Propuesta de implementación de un laboratorio de ingeniería de software 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

José Guillermo Fierro Mendoza

Instituto Tecnológico de Celaya

guillermo.fierro@itcelaya.edu.mx

Patricia Galván Morales

Instituto Tecnológico de Celaya

patricia.galvan@itcelaya.edu.mx

José Rosario Villanueva Morales

Instituto Tecnológico de Celaya

chayo777@msn.com

Resumen

En los procesos formativos de la educación superior, el desarrollo de competencias en actividades de gestión de proyectos de software requiere de condiciones específicas, de espacios de trabajo que suelen ser diferentes a las que se presentan en un laboratorio de programación tradicional. Es necesario que haya condiciones para llevar a cabo las diferentes actividades necesarias para los procesos de comunicación, análisis, diseño, codificación, verificación, integración y seguimiento a los proyectos. El presente documento contiene una propuesta para implementar un espacio adecuado para la práctica en los procesos de Ingeniería de Software, bajo el contexto de los

métodos ágiles, dadas las condiciones y características propias de una Institución de Educación Superior.

Palabras Clave: Métodos Ágiles, Laboratorio, Ingeniería de Software.

Abstract

In the formative processes of higher education, skills development activities of project management software requires specific conditions of workspaces that are often different from those presented in traditional programming lab. There must be conditions for carrying out the various processes necessary for communication, analysis, design, coding, verification, integration and project monitoring activities. This document contains a proposal to implement adequate space to practice the Software Engineering processes, in the context of agile methods, given the conditions and characteristics of an Institution of Higher Education features.

Keywords: *Agile Methods, Laboratory, Software Engineering.*

1. Introducción

Es común encontrar artículos manuales y otros materiales donde se describen las características de un laboratorio escolar de física, química o de temas relacionados con la ingeniería mecánica, pero cuando hacemos búsqueda sobre el tema para la conformación de un laboratorio de Ingeniería de Software para carreras profesionales y de posgrado relacionados a las tecnologías de la información, en donde se puedan reproducir ejercicios, realizar prácticas y proyectos o análisis de casos, en los que los estudiantes logren adquirir las competencias adecuadas para el desarrollo de software, sencillamente encontramos pocas referencias al respecto.

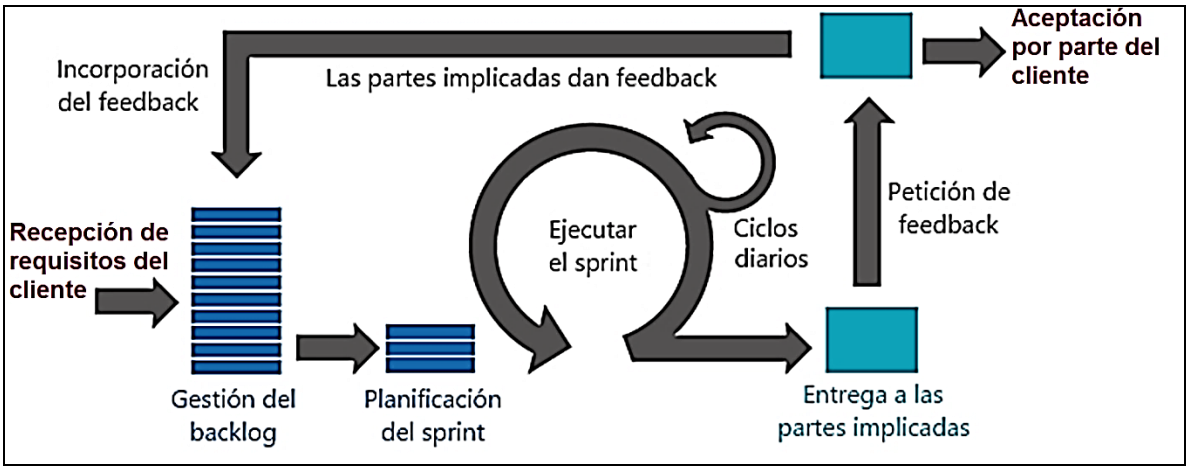
Hablar de la Ingeniería de Software implica una de las actividades más importantes de las Ciencias de la Computación. La construcción de un programa, sistema o solución implica una gran cantidad de aspectos a considerar, entre ellos la recopilación de requerimientos, obtenidos normalmente de una fuente externa (los usuarios o situaciones), su análisis, diseño e implementación, las pruebas unitarias y de integración, para finalmente generar una distribución y finalmente su puesta en operación. Por motivos de competitividad, las organizaciones que se dedican al software, consideran que es necesario construir los productos en el menor tiempo posible y con la mejor calidad, incluso como mencionan Alfredo Weitzenfeld y Silvia Guardati. “El desarrollo y mantenimiento de software debe realizarse con la misma seriedad y responsabilidad con la que se llevan a cabo cualquiera de las actividades propias de las ingenierías tradicionales” (Weitzenfeld, 2007).

La construcción de software es un proceso complejo y que generalmente se desarrolla en equipos de trabajo para lograr acelerar tanto su proceso de desarrollo como garantizar la calidad del mismo. Desde hace casi cinco décadas fue identificado un efecto denominado La Crisis del Software (OTAN, 1969), el cual en resumen identifica tres problemas principales que son:

- Los proyectos no terminan en el plazo indicado.
- El desarrollo excede el presupuesto inicial.
- El software desarrollado no hace lo que se esperaba de él.

Para atender estos problemas se han desarrollado varias metodologías para la construcción de programas de software, mismas que han ido evolucionando a través de los años. En el intento de mejorar, acelerar y garantizar el éxito el producto de software se han propuesto estrategias formales, metódicas y sistemáticas, que si bien en un sentido teórico resultan razonables, en la práctica no resuelven todos los problemas, en especial cuando se trata de productos de software que no han sido completamente definidos de inicio.

Cuando se trata de problemas complejos, donde la necesidad de adaptación es la constante, es requerido contar con un marco de trabajo ágil que pueda dar certidumbre a las acciones inmediatas que deberán desarrollarse a fin de garantizar un paso antes de dar el siguiente. Una de las estrategias más exitosa para este propósito es la conocida como SCRUM, el cual no es un proceso o una técnica, más que eso es un marco de trabajo dentro del cual se pueden emplear varias técnicas y procesos (Schwaber, 2013). Este marco de trabajo se distingue por ser ligero, fácil de entender pero extremadamente difícil de llegar a dominar. Es por eso que resulta importante para las instituciones de educación superior que deseen formar competencias en estos rubros contar con espacios de trabajo, donde estas prácticas no solamente puedan ser explicadas, sino también vividas y realizadas de inicio a fin, de manera que ofrezcan a los estudiantes la experiencia de realizar proyectos bajo condiciones similares a las presentadas en varias de las empresas líderes en el desarrollo de productos de software.



Basado en: http://programandonet.com/web/scrum-con-tfs/

Figura 1: El proceso SCRUM

En SCRUM la gestión de un proyecto inicia con la conformación de una pila de producto (backlog) compuesta por una serie de requisitos ligados a historias de usuario, de la cual se seleccionan algunas para conformar una carrera (sprint), esa carrera será ejecutada por un equipo de trabajo en un periodo de tiempo especificado, dentro del cual se harán seguimientos diarios para resolver dudas y atender problemas, una vez

concluida la carrera se hace la entrega al cliente, este decide si acepta o reformula su requisito (feedback) con lo cual se actualiza la pila de producto y se inicia una nueva carrera. Este proceso continua hasta que el proyecto se da por concluido (Escolar, 2013).

Como se mencionó en anteriormente, el proceso resulta bastante sencillo de entender, pero en la práctica surgen muchas dudas e inquietudes para su ejecución, en especial en ámbitos educativos como en el Instituto Tecnológico de Celaya (ITC) donde la disponibilidad de tiempo y atención de los estudiantes no es exclusiva para el proyecto, lo que apremia tener espacios adecuados donde se realicen prácticas a modo para que los estudiantes puedan desenvolverse en escenarios similares a los que tendrán en su actuar profesional.

Cabe aclarar que aunque por su nivel de difusión, SCRUM está considerado como el marco de trabajo base para el laboratorio, este no será exclusivo para esta estrategia de desarrollo, de manera que se tiene pensado que el laboratorio tenga características polivalentes para adaptarse a diferentes esquemas que deban desarrollarse, desde el uso de metodologías clásicas hasta sesiones teóricas grupales, de presentación, prácticas individuales o de trabajo por pares, según se necesite.

2. Métodos

Para la presente propuesta se revisaron bases metodológicas obtenidas de procesos de investigación documental sobre fuentes físicas y virtuales relacionadas con los aspectos tratados, para ello se recurrió a los documentos clave de las metodologías ágiles, fin de conformar una base de conocimiento sobre el estado del arte de la temática a tratar.

Para la parte demostrativa se establecieron tres fuentes de información:

- Referencias indirectas: Estas consisten en experiencias obtenidas sobre las condiciones de aplicación de las metodologías ágiles (SCRUM, Programación Extrema, Crystal Clear, entre otras), que hayan sido publicadas en los últimos diez años. El valor de esta información estriba en que son datos reconocidos y que reflejan la vivencia práctica de estas metodologías.
- Referencias directas: Son también experiencias en el uso de espacios de trabajo por parte de terceros, pero que han sido referidos directamente a los autores. Esta vinculación es muy importante ya que no solo refleja resultados y "cosas" que sirvieron, sino que también permite conocer las problemáticas particulares, estrategias que no funcionaron y los ajustes que debieron llevarse a cabo para que funcionen, enriqueciendo el sentido práctico de la propuesta presentada.
- Experiencias directas: Consisten en el proceso experimental de la propuesta de trabajo presentada, desarrollada y controlada directamente por los mismos autores. Aunque el marco de tiempo para estos trabajos es reducido, estas experiencias permiten examinar los resultados de elementos específicos de la propuesta, que no están cubiertos por las dos fuentes anteriores.

Con esta información como base es posible formular de forma sustentada una propuesta para la implementación del laboratorio de Ingeniería de Software para mejorar la competitividad de los estudiantes del ITC.

3. Resultados

Es interesante saber que las empresas dedicadas a la producción de software, al menos en nuestra región, adaptan un proceso propio donde normalmente toman las mejores características de algunos de los modelos. Desde esta perspectiva, se promueve la construcción de un laboratorio de Ingeniería de Software en donde los estudiantes puedan realizar siguientes las tareas prácticas.

1. Aplicar métodos apropiados en la recopilación de requisitos.
2. Aplicar técnicas para la generación del modelo del sistema.

3. Aplicar diversas técnicas en la elaboración del diseño de productos de software.
4. Utilizar herramientas para la generación de prototipos de un producto de software, incluyendo herramientas que permiten la generación de código automático.
5. Contar con plataformas donde el estudiante pueda realizar las pruebas de desempeño, funcionalidad y seguridad de las aplicaciones.
6. Tener una experiencia vivencial de trabajo en equipo conformado para la construcción de un producto de software utilizando procesos o marcos de desarrollo estandarizados para trabajar colaborativamente y obtener el mejor resultado posible de un proyecto.
7. Implementar el Proceso de Software Personal (PSP) y Proceso de Software por Equipo (TSP), a fin de poder medir y evaluar los factores clave que producen aciertos y errores en el trabajo.

Ahora bien, aunque las anteriores son las expectativas académicas, para la conformación de un espacio adecuado donde puedan desarrollarse estas habilidades, se han considerado también recomendaciones obtenidas de algunas empresas líderes en el desarrollo de software de la región, las cuales se sintetizan en los siguientes puntos.

- Los espacios para los equipos de desarrollo de software tienen una aportación especial en el alcance de los objetivos, dichos espacios deberán de estar diseñados en apoyar la creatividad, la comunicación abierta y la agilidad, de igual forma deberán de brindar comodidad, transparencia, Inspección y seguimiento, lo cual ayudara bastante en modelos como SCRUM, PSP, TSP, entre otros.
- Se debe tener presente la idea que a los equipos de desarrollo se les debe de dar autonomía y que estos equipos sean quienes tomen las decisiones correspondientes a diferentes situaciones que presentan los proyectos.
- Resulta muy útil poder formar islas de trabajo para equipos de un mismo proyecto esto contribuye a los aspectos citados en el primer punto.

- Es importante contar con espacios para hacer las reuniones de los equipos, sobre todo para el marco de trabajo SCRUM, así como contar con pizarrones, cañón, plumones, mesas modulares, tableros y demás implementos propios para las labores a desarrollar.
- Si bien ayuda bastante que los espacios sean agiles y prácticos, deberán existir políticas que estén orientadas al cuidado de los equipos y mobiliario, dado que como equipos de desarrollo es necesario disminuir el riesgo de daño en estos activos, que son de vital importancia para el funcionamiento del equipo. Algo que funciona para el cuidado de los equipos y mobiliario es que los integrantes del equipo adopten la idea que se trata de sus espacios.
- El espacio que asigne para el laboratorio debe permitir su reorganización, de manera que sea lo suficientemente flexible para la atención de proyectos con diferentes niveles de complejidad y con diferentes actividades a realizar a lo largo de la jornada de trabajo.

Como producto del análisis realizado con base a la información recabada, se determinó que el laboratorio de Ingeniería de Software debe contener elementos tecnológicos, mobiliario, instalaciones, servicios y herramientas, acordes con las metodologías que se van a desarrollar, para las cuales se identificaron los siguientes elementos:

Servidor de integración continua local: En donde se concentrarán diferentes herramientas para el almacenamiento de código, pruebas automatizadas, control de versiones e integración continua del software desarrollado, el contar con este equipo en el mismo laboratorio ofrece a los estudiantes la oportunidad de configurar y administrar estos servicios. El tener este servidor es necesario dada la relevancia de que sea tangible el producto del trabajo realizado, es decir, no es posible hablar de desarrollo de software si no hay un software terminado y en operación al final del proceso.

Infraestructura de red: Requerida para interconectar los equipos de desarrollo de software involucrados, esta deberá ser tanto alámbrica como con soporte

inalámbrico, ya que la realidad es que la mayoría de los estudiantes de estas asignaturas habitualmente prefieren desarrollar en sus propios equipos.

Estaciones de trabajo: Deberán ser apropiadas para desarrollar la codificación por pares, propia de los métodos ágiles. Para este caso se plantea utilizar mesas metálicas con espacio para el acomodo de dos estudiantes.

Mobiliario de trabajo: Preferentemente compacto y adaptable para las diferentes actividades a realizar bajo los marcos ágiles. En este caso se consideran mesas trapezoidales individuales, que puedan armar diferentes configuraciones según las necesidades de trabajo

Tableros: Cada equipo de desarrollo deberá tener su propio tablero para visualizar en todo momento el estado de las actividades y tomar oportunamente las acciones que correspondan. Una premisa de las metodologías ágiles es tener el trabajo "visible", para ello los tableros ayudan a evaluar los avances y las cargas de trabajo durante la corrida o Sprint.

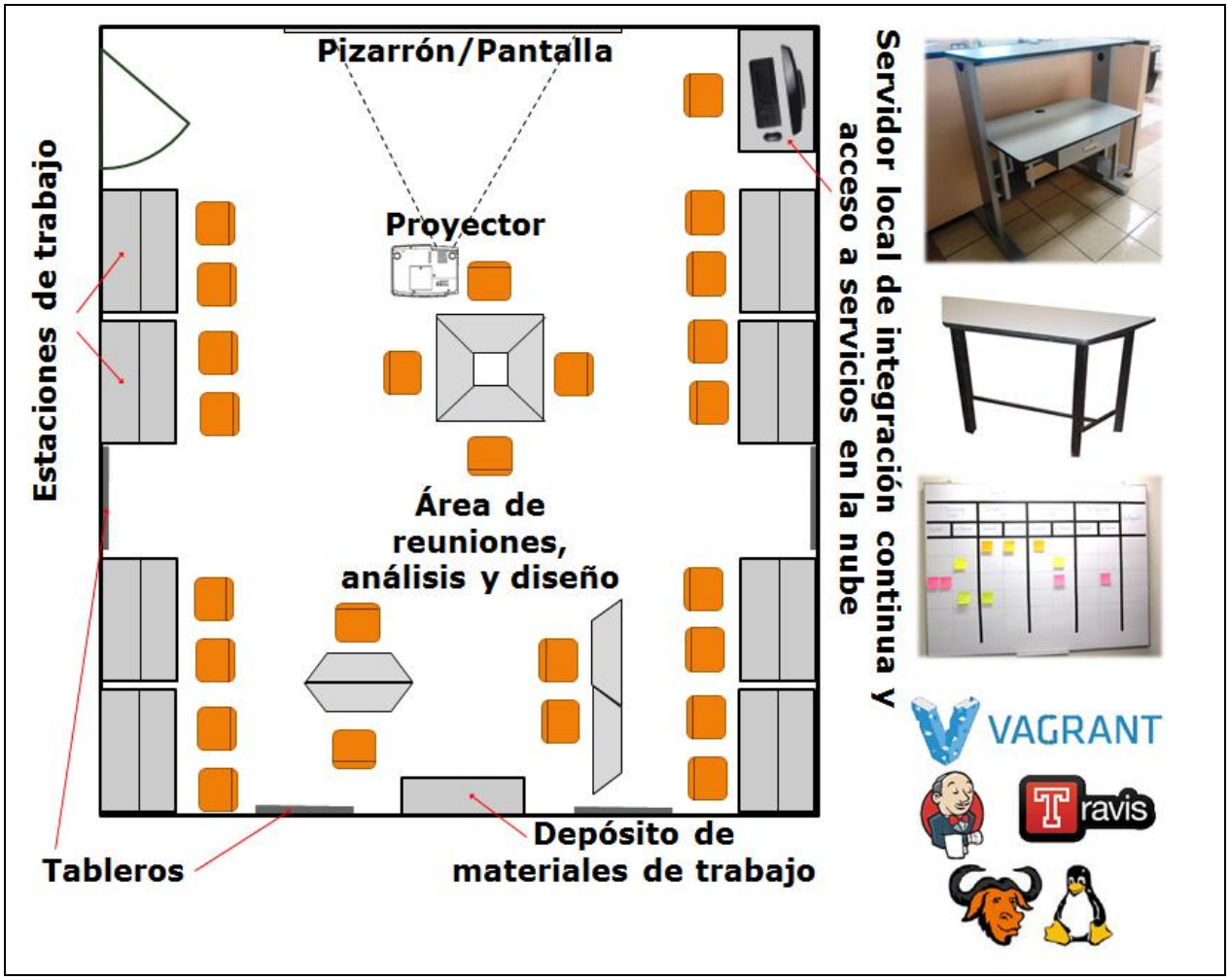
Materiales auxiliares: Adicional a lo anterior, son requeridos algunos materiales de apoyo como las cartas de planeación (*Planning Poker*), gráficos de control, formatos, etiquetas para los tableros y papelería diversa.

Cañón proyector: El escenario de trabajo deberá poderse convertir rápidamente en una sala de exposiciones tradicional, de manera que cuando los temas lo requieran no sea preciso hacer movimientos importantes para tener este tipo de procesos formativos.

Espacio y servicios en la nube: Complementario al servidor de integración continua local, es necesario contar con la posibilidad de distribuir las aplicaciones desarrolladas con el propósito de su almacenamiento, integración y pruebas, en los servicios que se ya ofrecen en la nube, como el caso de Azure de Microsoft, que les permite acceder a máquinas virtuales, implementar bases de datos, publicar aplicaciones y servicios web, entre otras cosas más.

Enlace de internet: Para el punto anterior resulta necesario contar con un ancho de banda suficiente es fundamental, además para las necesidades

Dados los elementos, características y expectativas citadas, la conformación del espacio físico del Laboratorio de Ingeniería de Software se sintetiza en la siguiente figura.



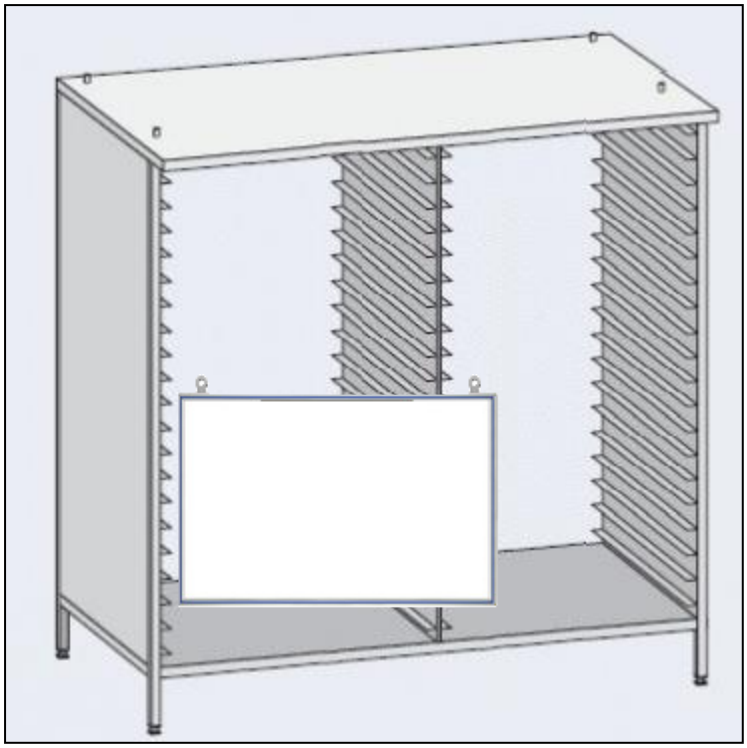
Fuente: Elaboración propia.

Figura 2: Diseño de planta del laboratorio de Ingeniería de Software.

A estas condiciones se añaden otros aspectos de carácter metodológico y formativo, ya que no todo está en el laboratorio físico. Como se citó en las recomendaciones de la industria, es necesario el establecimiento de políticas de operación, la elaboración de

prácticas y documentación complementaria que permitan el correcto desarrollo de las actividades.

Ahora bien, hay otro aspecto a considerar, este espacio no será de uso exclusivo para un cierto grupo de equipos de trabajo, al ser un laboratorio académico estará sujeto al uso compartido propio de ese tipo de instalaciones. Esto atañe nuevos retos, las políticas de operación pueden regular el uso de las instalaciones y espacios, la creación de cuentas individuales en el servidor y el uso de equipos propios de los estudiantes apoyan la fase de codificación, sin embargo, la necesidad de tener el trabajo visible en tableros de control implica un nuevo punto por atender. Esto se planea solventar con la incorporación de tableros desmontables, de manera que cada equipo tenga la certeza que su tablero estará tal cual lo dejó en la sesión anterior, para ello se dispondrá de un contenedor apropiado para la cantidad de tableros requeridos para todos los equipos de todos los grupos que hagan uso del laboratorio.



Fuente: Elaboración propia.

Figura 3: Esquema de contenedor de tableros.

También debe considerarse es el nivel de utilización del laboratorio dentro de los programas académicos actuales y futuros, como la incorporación de un posgrado en el área, donde las competencias específicas que distinguen a un administrador de proyectos y procesos de software implican que éste sea capaz de realizar las siguientes actividades:

1. Acta de constitución y Plan de administración del proyecto.
2. Gestión del alcance y requerimientos de un Proyecto.
3. Gestión de cambios.
4. Gestión de Personal y liderazgo.
5. Gestión y seguimiento a las actividades de un Proyecto.
6. Análisis y Estimación de tiempo y costo de proyectos de software.
7. Estimación de costos.
8. Gestión de la calidad de un Proyecto.
    a) Gestión de la configuración.
    b) Control de un Proyecto.
    c) Reporteo de Status y Medición.
    d) Verificación y Validación.
    e) Generación de lecciones aprendidas y aplicación en proyectos posteriores.
    f) Determinación de mejoras a procesos.
9. Manejo de Conflictos.
10. Negociación efectiva.
11. Cierre de Proyectos.
12. Redacción de informes y comunicación de estado de un Proyecto.
13. Seguimiento de Acuerdos.
14. Gestión de Riesgos.
15. Gestión de Incidencias de un Proyecto.
16. Herramientas de Gestión de Proyectos de software.
17. Gestión de proveedores y contratos.
18. Procesos de Desarrollo de Software.

Derivado de lo anterior, las asignaturas cuyas prácticas en equipo se relacionan con la gestión de proyectos, susceptibles a ser desarrollados en el laboratorio propuesto son:

Para Ingeniería Informática (IINF-2010-220):

- Fundamentos de Sistemas de Información
- Calidad de Sistemas de información
- Análisis y Modelado de Sistemas de Información
- Desarrollo e implementación de sistemas de información

Para Ingeniería en Sistemas Computacionales (ISIC-2010-224):

- Fundamentos de Ingeniería de Software
- Ingeniería de Software
- Gestión de proyectos de software

Maestría en Informática (próxima apertura):

- Gestión de proyectos
- Marcos de desarrollo ágil
- Planificación de proyectos
- Prácticas para el proceso de software
- Modelos de madurez de procesos

Para desarrollar las competencias descritas mediante el seguimiento y organización de los trabajos por grupo, se ha determinado que lo más adecuado es una configuración de cinco equipos de trabajo por grupo, con cinco personas cada uno. Estos rangos están pensados en la medida en que el docente pueda dar seguimiento puntual a los trabajos realizados, ya que es probable que tenga que actuar con el rol de dueño del producto, lo cual implica interacción directa y continua con los equipos. Por otra parte, el tamaño de los equipos corresponde a la recomendación SCRUM para el mismo, el cual establece un mínimo de tres y máximo de nueve integrantes (Schwaber, 2013), a fin de balancear la capacidad productiva y la agilidad de comunicación, de esta manera

el valor de cinco integrantes se encuentra en un punto medio a la recomendación formal del marco SCRUM y resulta afín a lo que en un sentido práctico deja la experiencia docente en lo relacionado a la conformación de equipos de trabajo estudiantiles.

4. Discusión

El seguimiento al proceso formativo también implica ciertos retos de carácter operativo tal como la elaboración de prácticas, lo que aunado a las limitantes de espacio físico y tiempo de los estudiantes representa motivo de estudio a mayor profundidad. Para los grupos que trabajen en este tipo de laboratorios se recomienda un tamaño de 20 a 25 estudiantes por grupo, a fin de poder conformar cuatro equipos de desarrollo los cuales trabajarán en sesiones de dos a cuatro horas continuas para obtener resultados significativos.

Otro aspecto que resulta relevante es que para la correcta operación de un laboratorio de Ingeniería de Software en una IES, es necesario que entre todos los participantes (estudiantes, docentes y “clientes”) exista un nivel de madurez y disposición para la aplicación de los marcos ágiles para el desarrollo de software, ya que con frecuencia se confunde la agilidad con informalidad o superficialidad.

Bibliografía

[1] Escolar, F. & Wigham, J. (2013). *Scrum con TFS*. E. U. A. Programando en .Net
Disponible en: http://programandonet.com/web/scrum-con-tfs/

[2] OTAN. (1969). *Software Engineering*, *Report on a conference sponsored by the NATO SCIENCE COMMITTEE*. Garmisch, Alemania. Organización del Tratado del Atlántico Norte.
Disponible en: http://www.scrummanager.net/files/nato1968e.pdf

[3] Schwaber, K. & Sutherland, J. (2013). *La Guía de Scrum, las reglas del juego*. E. U. A. Scrum.Org

Disponible en:

http://www.scrumguides.org/docs/scrumguide/v1/Scrum-Guide-ES.pdf

[4] Weitzenfeld, A., Guardati, S. (2007). *Introducción a la Computación*. Capítulo 12 Ingeniería de software: el proceso para el desarrollo de software. México: Cengage Learning.

Disponible en:

ftp://ftp.itam.mx/pub/alfredo/PAPERS/WeitzenfeldGuardatiComputacion2008.pdf