación bien detallada del qué hacer en cada diseño y en cada etapa del proceso. Esta metodología aparte de incorporar aspectos de diseño que se corresponden con la naturaleza educativa del producto, también atiende las especificaciones de representación que se han desarrollado a través de la ingeniería del software, lo cual representa una mejor opción metodológica en comparación con la primera citada en el estudio, esto, debido a su expresa preocupación para generar la documentación contentiva de la estructura del software en cada etapa de su desarrollo, facilitando por consiguiente su posterior estudio y mantenimiento; esto, además de adoptar implícitamente un modelo de proceso con tendencia evolutiva, puesto que no es lineal y las especificaciones se van desarrollando en forma creciente, se trata pues, de un proceso enfocado en el cliente.

3. Polo (2003), en su modelo ADITE, centra la producción de un software o medio, no sólo en el apoyo al proceso educativo, sino que éste pueda ser comprobable gracias a la evaluación exhaustiva al cual es sometido. También es un proceso que evidencia una tendencia evolutiva. Contempla con claridad los aspectos educacionales, pedagógicos, que fungen de sustento teórico a las formas y estrategias que adopta o sugiere el método de trabajo; se trata entonces de un aporte que hace explícita la importancia de la interdisciplinariedad en el grupo de desarrollo. Por otra parte, no se puede dejar de mencionar que esta alternativa ofrece instrucciones para ejecutar cada componente, también involucra en el diseño tecnológico, los instrumentos de representación de aspectos que constituyen las interfaces del medio, con lo cual se evidencia sutilmente apego a la ingeniería del software, con poco desarrollo, pero que seguramente, será punto de partida para que expertos en el área complementen la metodología con el lenguaje unificado de modelado de sistemas, a fin de proveer mayor documentación factible de examinar acerca del medio producido.

4. Nótese que en MOOMH Benigni (2004), se emplean cuidadosamente las buenas prácticas de la ingeniería del software, dejando en manos de los especialistas en el área educativa, la preparación de materiales y contenidos, incluyendo con esto el detalle de estrategias instruccionales y evaluativas que se implementarán, es decir, se aboca al desarrollo del software como tal, tomando en cuenta que, durante todo el proceso interviene un grupo interdisciplinario que en conjunto colaborarán cada quien desde su especialidad, para producir un software en correspondencia con sus objetivos. Un software desarrollado bajo esta metodología, contará de seguro con una buena documentación de sistemas; el proceso de desarrollo es iterativo e incremental, en correspondencia con la clasificación de los modelos de desarrollo de software.

CONSIDERACIONES FINALES

El desarrollo de soportes multimediales para la educación es una disciplina que ha evolucionado a lo largo del tiempo, por ende, se han dedicado múltiples esfuerzos para que la educación y la informática se vinculen adecuadamente, explotando por esta vía, todos los recursos que la informática ha puesto a la orden del mundo. El desarrollo de software implica un proceso donde interviene la logística, el recurso humano y tecnológico, necesarios para la producción de éstos. En un desarrollo de software ideal, se debe seleccionar un proceso cónsono con la ingeniería de software (Fig. 3), lo que se espera es un software de calidad y que el mismo pueda ser actualizado a consecuencia de los cambios continuos a los que se someten los procesos automatizados, que en este caso trata sobre lecciones, ejercicios, simulaciones, entre otros; es por ello, que la selección de una metodología es muy significativa puesto que de allí depende la representación lógica del sistema, de forma tal que se exponga toda la información de la entidad desarrollada (software) en momentos futuros. Actualmente se pueden encontrar múltiples referencias de métodos que soportan este tipo de desarrollo, sin embargo, el estudio consideró estas cuatro metodologías debido a su claridad, documentación y fiabilidad, aportando como resultado y en función del análisis aplicado a éstas, que una selección acertada pueden ser las metodologías propuestas por Galvis (1998) o Benigni (2004); ambas proporcionan un soporte documental que facilita su aplicación, en correspondencia con los procedimientos que se deben seguir y los artefactos de modelado que se derivan de esta práctica.

Autor	Proceso de Desarrollo	Equipo de Trabajo	Documentación	Prueba y Validación
Marqués (2005)	En Cascada	Multidisciplinario	Diseño instructivo, Estudio de viabilidad, guión multimedia.	Uso de prototipo, Alfa-Test, Beta-Test, Evaluación Externa
Galvis (1998)	Evolutiva	No lo establece explícitamente	Contexto, Requerimientos, diseño computacional (sin precisar su forma de representación)	Prueba piloto, Validación por Expertos, Pruebas a lo largo y al final del proceso de desarrollo
Polo (2003)	Evolutiva	Interdisciplinario	Diagramas de interacción del producto (sin especificar en el componente tecnológico los artefactos a utilizar)	Evaluación en todo el proceso, Evaluación centrada en el alumno como usuario del producto.
Benigni (2004)	Iterativa e Incremental	Interdisciplinario	Casos de Uso, Estudio de factibilidad, Mapa de navegación, Diseño de objetos, Prototipos de pantalla, Diseño de Base de datos, Modelado en la Web, Manual del sistema.	Pruebas alfa y beta, Evaluación de usabilidad (en todo el proceso).

Figura 3. Resumen comparativo de las metodologías.

REFERENCIAS BIBLIOGRÁFICAS

Benigni G. 2004. Una metodología orientada a objetos para la producción de software multimedia. Saber. 16(1):26-32.

Fernández R, Delavaut M. 2008. Educación y tecnología. Un Binomio Excepcional. Grupo K, España, pp. 247.

Galvis A. 2000. Ingeniería de software educativo. Universidad de Los Andes. Colombia, pp. 359.

Galvis A. 1998. Modelos de Desarrollo de MDCS. Ingeniería de Software Educativo. Modelo Propuesto por Galvis. Disponible en línea en: http://modelosdesarrollomdc.blogspot.com/search/label/Galvis (Acceso 30.01.2013).

Marquès P. 2005. Modelos de Desarrollo de MDCS. La Metodología de Pere Marquès. Disponible en línea en: http://www.peremarques.net/disdesa.htm (Acceso 10.03.2013).

Polo M. 2003. Aproximación a un Modelo de Diseño: ADITE. Docencia Universitaria. 1(4):67-83. Disponible en línea en: http://www.ucv.ve/fileadmin/user_upload/sadpro/Documentos/docencia_vol4_n1_2003/7_art._4Marina_Polo.pdf (Acceso 15.02.2013).

Rodríguez R. 2000. Introducción a la Informática Educativa. Universidad de Pinar del Río. "Hermanos Sainz", Cuba, pp. 54.

Sommerville I. 2007. Ingeniería del Software. Pearson y Addison Wesley, España, pp. 687