Disciplinando Equipos Pequeños con Prácticas Ágiles

Erick Orlando Matla Cruz, Miguel Ehécatl Morales Trujillo, David Velázquez Portilla

División de Estudios de Posgrado de la Facultad de Medicina, Universidad Nacional Autónoma de México.
Unidad de Posgrado , Circuito de Posgrados, Ciudad Universitaria (Zona Cultural), Distrito Federal, México, 04510.
ematla@fmposgrado.unam.mx , davepo@fmposgrado.unam.mx ,http://www.sidep.fmposgrado.unam.mx
Grupo de Investigación KUALI-KAANS, Universidad Nacional Autónoma de México.
Ciudad Universitaria, Distrito Federal, México, 04510.
migmor@ciencias.unam.mx , http://www.kuali-kaans.mx

2014 Published by $\mathcal{DIFU}_{100}ci@$ http://nautilus.uaz.edu.mx/difu100cia

Resumen

En este trabajo se presenta una metodología de desarrollo de software híbrida, basada en SCRUM y la norma ISO/IEC 29110, diseñada por el Departamento de Cómputo de la División de Estudios de Posgrado de la Facultad de Medicina de la UNAM. Esta metodología híbrida pretende guiar a equipos pequeños que desean desarrollar software inmerso en un entorno con requerimientos cambiantes y cliente en sitio, sin descuidar la generación de la documentación adecuada.

Palabras clave: Equipos pequeños, Historia de Usuario, Metodología híbrida, SCRUM.

1. Introducción

En la actualidad la Ingeniería de Software ha cobrado gran importancia debido a la presencia del software en prácticamente cualquier entorno en el que la información debe ser manipulada. La diversidad de organizaciones que basan o apoyan sus procesos y lógicas de negocio en software hace de éste una herramienta indispensable para atender las cada vez más numerosas y específicas necesidades de los clientes.
En consecuencia la creación de software a la medida, desarrollado con el fin de apoyar las actividades de administración y control de datos específicas de un cliente, es un servicio en constante crecimiento al que recurren las organizaciones, ya sea de manera externa, al contratar el desarrollo o de manera interna, asignando el proyecto a un área perteneciente a la misma organización.
Tal es el caso de la División de Estudios de Posgrado de la Facultad de Medicina (DEP-FM) de la Universidad Nacional Autónoma de México, siendo esta División la institución que brinda el mayor número de programas de especializaciones médicas en México. La DEP-FM cuenta con 78 especialidades y 106 sedes académicas en donde son impartidos 601 cursos de especialización por 1539 profesores. Entre sus principales operaciones se tienen: el control de sedes, cursos, profesores y alumnos además de procesos administrativos como el pago de nómina, registro de currículum vitae y otros servicios internos.

Debido al volumen de información y procesos a los que debe hacer frente la DEP-FM, ésta tuvo que recurrir a la alternativa del software como herramienta de apoyo para sus actividades, razón por la cual consolidó el Departamento de Cómputo (DC), organismo interno de la División, quien además de administrar los sistemas existentes y analizar la problemática de los mismos, tiene a su cargo la tarea de desarrollar nuevos sistemas que cumplan con la creciente gama de actividades desarrolladas en la División.
Desde el inicio de su labor, el DC se enfrentó a que sistemas, cuya administración le fue heredada, presentaban oportunidades de mejora en cuanto a cuestiones de integridad de datos, funcionalidad e interacción con el usuario. Estas cuestiones a lo largo del tiempo derivaron en dificultades para sus usuarios y sobre todo para los nuevos administradores/desarrolladores, entre las que destacan:

- La dificultad de encontrar a los stakeholders/usuarios adecuados.
- La poca o nula documentación de sistemas heredados por otras dependencias al DEP-FM
- La alta rotación de personal dedicado al desarrollo de software
- La invariable de contar con un equipo reducido, nunca más de 3 personas, dentro del DC.

Siendo así, la inquietud constante dentro del DC que motivó este trabajo fue buscar una respuesta a:

- ¿Cómo desarrollar “buen” software a la medida?
- ¿Cómo identificar necesidades de cada uno de los clientes y usuarios?
- ¿Cómo responder adecuadamente a los constantes cambios? este último fuertemente ligado al mantenimiento de dichos sistemas.

Bajo el contexto anterior, este trabajo presenta una metodología de desarrollo de software, concebida por el DC de la DEP-FM, que actualmente sirve como guía para analizar, diseñar y construir software bajo las restricciones de requerimientos difusos y cambiantes, cliente en sitio, equipo reducido y necesidad de documentación precisa y actualizada.
La estructura de este artículo incluye en la sección 2 una descripción de las metodologías de desarrollo de software vistas desde la perspectiva tradicional y ágil. En la sección 3 se describe la metodología híbrida de desarrollo propuesta, en la sección 4 son presentados los resultados obtenidos de la aplicación de esta metodología, finalmente se concluye y se proyecta el trabajo futuro.

2. Metodologías de Desarrollo de Software

La Ingeniería de Software es la disciplina de la ingeniería que estudia la naturaleza, los enfoques y metodologías de desarrollo a gran escala del software [1]. La Ingeniería de Software es un proceso que involucra métodos, herramientas y técnicas que guían a un equipo de trabajo a lo largo del proceso de desarrollo de software y que tienen como fin la construcción de software.
De manera genérica, un proceso de desarrollo de software se compone de 6 etapas: Análisis, Diseño, Construcción, Integración, Pruebas y Liberación. Cada una de las cuales agrupa actividades específicas que son ejecutadas de acuerdo a la metodología elegida por el equipo de trabajo.
Una metodología de desarrollo de software es el marco utilizado para estructurar, organizar y controlar el proceso para producir software, las metodologías de desarrollo de software pueden agruparse de acuerdo a dos grandes enfoques: Ágil o Tradicional. En las siguientes subsecciones se explicará cada una de ellas.

2.1. Metodologías Ágiles

Diseñadas principalmente para equipos reducidos en contextos donde es posible involucrar al cliente como parte integral del equipo de desarrollo, estas metodologías siguen un desarrollo iterativo e incremental en donde el progreso es medido de acuerdo al software funcional liberado al término de cada iteración.
Las iteraciones se dan en periodos cortos de tiempo y buscan minimizar el impacto de los cambios a los largo del proyecto. La filosofía ágil potencializa al equipo y le otorga un mayor control sobre sus actividades, permitiéndole a sus integrantes concentrarse en entregar software funcional de calidad de manera ágil.
Las ideas anteriores se derivan del Manifiesto Ágil [2], escrito que concentra la filosofía en la que se basan metodologías ágiles como SCRUM [3], XP [4], KANBAN [5] o Lean [6], siendo SCRUM la más popular y el ejemplo más representativo.
SCRUM se define como un proceso iterativo e incremental, dirigido por un conjunto de prácticas y roles que sirven como base para la planeación de un proyecto.
De acuerdo con [3] estos roles son:

- Product Owner (PO) quien representa la voz del

cliente y se asegura de que el equipo trabaje de forma adecuada desde la perspectiva del negocio.

- SCRUM Master (SM) cuyo trabajo es eliminar los obstáculos que impiden que el equipo alcance el objetivo del sprint.

- Equipo de Trabajo que tiene la responsabilidad de entregar el producto de software funcionando.

El desarrollo en SCRUM se divide en iteraciones cortas llamadas *sprints*, tras cada sprint el equipo de desarrollo entrega un producto funcional al cliente. Las características que debe cumplir dicho entregable se encuentran especificadas en un documento llamado: *Product Backlog*. Además existen reuniones diarias *Daily SCRUM* o *Stand-up meeting* en las cuales los integrantes del equipo de desarrollo hacen del conocimiento del SM los problemas encontrados o bien los avances que se van dando en el desarrollo del proyecto. El *Daily SCRUM* como su nombre lo indica, se realiza todos los días, su duración no excederá los 15 minutos y cada miembro del equipo debe responder 3 preguntas específicas:

- ¿Qué has hecho desde ayer?

- ¿Qué es lo que harás hasta la reunión de mañana?

- ¿Has tenido algún problema quen te haya impedido alcanzar tu objetivo?

Otra particularidad de las metodologías ágiles es la manera en que se da el levantamiento de requerimientos, ya que en este tipo de metodologías se aprovecha el hecho de que el cliente es parte del equipo de trabajo. Una técnica popular para realizar esta actividad son las Historias de Usuario (HU) [7], las cuales son capturadas en tarjetas que contienen por un lado, la descripción del requerimiento en lenguaje coloquial debido a que son narradas y escritas por el cliente que requiere la funcionalidad. Mientras que por el otro lado se indica cómo será probada dicha funcionalidad ya que ésta sea implementada.
Para fines de este trabajo, se utilizaron varios aspectos de SCRUM para realizar las actividades de administración de proyectos del DC, tales como la adopción de los roles de PO y SM, el control basado en reuniones diarias, inicialmente el *Daily SCRUM*, así como una propuesta de variante de las HU que es utilizada para delimitar el alcance funcional del sistema así como para proveer de trazabilidad a cada uno de los elementos que se construyan.

2.2. Metodologías Tradicionales

Las metodologías tradicionales se centran en el proceso, guían sobre el qué hacer en cada una de las etapas de desarrollo, concentrando gran parte del desarrollo en la generación de documentación exhaustiva. Estas metodologías asignan un alto costo a la implementación de cualquier cambio, principalmente causado por el ciclo de vida en el que se desenvuelven. El ejemplo más conocido de este tipo de metodologías es el Proceso Unificado (PU) [8], el cual se define como un proceso basado en componentes y centrado en la arquitectura, que utiliza el Lenguaje Unificado de Modelado (UML) para detallar todos los esquemas del software a desarrollar. Además del PU, existen modelos de referencia de procesos como CMMI [9] y estándares como la ISO/IEC 12207 [10] que definen al conjunto de procesos relacionados con el desarrollo de software. Sin embargo desde sus orígenes este tipo de modelos y estándares han sido recibidos de mejor manera por organizaciones grandes y no así por las pequeñas. Debido a esto, en los últimos años se han realizado esfuerzos para acercar la definición de estándares a organizaciones pequeñas con el fin de apoyar sus procesos, el más significativo de estos esfuerzos ha sido la norma ISO/IEC 29110 [11]. La ISO/IEC 29110 es un estándar internacional desarrollado específicamente para entidades muy pequeñas (VSE-Very Small Entities) encargadas de construir software. Una VSE se define como una entidad que tiene menos de 25 personas. Esta norma se divide en dos procesos, definidos de la siguiente manera [11]:

- **Implementación del Software:** Describe conceptos de procesos, ciclo de vida y estandarización para la construcción del software.

- **Administración del proyecto:** Describe el desarrollo de software de una sola aplicación por un solo equipo de proyecto sin riesgos o factores situacionales especiales.

Para fines de este trabajo, la norma ISO/IEC 29110 sirvió como base para la definición de qué actividades y productos de trabajo que realizaría el DC a lo largo del desarrollo de software para la DEP-FM.

3. Metodología Híbrida DC-DEP-FM

Si bien las metodologías tradicionales indican qué actividades realizar y qué productos de trabajo generar, no todos los proyectos a los que una organización hace frente tienen las mismas restricciones de rigurosidad en cuanto al control de actividades o utilidad de la documentación.

Por otra parte, una metodología ágil podría relajar demasiado las cosas ocasionando que el control y orden se pierda. De acuerdo con esto y bajo el contexto presentado en la DEP-FM se buscó una solución para lograr abarcar las necesidades locales de documentación, mantenimiento y desarrollo de software. Lo anterior llevado a cabo por un equipo reducido y tomando ventaja de que la mayoría de los usuarios y clientes de los sistemas en cuestión se encuentran físicamente en el mismo lugar de trabajo, resultando en la creación de una metodología híbrida con las siguientes características:

- El desarrollo de software estará guiado por la norma ISO/IEC 29110.
- Las actividades de la norma se podrán llevar a cabo de manera: Ágil, Tradicional o Híbrida.
- Las actividades de Implementación del Software que involucren documentación se llevarán a cabo siguiendo el PU.
- Las actividades de Administración del Proyecto que involucran reuniones o acuerdos se llevarán a cabo de manera ágil durante el Daily SCRUM.
- Se involucrará como parte del equipo a los clientes y usuarios, tomando el rol de PO. Serán ellos quienes comiencen con la creación de las historias de usuario adaptadas.
- Los productos de trabajo se concentrarán en las historias de usuario adaptadas, cuyo formato permitirá incluir lo requerido por los productos de trabajo de la norma ISO/IEC 29110 pero su creación contará con la participación activa de los usuarios.

De acuerdo a los puntos expuestos, esta propuesta se trata de una metodología híbrida, ya que la guía será la norma ISO/IEC 29110, que es tradicional, pero las actividades de administración del proyecto así como la especificación de requerimientos, pruebas y liberación se llevarán a cabo mediante SCRUM e HU adaptadas, esto último sustentándose en lo que una metodología ágil señala.

3.1. Historias de Usuario Adaptadas

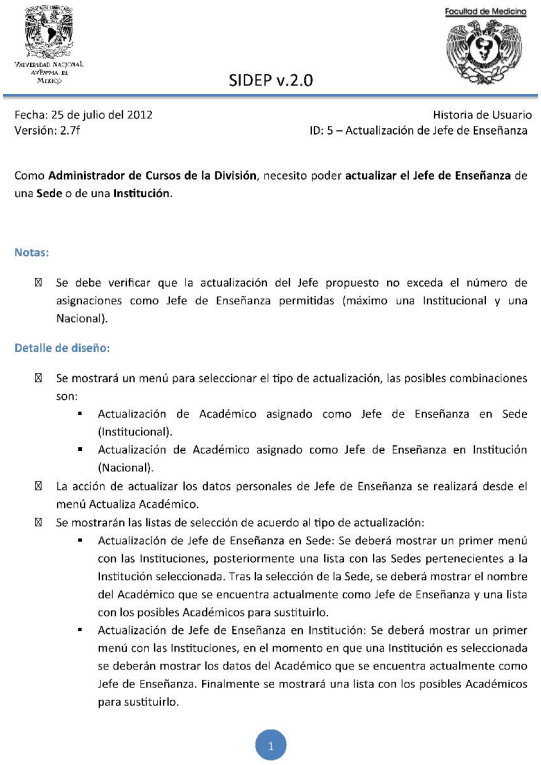
Inspiradas en las HU y reutilizando la idea original de Cohn de representar un requisito de software escrito en una o dos frases utilizando el lenguaje común del usuario [7], se crearon las HU adaptadas. Las HU adaptadas incluyen apartados que permiten atender las necesidades de documentación del equipo y aquellas solicitadas por la norma ISO/IEC 29110. La estructura resultante de las HU adaptadas es la siguiente, ver Figura 1:

- **Fecha, Identificador y Versión.**
- **Requerimientos:** Simulará la parte frontal de una HU convencional, de esta manera, es aquí donde el cliente detallará con sus propias palabras la funcionalidad solicitada.
- **Notas:** En este espacio el equipo de trabajo realizará anotaciones para aclarar el requerimiento o bien registrar observaciones propias.
- **Detalle de diseño:** En esta zona se podrán mostrar bocetos de lo que será el sistema, el equipo de trabajo explicará mediante narrativa o prototipos el cómo se pretende realizar la implementación.
- **Pruebas:** Indicará las fechas en las que se obtienen los vistos buenos del equipo de trabajo y del cliente para cada módulo.
- **Notas de rechazo:** En caso de que el cliente rechace el módulo, se detallará el motivo y, en caso de ser necesario, se incluirá el error o defecto presentado.
- **Detalle de la prueba:** Esta área de la HU simulará la parte posterior de la tarjeta de HU, aquí el cliente detallará en su lenguaje la manera en la que llevará a cabo la prueba del módulo que implementa la HU.
- **Registro de rastreo:** Servirá para que el equipo de trabajo detalle con qué otras HU se encuentra relacionada.

Apartados como *Requerimientos* y *Detalle de la Prueba* son preservados de las HU originales, sin embargo secciones como *Detalle del Diseño* y *Registro de Rastreo* fueron agregadas para dar orden al desarrollo y facilitar futuros mantenimientos del sistema a construir. Finalmente los apartados de *Notas* y *Notas de Rechazo* permitieron clarificar y mejorar el entendimiento de las necesidades del cliente/usuario con el equipo de desarrollo, ocasionando que se obtuviera un mejor producto de software.

3.2. Reuniones Diarias

ISO/IEC 29110 sugiere, al inicio de cada una de sus fases, asignar las tareas al equipo de trabajo de acuerdo al rol que desempeñan, dado que en el DC se tiene un equipo reducido y que todos se encuentran en el mismo lugar de trabajo, se decidió implementar el *Daily SCRUM*.



SIDEP v.2.0

Fecha: 25 de julio del 2012
Versión: 2.7f

Historia de Usuario
ID: 5 – Actualización de Jefe de Enseñanza

Como **Administrador de Cursos de la División**, necesito poder **actualizar el Jefe de Enseñanza** de una **Sede** o de una **Institución**.

Notas:

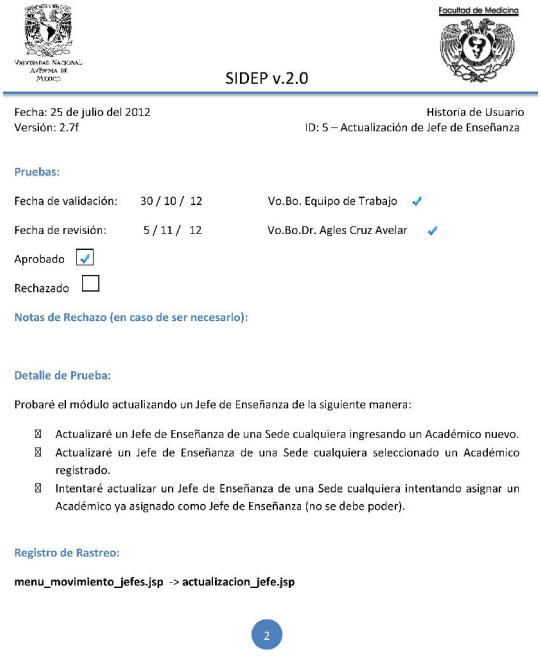
- Se debe verificar que la actualización del Jefe propuesto no exceda el número de asignaciones como Jefe de Enseñanza permitidas (máximo una Institucional y una Nacional).

Detalle de diseño:

- Se mostrará un menú para seleccionar el tipo de actualización, las posibles combinaciones son:
  - Actualización de Académico asignado como Jefe de Enseñanza en Sede (Institucional).
  - Actualización de Académico asignado como Jefe de Enseñanza en Institución (Nacional).
- La acción de actualizar los datos personales de Jefe de Enseñanza se realizará desde el menú Actualiza Académico.
- Se mostrarán las listas de selección de acuerdo al tipo de actualización:
  - Actualización de Jefe de Enseñanza en Sede: Se deberá mostrar un primer menú con las Instituciones, posteriormente una lista con las Sedes pertenecientes a la Institución seleccionada. Tras la selección de la Sede, se deberá mostrar el nombre del Académico que se encuentra actualmente como Jefe de Enseñanza y una lista con los posibles Académicos para sustituirlo.
  - Actualización de Jefe de Enseñanza en Institución: Se deberá mostrar un primer menú con las Instituciones, en el momento en que una Institución es seleccionada se deberán mostrar los datos del Académico que se encuentra actualmente como Jefe de Enseñanza. Finalmente se mostrará una lista con los posibles Académicos para sustituirlo.

1

(a) Parte 1



SIDEP v.2.0

Fecha: 25 de julio del 2012
Versión: 2.7f

Historia de Usuario
ID: 5 – Actualización de Jefe de Enseñanza

Pruebas:

Fecha de validación: 30 / 10 / 12 Vo.Bo. Equipo de Trabajo ✓

Fecha de revisión: 5 / 11 / 12 Vo.Bo.Dr. Agles Cruz Avelar ✓

Aprobado ☑

Rechazado ☐

Notas de Rechazo (en caso de ser necesario):

Detalle de Prueba:

Probaré el módulo actualizando un Jefe de Enseñanza de la siguiente manera:

- Actualizaré un Jefe de Enseñanza de una Sede cualquiera ingresando un Académico nuevo.
- Actualizaré un Jefe de Enseñanza de una Sede cualquiera seleccionado un Académico registrado.
- Intentaré actualizar un Jefe de Enseñanza de una Sede cualquiera intentando asignar un Académico ya asignado como Jefe de Enseñanza (no se debe poder).

Registro de Rastreo:

menu_movimiento_jefes.jsp -> actualizacion_jefe.jsp

2

(b) Parte 2

Figura 1. Historia de Usuario Adaptada

Los *Daily SCRUM* son aprovechados para asignar tareas y mantener informado al resto del equipo sobre el progreso del proyecto, así como para expresar irregularidades u obstáculos en el desarrollo del mismo. En particular, en el DC, dichas reuniones son llevadas a cabo de manera diaria a las 10:00 en la Sala de Juntas de la Subdivisión de Especialidades Médicas de la DEP-FM, participando los miembros del equipo de desarrollo y en caso de ser necesaria la toma de acuerdos o resolución de dudas, se involucra al cliente o a aquellos usuarios significativos.

4. Resultados

Como resultado de la aplicación de la metodología híbrida planteada el DC ha desarrollado dos sistemas nuevos: Profesores.v.2.0 que permite la administración de cursos, sedes y profesores de cursos de Especialidad y Alta Especialidad Médica; y PNPC.v.1.0 que permite la consulta de los programas que solicitan evaluación en el Programa Nacional de Posgrados de Calidad del CONACYT.
Ambos sistemas han tenido un impacto positivo en sus usuarios y administradores, ya que se ha logrado que cumplan con las siguientes características:

- **Mantenimiento y Operación:** Los sistemas cuentan con documentación adecuada, lo que ha permitido responder de manera eficiente al mantenimiento y operación de los mismos.
- **Usabilidad:** Los usuarios, nuevos y existentes, han recibido de mejor manera a los sistemas, ya que al incluirlos en el proceso de desarrollo, ha permitido al DC construir soluciones más cercanas a los usuarios.
- **Funcionalidad:** se ha logrado desarrollar sistemas con las funcionalidades que realmente apoyan al usuario en su labor diaria, debido a que éstas fueron concebidas en conjunto entre el equipo y el cliente.

En conjunto, los puntos anteriores permitieron responder satisfactoriamente a las tres preguntas que guiaron esta propuesta.
Por otro lado, el DC ha identificado las siguientes mejoras en el equipo de trabajo:

- Los alumnos prestadores de servicio social en la DEP-FM se familiarizan con mayor rapidez a los procesos del DC. Cabe resaltar que su inclusión en los Daily SCRUM los mantiene informados y les provoca un sentido de pertenencia al DC.

- Las HU adaptadas han resultado ser una herramienta poderosa para evitar inconsistencias entre la lógica del negocio y su implementación, incluso han permitido utilizar tiempos de desarrollo para la inclusión de nuevas funcionalidades en lugar de la corrección de las existentes por no "entende" los requerimientos del cliente.

Finalmente como oportunidades de mejora para la metodología, se puede mencionar que sería deseable contar con un sistema para la captura y administración de las HU adaptadas, ya que conforme el número de éstas se incrementa se dificulta su manejo. Por otra parte, para fines estadísticos solicitados por la institución, sería conveniente tener un registro y seguimiento de métricas de esfuerzo empleado.

5. Conclusiones

Tras el desarrollo de este proyecto se puede reafirmar la importancia que hoy en día tiene la Ingeniería de Software para el correcto desarrollo de software y sistemas. Este trabajo demostró que la aplicación de una metodología híbrida permitió transformar problemáticas específicas del DC como lo eran el personal reducido, sistemas sin documentación y la frecuente presencia del cliente solicitando cambios, en ventajas: equipo pequeño bien comunicado, ágil asignación de actividades, documentación adecuada y estandarizada, integración del cliente al equipo de trabajo y productos funcionales acorde a las necesidades de los usuarios.
Siendo lo anterior consecuencia de haber aprovechado las bondades de ambos tipos de metodologías, tradicionales y ágiles, culminando los proyectos con productos de mejor calidad, reflejándose en aprendizaje y satisfacción para las partes involucradas.
Como trabajo futuro, el DC continuará aplicando la metodología propuesta en sus proyectos de desarrollo, lo que permitirá enfrentar sus debilidades y fortalezas, se reforzará la parte correspondiente a métricas y se capacitará a los prestadores de servicio social dotándolos de técnicas para el uso adecuado de las historias de usuario y el manejo de clientes.

Referencias

[1] Y. Wang, *Software Engineering Foundations: A Software Science Perspective*. Auerbach Publications, 2008

[2] Agile Alliance (2011) "The Manifesto for Agile Software Development". http://www.agilealliance.org/the-alliance/the-agile-manifesto/

[3] K. Schwaber, J. Sutherland. (2011) "The Scrum Guide - The Definitive Guide to Scrum: The Rules of the Game". http://www.scrum.org/

[4] M. Stephens, D. Rosenberg, *Extreme Programming Refactored: The Case against XP*. Apress, 2003

[5] D. Anderson, *Kanban: Successful Evolutionary Change for Your Technology Business*. Blue Hole Press, 2010

[6] M. Poppendieck, M. Cusumano. (Septiembre-Octubre, 2012) "Lean Software Development: A Tutorial". IEEE Software, No. 5, Vol. 29, pp. 26-32. Septiembre-Octubre 2012.

[7] M. Cohn, *Agile Estimating and Planning*. Prentice Hall, 2006.

[8] J. Rumbaugh, I. Jacobson, G. Booch, *El Proceso Unificado de Desarrollo de Software*. Pearson, 2000

[9] C. Beth, M. Konrad, S. Shrum, *CMMI for Development: Guidelines for Process Integration and Product Improvement*. Pearson, 2012

[10] ISO/IEC 12207:2008 Systems and software engineering - Software life cycle processes. International Organization for Standardization, 2008.

[11] ISO/IEC 29110-5-1-2:2012 Software engineering - Lifecycle profiles for Very Small Entities (VSEs) - Management and Engineering Guide: Generic profile group: Basic Profile. International Organization for Standardization, 2012.