IX CONGRESO ARGENTINO DE CIENCIAS DE LA COMPUTACIÓN 2003
II Workshop de Tecnología Informática Aplicada en Educación

¿INSTRUCTIVISMO O CONSTRUCTIVISMO?: GUÍA MULTIMEDIA PARA EL DESARROLLO DE SOFTWARE EDUCATIVO[1]

Doctor Carlos Vargas Castillo
Escuela de Ciencias de la Computación e Informática
Universidad de Costa Rica

cvar@costarricense.cr

RESUMEN

Este artículo se enfoca en aquellos elementos teórico-metodológicos que deben guiar la elaboración de software con fines educativos. Se propone una guía metodológica, con la que se persigue propiciar un enfoque crítico, reflexivo, interdisciplinario e integrador, respecto de los conceptos y constructos que resultan claves en el desarrollo de software educativo. En primer lugar, se presenta la estructura completa de categorías temáticas utilizada en los contenidos de la guía. A continuación, se ofrecen una serie de reflexiones, enfatizando la relación entre pedagogía e ingeniería del software. Luego, se profundiza acerca de los principios subyacientes en los dos paradigmas educativos, instructivismo y constructivismo, los cuales con frecuencia, de manera implícita o explícita, permean el diseño de productos de software educativos. Finalmente, se muestran algunas pantallas de la guía metodológica.

Palabras claves: Software educativo, instructivismo, constructivismo, guía metodológica, herramienta de software multimedial

1. Introducción

El auge que las tecnologías en informática y en comunicaciones han tenido en la última década es bastante conocido. Es tal su magnitud, que prácticamente han logrado permear todas las esferas del quehacer humano. Incluso se habla de la revolución informática. Desde los inicios de esta revolución, en forma creciente se han venido utilizando estas tecnologías en el campo de la educación, en cuyo caso se suele denotar por tecnología educativa: "área del saber que relaciona la informática con la educación" (Galvis, 1995, p.15). En Costa Rica, así como en otros países de Iberoamérica, a veces se utiliza el término informática educativa, como sinónimo. De acuerdo con Seas (1998), "La informática educativa se ha desarrollado como un nuevo énfasis en las Ciencias de la Educación, que procura la integración de la tecnología computacional con el quehacer educativo, tanto el proceso de enseñanza como el de aprendizaje, como una opción para el cambio cualitativo de la educación, ofreciendo a los docentes la posibilidad de expandir y desarrollar nuevos escenarios para el aprendizaje". (p.6) Por lo tanto, se busca incorporar las tecnologías informáticas al quehacer educativo con el fin de lograr un cambio cualitativo en la educación, al potenciar nuevos escenarios para el aprendizaje. El uso de las nuevas tecnologías informáticas ofrecen mayores posibilidades de enriquecer las prácticas pedagógicas que los medios audiovisuales convencionales.

En Costa Rica, diversas instituciones e individuos han contribuido con investigaciones relacionadas con la informática educativa, con producción de herramientas de software educativo, y con material didáctico en formato

[1] Elaborado dentro del proyecto **Desarrollo de Software Educativo**, y el **Programa de Cooperación Académica para el uso de las tecnologías de información en el mejoramiento de la docencia universitaria**, ambos inscritos ante las Vicerrectorías de Investigación y de Docencia de la Universidad de Costa Rica.

digital. Se destacan el Instituto para el Mejoramiento de la Educación Costarricense (IIMEC), y la Escuela de Ciencias de la Computación e Informática (ECCI), ambos de la Universidad de Costa Rica (UCR), Universidad Estatal a Distancia (UNED), la Universidad Nacional (UNA), el Instituto Tecnológico de Costa Rica (ITCR), el Programa de Educación Ambiental del Instituto Nacional de Biodiversidad (INBio) y la Fundación Omar Dengo (FOD).

En un estudio que realizaramos para conocer las orientaciones y tendencias en producción de software educativo en Costa Rica (Vargas, 2003), se censó la producción de software educativo desarrollado en las universidades estatales costarricenses, a partir de 1990. Se comprobó una tendencia creciente en el desarrollo de las aplicaciones de software educativo. Por otra parte, también se evidenció que una parte sustancial del software producido, aunque bien elaborado desde el punto de vista técnico, en algunos casos exhibe debilidades pedagógicas. Además, en dicho estudio se encontró que una cantidad importante del software educativo producido, proviene de iniciativas de las Escuelas en Ciencias de la Computación y de sus institutos de investigación (tesis y proyectos de investigación), lo que ayuda a explicar, en parte, las deficiencias pedagógicas de algunas aplicaciones educativas. Se revela entonces que, en la educación superior costarricense, el diseño de software educativo algunas veces no toma en consideración investigaciones disponibles en el campo educativo, las cuales ofrecen marcos conceptuales acerca de la enseñanza y el aprendizaje en general, y acerca de disciplinas particulares.

El desarrollo de software educativo enfocado exclusivamente en el producto, se aparta de la manera en que usualmente se suele llevar a cabo la investigación en la educación superior. Al respecto, Kennedy indica que "El desarrollo de software educativo ha tendido a enfocarse en aspectos de *hardware* (el tipo de plataforma, velocidades de los procesadores, etc.) y del *software* (diseño de pantallas, herramientas de desarrollo, uso del color, navegación, presupuesto y restricciones de tiempo), en vez de la perspectiva educativa." (p.1). El diseño e implementación de software, regido por modelos propios de la ingeniería del software, pone gran énfasis en la calidad y en la eficiencia (Galvis, 1995). Se busca obtener la calidad del producto mediante la adopción de estándares internacionales[2]. Cuando los diseñadores de software son ingenieros de sistemas, lo que intentan evaluar son los aspectos computacionales, la relación hombre-máquina y la facilidad o funcionalidad de uso del programa. Sin embargo, en el caso de una aplicación educativa, además de buscarse la calidad técnica, se deben tomar muy en cuenta aspectos educacionales: pedagógicos, didácticos y curriculares.

Partimos de la premisa de que el software educativo, no solamente debe ser sólido desde el punto de vista técnico, sino que además, es indispensable partir de una fundamentación pedagógica. Es importante que el equipo de trabajo, conozca y establezca de modo explícito, desde cual modelo pedagógico se persigue fundamentar el software, previo al diseño e implementación del mismo. En este artículo analizamos los dos paradigmas educativos más comunes en que se sustenta el software educativo: el instructivismo y el constructivismo. Consideramos que es importante conocer y explicitar la perspectiva teórica educativa, que sirve de fundamento a un producto de software educativo concreto, porque de lo contrario, su implementación podría resultar en una mezcla de técnicas conflictivas, al derivarse de concepciones sobre enseñanza y aprendizaje antagónicas entre sí.

Proponemos una guía metodológica enmarcada dentro de la Ingeniería de Software Educativo. Esta trata los aspectos propios de la ingeniería de software, y además, enfatiza aspectos educativos en relación con la pedagogía, la didáctica y la integración curricular. Por razones de espacio, en este artículo nos limitamos a la pedagogía, la cual constituye el primer elemento a considerarse, cuando se busque desarrollar una aplicación verdaderamente educativa. Diferentes concepciones acerca de la enseñanza y el aprendizaje conducen a diferentes diseños de software. La metodología propuesta se constituye en una herramienta que facilita el comprender como el componente educacional guía y encaja en el diseño de las aplicaciones de software educativo.

La metodología propuesta busca desarrollar un enfoque crítico, reflexivo, interdisciplinario e integrador, mediante aquellos conceptos claves, fundamentales del desarrollo de software educativo. La propuesta está orientada a estudiantes de posgrado en Ciencias de la Computación, de la Universidad de Costa Rica, del curso Diseño de Interfaces de Usuario. Sin embargo, la guía es de propósito general.

En las siguiente sección se muestra la estructura completa de categorías temáticas utilizadas en los contenidos de la guía. Luego se ofrecen una serie de reflexiones, enfatizando la relación pedagogía e ingeniería del software. Se

[2] ISO/IEC 9001 (1991). *Quality Systems –Model for quality assurance in design/development, production, installation an servicing.*

profundiza acerca de los principios subyacientes en los paradigmas antagónicos: instructisvo y constructivismo. Finalmente, se muestran algunas pantallas de la guía metodológica. La figura No 1. muestra la pantalla principal de la guía.

Figura No. 1: Pantalla inicial de la guía metodológica.

2. Procedimientos metodológicos para derivar los contenidos temáticos

Para derivar los contenidos temáticos de la guía metodológica para desarrollar software educativo, se seleccionó una muestra representativa del software producido en las universidades estatales, atendiendo al criterio de representatividad y pertinencia, respecto de las múltiples dimensiones del proceso de elaboración de un producto de software educativo. En la selección de la muestra se consideró: la diversidad de tipos de software producidos y los distintos niveles educativos.

Mediante el método de análisis de contenido, se logró identificar aquellos aspectos teórico-metodológicos y práticos, más significativos, concernientes con la producción de software educativo. Los dominios en el nivel más general son: la descripción del proyecto, el enfoque pedagógico, la propuesta didáctica, el diseño de la interfaz, la producción, la evaluación, la integración curricular, y la conformación del equipo humano.

El tipo de análisis de contenido llevado a cabo fue de tipo abierto, temático, y cualitativo, en el cual, no se parte de categorías previamente definidas, sino, que estas van surgiendo desde los mismos contenidos. Bermúdez (1982, 73), señala las características más importantes que deben poseer las categorías, las cuales se observan en la Figura No. 2.

Exclusividad	Significa que los mismos elementos no deben clasificarse en las mismas categorías.
Fiabilidad	Las categorías deben definirse de una manera suficientemente clara para que elementos semejantes del contenido se clasifiquen en las mismas categorías.
Pertinencia	Las categorías deben estar en relación con el objetivo perseguido y al contenido tratado.
Marco de referencia	Debe establecer una relación entre las categorías y el contexto.
Evitar extremos	No es aconsejable imponer un esquema muy rígido de clasificación, ni uno muy superficial. Deberán evitarse las categorías muy detalladas y numerosas, creadas con el fin de no dejar escapar nada. Tampoco es aconsejable utilizar categorías demasiado "amplias" en las que cualquier elemento pueda clasificarse.

Figura No. 2: Características de las categorías de análisis.

Procediendo de lo particular hacia lo general, es decir, las subcategorías conceptuales emergen de lo más concreto hacia lo general, se obtuvo un sistema categorial. La Figura No. 3, muestra el árbol de categorías temáticas para el desarrollo de software. Estas en conjunto definen una estrutura conceptual para el diseño y desarrollo de software educativo, de manera integral, que constituye el árbol de contenidos temáticos, de la guía metodológica titulada: "Software Educativo: una Guía para su Elaboración".

3. Pedagogía e Ingeniería del Software

Resulta común, y deseable, que durante el proceso de desarrollo de un producto de software educativo, se forme un equipo humano multidisciplinario, usualmente formado por educadores e ingenieros en sistemas, así como especialistas en diseño grafico, en sonido, en animación, e ilustradores y locucionistas, de acuerdo con los medios seleccionados para construir la interfaz de usuario. Los últimos miembros juegan un papel importante para interfaces de usuario sofisticadas. Actualmente, a la interfaz se le presta bastante atención, quizá porque se considera que "No importa que tan potentes o elegantes sean las tecnologías subyacientes en la aplicación educativa, es la interfaz de usuario la que en último término determina como esos sistemas serán usados". (MacIntyre, 1996, p.251).

El coordinador del equipo, y responsable del desarrollo del producto educativo, puede perfectamente ser un especialista en computación, pero deberá incorporar en su forma de pensar la contribución de la pedagogía (conceptos, aportes, y principios). Bajo cualquier modelo de desarrollo de software por el que se incline (por ejemplo, modelo de desarrollo en cascada, o de desarrollo rápido por prototipos), a nuestro juicio, necesita considerar la fundamentación conceptual que ofrecen los paradigmas pedagógicos.

Existe gran variedad de modelos sistémicos, caracterizados por la descomposición del desarrollo del software educativo en tareas y procesos. Los más populares son el modelo de desarrollo lineal, también conocido como modelo en cascada, y el modelo de desarrollo por prototipos. El modelo lineal considera la elaboración del software educativo como un proceso de cinco fases independientes: análisis, diseño, desarrollo, evaluación e implementación. Mientras que el modelo de desarrollo rápido de prototipos "se caracteriza por un factor de revisión continúa y actualización del producto" (Gros, 1997, p.23). Este modelo está basado en cinco fases: formulación de objetivos, diseño del programa, soluciones, prototipos, revisión de la solución y revisión de objetivos. Dentro del modelo de desarrollo rápido de prototipos, la tarea de desarrollo de software educativo se concibe como un proceso de resolución de problemas.

La participación de los educadores tiende a ser más intensa durante la fase de diseño del software educativo, mientras que la de los especialistas en computación se acentúa durante la fase de desarrollo. De modo que el desarrollo de aplicaciones educativas requiere de una metodología que permita una verdadera cooperación entre los distintos tipos de agentes involucrados: principalmente los educadores y los expertos en software. Una disciplina dentro del campo de

la informática está emergiendo, denominada Ingeniería de Software Educativo (Nodenot, 1998), la cual usa modelos provenientes del campo educativo, conjuntamente con principios propios de la ingeniería del software.

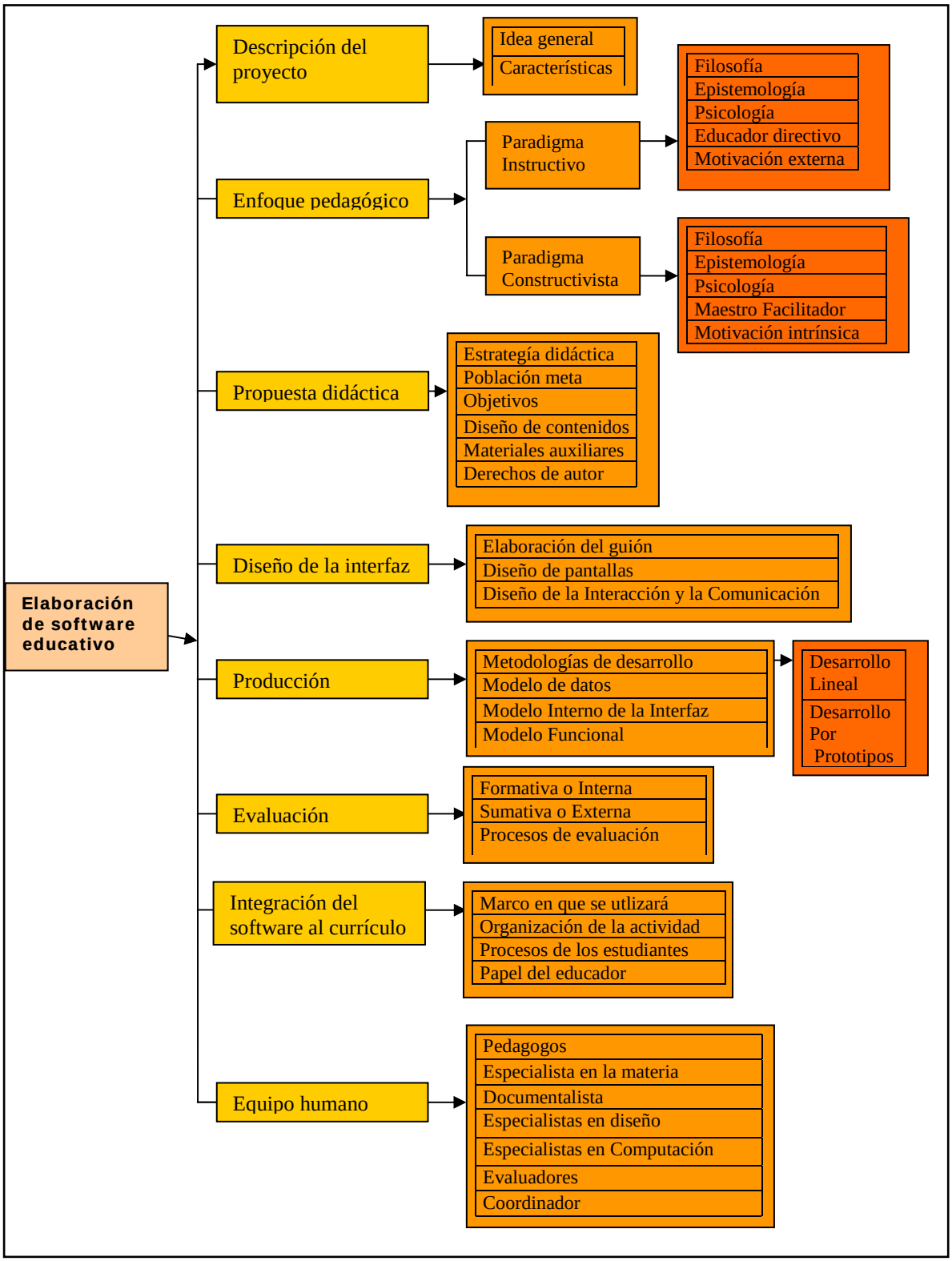

Figura No. 3: Categorías temáticas para el desarrollo de software educativo

La guía metodológica propuesta, intencionalmente sitúa los asuntos pedagógicos como elementos claves que no pueden obviarse, y que deben considerarse desde el inicio de cualquier proyecto que desarrolle software educativo. Podría pensarse, que una manera sencilla de asegurar flexibilidad pedagógica se lograría si se excluyen todas las consideraciones pedagógicas en el diseño del software educativo, resultando así en un software con neutralidad pedagógica. Sin embargo esta es una actitud muy simplista. Por nuestra parte, compartimos con Bruner (1996), que "la neutralidad pedagógica es un mito, por cuanto esta es basada sobre la falacia que a menos que una posición pedagógica se haga explícita (ya sea en términos de las creencias de un persona o la funcionalidad de un sistema), esta no existe. En efecto, lo que es usualmente enseñado como neutralidad pedagógica por los vendedores de software es meramente ingenuidad pedagógica"(p.13).

El rasgo distintivo de una aplicación educativa respecto de una meramente informativa, es que la primera se halla inmersa en un diseño concreto o plan determinado para que alguien aprenda algo nuevo. Incorporar este rasgo en el diseño incrementa la complejidad del producto. Por aplicación educativa se entiende, en términos amplios, cualquier producto de software, realizado con una finalidad educativa: "Una aplicación educativa debe tener la intención de educar y debe hacerlo de manera sistemática." (Bauzá, 1997,253). Por lo tanto, una aplicación orientada a la educación deberá considerar, además del guión, el discurso, y el mensaje, el añadir una estrategia de formación.

Figura No. 4 Menú principal.

4. El enfoque pedagógico: instrucción vrs construcción

Con el término pedagogía se suele hacer referencia a un cuerpo de conocimientos teóricos y prácticos, fruto de la reflexión sobre el fenómeno de la educación intencional. Existe una gran diversidad de enfoques pedagógicos, que se fundamentan en diversas concepciones, teorías e investigaciones, provenientes de diversos dominios del

conocimiento, principalmente de la Filosofía, la Psicología, la Antropología y la Epistemología. El diseño y desarrollo de software educativo no es neutral, ya que se ve permeado por las concepciones de quienes lo construyen y usan. Generalmente no se da en la práctica un enfoque pedagógico puro. Encontramos que el software educativo desarrollado en las universidades estatales costarricenses, tiende a polarizarse hacia alguno de dos paradigmas educativos: enfoque instructivo o enfoque constructivista, lo cual es coincidente con el planteamiento teórico de Reeves (2000). Las figura No 4. muestra la pantalla que trata acerca de estos dos enfoques anatagónicos.

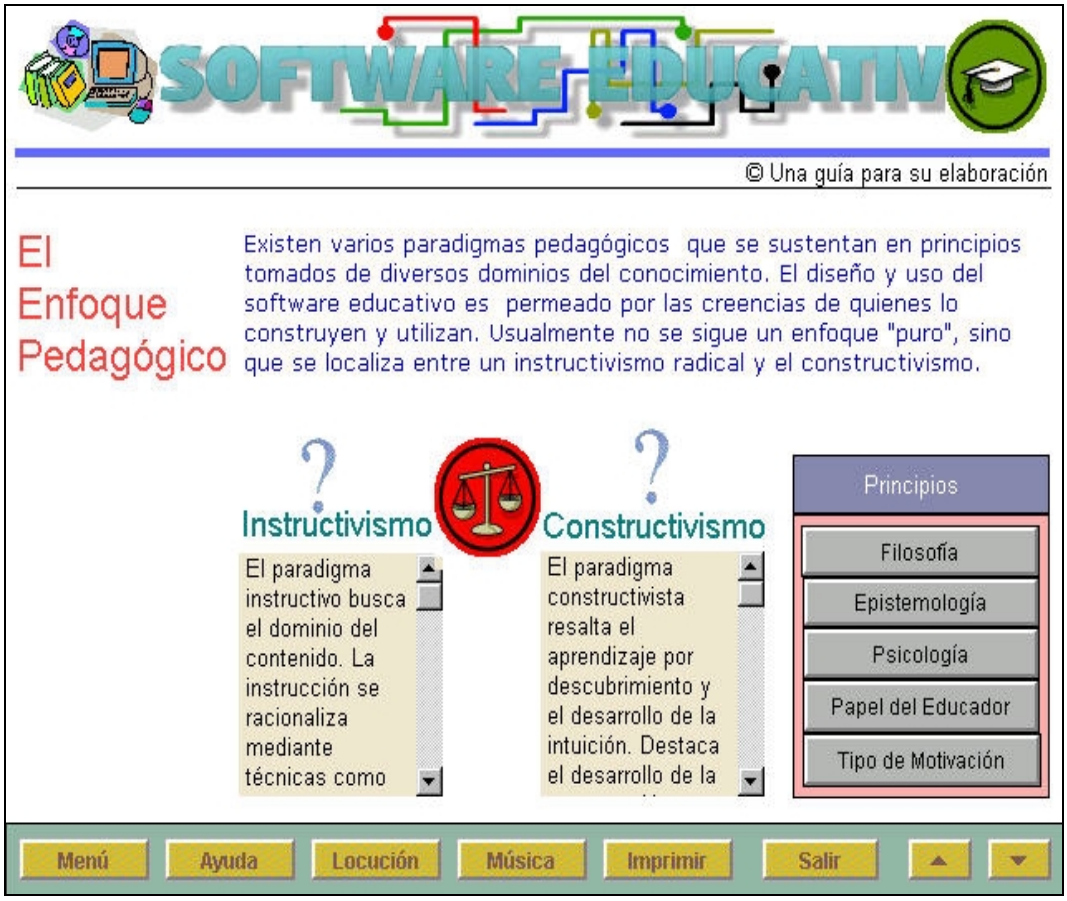

Figura No. 5: El enfoque pedagógico.

5. Enfoque pedagógico instructivo

El enfoque instructivo busca el dominio del contenido. La instrucción se racionaliza mediante técnicas como el establecimiento de la sucesión de los aprendizajes, la presentación y el reforzamiento mediante la información sobre las consecuencias. El software educativo, guiado por el paradigma Instructivo pretende, explícitamente, enseñar determinado material, mediante su división en partes más pequeñas e ir, gradualmente, presentándolas a los estudiantes. El alumno tiene la oportunidad de responder a preguntas, recibiendo información sobre la precisión de sus respuestas. El software se encarga de presentar los contenidos, de proporcionar la información sobre los resultados y de asignar tareas. Ejemplos de este software son los programas de ejercicio y reconocimiento de problemas aritméticos y los sistemas de aprendizaje dirigido.

La filosofía que guía el enfoque pedagógico instructivo se centra en los objetivos, separados del estudiante, a quien se le considera un recipiente pasivo de la instrucción. Los objetivos son extraídos por expertos de algún área del conocimiento particular y programados como una secuencia de aprendizajes estructurados de lo más simple a lo más complejo. La filosofía instruccionista parte de que el estudiante es un recipiente que debe ser llenado con aprendizajes. Por ello, el software educativo es diseñado de acuerdo con estrategías de enseñanza que promueven una instrucción muy directiva, con contenidos preestablecidos organizados en una jerarquía que cubran los objetivos propuestos. Los objetivos constituyen el elemento más importante por considerarse en el diseño del software.

La epistemología trata con teorías acerca de la naturaleza del conocimiento. En educación, una categoría importante es la teoría del conocimiento, o de la realidad, en que se sustenta el diseño del software. La perspectiva epistemológica objetivista considera que:

- El conocimiento existe separado del conocedor.
- La realidad existe independiente de los individuos.
- Los individuos adquieren conocimiento en una manera objetiva mediante los sentidos.
- El aprendizaje consiste de la adquisición de la verdad.
- El aprendizaje puede medirse de manera precisa mediante pruebas.

Desde la perspectiva de la Psicología Conductista, se considera que no son las construcciones internas del estudiante lo que realmente importa en el aprendizaje, sino el comportamiento directamente observable. El diseño de software educativo, de acuerdo con el conductismo, conlleva a generar un estímulo a través de algún ítem elemental en la pantalla, luego una respuesta es requerida, lo cual resulta en retroalimentación que, dependiendo de la exactitud de la respuesta, es seguida de un refuerzo positivo, cuando la respuesta sea la correcta, o, de lo contrario, una repetición del estímulo original,o una versión simplificada del mismo, y el ciclo comienza otra vez.

En el enfoque pedagógico instruccionista el educador ejerce el control del aprendizaje. El papel del educador consiste en trasmitir un contenido a los estudiantes. Se espera que todos los estudiantes aprendan de la misma manera. Además, en el enfoque instruccionista, la motivación para utilizar el software y aprender con ayuda del mismo, no es un aspecto relevante al diseño. Por tanto, se requiere que la motivación provenga del exterior al ambiente de aprendizaje proveído por el software.

6. Enfoque pedagógico constructivista

El paradigma constructivista resalta el aprendizaje por descubrimiento y el desarrollo de la intuición. Destaca el desarrollo de la comprensión, mediante la construcción activa del conocimiento así como la articulación y la manipulación de ideas. El software se construye con el fin de que proporcione ambientes para la exploración y el descubrimiento. Los estudiantes investigan el modelo introduciendo datos y observando los resultados que se producen, de modo que las ideas y los conceptos importantes comienzan a surgir naturalmente. El software proporciona ambientes en los que se articulan y exploran ideas mediante la creación de modelos, programas, planes y estructuras conceptuales. Ejemplos de este software son los programas de simulación y las aplicaciones multimediales en que se plantean situaciones ricas en interacción. Otros ejemplos de este software son los micromundos informáticos que permiten que los estudiantes expresen sus ideas y construyan soluciones para los problemas, modificando el estado de un objeto por medio de la programación.

La filosofía constructivista se enfoca en el conocimiento anterior del estudiante y en los modelos mentales. El ambiente de aprendizaje debe ser lo más rico posible para mejorar la habilidad de los estudiantes para construir conocimiento y resolver dificultades conceptuales. El software educativo constructivista se diseña de modo que no fomente una enseñanza directiva y, por el contrario, que estimule la exploración autodirigida y el aprendizaje por descubrimiento. El énfasis del diseño de software se orienta a construir ambientes de aprendizaje que puedan ser adaptados a las necesidades específicas de cada estudiante.

La perspectiva epistemológica constructivista permite múltiples perspectivas de un fenómeno, desde las cuales los estudiantes construyen su propio conocimiento. El constructivismo considera que:

- El conocimiento no existe afuera de las mentes de los seres humanos.
- Los seres humanos construyen el conocimiento subjetivamente con base en experiencias anteriores y procesos metacognocitivos o de reflexión.

- El aprendizaje consiste en la adquisición de constructos viables o estrategias.
- El aprendizaje a lo sumo puede ser estimado, mediante la observación y el diálogo.
- El conocimiento no existe fuera de las mentes de los seres humanos.

A diferencia de la psicología conductista, cuyo interés se enfoca en el comportamiento observable, la psicología cognoscitiva pone mayor énfasis en los estados mentales internos. Estos estados van desde simples proposiciones, pasando por esquemas, reglas, habilidades, habilidades generales, hasta los modelos mentales. Por ello, de acuerdo con la psicología cognoscitiva, el diseño del software requiere implementar o facilitar una gran variedad de estrategias de aprendizaje, dependiendo del tipo de conocimiento que se busque construir. Las estrategias de aprendizaje incluyen la deducción y la inducción.

El papel del docente en la utilización del software se reduce al mínimo necesario. El alumno es quien ejerce el control de su propio aprendizaje. El docente debe limitarse al papel de facilitador. El diseñador de software busca que las creencias del docente no sesguen los fines del software constructivista, que se espera sea principalmente utilizado por los alumnos.

Para el enfoque constructivista, la motivación se considera un aspecto muy importante en los procesos de aprendizaje. Por ello, en el diseño del software se busca proveer un contexto de aprendizaje significativo que soporte motivación intrínsica. El software multimedial facilita la motivación al ofrecer un ambiente de aprendizaje altamente interactivo, que integra música, voz, texto, imágenes, animaciones, y una interfaz amigable. También los micromundos, los juegos educativos y las simulaciones incorporan la motivación intrínsica. Las figuras No. 6 hasta la no.10 muestran algunas pantallas en que se contrastan la perspectiva instructivista con la constructivista.

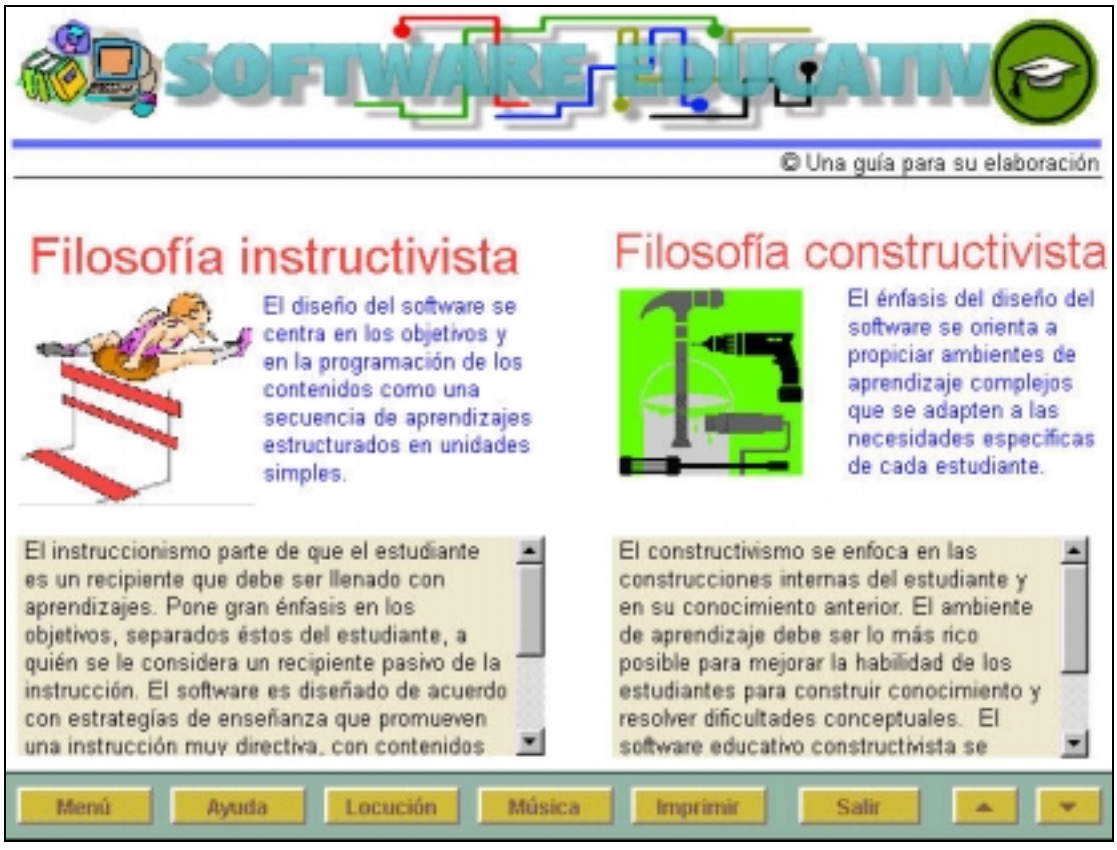

Figura No. 6 : Tipo de Filosofía

Figura No. 7 : Tipo de Psicología

Figura No. 8 : Tipo de educador.

No. 9 : Tipo de motivación.

Figura No. 10 : Diseño de pantallas.

7. Conclusiones

Consideramos que un aporte importante de esta investigación lo constituye la construcción de la tipología de categorías en que se contemplan todos aquellos aspectos significativos del desarrollo de software educativo, vistos desde la perspectiva pedagógica, didáctica y curricular, así como todos aquellos elementos técnicos pertinentes.

Cuando el diseño de software educativo es realizado exclusivamente por ingenieros en sistemas, o especialistas en computación, quienes por lo regular carecen de preparación formal en el campo de la educación, entonces sus concepciones personales acerca de la enseñanza y el aprendizaje se materializa en el producto final, lo que en algunos casos le acarrea debilidades en su función educativa. Ante tal situación, elaboramos una guía metodológica, con la que se persigue desarrollar un enfoque crítico, reflexivo, interdisciplinario, e integrador, mediante aquellos conceptos claves, fundamentales del desarrollo de software educativo. Conluimos que el paradigma pedagógico no es neutral debido a que permea al producto de software educativo.

La guía propuesta se desarroló para estudiantes de posgrado en Ciencias de la Computación, de la Universidad de Costa Rica, del curso Diseño de Interfaces de Usuario. Sin embargo, esta es bastante general y se anticipa su utilidad en contextos mas amplios. Concluimos que la guía es una herramienta valiosa, que sirve como canal de comunicación entre los distintos miembros del equipo desarrollador, quienes generalmente poseen muy diversos bagajes profesionales.

8. Bilbiografía

Bauza, G. (1997). El guión mutimedia. Madrid. Ediciones Anaya Multimedia, S.A.

Bermúdez, M. (1982). El Análisis del contenido: procedimiento y aplicaciones. Revista de Ciencias Sociales. No. 24, pp. 71-80. San José, C.R. Editorial de la Universidad de Costa Rica.

Bruner, J. (1996). *The Culture of Education*. Cambridge, Massachusetts & London, England, Harvard University Press.

Galvis, A.(1995). Fundamentos de tecnología educativa. Editorial de la Universidad Estatal a Distancia. San José, Costa Rica.

Gros, B. (1997). Diseños y Programas Educativos. Editorial Ariel S.A. Barcelona, España.

Kennedy (1997*). Design elements for interactive multimedia*. Astralian Journal of Educationa Technology. 13(1): 1-22.

MacIntyre, B. (1996). Mulimedia Systems. Springer-Verlag, 1996, 4:250-258.

Nodenot, T. (1998). *Educational software engineering: a methodology based on cooperative developments*. In: Tomek, I. (Ed). Computer Assisted Learning: Proceedings of 4th International Conference, ICCAL-98. Berlin: Springer-Verlag, 1998, pp. 529-541.

Reeves, T. (2000). *Evaluating What Really Matters in Computer-Based Education*, Dr. Tom Reeves. 2003. hhtp://www.educationau.edu.au/archives/cp/reeves.htm.

Seas, J. et al. (1998). Informática educativa: Ampliando escenarios para el aprendizaje. San José, C.R.
Editorial de la Universidad Estatal a Distancia.

Vargas, C. (2003). Software educativo: aporte de las universidades estatales costarricenses durante el periodo 1900-2001. Revista de Ingeniería. Universidad de Costa Rica. Vol. 12, No.2, pp.100-111.