La enseñanza de ingeniería de software para hacer que los estudiantes estén listos para el mundo real. UN PREÁMBULO A LA DIDÁCTICA UNIFICADA DEL DISEÑO DE SOFTWARE

Fecha de recepción: septiembre 30 de 2008
Fecha de aprobación: noviembre 21 de 2008

Por: Robinson Jiménez
Ingeniero de Sistemas
Docente Programa de Ingeniería de Sistemas,
Universidad Mariana - Colombia
rjimenez@umariana.edu.co

Álvaro Martínez
Ingeniero de Sistemas
Especialista en Docencia Universitaria
Docente Programa de Ingeniería de Sistemas,
Universidad Mariana - Colombia
amartinez@umariana.edu.co

Jesús Insuasty
Ingeniero de Sistemas
Especialista en Docencia Universitaria
Docente Departamento de Sistemas,
Universidad de Nariño - Colombia
insuasty@udenar.edu

RESUMEN

El presente artículo pretende dar una primera explicación de las bases generales que soportarán la didáctica del diseño de software propuesta, a saber: desarrollo basado en equipos, definición de problemas reales, procesos de desarrollo administrados, utilización de herramientas de vanguardia, aprendizaje con base en experiencias de expertos, presentación profesional de productos y aprender a aprender; que les permita a los actores del proceso enseñanza – aprendizaje – evaluación acercarse lo más posible al ambiente profesional y laboral.

PALABRAS CLAVE

Didáctica, diseño de software

ABSTRACT

This article intends to provide a preliminary explanation of the general basis that support the teaching of software design a methodology for embedded software**.** development and platform-based design, such as computer software development ,the Managed Model Development Process, the use of cutting edge technology , experiences of subject matter experts to facilitate learning , professional presentation of products and learn to understand, allowing to the actors of the teaching-learning process and assessment play an important role Into a professional and working environmental.

KEY WORDS

Didactic, software design

INTRODUCCIÓN

Muchos graduados requieren una gran cantidad de entrenamiento en las competencias laborales antes de presentarse a ejercer sus profesiones debido a que muchos de ellos no han tenido la oportunidad de desarrollar software comercial durante su período de estudio en la universidad o institución de educación superior; haciendo un análisis sobre los contenidos curriculares ofertados para el área de diseño de software (o ingeniería de software), es evidente que a nivel internacional su contenido trata de ser homogenizado bajo los lineamientos propuestos por la

ACM Computing Curricula a lo cual se debe entender que se trabaja de alguna forma "estándar" dentro de las cátedras si se puede llamar así.

Para cambiar esta situación, se ha planteado la propuesta teórico-práctica acerca de la construcción de la Didáctica Unificada del Diseño de Software que estará cimentada sobre las siguientes directrices:

- Desarrollo basado en equipos
- Definición de problemas de los clientes del mundo real
- Procesos de desarrollo administrados
- Utilización de herramientas profesionales de vanguardia para el desarrollo de software
- Aprendizaje basado en expertos de la industria del software
- Presentaciones profesionales de los productos construidos
- Énfasis en la solución de problemas y aprender a aprender.
- Mejoramiento significativo de la creatividad soportada en los procesos de abstracción

Sin lugar a duda es importante tener presente que cada universidad o institución de educación superior, y específicamente, cada programa de ingeniería en computación, informática o sistemas tiene un perfil profesional que está latente dentro de la formación de sus estudiantes, lo que es difícil negar es que en la mayoría de los casos, la competencia laboral en el campo del diseño y desarrollo de software es la que prevalece sobre las demás.

El presente artículo pretende explicar las bases generales donde la nueva propuesta didáctica para la enseñanza del diseño de software se encuentra soportada.

Desarrollo basado en equipo

El primer ítem de acción es la formación de equipos. Tres o cuatro estudiantes estarán localizados en un equipo, es importante reconocer que para muchos estudiantes es difícil trabajar en equipo y sobre todo si no ha sido a voluntad, pertenecer a uno de ellos, es menester hacer de esta acción una experiencia constructiva y participativa tratando en lo posible de no ser un obstáculo de rendimiento.

Es importante que los estudiantes trabajen en equipo puesto que en la vida real, el software de misión crítica por lo general no es construido por un solo individuo, hoy por hoy la realidad sobre el diseño y construcción profesional de software es muy distinta a la de hace algunas décadas.

Pero, ¿cuáles serían las estrategias para la conformación de los equipos de trabajo en el diseño y la construcción de software? Quizás este sea uno de los interrogantes más complejos de resolver, pero a través de la didáctica unificada del diseño de software se realizan las siguientes apreciaciones:

En el pasado se ha experimentado con diferentes tipos de conformación de grupos, tales como en forma aleatoria, mezclando capacidades y el equipo con compañeros de clase en términos de afinidad demostrada previamente.

Equipos aleatorios

Tres o cuatro estudiantes son ubicados en forma arbitraria en un grupo, el problema de esto es que generalmente los estudiantes con mejores habilidades llevan el grupo, mientras otros se dejan llevar por él. Cuando éste método ubica amigos en un grupo, éstos tienden a formar subgrupos que dominan el proceso de toma de decisiones y esto no es una situación deseable.

Equipos con habilidades mezcladas

Un grupo es formado con el propósito de ubicar a estudiantes destacados con otros que no lo son tanto. Un propósito de esta estrategia es que las debilidades de algunos estudiantes puedan ser superadas mediante el aprendizaje de las fortalezas de otros estudiantes. El resultado es que, en la mayoría de los casos, los estudiantes aventajados llevan la responsabilidad del trabajo del equipo y los otros suelen dejarse arrastrar por él sin la producción de aprendizaje significativo.

Equipos a través de compañeros de clase con grado de afinidad previamente demostrado

Quizás la didáctica unificada del diseño de software apunta más hacia la formación de este tipo de grupos; esto debería ocurrir desde el claustro de profesores que mediante debate y discusión sean los profesores que

determinen y se "den cuenta" del grado de afinidades de los estudiantes. Esto significa un cambio radical en el grado de conocimiento de los estudiantes por parte de los profesores, significa además la realización de ejercicios de observación atenta y no-técnica de los estudiantes, representa la reivindicación de la conciencia de la labor eminentemente humana de la docencia puesto que se trata de observar la integralidad y la complejidad de esos individuos llamados estudiantes y ser conscientes de que se trata de seres humanos que piensan pero que también sienten.

Después de la conformación de los grupos, cada equipo deberá funcionar como una pequeña compañía de desarrollo de software que es administrada y guiada por el profesor. Los estudiantes seleccionarán un nombre y crearán un logo de la compañía, información general y hasta incluso un Web Site.

Definición de problemas de los clientes del mundo real

Inmediatamente después de la formación de los equipos, cada uno tiene un determinado período de tiempo para conseguir un cliente real, por lo general una compañía cercana al campus universitario con algún tipo de proyecto del mundo real.

Los clientes proveen los requerimientos del proyecto y las definiciones de las tareas de las cuales los estudiantes deberán elaborar los requerimientos del sistema. En esta parte es importante resaltar que los estudiantes se enfrentarán directamente con el trato interpersonal y vivirán qué tan difícil puede llegar a ser esta fase, porque sin lugar a dudas, la interacción humana es muy compleja.

La didáctica unificada del diseño de software promueve una serie de restricciones en la selección de los clientes del mundo real. Una es que los clientes no pueden estar directamente relacionados con algún miembro de los equipos, por lo general cuando se presentan estos casos, la experiencia dice que los miembros del equipo que tienen una cercana relación con los clientes del mundo real, finalmente terminan haciendo más de lo planteado en los requerimientos generales del sistema.

En la búsqueda de los clientes, es importante aclarar que los estudiantes no "vendan" el producto ya que deberá estar soportado mediante una carta del departamento o del programa que representa a la universidad donde se aclare que se generará un prototipo de aplicación y no un producto 100% terminado. Otro aspecto crítico es el grado de complejidad del problema real a resolver, es vital la visión que tenga el docente sobre el alcance de cada proyecto de tal forma que sea viable su culminación en el período de duración de la cátedra.

Procesos de desarrollo administrados

Es preciso tener en cuenta que las temáticas dentro del campo de la computación y la informática sufren transformaciones en períodos de tiempo relativamente cortos. Los sistemas y la computación permanentemente están acompañados de procesos de actualización, por lo tanto es lógico pensar que las temáticas de las cátedras de ingeniería de software o diseño de software estarán sujetas a las tendencias internacionales del momento.

Respecto a lo anterior la base teórica conceptual de la cátedra estará sujeta al paradigma y propuesta teórica-tecnológica actual, esto es: quizás en la década de los 80s se hablaba de enfoques de cliente-servidor, en los noventas se hablaba sobre sistemas distribuidos, en la primera década del siglo 21 obviamente se sigue hablando de los anteriores enfoques pero quizás las miradas apuntan hacia los ambientes Internet y las arquitecturas orientadas al servicio y desarrollo de componentes, y con seguridad en la próxima década sea otra la visón que tenga el mundo sobre el desarrollo de software.

Cierto es que todos los procesos de diseño y desarrollo de software, indiferentemente del paradigma o enfoque utilizado deberá estar administrado mediante metodologías claras, haciendo que los diseños y las construcciones de software estén fuertemente armadas con las principales características que garantizan longevidad al software hoy por hoy, características tales como:

- Robustez
- Escalabilidad
- Interoperabilidad
- Desempeño

Utilización de herramientas profesionales de vanguardia para el desarrollo de software

Evidentemente, el diseño de software conlleva o desemboca en la creación de productos informáticos que tendrán que ser realizados a través de herramientas. Sin el temor a ser excluyentes, es claro que en la actualidad existe un gran número de programas informáticos relacionados al campo del desarrollo de software como lenguajes meramente de programación, gestores de aplicaciones, suites completas de desarrollo e incluso herramientas CASE, sin embargo forma parte de la filosofía de la didáctica unificada del diseño de software el uso de herramientas que potencien la fase de construcción de software haciendo que ésta no sea la fase más relevante de todo el proceso.

Entender que el diseño es lo primordial, se convierte en la esencia de la didáctica propuesta. Se busca el rescate de los principios que fundamentan el concepto de ingeniería, es evidente que cuando se habla de ingeniería, se hace relación directa a esa facultad propia del ser humano que ha sido la causa nuclear de su propio desarrollo y evolución, se trata del Ingenio, al igual que la Imaginación y la Creatividad como motores intrínsecos que mueven las acciones del verbo diseñar. Bajo este principio es preciso que la construcción de software pase a un segundo plano frente al diseño de software y es por esta razón que es requerido el uso de herramientas modernas de última tecnología para la construcción de sistemas informáticos donde las labores triviales de programación ya existan y se busque la solución puntual de los problemas planteados dentro de la cátedra.

Aprendizaje basado en expertos de la industria de software

Aún cuando se asume que el docente responsable de la cátedra de diseño de software o de ingeniería de software dispone de los conocimientos y la experiencia suficiente para garantizar el éxito de la misma, es una realidad que los expertos en la industria de software se convierten en un punto de referencia bastante importante para el enriquecimiento de aprendizajes.

La didáctica unificada del diseño de software propone la invitación a expertos (por cualquier medio, bien sea presencial o virtual) en el campo del diseño y la construcción de software con el ánimo de alimentar las experiencias de los estudiantes frente a esta temática.

Presentaciones profesionales de los productos construidos

Al finalizar la cátedra en el período académico respectivo, la didáctica unificada del diseño de software propone la realización de la presentación profesional de los productos construidos como estrategia de socialización y evaluación parcial y colectiva de los logros académicos de los estudiantes.

Con esta actividad se pretende que cada grupo presente de manera profesional, "simulando" que están frente a una junta directiva de la empresa (cliente) y que tratan de vender su producto no con el ánimo de capacitar a los estudiantes en marketing sino que se trata de establecer los niveles de claridad y eficiencia de los resultados obtenidos como conclusión del proyecto planteado.

Énfasis en la solución de problemas y aprender a aprender

La mayor meta de la didáctica unificada del diseño de software es crear aprendices autosuficientes. Es primordial que la cátedra a partir de la fundamentación teórica se complemente con una buena práctica bajo el lema "hágalo usted mismo", este componente es importante en la fase de implementación de software por tal razón este curso no necesariamente deberá requerir grandes conocimientos en programación de software.

CONCLUSIONES

Se ha presentado una primera aproximación, un preámbulo de lo que será la didáctica unificada del diseño de software, la idea central del presente artículo gira sobre el potenciamiento de las competencias aptitudinales que se desea potenciar en el estudiante a través de la puesta en marcha de la didáctica.

Pretender construir una didáctica que funcione en todos los contextos es quizás una postura muy ambiciosa, quizás utópica, sin embargo, no se busca la

creación de una propuesta de aplicación dogmática o de rigurosidad extrema, tan solo se trata del compendio de reflexiones fundamentadas en experiencias de las más prestigiosas universidades del mundo en el campo de la computación y la informática y específicamente de la cátedra de diseño de software.

REFERENCIAS

ACM Special Interest Group for Computing Education - ACM Computing Curricula, 2005

Beyer & Holtzblatt (1998). Contextual Design, Defining Customer-Centered Systems. San Francisco: Morgan Kaufman.

IBM, Object-Oriented Technology Center. (1997). "Developing Object-oriented Software". New Jersey: Prentice Hall.

Demarco, T & Lister T (1999). "Peopleware—Productive Project and Teams". New York, NY: , Dorset House Publishing Co.

Sanjiv P, Sawh D. & Shah B (1995). "How to Manage a Successful Software Project". New York: John Wiley & Sons, Inc.