Desarrollo de competencias en Ingeniería de Software: una experiencia académica en el ITC

Julio Armando Asato España

Instituto Tecnológico de Celaya

julio.asato@itcelaya.edu.mx

Elda Ramírez González

Instituto Tecnológico de Celaya

elda.ramirez@itcelaya.edu.mx

Dolores Berenice Hernández Jáuregui

Instituto Tecnológico de Celaya

avi_dbhj@hotmail.com

Jonathan Abraham Sánchez Arias

Instituto Tecnológico de Celaya

jabrahamsa55@gmail.com

Resumen

El proceso de desarrollo de competencias en disciplinas que implican el uso de metodologías, estándares y normas presenta un nivel de complejidad que frecuentemente trasciende el alcance de un curso universitario. En el caso particular de la Ingeniería de Software, para la carrera de Ingeniería en Sistemas Computacionales de los Institutos Tecnológicos, se tiene la ventaja que hay tres asignaturas que contemplan este proceso, sin embargo, es necesario por una parte ajustar los contenidos temáticos para que tengan continuidad en el proceso global, y en otro sentido, es requerido establecer las estrategias didácticas necesarias para que esta formación sea efectivamente adquirida por los estudiantes, a fin de combatir el efecto denominado la Crisis del Software, que implica tener productos que no cumplen lo requerido, resultan más costosos de lo esperado y que adicionalmente se entregan de manera tardía.

La Ingeniería de Software

Típicamente la palabra software se percibe como una referencia directa a un programa de computadora, el sentido de este término suele ser atribuido a John W. Tukey quien lo utilizó por primera vez en un artículo para la revista *American Mathematical Monthly* en 1957. Hoy en día este concepto toma un sentido más amplio, refiriéndose a un conjunto completo de programas, procedimientos y la documentación asociada a un sistema que opera en dispositivos de cómputo (Sánchez, 2012: 13).

Ante la necesidad creciente del procesamiento automatizado de datos, en los años 60s hubo una alta demanda productos de software, lo que propició el desarrollo de sistemas de información, esto aunado a la disminución en el costo del equipo de cómputo (hardware) que puso al alcance de numerosas empresas la posibilidad de contar con sus propios centros para el proceso de datos, de forma que las aplicaciones computacionales se multiplicaron. Sin embargo, ante la falta de estándares metodológicos para el desarrollo de software surgieron ciertos problemas característicos, que fueron resumidos en una conferencia sobre esta materia, convocada en 1968 por la Organización del Tratado Atlántico Norte (OTAN), en lo que se identificó como la "Crisis del Software", la cual se resume en tres problemas principales (Sommerville, 2011: 5):

- Productos no confiables.
- Con un costo superior a lo presupuestado.
- Que son entregados de manera tardía.

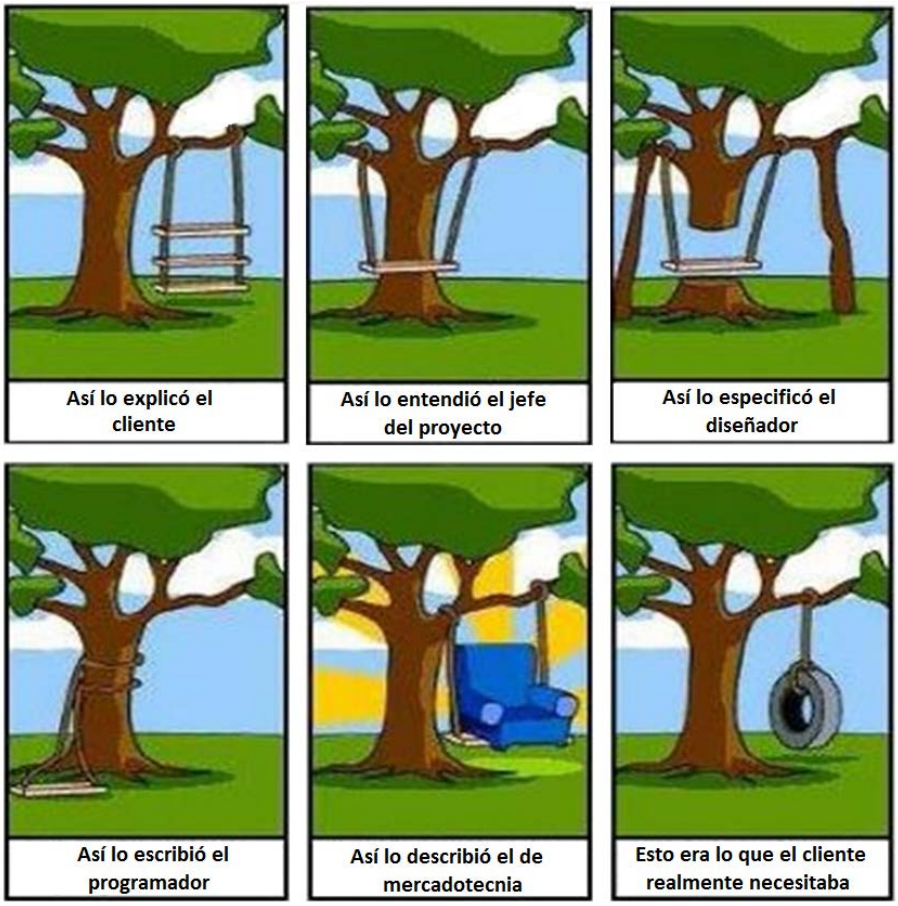
Posteriormente resultó evidente un nuevo problema, la dificultad de mantenimiento y actualización debido a la carencia de documentación de los sistemas. Como respuesta a esta crisis, en la misma conferencia recibió su denominación oficial la disciplina conocida como Ingeniería de Software (Sánchez, 2012: 16). De acuerdo a una de las definiciones más ampliamente utilizadas, la del Instituto de Ingenieros Eléctricos y Electrónicos (IEEE por sus siglas en inglés), la Ingeniería de Software puede definirse como:

"La aplicación de un enfoque sistemático, disciplinado y cuantificable al desarrollo, la operación y el mantenimiento del software; es decir la aplicación de la ingeniería al software." (IEEE, 1990: 67)

Esta disciplina tiene como fin principal realizar y revisar un análisis previo de las necesidades o requisitos que atenderá el software, el diseño, la construcción, prueba y mantenimiento, mediante alguna de las metodologías establecidas. Pero sobre todo, da la pauta de realizar una correcta documentación de cada una de las fases de desarrollo, facilitando en gran medida lo que necesita para su conclusión en tiempo, forma y costo.

Las buenas prácticas de la ingeniería de software previenen la ocurrencia de los problemas citados en la Crisis del Software, hasta cierto punto. Hoy en día, a más de cuatro décadas de aquella conferencia de la OTAN, todavía son persistentes los problemas en el desarrollo de software ya que académicamente, en las Instituciones de Educación Superior (IES), resulta complicado formar profesionistas con la experiencia de haber experimentado las condiciones de una organización real de desarrollo de software. Obteniendo como resultado la formación de profesionistas muy capacitados en labores específicas, pero con deficiencias en la visión global del problema y en la gestión de requisitos, como se representa humorísticamente en la Figura 1.

Cualquier enfoque de la ingeniería (incluido el de la Ingeniería del Software) debe estar sustentado en un compromiso con la calidad el cual es un enfoque ineludible en cualquier disciplina. Por otra parte, la esencia de la ingeniería del software es el estrato del proceso (Pressman, 2005: 23) como puede apreciarse en la Figura 2. El proceso de desarrollo de software es definido como: "Un conjunto de personas, estructuras de organización, reglas, políticas, actividades y sus procedimientos, componentes de software, metodologías, y herramientas utilizadas o creadas específicamente para definir, desarrollar, ofrecer un servicio, innovar y extender un producto de software" (Ruvalcaba, 2009: 20). El proceso de la ingeniería del software es el elemento que mantiene juntos los estratos metodológicos y de herramientas tecnológicas, lo que permite el desarrollo racional y a tiempo de software de computadora de calidad.



Fuente: Basado en (Peves, 2009)

Figura 1 Alegoría del proceso de gestión de requisitos de software.

Fuente: Elaboración propia.

Figura 2 Estratos de la Ingeniería de Software.

Las normas de calidad, los procesos definidos, así como la relación de comunicación entre el cliente y el equipo de desarrollo son muy importantes, ya que de ellos se deriva el éxito o fracaso de un producto de software. Sin embargo, la actividad de realizar el

seguimiento de cada fase se puede transformar en una tarea abrumadora, especialmente si se trata de un desarrollo complejo. Por esta razón se cuentan con diversas alternativas metodológicas de desarrollo, las cuales se adaptan a las condiciones del proyecto y a las particularidades tanto de clientes como del equipo de trabajo. De esta manera existen metodologías estructuradas para situaciones bien especificadas, de tipo evolutivo cuando los requisitos o condiciones son un tanto inciertas, orientadas a objetos para apegarse a las características de la programación, o bien las metodologías ágiles para proyectos que no ameritan el uso de metodologías más complejas y formales. Sin embargo, estas prácticas con frecuencia no son utilizadas por algunos programadores, debido a la carencia de una formación adecuada desde la carrera universitaria, en donde los estudiantes son abrumados con los aspectos técnicos de las herramientas de programación y el énfasis en el producto más que en el proceso.

Antecedentes curriculares

La función principal del Sistema Nacional de Institutos Tecnológicos (SNIT) es elevar la calidad de la educación, para que los estudiantes mejoren su nivel de logro educativo, por este motivo siempre se busca de la innovación y mejora del programa de estudios (SNIT, 2013A).

A finales de siglo pasado e inicios de este, el SNIT ha desarrollado tres procesos importantes en torno a las actividades de diseño, seguimiento y evaluación curricular.

- El Programa para la Modernización Educativa.
- La Reforma de la Educación Superior Tecnológica.
- El Programa de Evaluación Curricular.

En estas mejoras y ajustes se aplicó a partir del 2009 y 2010 un nuevo plan de estudios basado en competencias. Para el programa de estudios de la carrera de Ingeniería de Sistemas Computacionales se establecieron un conjunto de asignaturas que son clasificadas según el organismo acreditador CONAIC (Consejo Nacional de Acreditación en Informática y Computación A. C.) y por la ANIEI (Asociación Nacional

de Instituciones de Educación en Tecnologías de la Información A. C.) en las siguientes áreas: entorno social, arquitectura de computadoras, redes, software de base, programación e ingeniería de software, tratamiento de la información y por último, interacción hombre – máquina (ANIEI, 2013). Las asignaturas de la carrera de Ingeniería en Sistemas Computacionales que competen a la disciplina de ingeniería de software son:

- Fundamentos de ingeniería de software (SCC-1007).
- Ingeniería de software (SCD-1011).
- Gestión de proyectos de software (SCG-1009).

Estas asignaturas están ubicadas respectivamente en los semestres quinto, sexto y séptimo de la retícula de la carrera, la cual tiene un perfil de egreso con competencias genéricas y específicas, en donde algunas de estas últimas están ligadas directamente con la Ingeniería de Software.

Una de las competencias que debe adquirir un egresado de la Ingeniería en Sistemas Computacionales es la de "Desarrollar, implementar y administrar software de sistemas o de aplicación que cumplan con los estándares de calidad con el fin de apoyar la productividad y competitividad de las organizaciones" (SNIT, 2013B). Por este motivo se ha buscado la mejora en la labor docente respecto a las prácticas de la Ingeniería de Software, aprovechando el hecho de que las asignaturas implicadas llevan un proceso de continuidad reticular.

Dentro del marco que otorga el Modelo Educativo para el Siglo XXI, se faculta a las Academias de las IES para contribuir en el proceso de revisión curricular y contenidos temáticos (DGEST, 2012: 50). Con este propósito, los profesores que conforman la vocalía de Ingeniería de Software en la Academia de Ingeniería en Sistemas Computacionales e Informática, del Instituto Tecnológico de Celaya (ITC), realizaron la revisión de los contenidos temáticos y obtuvieron las siguientes observaciones:

- En las tres asignaturas ya mencionadas se detectó discordancia en la continuidad de los temas, en donde se presentaba a grandes rasgos: conceptos básicos, obtención de datos, modelado, metodologías, más

temas de modelado, seguridad, planificación, calidad y ejecución del proyecto.

- No se encontró un orden adecuado para la formación de competencias, en la asignatura Fundamentos de Ingeniería de Software ya se estaba trabajando con los modelos de análisis, diseño e implementación sin haber estudiado alguna metodología de desarrollo.
- En la siguiente asignatura, llamada Ingeniería de Software se iniciaba el temario con el modelado de negocios, no ofreciendo continuidad con la asignatura anterior, además, los temas relacionados a las metodologías de desarrollo de software se trataban hasta después en este temario.
- En la asignatura final denominada Gestión de Proyectos de Software, se encontraba ubicado el único tema relacionado con la calidad del software. En la experiencia de los planes curriculares anteriores, se observó que al formar al estudiante en la parte práctica primero y luego en los tópicos de calidad, daba como resultado que los estudiantes tomaran a esta última como un anexo, hasta cierto punto optativo o no deseado del proceso, cuando la realidad es que debe ser un aspecto inmerso en las actividades y la práctica cotidiana a lo largo del proceso de desarrollo de software.

Ante esta situación, se buscó resolver estos problemas de continuidad en las asignaturas mediante las siguientes acciones:

- Se reorganizó el contenido de las tres asignaturas para llevar un orden progresivo en la formación del estudiante.
- La propuesta para que en las asignaturas de Ingeniería de Software y Gestión de Proyectos de Software se integren las etapas de los modelos de análisis, diseño, implementación, arquitectura y seguridad, a nivel de diseño en la primera y de ejecución en la segunda asignatura.
- Las competencias genéricas se ajustaron de acuerdo a los cambios en los contenidos temáticos, cabe destacar la inclusión de la competencia interpersonal de Compromiso Ético, principalmente por la unidad de Calidad del Software que

fue trasladada de la última a la primera de las tres asignaturas, esta competencia se juzgó pertinente por el carácter de confidencialidad propio del análisis de sistemas.

Al realizar los cambios a los contenidos temáticos, se conserva la idea que las competencias que adquirirá el estudiante una vez que acredite las tres asignaturas del grupo de Ingeniería de Software serán equivalentes a las del programa original, pero ahora desarrolladas de manera progresiva.

Con los ajustes propuestos, queda la clara la intención de que las asignaturas del grupo de Ingeniería de Software apoyarán en la aplicación práctica del conocimiento científico, a través de las metodologías, técnicas y estándares adecuados, para el desarrollo de software de calidad ahora con un enfoque teórico-práctico.

La formalización de los ajustes a los programas de estudios para estas tres asignaturas se llevó a cabo en el ITC, donde participó la Academia de Sistemas e Informática, a través de la vocalía de Ingeniería de Software, conformando la justificación del ajuste y actualización de contenidos temáticos de acuerdo a la revisión global de las asignaturas relacionadas. Estos ajustes fueron expuestos en el pleno de la Academia siendo aprobados por el mismo, quedando sustentado en el acta correspondiente con fecha de 06 de enero de 2012, para posteriormente proponerse en la Reunión Nacional de Seguimiento Curricular del SNIT, llevada a cabo del 22 al 25 de octubre del 2012 en la ciudad de Querétaro, Querétaro. Encontrándose apoyo en los pares académicos de otras instituciones para que estos cambios sean formalizados en los programas oficiales de la carrera a nivel nacional.

Método de trabajo y experiencias adquiridas

El ajuste en los programas de estudio es sólo un primer paso, la formación efectiva de los estudiantes en la disciplina de la Ingeniería de Software implica la definición y ejecución de estrategias de aprendizaje específicas. Para ello se establecieron ciertas directrices en la impartición de las tres asignaturas involucradas, para los estudiantes de la generación puntera de la carrera de Ingeniería en Sistemas Computacionales plan

2010. Las políticas empleadas en este primer ejercicio se resumen en los siguientes puntos:

1. Dar continuidad en la formación procurando conservar el mismo proyecto y el mismo docente para una misma generación de estudiantes.
2. Aplicación de metodologías, estándares y situaciones reales en el desarrollo de los trabajos en las tres asignaturas.
3. Definición de un proyecto a tres semestres, el cual implique el desarrollo de un sistema de información con los componentes típicos que a veces no son considerados en los proyectos académicos, como reportes impresos, respaldos, control de usuarios y documentos de gestión como lo es el contrato.
4. La formación de equipos de trabajo relativamente numerosos, acordes a la magnitud del proyecto, de manera que surja de manera natural la necesidad de comunicación y colaboración (así como la problemática típica de organización), para que se fomente el efectivo trabajo en equipo. Estos equipos debieron definir su identidad como empresa, estableciendo su nombre, logotipo, estructura y filosofía organizacional.
5. El uso de un repositorio compartido de configuración del software, en donde se tendrá la base de conocimiento de la empresa, conteniendo los materiales de apoyo, referencias de trabajo e información técnica, registros de seguimiento y productos obtenidos, de manera que al final del proceso todos cuenten con la misma información y de esta manera se subsane la natural diferencia de aprendizaje de las actividades en donde se participó directamente y donde no fue así.
6. Definición de una estructura operativa, en donde los integrantes tengan un rol definido el cual puede rotarse periódicamente.
7. Supervisión externa, de manera que las actividades de un grupo de trabajo dentro del equipo de desarrollo de software sean evaluadas por una persona ajena a ese grupo.

Durante la ejecución práctica de estas estrategias, se ha tenido como experiencia una vivencia más profunda del proceso de desarrollo de software en los alumnos, ya que al

tratarse de un proyecto a largo plazo (académicamente hablando) es posible incluir la complejidad propia de un proyecto real. También han surgido los problemas típicos de un grupo de trabajo estudiantil en cuanto a participación, cumplimiento y comunicación, sin embargo estas experiencias "negativas" también son deseables con la finalidad de que ellos mismos participen en la búsqueda de estrategias para resolverlas.

El trabajo se ha desarrollado en términos que pretenden emular a la realidad, teniendo como marco una metodología y principios de calidad, se tuvo un primer acercamiento con una necesidad de sistemas de información para una empresa; después se realizó el análisis de requisitos y la propuesta de sistemas para recibir el visto bueno del "cliente", el cual en esta experiencia fue el profesor; se determinó el costo del proyecto y la planificación del mismo; lo cual fue tomado como base para la redacción de un contrato formal para el desarrollo, con cláusulas de costo, tiempo de entrega, características del producto, garantía y soporte; se trabajó en el diseño del sistema; así como la codificación, prueba, integración y entrega de una versión demo del mismo, de manera que los estudiantes hayan desarrollado al menos un producto de software tangible de todo el proceso.

Debido a que las asignaturas del grupo de Ingeniería de Software no comprenden actividades específicas de programación, no resulta conveniente tratar de desarrollar el sistema completo, aunado esto al hecho de que las asignaturas de este grupo no son las únicas que llevan en el semestre, es preciso acotar las labores de programación a niveles factibles que a la vez, permitan generar un producto entregable y no representen una carga demasiado fuerte a las labores académicas del estudiante en cada semestre del proceso.

Conclusiones y recomendaciones

El desarrollo de proyectos de software complejos en el ámbito académico es posible con una adecuada coordinación tanto de docentes como de estudiantes. Es necesario establecer una seriación en la impartición de las asignaturas para lograr el objetivo de seguimiento a los proyectos por generación estudiantil.

En la práctica, las dificultades en la definición de requisitos han resultado un punto a mejorar, ya que si bien fue una experiencia muy importante afrontar estos problemas, se llevó más tiempo de clase del estimado lo cual repercutió en el avance en los contenidos temáticos de la asignatura intermedia de Ingeniería de Software.

Es deseable que en nuevas experiencias se realice un trabajo más profundo por parte del docente en la definición previa del proyecto a abordar, con el fin de proporcionar al estudiante elementos suficientes para conservar el ritmo de trabajo que se puede desarrollar un equipo grande.

Pese a las dificultades en este primer ejercicio de las prácticas expuestas, el propósito final ha sido cumplido, la experiencia adquirida por los estudiantes en el trabajo dentro de una organización de desarrollo de software ha proporcionado elementos que permiten vislumbrar en el equipo los propósitos de madurez del proceso y mejora continua, todo ello en aras de formar profesionistas capaces de afrontar el reto todavía persistente de la Crisis del Software.

Referencias

[1] ANIEI (2013). “Catálogo de áreas de conocimiento”. Asociación Nacional de Instituciones de Educación en Tecnologías de la Información A. C. México: http://aniei.org.mx /portal/modules.php?&name=modeloslic2&op=areas&func=all.

[2] DGEST (2012). “Modelo educativo para el siglo XXI”. Dirección General de Educación Superior Tecnológica, Secretaría de Educación Pública. México.

[3] IEEE (1990). “Standard glossary of software engineering terminology”. Instituto de Ingenieros Eléctricos y Electrónicos. EUA: IEEE.

[4] Peves, Alberto (2009). “Ingeniería de software (eBook en español)”. Blog vidainformático.com, publicado el 18 de septiembre del 2008: http://www.vidainformatico.com/2009/09/ingenieria-de-software-ebook-en-espanol.html.

[5] Ruvalcaba, Mara (2005). “Procesos de software, guía del viajero”. Revista Software Guru, año 1 número 1, enero-febrero del 2005. México: http://issuu.com /softwareguru/docs/sg01?mode=window&backgroundColor=%23222222.

[6] Pressman, Roger S. (2005). “Ingeniería de Software”, 6ª edición. México: Mc. Graw Hill.

[7] Sánchez, Salvador y otros (2012). “Ingeniería de Software. Un enfoque desde la guía SWEBOK”. México: Alfaomega.

[8] SNIT (2013A). “Breve historia de los Institutos Tecnológicos”. Sistema Nacional de Institutos Tecnológicos. México: http://www.snit.mx/informacion/sistema-nacional-de-educacion-superior-tecnologica.

[9] SNIT (2013B). “Ingeniería en Sistemas Computacionales, perfil de egreso”. Sistema Nacional de Institutos Tecnológicos. México: http://www.snit.mx /licenciatura_2009_2010/ingenieria-en-sistemas-computacionales.

[10] Sommerville, I. (2011). “Ingeniería de Software”. 9ª edición. México: Pearson.