Artículo de revisión

Comportamiento histórico de la enseñanza del diseño de software para la carrera Sistemas de Información en Salud

Historical Behavior of Software Design Teaching for Health Information Systems Career

Mayenny Linares Río [1*]

Milagros Aleas Díaz [2]

Juan A. Mena Lorenzo [2]

Darianna Cruz Marquez [1]

Deivy Rosales Quintana [1]

[1] Filial de tecnología de la Salud. Universidad de Ciencias Médicas de Pinar del Rio, Cuba.

[2] Universidad de Ciencias Pedagógicas "Rafael María de Mendive". Pinar del Rio, Cuba.

[*]Autor para la correspondencia: mayenny@infomed.sld.cu

RESUMEN

Introducción: El Sistema Nacional de Salud, por su misión, cobertura y características, requiere de un constante flujo informativo, por lo que necesita un personal especializado en el uso de tecnologías que les permita dirigir y gestionar la infraestructura necesaria para la efectiva toma de decisiones en el Sector.

Objetivo: Analizar el comportamiento histórico de la enseñanza del diseño de software en la carrera Sistemas de Información en Salud.

Métodos: Para la realización de la presente investigación de tipo descriptiva, se utilizaron como principales métodos los de nivel teórico: Histórico-lógico, Análisis-síntesis, Inducción-deducción y Enfoque sistémico, dentro de los de nivel empíricos el de Análisis documental.

Resultados: Se obtiene una valoración del análisis realizado a los documentos normativos que rigen el Proceso de Enseñanza y Aprendizaje del diseño de software en el mundo, América Latina y Cuba que puede ser utilizado como medio complementario de apoyo a la preparación metodológica de los profesores de la disciplina Informática.

Conclusiones: Existen elementos curriculares que deben ser atendidos con el fin de aumentar la calidad del proceso docente, a pesar de que se han alcanzado grandes avances cualitativos y cuantitativos en todos los aspectos de este proceso.

Palabras clave: diseño de software; sistemas de información en Salud

ABSTRACT

Introduction: The National Health System, its mission, coverage and characteristics, requires a constant flow of information in order to maintain a high level of knowledge of each of the activities undertaken at all levels for driving management processes and services. A specialist in the use of technologies is necessary in order to allow lead and manage effective decision making in the Health Sector infrastructure.

Objective: To analyze the historical behavior of Software Design teaching in Health Information Systems program.

Methods: as the main methods of theoretical level: Historical and logical, analysis-synthesis, induction, deduction and systemic approach, within the empirical level: documentary analysis.

Results: An assessment of the regulatory documents governing the teaching and learning of software design in the world, Latin America and Cuba, that can be used as a supplementary means of support for the methodological training of teachers of the discipline.

Conclusions: The current medical education in Cuba, has made great qualitative and quantitative progress in all aspects of the teaching process, but there are curricular elements that must be addressed in order to increase the quality of the teaching process.

Keywords: software design; health information systems.

Introducción

Probablemente la definición más formal de software es la atribuida al Institute of Electrical and Electronic Engineers (IEEE) en su estándar 729: "la suma total de los programas de cómputo, procedimientos, reglas [,] documentación y datos asociados que forman parte de las operaciones de un sistema de cómputo". (1)

Bajo esta definición el concepto de software va más allá de los programas de cómputo en sus distintas formas: código fuente, binario o código ejecutable, además de su documentación.

Pressman, en su texto Ingeniería de Software: un enfoque práctico, plantea que "el software de computadora es el producto que los ingenieros de software construyen y después mantienen en el largo plazo. Incluye los programas que se ejecutan dentro de una computadora de cualquier tamaño.

A la tecnología que comprende un proceso, un juego de métodos y un conjunto de herramientas, que permiten desarrollar un software se le designa como Ingeniería de Software." (2)

Según Fritz Bauer, quien fue el primero que definió la ingeniería de software, la especifica como: el establecimiento y uso de principios de ingeniería robustos, orientados a obtener económicamente software que sea fiable y funcione eficientemente sobre máquinas reales. (3)

El IEEE ha desarrollado una definición más completa, basado en el estudio de enfoques y define la ingeniería de software como:

La aplicación de un enfoque sistemático, disciplinado y cuantificable hacia el desarrollo, operación y mantenimiento del software; es decir, la aplicación de la ingeniería al software. (4)

Los autores consideran que cada una de estas definiciones de Ingeniería de software concuerdan en sus elementos esenciales que es un conjunto de pasos a seguir para lograr un software con la calidad requerida, estos pasos son definidos como paradigmas (modelos), al llevar a cabo cada uno de estos pasos se sigue un procedimiento a través de cual se aplican métodos que permiten la correcta realización de las tareas descritas, para lo cual se utilizan herramientas.

La creciente importancia del software en la vida diaria de las personas ha generado en los últimos años una fuerte y creciente demanda mundial de ingenieros de software que ayuden a producir software de calidad, en el plazo y dentro del presupuesto especificado.

En el sector de la salud, no se preparan informáticos que puedan ejercer como Ingenieros de software, pero si se imparte esta como asignatura a los estudiantes de la carrera Sistemas de Información en Salud (SIS), por ser estos futuros integrantes de grupos de desarrollo de software como gestores de la información y especialistas en los SIS, encargándose estos de determinar, diseñar, evaluar las interfaz gráfica, componentes, datos y relaciones de estos sistemas automatizados. (5)

Los métodos empleados en la enseñanza de la Ingeniería de software varían según las instituciones que la imparten y los conocimientos que cada educador tenga de estos, por lo que se hace necesario llegar a una unicidad en este tema por la importancia que tiene para el país, aportar software que optimicen los Servicios de Salud. Esto solo puede lograrse si se conoce la historia de esta enseñanza, de ahí que el objetivo de nuestro

trabajo sea: Analizar el comportamiento histórico de la enseñanza del diseño de software para los estudiantes de la carrera Sistemas de Información en Salud.

Métodos

La presente investigación de tipo descriptiva, se basa en el enfoque metodológico general dialéctico-materialista, el cual se asume como la base filosófica de los elementos tratados en el documento, permitiendo la selección de los métodos, procedimientos y técnicas de investigación, basados en la teoría de Carlos Álvarez de Zayas (1996), [6] que se utilizaron en el proceso de la investigación con el fin de cumplir el objetivo planteado.

Dentro de los métodos teóricos se emplearon:

- Histórico y lógico: se utilizó para el estudio de las etapas por la que ha transitado el proceso de enseñanza aprendizaje de la Ingeniería de software en la carrera Sistemas de Información en Salud, así como en su evolución, desarrollo y perfeccionamiento, que permitió investigar sus tendencias y regularidades.
- Análisis y síntesis: se aplicó durante todo el proceso investigativo para llegar al conocimiento multilateral del proceso de enseñanza aprendizaje en la asignatura Ingeniería de Software en la carrera Sistemas de Información en Salud, delimitar los elementos esenciales que lo conforman así como los nexos existentes entre ellos y sus características más generales.
- Inducción y deducción (como procedimientos): se empleó desde la recogida del material empírico para obtener conclusiones generalizadoras, que unido al estudio teórico permitió la elaboración del documento.

Dentro de los métodos empíricos se tienen:

- Análisis documental: permitió el estudio de los documentos normativos que avalan la enseñanza de la asignatura Ingeniería de Software, así como artículos asociados a este tema.

Resultados

Periodización de la enseñanza de la Ingeniería de software

El aprendizaje de la Ingeniería del Software comienza en la década de 1980 a 1990, que según Shepperd, estuvo caracterizada por el aprendizaje autónomo de la Ingeniería de Software de forma empírica, lo que marcó un hito en el crecimiento de la industria, considerándose esta como una primera etapa. [7]

Dentro de los principales aportes en la enseñanza de la Ingeniería de software en esta etapa podemos encontrar:

En el 1980-1995 se comienza la formación en Sistemas de Información en Salud en la Escuela Nacional de Salud Pública de la Universidad de Antioquia, Colombia. En el que se centra la formación para el tratamiento de los Sistemas de Información en Salud.

En Cuba, en 1961 se comienza el curso de Auxiliares de Estadística, en la Escuela Sanitaria pero no es hasta el año 1987, se formaron en los institutos politécnicos de la salud dos especialidades técnicas dirigidas a satisfacer las necesidades en estas áreas de la actividad en el contexto de aquella etapa, éstas eran: Técnico en Estadísticas de Salud y Técnico en Bibliotecología Médica. Ambas comprendían el desarrollo de habilidades en las

herramientas informáticas que existían para la época, pero no en Ingeniería de software, de ahí que la incorporación de técnicos comunes a los servicios de salud no satisfizo la demanda de estos recursos humanos y si bien estaban formados en contenidos generales, no disponían de preparación alguna en materias de salud y automatizaciones.[8-10]

Esta etapa de forma general estuvo caracterizada por la mejora continua de los ciclos de vida para el desarrollo del software pero no se hace referencia alguna a su enseñanza en la Ingeniería de software, en el mundo.

Se puede hablar entonces de una segunda etapa (1991-2000) caracterizada por el comienzo de la enseñanza curricular de la Ingeniería de software, el continuo perfeccionamiento de su ciclo de vida y procederes didácticos en función de la enseñanza de estas metodologías.

En 1998, la Sociedad de Profesionales del Instituto de Instituto de Ingenieros Eléctricos y Electrónicos (Institute of Electrical and Eletronics Engineers.Computer Society. (IEEE-CS)) y la Asociación de los Sistemas Informáticos (Association for Computing Machinery (ACM)) establecieron un grupo de trabajo conjunto para crear una nueva versión de las guías curriculares para programas de pregrado en computación, debido a que desde comienzos de 1990 esta área presentaba un desarrollo continuo y se estaban conformando disciplinas dependientes e independientes de la misma. [11]

En 2001, el proyecto Computing Curricula 2001 (CC2001) elaboró cuatro volúmenes de guías curriculares para disciplinas relacionadas con la computación: 1) Ciencias Computacionales, 2) Sistemas de Información, 3) Ingeniería en Computación y 4) Ingeniería de Software, a los que posteriormente se agregó otro relacionando la guía curricular para la disciplina de Tecnologías de la Información. [12]

El volumen Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering 2004 (SE2004), se convirtió rápidamente en la guía curricular para los programas desarrollados por IEEE-CS y ACM, a la vez que sirvió de base para que las instituciones académicas y organismos de acreditación estructuraran la formación de pregrado. [13]

A partir de la concepción de SE2004, diferentes universidades del mundo han utilizado la guía para crear nuevas carreras de pregrado en Ingeniería de Software, y para adaptar y comparar los planes de estudio existentes. [14-17] Latinoamérica se ha movido a un ritmo lento en este sentido, y no es posible conseguir publicaciones que presenten adopciones del SE2004; pero, en 2011, en la Conferencia Latinoamericana de Informática (CLEI) se creó el Workshop en Nomenclatura y Acreditación en Programas de Computación. En la tercera edición se presenta un trabajo que estudia la evolución de los currículos de Latinoamérica. [18]

Dentro de las carreras de grado orientadas al software, la informática, la computación o a los sistemas, en las principales universidades del Cono Sur existen en los planes de estudio materias específicas dedicadas a la ingeniería de Software, como Análisis de Sistemas, Diseño de Sistemas, Ingeniería de Software, Calidad del Software y otras. En algunas también existen acreditaciones nacionales a nivel de carrera, que brindan una estructura homogénea. Por otro lado, en los estadios de formación universitaria la Ingeniería de Software se suele trabajar la teoría y la práctica de una determinada metodología, y actualmente la más extendida es la propuesta por Rumbaugh, Jacobson y Booch, conocida

como Proceso Unificado de Desarrollo, acompañada del Unified Modeling Language (UML), que propagó rápidamente sus conceptos y sentó las bases para una nueva forma de comprender un paradigma particular para desarrollar software: la Programación Orientada por Objetos (POO). [19]

Un ejemplo de países que aportan a esta docencia son:

Estonia que imparte programación a los adolescentes de primer grado; [20] en Estados Unidos se crearon escuelas secundarias, como Academy for Software Engineering (AFSE), para orientar a los estudiantes en carreras relacionadas con las Ciencias Computacionales o la Ingeniería de Software, [21] y en Uruguay se creó la opción de Bachillerato Tecnológico en Informática, para introducir a los estudiantes en temas relacionados con la computación, como la programación, los sistemas operativos y las bases de datos.

En el caso de Facultad de Informática Universidad Politécnica de Madrid se han adoptado un conjunto de estrategias que pretenden: -Incorporar, dentro de los planes y programas de estudio, asignaturas con programas abiertos que permitan introducir tópicos selectos correspondientes al desarrollo de las Ciencias de la Computación. -Contemplar, dentro de los planes y programas de estudio, un conjunto de materias electivas, con programas que permitan la especialización y actualización en las diversas líneas del conocimiento de las Ciencias de la Computación. -Promover la participación de los profesores en diversas actividades de actualización disciplinaria, profesional y docente. Tales como su incorporación a programas de especialización, de diplomado y de certificación profesional, entre otros. [22]

Existe un grupo grande de universidades americanas, que contemplan dentro de su diseño curricular, la formación en Ingenierías, técnicos universitarios, maestrías y especialidades asociadas a la Computación, la Ingeniería de sistemas y de software como es el caso de Argentina (7 universidades), Chile (29 centros de formación en técnico de nivel superior, ingenierías y licenciaturas en análisis de sistema, diseño de sistemas y programación), Uruguay (3 universidades), Brasil (11 universidades), Venezuela (6 universidades) además cuenta con el Grupo de Investigación en Ingeniería del Software (INGESOF) realiza sus procesos en las áreas de: Ingeniería de Requisitos, Gestión de Proyectos en Sistemas de Información. Gestión de la Calidad del Software. Diseño de Sistemas de Información. Programación y Pruebas de Sistemas de Información. Ingeniería del Software Libre y Sistemas de Información Gerencial, Ecuador (5 universidades), Perú (8 universidades), Bolivia (5 universidades), Colombia (7 universidades). En Centroamérica podemos mencionar a Costa Rica (4 universidades), Guatemala (4 universidades), México (5 Universidades y 1 Instituto politécnico central) y Panamá (3 universidades). [23]

Los principales aportes en el área de la enseñanza de la Ingeniería de software en esta etapa fueron:

Wankat, P. C. & Oreovicz, F.S., (1993): Presentaron un conjunto de estrategias didácticas para ingeniería en general, en la enseñanza incluyen las clases magistrales y los proyectos prácticos, basados en juegos. [24]

Marqués P. (1995). Plantea un ciclo de desarrollo para software educativo de programas en diez fases y hace una descripción detallada de las actividades y recursos fundamentales para cada una de las etapas. [25]

En el 1995 se comienza la formación en Gerencia de Sistemas de Información en Salud en la Escuela Nacional de Salud Pública de la Universidad de Antioquia, Colombia. La formación del Gerente de Sistemas de Información está atravesada por los siguientes ejes que guían el proceso de enseñanza - aprendizaje: La Salud Pública, La Seguridad Social, La Informática, Los Sistemas, La Investigación Científica y La Gerencia, pero dentro de sus funciones no está la de desarrollar software. [26]

Como principales tendencias en esta etapa se pueden encontrar: las mejoras en el proceso evolutivo del ciclo de vida del software, el perfeccionamiento de estos ciclos por parte de los docentes para una mejor enseñanza del mismo y se comienza a utilizar el aprendizaje por proyecto pero solo en forma de juegos como parte del uso de métodos activos pero no se evidencia la utilización de las habilidades investigativas para este aprendizaje por proyecto. Aún en esta etapa no se consideraba su enseñanza en el sector de la salud.

Se puede mencionar una tercera etapa (del 2000 al presente) caracterizada por el uso de Metodologías ligeras para el diseño del software y la explotación del uso de las redes y la tecnología para el tratamiento de estos contenidos en clases a través del uso de medio de enseñanza tecnológicos.

Principales aportes en la enseñanza de la Ingeniería de software en esta etapa en el mundo:

Mariño, M. et al. (2001). Desarrollaron un software interactivo orientado a la enseñanza de un método de programación por camino crítico. [27]

Baker A. et al. (2005). Presenta una simulación para la especificación de requisitos en la ingeniería de software, con el fin de entregar una aplicación de software con juegos de cartas, usado para ejemplificar el proceso de desarrollo de software. [28]

Gayo, J.E. et al. (2006): Describen una experiencia, desarrollada en una asignatura utilizando herramientas colaborativas en el desarrollo de software libre utilizando la herramientas Sourceforgo, creado un proyecto común entre todos los estudiantes, con el objetivo de facilitar un aprendizaje basado en proyectos.[29]

Guitart, I. et al. (2006): Estructuran y hacen una elección del modelo de evaluación como caso práctico para asignaturas de ingeniería del software, partiendo de un planteamiento inicial del proceso de evaluación, que debe de ser coherente con los objetivos de aprendizaje. [30]

Díaz M. et al. (2006): Estructuraron elementos teórico-metodológicos con el fin de guiar la elaboración de software con fines educativos, haciendo referencia a una guía metodológica, con la que se persigue propiciar un enfoque crítico, reflexivo, interdisciplinario e integrador, de los conceptos clave en el desarrollo de software educativo. [31]

Taran, G. (2007): La Universidad de Carnegie Mellon desarrolló un juego para la enseñanza de los conceptos básicos de la gestión de riesgos en el proceso de desarrollo de software, para el programa de Ingeniería, con el objetivo de afianzar la toma de decisiones.[32]

La Ruiz, F. (2007): Afirma la importancia y el papel que juega la ingeniería del software en la informática y argumenta que es necesaria una reforma en los contenidos en la manera de enseñar y aprender.[33]

Zapata, C.M. & Awad, A. (2006): Proponen un juego para la ingeniería de requisitos a partir de roles y orientado al desarrollo de una aplicación basada en simular aspectos en el cumplimiento de requisitos, en un entorno competitivo. [34]

Zapata, C.M. & Duarte, M. (2007): Crearon un juego de la consistencia como herramienta didáctica para usar en las aulas de clase, que le permite al estudiante afianzar, estructurar, analizar los conocimientos sobre modelado, métodos de desarrollo de software, trabajo en equipo y comunicación en la ingeniería de software.[35]

Zapata, C.M. & Carmona, N. (2010): Estructuran un modelo de diálogo para la ingeniería de software basado en requisitos, esta es una técnica para recolección, procesamiento y especificación de los requisitos de los interesa dos en el desarrollo de una aplicación de software con base a preguntas durante una entrevista, aplicado a un caso de estudio.[36]

Cuervo, M. et al. (2010): Presentan la realización de herramienta HELER (Herramienta Libre para la Especificación de Requisitos) 5 módulos: proyecto, stakeholder, actores, casos de uso, y requisitos. [37]

Monsalve, E. Werneck, V. & Leite, J. (2010) Ecuador. Han efectuado estudios sobre la importancia de los procesos de captura, definición, validación e implementación en la ingeniería de requisitos y se ha encontrado que una de las debilidades del desarrollo de software se encuentra comúnmente en el análisis de un problema; para esto se necesita buscar aproximaciones concretas aplicando los modelos que permitan mejorar la consistencia, la completitud, viabilidad en la comprensión del sistema y la gestión de los cambios. [38]

Anaya, R. (2012): Propone una visión de la enseñanza en la ingeniería de software como apoyo al mejoramiento de las empresas con un fin: la integración de factores técnicos, gerenciales y organizacionales permitiendo mejorar la práctica del desarrollo en las organizaciones como principios y estrategias.[39]

Existen varios problemas para la enseñanza en las aulas de las técnicas de entrevistas para la definición de requisitos. De acuerdo a Oliveros A. et al. (2012), una de ellas es la dificultad de ejecutar una práctica real: las técnicas son meramente descriptas y en la mejor de las situaciones practicadas en un caso simplificado. Para abordar el problema, tuvo en cuenta cada uno de los pasos identificados por. Bandura, A. & McClelland, D. C. (1977) en el proceso de aprendizaje por observación quien desarrolló la experiencia basado en, las cuatro fases de aprendizaje usando la técnica de requisitos: entrevista a dos grupos de estudiantes y obtuvo muy buenos resultados. [40]

Godoy, L. A. (2013): Hace una revisión de las contribuciones de Roger C. Schanik y propone desarrollar actividades en un ambiente virtual. Este autor afirma que el estudiante solo aprende a través de intentar cosas que requieren conocimiento y no escuchando narrar a un docente reglas de oficio. [41]

González, A.M. et al. (2013): Proponen el diseño y aplicación para la enseñanza de ingeniería de software. “Ingeniería de Información” realizado en tres fases:

1. Video casero sobre catástrofes de software.
2. Los empresarios en el aula.
3. Concurso para la asimilación de conceptos básicos de la Ingeniería de Requisitos. (Montoya-Suárez L.M., Pulgarín-Mejía E. 2013) [42]

En Cuba la enseñanza de la Ingeniería de software se desarrolla en esta etapa en Las Universidades de todo el país para la formación de Ingenieros Informáticos (Universidad de las Ciencias Informáticas, Facultad de matemática de la Universidad de la Habana, Villa Clara entre otras e Instituto Superior Politécnico José Antonio Echeverría). También se imparte en las Universidades de Ciencias Médicas para la formación en la Carrera Licenciatura en Tecnología de la Salud, pero solo como asignatura básica que los prepara para desempeñarse como miembros de un equipo de desarrollo de software.

Durante el curso escolar 2002-2003, se inició una experiencia para desarrollar un Técnico en Registros, Información e Informática de Salud, a partir de dos fuentes de ingreso de la educación general: 9no. y 12mo. grado; estos recibían la Ingeniería de Software como materia pero dividida en dos asignaturas en las que se evidenciaban que los contenidos referidos al diseño de software se trataban en muy pocas horas clases, rompiendo la sistematicidad necesaria en la adquisición de conocimientos, sin explotar las habilidades investigativas, la interdisciplinariedad y el aprendizaje por proyectos [(43)]

Según el Plan de estudios de la carrera SIS, 2010 [(44)] a partir del 2009, teniendo en cuenta que, el Sistema Nacional de Salud (SNS), por su misión, cobertura y características, así como su enfoque estratégico y programático, requiere de un personal especializado en el uso de tecnologías caracterizadas en los Sistemas de Información de Salud (SIS) y Tecnologías de la Información y las Comunicaciones (TIC), que les permita dirigir y gestionar la infraestructura necesaria para la efectiva toma de decisiones en el Sector de la Salud, se hizo un último rediseño para los planes D (D, D1, D2). Estos recibían la Ingeniería de software ya unificada en un programa pero mantenía las características de los anteriores en cuanto a su enseñanza. Estos planes de estudios debido a la colaboración que se lleva a cabo por el Ministerio de Salud Pública, se lleva a cabo en Venezuela.[(45)]

Esta etapa estuvo caracterizada por el uso de metodologías ligeras para el desarrollo del software, así como de medios de enseñanza tecnológicos para su enseñanza, comienzan los avances en cuanto a que el profesor deje ser un facilitador del conocimiento sino que el alumno llegue a construir su propio aprendizaje a través del uso de estos medios de enseñanza ya mencionados; pero aún enfocado a la explicación de los contenidos teóricos, de forma expositiva por parte del profesor y a la realización de proyectos de clase por parte de los estudiantes, que distan de las características y de la magnitud de una situación ubicada en un contexto real, ya que a pesar de ser estos estudiantes de una carrera de perfil amplio, deben participar en equipos de trabajo en los que no solo gestionen la información sino también que aporten elementos al diseño del sistema, pero no favorece el desarrollo de habilidades de planeación y diseño en el futuro profesional ya que no se explotan las habilidades investigativas para el estudio del negocio a profundidad y la obtención de requisitos. En el mundo se concibe la formación de un Ingeniero en el área de la Ingeniería de software que presta servicios a todos los sectores en su ubicación laboral, pero en el caso de la Salud se ve más orientado a la formación de especialistas en equipos médicos, no así en Cuba, pues el Ingeniero, tiene campo en el Sector de la Salud en Grupos de Informática y el Centro de Desarrollo de Salud Pública (CEDISAP), que son los encargados de las soluciones complejas.

Conclusiones

De forma general se analizó el comportamiento histórico de la enseñanza del diseño de software, en especial para los estudiantes de la carrera Sistemas de Información en Salud, de lo cual se puede concluir como principales tendencias de la enseñanza de la Ingeniería del software:

- Las direcciones en las que evoluciona la ingeniería del software hoy en día pueden agruparse de la siguiente manera: Metodologías ágiles, Experimentación, Desarrollo dirigido por modelos y Líneas de productos software, en lugar de productos individuales.[46]
- La actividad laboral-investigativa se diseña principalmente para ejecutarse en proyectos de curso docentes y en insuficiente vínculo con la industria de software, con la sistematicidad que se requiere, en los que no se tiene en cuenta la interdisciplinariedad y la relación entre las asignaturas que componen la disciplina Informática para la formación del profesional, sin que se explote el desarrollo de habilidades investigativas y/o el aprendizaje por proyectos.

Referencias

1. The Institute of Electrical and Eletronics Engineers. EE UU: IEEE Standard Glossary of Software Engineering Terminology; 1990. 83 p.
2. Pressman R. Ingeniería de Software. Un enfoque práctico. 5ª Ed. EE UU: McGraw Hill; 2001. 30-34 p.
3. Chavarriaga J, LIDIS G. La Ingeniería de Software como profesión tecnológica: Implicaciones en la investigación. Métodos de Investigación y Fundamentos Filosóficos en Ingeniería del Software y Sistemas de Información. Madrid: Universidad Rey Juan Carlos; 2003. 162-78 p.
4. Lehman MM. A further model of coherent programming processes. Proceedings of the IEEE Software Process Workshop; 1984.
5. Vidal Ledo M, Rodríguez Díaz A, Delgado Ramos A, Manrique García E. Estrategia educativa para la formación de recursos humanos en la gestión de información en salud. Rev cuban salud púb [Internet]. 2009 Jul-Sep [citado 22 Nov 2016];35(3):[aprox. 10 pantallas]. Disponible en: http://scielo.sld.cu/scielo.php?script=sci_arttext&pid=S0864-34662009000300011&lng=es.
6. Álvarez de Zayas C, Sierra Lombardía V. Metodología de la investigación científica. La Habana: Editorial Ciencias Sociales; 1995. 80 p.
7. Shepperd M. Empirically-based Software Engineering. European Journal for the Informatics Professional. 2003 Aug;4(4):37-41.
8. Ríos NE, Fernández RM, Jorge LR. Los registros médicos en Cuba. Rev cuban salud púb [Internet]. 2005 Sep.-Dic [citado 22 Nov 2016];31(4):[aprox. 14 pantallas]. Disponible en: http://scielo.sld.cu/scielo.php?script=sci_arttext&pid=S0864-34662005000400013.
9. Montero García1 I, Sánchez Sánchez C, Manso Fernández E, Llano Gil EA, Dávila Expósito N. Gestión de la información en los servicios de salud. Gaceta Médica Espirituana [Internet]. 2009 [citado28 Nov 2019];11(3):[aprox. 6 pantallas]. Disponible en: http://bvs.sld.cu/revistas/gme/pub/vol.11.%283%29_12/p12.html.

10. Vidal Ledo M. Evaluación del diseño curricular del perfil de Gestión de Información en Salud de la carrera de Tecnología de la Salud. BVS [Internet]. 2008 [citado 26 Nov 2019]:[aprox. 15 pantallas]. Disponible en: http://bvs.sld.cu/revistas/ems/vol22_1_08/ems06108.htm.
11. Mishra A, Yazici A. An Assessment of the Software Engineering Curriculum in Turkish Universities: IEEE/ACM Guidelines Perspective. Croatian Journal of Education. 2011;13(1):188-219.
12. ACM, AIS, IEEE-CS. Computing Curricula 2005 – The Overview Report covering undergraduate degree programs in Computer Engineering, Computer Science, Information Systems, Information Technology, Software Engineering [Internet]. EE UU: Association for Computing Machinery and IEEE Computer Society; 2005 [cited 2019 Dic 5]. 62 p. Available en: https://www.acm.org/binaries/content/assets/education/curricula-recommendations/cc2005-march06final.pdf.
13. IEEE-CS. Computer Engineering 2004 – Curriculum Guidelines for Undergraduate Degree Programs in Computer Engineering. EE UU: Association for Computing Machinery; 2004.
14. Ramakrishnan S. Accreditation of Monash University Software Engineering (MUSE) Program. International Journal of Issues in Informing Science and Information Technology. 2007;4:73-89.
15. Ding E. Research and practice on software engineering undergraduate curriculum NJU-SEC. In: IEEE-CS Conference on Software Engineering Education and Training; 2011. p. 492-6.
16. Cuadros Vargas E. Evolution of the Computing Curricula for Computer Science in Latin America; 2013. p. 1-10.
17. Baum G, Artopoulos A. Libro Blanco de la Prospectiva TIC – Proyecto 2020. Buenos Aires: Ministerio de Ciencia, Tecnología e Innovación Productiva.; 2008. p.51-79.
18. Rositas J, Torres G. Diseño de planes educativos bajo un enfoque de competencias. México: Editorial Trillas; 2011.
19. Rumbagh J. Over the waterfall and into the Whirlpool. Journal of Object-Oriented Programming. 1992;5(2): 23-6.
20. Ding E. Research and practice on software engineering undergraduate curriculu. In: Conference on Software Engineering Education and Training; 2011. p. 492-6.
21. Thiruvathukal G. The Education Issue. Computing Now; 2013.
22. Ardis M, Henderson P. Software Engineering Education (SEEd). Software Engineering Notes. 2012;37(3):8-9.
23. Baum G, Artopoulos A. Libro Blanco de la Prospectiva TIC – Proyecto 2020. Buenos Aires: Ministerio de Ciencia, Tecnología e Innovación Productiva; 2008. p. 46.
24. Wankat PC, Oreovicz FS. Teaching engineering. New York :McGraw-Hill; 1993.
25. Marqués P. Metodología para la elaboración de software educativo. Barcelona: Estel; 1995.
26. Montoya Suárez LM, Pulgarín Mejía E. Colombia. Enseñanza en la Ingeniería de Software: Aproximación a un estado del arte. Revista Lámpsakos. 2013 Jul-Dic;10:76-91.
27. Mariño SI, López MV, Golobisky MF. Un software interactivo orientado a la enseñanza del Método de Programación por Camino Crítico. VII Congreso Argentino de Ciencias de la Computación; 2001.
28. Baker A, Navarro E, Van Der Hoek A. An experimental card game for teaching software engineering processes. Journal of System Software. 2005;75(1):3–16.

29. Labra Gayo JE, Fernández Lanvin D, Calvo Salvador J, Cernuda del Río A. Una experiencia de aprendizaje basado en proyectos utilizando herramientas colaborativas de desarrollo de software libre. XII Jornadas de Enseñanza Universitaria de la Informática [Internet]. 2006 [citado 12 Ago 2015]:[aprox. 8 p.]. Disponible en: http://di002.edv.uniovi.es/~labra/FTP/Papers/LabraJenui06.pdf.
30. Guitart I, Rodríguez ME, Cabot J, Serra M. Elección del modelo de evaluación: caso práctico para asignaturas de ingeniería del software. Actas las XII Jornadas Enseñanza Universitaria Informática; 2006. p.191–8.
31. Díaz-Antón MG, Pérez M, Grimmán A, Mendoza L. Propuesta de una metodología de desarrollo de software educativo bajo un enfoque de calidad sistémica. Caracas, Venezuela: Universidad Simón Bolívar; 2006.
32. Taran G. Using games in software engineering education to teach risk management. In: Software Engineering Education & Training; 2007. p. 211–20.
33. La Ruiz F. Enseñanza de la Ingeniería del Software en el marco del Espacio Europeo de Educación Superior. Ponencia en el II Congreso Español de Informática; 2007. p.1–20.
34. Zapata Jaramillo CM, Gelbukh A, Arango F, Hernández A, Zechinelli JL. UN-Lencep: Obtención automática de diagramas UML a partir de un lenguaje controlado. Dyna [Internet]. 2007 [citado 28 Nov 2019];74(153):223-36. Disponible en: https://www.redalyc.org/pdf/496/49615309.pdf.
35. Zapata Jaramillo CM, Awad Aubad G. Requirements Game: Teaching Software Project Management. CLEI Electronic Journal [Internet]. 2007 [cited 2019 Nov 28];10(1):[aprox. 11 p.]. Available from: https://pdfs.semanticscholar.org/b795/8cf1eb953c1f988f695320b6bdd07cfbdbf6.pdf.
36. Zapata CM, Carmona N. Un modelo de diálogo para la educación de requisitos de software. Dyna [Internet]. 2010 [citado 28 Nov 2019];77(164):209–19. Disponible en: https://revistas.unal.edu.co/index.php/dyna/article/view/25591/26062.
37. Callejas Cuervo M, Castillo Estupiñán LY, Fernández Álvarez RM. Heler: Una herramienta para la ingeniería de requisitos automatizada. Entramado [Internet]. 2010 Jul-Dic [citado 28 Nov 2019];6(2):184-200. Disponible en: https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&cad=rja&uact=8&ved=2ahUKEwjT_5-N8qjmAhUww1kKHV2NC7kQFjAAegQIBxAC&url=https%3A%2F%2Fdialnet.unirioja.es%2Fdescarga%2Farticulo%2F3644325.pdf&usg=AOvVaw1v79po9Suw77rxrgjaacT6.
38. Monsalve E, Werneck V, Leite J. Evolución de un Juego Educacional de Ingeniería de Software a través de Técnicas de Elicitación de Requisitos. En: Proceedings of XIII Workshop on Requirements Engineering. Cuenca, Ecuador; 2010. P. 12–23.
39. Anaya R. Una visión de la enseñanza de la ingeniería de software como apoyo al mejoramiento de las empresas de software. Revista Universidad Eafit. 2012;42(141):60–76.
40. Oliveros A, Zuñiga JA, Wehbe R, Rojo SV, Sardi F. Enseñanza de elicitación de requerimientos. XVIII Congreso Argentino de Ciencias de la Computación; 2012.
41. Godoy LA. Una revisión del programa de investigación sobre aprendizaje activo en un ambiente simulado desde la perspectiva de la educación en ingeniería. LatinAmerican and Caribbean Journal on Engineering Education. 2013;3(2).
42. Montoya Suárez LM, Pulgarín Mejía E. Enseñanza en la Ingeniería de Software: aproximación a un estado del arte. Revista Lámpsakos. 2013 Jul-Dic [citado 2 Dic 2019];(10):76-91. Disponible en: https://www.funlam.edu.co/revistas/index.php/lampsakos/article/download/1338/1216.

43. Universidad de Pinar del Río, Facultad de Ciencias Técnicas. Plan de estudio de la carrera Ingeniería en Informática. 2012.
44. Vidal Ledo M. Diseño curricular del perfil de Información, Informática y Estadística de Salud de la carrera de Tecnología de la Salud [tesis maestía]. La Habana: Centro de Enseñanza de Cibernética Aplicado a la Medicina (CECAM); 2005.
45. Hislop GW. Software Engineering Education: Past, Present, and Future. In: Ellis H, Demurjian S, Naveda JF. Software Engineering – Effective Teaching and Learning Approaches and Practices. EE UU:IGI Global; 2009. p.1-13.
46. Ciudad Ricardo FA, Ruiz Jhones A. El proceso de enseñanza – aprendizaje de la disciplina Ingeniería y Gestión de Software desde los proyectos industriales. Revista Pedagogía Universitaria [Internet]. 2012 [citado 2 Dic 2019];17(3):18-44. Disponible en: http://cvi.mes.edu.cu/peduniv/index.php/peduniv/article/download/29/28.