# DESARROLLANDO LA INNOVACIÓN Y LA CREATIVIDAD EN ESTUDIANTES DE INGENIERÍA DE SOFTWARE

José Miguel Rubio, Universidad de Playa Ancha, jose.rubio.l@upla.cl
Franklin Johnson, Universidad de Playa Ancha, franklin.johnson@upla.cl
Mario Bruno, Universidad de Playa Ancha, mbruno@upla.cl

## RESUMEN

El desarrollo de software es una actividad que requiere de conocimientos y su éxito depende de la creatividad y capacidad de innovación de los desarrolladores. En los últimos años la perspectiva tradicional de desarrollo de software está cambiando y los métodos agiles han recibido una atención considerable. Entre otros atributos, sus precursores afirman que el conocimiento fomentando la innovación y la creatividad es una de las claves para dar respuesta a los problemas y retos del desarrollo de software en la actualidad. El desarrollo de nuevos productos de software requiere de la generación de ideas nuevas y útiles. El propósito de este documento es explicar la innovación y la creatividad en relación con las nuevas tendencias y la formación en ingeniería de software. Se consideran las implicaciones de estos hallazgos, y se sugieren algunas directrices para investigación futura.

PALABRAS CLAVES: Ingeniería de software, Metodologías agiles, Innovación y creatividad, Trabajo colaborativo, Gestión del conocimiento.

## INTRODUCCIÓN

La Ingeniería de Software es una disciplina basada en el conocimiento, donde sus actividades requieren el uso y el intercambio de conocimientos entre los grupos de interés. Por lo tanto, una mejor transferencia y aplicación del conocimiento permite fortalecer los procesos de software, ya sea al hacer uso de enfoques tradicionales o ágiles. En las organizaciones de desarrollo de software el conocimiento en manos de los empleados es el principal activo y los proyectos de desarrollo de software dependen en gran medida del rendimiento del equipo: "el Software es desarrollado para las personas y por las personas" (John et al., 2005). Pero, sorprendentemente, la mayoría de la formación en Ingeniería de Software es técnica y con menos énfasis en aspectos humanos y sociales. Por otro lado, el proceso tradicional de desarrollo de nuevos productos, que es parte fundamental del marketing, también ha sido criticado por Kotler y Trías de Bes (Kotler y Trías de Bes, 2004). Ellos señalan que no se consideran en absoluto aspectos creativos fundamentales y, como consecuencia, este desarrollo no es útil, viable o innovador. En este contexto, es interesante tener en cuenta las nuevas propuestas de las metodologías ágiles para el desarrollo de software con el fin de analizar y evaluarlas a la luz de las propuestas de creatividad existentes, teniendo en cuenta sobre todo las prácticas de trabajo en equipo para la formación en ingeniería de software.

Los principios y valores ágiles han enfatizado la importancia de la colaboración y la interacción en el desarrollo de software y, por otro lado, el trabajo creativo generalmente implica la colaboración de alguna forma y se puede entender como la interacción entre una persona y un contexto sociocultural. En relación al trabajo conjunto entre los usuarios y los desarrolladores de software, hay casos muy interesantes en empresas de salud y en la medicina. La relación entre los enfoques ágiles y el sector de la salud tiene un caso notable con el trabajo de Jeff Sutherland, el inventor del popular proceso de desarrollo ágil Scrum (Sutherland, 2001). Scrum, es el competidor más destacado de eXtreme Programming XP (Beck, 2000), que ha alcanzado fama

mundial por su capacidad de aumentar la productividad de los equipos de software a través de potenciar a las personas, el fomento de un ambiente orientado al equipo, y centrado en la transparencia y resultados del proyecto. Hay estudios recientes que reportan los esfuerzos para mejorar el proceso ágil. El desarrollo de software ágil se ocupa de la mejora de los procesos de software mediante los equipos. El trabajo en (Ringstad et al., 2011) y (Moe et al., 2010) propone el uso del diagnóstico y la planificación de acciones para mejorar el trabajo en equipo en el desarrollo de software ágil. La planificación de acciones se centra en mejorar el intercambio del liderazgo, la orientación hacia el equipo y el aprendizaje. El mejoramiento del proyecto prevé una más nueva visión para un equipo maduro. La innovación y el desarrollo de nuevos productos es un tema interdisciplinario (Takeuchi y Nonaka, 1986) (Nonaka y Takeuchi, 1995), que interesa en el estudio del potencial de nuevos conceptos y técnicas para fomentar la gestión del conocimiento y la creatividad en ingeniería de software (Gu y Tong, 2004).

Este trabajo está organizado de la siguiente manera: la sección 2 presenta antecedentes y conceptos generales sobre creatividad en el desarrollo de software. La sección 3 se dedica a la presentación de la gestión del conocimiento en Ingeniería de Software. Se incluye una breve introducción a los conceptos básicos del área de gestión del conocimiento en la sección 4, y se presentan sus dos enfoques: productos y procesos. En la sección 5 se describe el proceso creativo propuesto. Por último, en la sección 6 se concluye el artículo y se entregan algunas perspectivas para futuras investigaciones.

## CREATIVIDAD EN EL DESARROLLO DE SOFTWARE

La Ingeniería de Software es un proceso intensivo en conocimientos que incluye algunos aspectos de gestión del conocimiento y de creatividad en todas las fases: especificación de requisitos, diseño, construcción, pruebas, puesta en marcha, mantenimiento y gestión del proyecto (John et al., 2005). Ningún trabajador de un proyecto de desarrollo posee todos los conocimientos necesarios para cumplir con todas las actividades. Esto destaca la necesidad de comunicación, colaboración y apoyo para el intercambio de conocimientos, y para compartir la experiencia del dominio entre el cliente y el equipo de desarrollo (Chau et al., 2003).

Los enfoques tradicionales (a menudo denominados como dirigidos por el plan, basados en tareas o burocráticos), como el modelo cascada y sus variantes, facilitan el intercambio de conocimiento principalmente a través de la documentación. También promueven el uso de equipos basados en roles y planes detallados de todo el ciclo de vida de desarrollo de software. Esto cambia el enfoque desde los individuos y sus habilidades creativas hacia los procesos. Por el contrario, los métodos ágiles hacen hincapié en el valor de los individuos y las interacciones por sobre los procesos. Los métodos tradicionales usan fuerte y rigurosamente la documentación para capturar el conocimiento adquirido en las actividades del ciclo de vida del proyecto de software (Chau y Maurer, 2004). En contraste, los métodos ágiles sugieren que la mayoría de la documentación escrita puede ser reemplazada por el mejoramiento de las comunicaciones informales internas entre los miembros del equipo y entre el equipo y los clientes, con un mayor énfasis en el conocimiento tácito en lugar del conocimiento explícito (Beck et al, 2001).

Dado que la creatividad humana se considera como la fuente para resolver problemas complejos o crear productos innovadores, una de las posibilidades para mejorar el proceso de desarrollo de software es diseñar un proceso que pueda estimular la creatividad de los desarrolladores. No son pocos los estudios sobre la importancia de la creatividad en el desarrollo de software. En gestión y en negocios, los investigadores han trabajado mucho sobre la creatividad y se ha obtenido evidencia de que los empleados que tenían características apropiadas de creatividad, trabajaron en tareas complejas y desafiantes, y fueron supervisados con apoyo, de forma no controlada,

produciendo trabajo más creativo. Por lo tanto, de acuerdo con las ideas anteriores, el uso de la creatividad en el desarrollo de software es innegable, pero la ingeniería de requisitos no es reconocida como un proceso creativo en todos los casos (Maiden et al., 2004). En sólo algunas publicaciones la importancia de la creatividad se ha investigado en todas las fases del proceso de desarrollo de software (Gu y Tong, 2004) (Glass, 1995) (Crawford y León de la Barra, 2007) (León de la Barra y Crawford, 2007) (Crawford et al., 2008) (Crawford y León de la Barra, 2008) y más focalizada en la ingeniería de requisitos en (Robertson, 2005) y (Mich et al., 2005). Sin embargo, el uso de técnicas para fomentar la creatividad en la ingeniería de requisitos está todavía poco desarrollado. No es de extrañar que el papel de la comunicación y la interacción es central en muchas de las técnicas de creatividad. La técnica de creatividad más utilizada popularmente para la identificación de requisitos es la "lluvia de ideas" clásica (brainstorming) y, más recientemente, se han aplicado como un intento de traer más creatividad para la obtención de requisitos, los escenarios basados en juegos de roles, escenarios basados en historias de usuario, la simulación y visualización. Estas técnicas tratan de abordar el problema de la identificación de los puntos de vista de todas las partes interesadas (Mich et al., 2005).

Sin embargo, en la ingeniería de requisitos las respuestas no llegan por sí solas, es necesario preguntar, observar, descubrir y crear cada vez más requisitos. Si el objetivo es construir productos más competitivos y originales, la creatividad debe formar parte del proceso de requisitos. De hecho, la importancia del pensamiento creativo se espera que aumente en la próximos década (Maiden y Gizikis, 2001) (Brown, 2008). En (Robertson, 2005) (Robertson, 2002) se proponen preguntas abiertas muy interesantes: ¿Se está inventando parte de la actividad de requisitos? Si se trata de querer avanzar. Entonces, ¿quién hace la invención? No podemos confiar en que el cliente sabe qué inventar. El diseñador ve su tarea como la de encontrar la solución óptima a los requisitos establecidos. No podemos depender de los programadores, ya que están muy lejos del trabajo del cliente para entender lo que hay que inventar. Los analistas de requisitos están en una posición ideal para innovar. Ellos entienden el problema de negocio, tienen conocimientos de tecnología actualizados, serán culpados si el nuevo producto no complace al cliente, y saben si las invenciones son apropiadas para el trabajo que se está estudiando. En resumen, los analistas de requisitos son las personas cuyas habilidades y posición permite, de hecho, fomenta la creatividad. En (Boden, 2004) el autor, una autoridad líder en creatividad cognitiva, identifica los tipos básicos de procesos creativos: la creatividad exploratoria explora una posible solución en el espacio y descubre nuevas ideas, la creatividad combinatoria combina dos o más ideas que ya existen para crear nuevas ideas y la creatividad transformacional cambia el espacio de soluciones para hacer posibles cosas imposibles. Por lo tanto, la mayoría de las actividades de la ingeniería de requisitos son de exploración, adquisición y descubrimiento de requisitos y conocimientos sobre el dominio del problema. Los profesionales de la ingeniería de requisitos están centrados explícitamente en la creatividad combinatoria y transformacional.

## GESTIÓN DEL CONOCIMIENTO EN INGENIERÍA DE SOFTWARE

El principal argumento para la Gestión del Conocimiento en la ingeniería de software es que es una actividad centrada en el conocimiento y en el ser humano. El desarrollo de software es un proceso en el que cada persona involucrada tiene que tomar un gran número de decisiones y el conocimiento individual tiene que ser compartido y movilizado hacia un proyecto y nivel organizacional, y esto es exactamente lo que propone la gestión del conocimiento. Las personas en estos grupos deben colaborar, comunicar y coordinar su trabajo, lo que hace la gestión del conocimiento una necesidad. En el desarrollo de software se pueden identificar dos tipos de conocimiento: El conocimiento incorporado en los productos o artefactos, ya que son el resultado de actividades altamente creativas y el meta-conocimiento, que es el conocimiento acerca de los

productos y procesos. Algunas de las fuentes del conocimiento (artefactos, objetos, componentes, patrones, plantillas y contenedores) se almacenan en forma electrónica. Sin embargo, la mayoría de los conocimientos es tácita, y reside en el cerebro de los empleados. Una manera para hacer frente a este problema puede ser el desarrollo de una cultura de intercambio de conocimiento, así como el apoyo de tecnología para la gestión del conocimiento. Hay varias razones para creer que la gestión del conocimiento en la ingeniería de software sería más fácil de implementar que en otras organizaciones: la tecnología no es intimidante para los ingenieros de software y creen que las herramientas los ayudarán a hacer un mejor trabajo; todos los artefactos ya están en formato electrónico y pueden ser fácilmente distribuidos y compartidos; y el hecho de que el intercambio de conocimientos entre ingenieros de software ya se produce en gran medida en muchos proyectos exitosos de software colaborativo (Rus y Lindvall, 2002) .

## ESTRATEGIA PARA LA GESTION DEL CONOCIMIENTO

La Gestión del Conocimiento se centra en el conocimiento corporativo como un activo fundamental de la empresa y sus objetivos son el uso óptimo y el desarrollo de este activo, ahora y en el futuro. La gestión del conocimiento ha sido tema de mucha discusión y se han propuesto diferentes ciclos de vida y estrategias de gestión del conocimiento. Uno de los métodos más aceptados para clasificar el conocimiento desde una perspectiva de gestión del conocimiento es la matriz de conocimiento de Nonaka y Takeuchi (Nonaka y Takeuchi, 1995). Esta matriz clasifica conocimiento, ya sea explícito o tácito, tanto individual como colectivo. Nonaka y Takeuchi también proponen los procesos de conocimiento correspondientes que transforman conocimientos de una forma a otra: la socialización (de tácito a tácito, por el cual un individuo adquiere conocimiento tácito directamente de otros a través de la experiencia compartida, la observación, la imitación y así sucesivamente); la externalización (de tácito a explícito, a través de la articulación del conocimiento tácito en conceptos explícitos ); la combinación (de explícito a explícito, a través de una sistematización de los conceptos materializados en diversos cuerpos de conocimiento explícito), y la internalización (de explícito a tácito, a través de un proceso de aprendizaje por la práctica y a través de la verbalización y documentación de experiencias). Nonaka y Takeuchi modelan el proceso de creación de conocimiento organizacional como una espiral en la que el conocimiento se amplifica a través de estos cuatro modos de conversión del conocimiento (Ver Figura 1). Aquí también se considera que el conocimiento se comienza a cristalizar dentro de la organización en los niveles superiores moviéndose desde lo individual hacia lo grupal a través de la organización e incluso los niveles inter-organizacionales (Bueno, 2003).

Como se mencionó anteriormente, los métodos tradicionales de desarrollo de software utilizan una gran cantidad de documentación para capturar el conocimiento adquirido en las actividades del ciclo de vida del proyecto. En contraste, los métodos ágiles sugieren que la mayor parte de la documentación escrita puede ser reemplazada por una mejor comunicación informal entre los miembros del equipo y los clientes, con un mayor énfasis en el conocimiento tácito más que en el conocimiento explícito. En el ámbito de la gestión del conocimiento existe una situación similar y se han empleado principalmente dos enfoques, nos referiremos a ellos como el enfoque en el Producto y el enfoque en el Proceso. Estos enfoques adoptan diferentes perspectivas en relación a la documentación y las interacciones entre los interesados (Mentzas, 2000):

a) El conocimiento como un producto. El enfoque en el producto implica que el conocimiento puede ser localizado y manipulado como un objeto independiente. Los defensores de este enfoque señalan que es posible capturar, distribuir, medir y gestionar el conocimiento. Este enfoque se centra principalmente en los productos y objetos que contengan y representen el conocimiento.

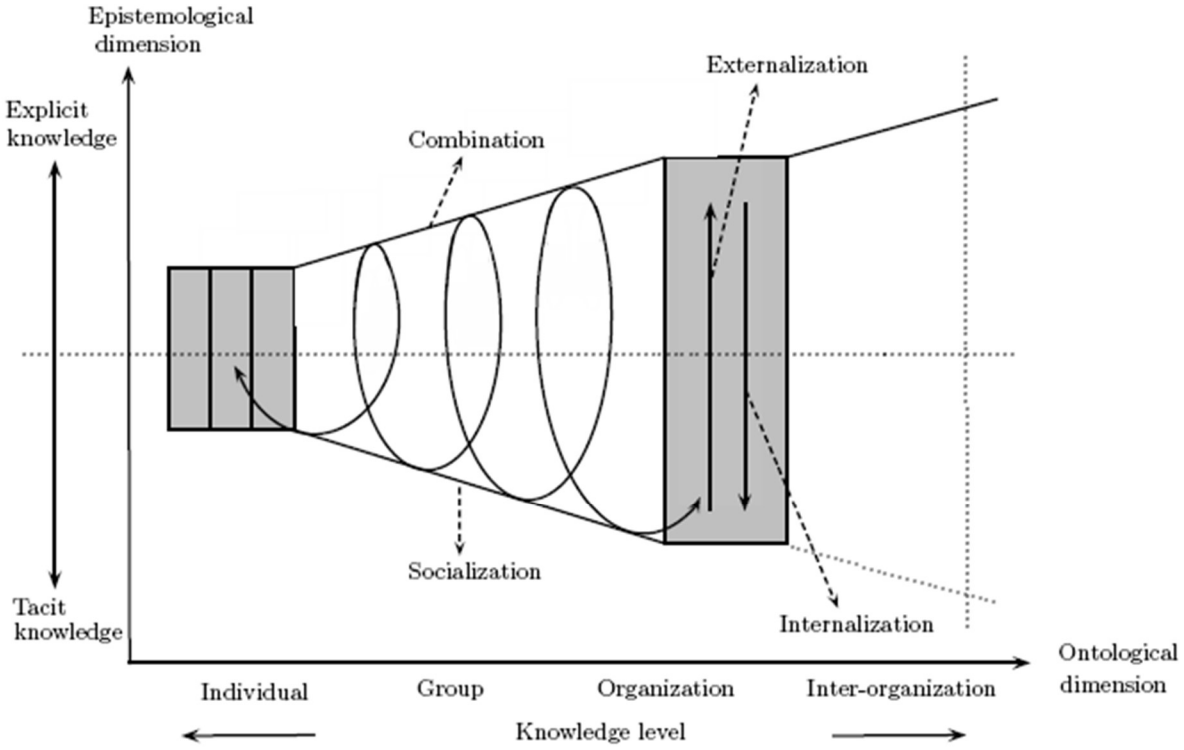


Figura N° 1. Espiral de creación de conocimiento organizacional.

b) El conocimiento como un proceso: El enfoque basado en el proceso pone énfasis en las formas de promover, motivar, animar, alimentar o guiar el proceso de aprendizaje, y elimina la idea de tratar de capturar y distribuir conocimiento. Este punto de vista entiende principalmente la gestión del conocimiento como un proceso de comunicación social, que puede mejorar con la colaboración y cooperación de herramientas de apoyo. En este enfoque, el conocimiento está estrechamente vinculado a la persona que lo desarrolló y se comparte de persona a persona principalmente por medio de la interacción. Este enfoque también se ha denominado como el enfoque de "colaboración" o "personalización".

La elección de uno u otro enfoque debe ser en relación con las características de la organización, el proyecto y las personas involucradas en cada caso (Apostolou y Mentzas, 2003).

## RESULTADOS

Un proceso creativo constituye el aspecto central del rendimiento del equipo, ya que supone una serie de fases claramente diferenciadas que tienen que realizarse por uno o más de los miembros del equipo con el fin de obtener un resultado concreto.

Las fases -basándose en Wallas (Wallas, 1926) y Leonard y Swap (Leonard y Swap, 1999)- son las siguientes:

a) Preparación inicial: la creatividad florecerá cuando el terreno mental, es profundo, fértil y tiene una preparación adecuada. Por lo tanto, el conocimiento profundo y relevante y la experiencia precede a la expresión creativa.
b) Encuentro: los descubrimientos correspondientes a la percepción de una situación problemática. Para esta situación no existe una solución. Se trata de un problema nuevo.
c) Preparación final: corresponde al entendimiento y los fundamentos del problema. Es la inmersión en el problema y el uso del conocimiento y la capacidad de análisis. Esto incluye la búsqueda de datos y el análisis detallado de factores y variables.
d) Generación de opciones: se refiere a producir un menú de posibles alternativas. Supone pensamiento divergente. Se incluye la búsqueda de principios, líneas o direcciones, por un lado, para hacer asociaciones y unir diferentes marcos de referencia y, por otra parte, para generar posibles soluciones, combinaciones e interpretaciones.
e) Incubación: corresponde al tiempo requerido para reflexionar sobre las alternativas elaboradas, y "ponerlas a prueba mentalmente".
f) Elección de opciones: corresponde a la evaluación final y la selección de las opciones. Supone pensamiento convergente.
g) Persuasión: cierre del proceso creativo y comunicación con otras personas.

Teniendo en cuenta la creatividad como un proceso "no lineal ", son necesarios algunos ajustes, redefiniciones o descartes que obligan a regresar a fases previas, en una dinámica creativa compleja. Por lo tanto, para cada una de las fases definidas es posible asociar retroalimentaciones cuyo "destino" puede ser cualquiera de las fases anteriores en la secuencia señalada.

Por otro lado, Lumsdaine y Lumsdaine (Lumsdaine y Lumsdaine, 1995) plantean el tema de las habilidades cognitivas requeridas (mentalidad) para la resolución creativa de problemas. Su tipología es excelente para el equipo creativo, y los diferentes roles a considerar. Estas funciones son las siguientes:

a) El *Detective* es el encargado de recoger la mayor cantidad de información relacionada con el problema. Este tiene que recoger datos sin hacer juicios, incluso cuando se piensa que ya ha entendido el problema completamente.
b) El *Explorador* detecta lo que puede ocurrir en el área del problema y su contexto. Piensa en efectos a largo plazo y se anticipa a ciertos desarrollos que pueden afectar el contexto (en este caso, al equipo). El explorador percibe el problema en un sentido amplio.
c) El *Artista* crea cosas nuevas, transformando la información. Debe ser capaz de romper sus propios esquemas para generar ideas excéntricas, con imaginación y sentimiento.
d) El *Ingeniero* es el encargado de evaluar nuevas ideas. Se debe hacer converger las ideas, con el fin de aclarar los conceptos y obtener ideas prácticas que se pueden implementar para la resolución de problemas.
e) El *Juez* debe hacer una jerarquía de ideas y decidir cuál de ellas se llevarán a cabo (y también, cuáles deben ser desechadas). Además, se deben detectar posibles fallos o inconsistencias, así como plantear las soluciones correspondientes. Su papel debe ser crítico e imparcial, teniendo que buscar la mejor idea, evaluando los riesgos asociados.
f) El *Productor* es el responsable de la implementación de las ideas escogidas.

Leonard y Swap (Leonard y Swap, 1999) han mencionado posibles funciones adicionales, que es posible integrar con las anteriores, porque tratan de hacer más fructífera la divergencia y la convergencia en el proceso creativo:

a) El *Provocador* que toma los miembros del equipo "para romper" los esquemas mentales y de procedimiento habituales para permitir la divergencia mencionada (en el caso del "Artista") o incluso una mejor convergencia (en el caso del "Ingeniero").

b) El *Pensador Fuerte* que es invitado a las sesiones del equipo para dar una visión renovada de la situación problema sobre la base de su experticia y experiencia.
c) El *Facilitador* cuya función consiste en ayudar y apoyar el trabajo en equipo, en su tarea creativa, en diferentes etapas.
d) El *Gerente* que se preocupa por el rendimiento y especialmente por los resultados del equipo creativo tratando de ajustarlos a los criterios y normas de la organización (uso de los recursos, fechas de vencimiento, etc.).

Finalmente, Kelley y Littman (Kelley y Littman, 2005), se han planteado una tipología basada en roles similar a Lumsdaine y Lumsdaine (Lumsdaine y Lumsdaine , 1995), siendo interesante la agrupación de estos roles en tres categorías: las dirigidas al aprendizaje del equipo creativo (susceptibles de relacionar con los roles del detective, explorador, artista, provocador y pensador fuerte) , otros dirigidos a la organización interna y el éxito del equipo (similar al rol del juez, facilitador y gerente) y, por último, las funciones que tienen por objeto la construcción de la innovación (posiblemente relacionadas con el rol del ingeniero y juez).

## CONCLUSIONES

En ingeniería de software muchos enfoques de desarrollo trabajan repitiendo el modelo lineal básico en todas las iteraciones, y en una gran cantidad de casos se utiliza un enfoque de desarrollo iterativo para proporcionar una retroalimentación rápida y un aprendizaje continuo en el equipo de desarrollo. Para facilitar el aprendizaje entre los desarrolladores, los métodos ágiles defienden el uso de reuniones diarias o semanales, la programación en parejas y la propiedad colectiva. Los métodos agiles enfatizan en las personas, comunidades de práctica, la comunicación, y la colaboración para facilitar la práctica de compartir el conocimiento tácito a nivel de equipo. Un importante descubrimiento es la necesidad de no centrarse exclusivamente en el conocimiento explícito, sino también en el conocimiento tácito.

También fomentan una cultura de equipo de intercambio de conocimientos, confianza y cuidado mutuo. El desarrollo ágil no es definido por un pequeño conjunto de prácticas y técnicas. El desarrollo ágil define una capacidad estratégica, una capacidad para crear y responder al cambio, una capacidad para equilibrar la flexibilidad y la estructura, una capacidad de fomentar la creatividad y la innovación en un desarrollo de equipo, y la capacidad para dirigir las organizaciones a través de la turbulencia y la incertidumbre. Estos se desarrollan fuera de los planes (modelos), pero se concentran en el trabajo de creación de software. Se centran en los individuos y sus habilidades y sobre la intensa interacción de los miembros del equipo de desarrollo entre sí y con los clientes y la gerencia.

El uso de herramientas de software para la creación rápida de prototipos de los algoritmos ahorraría recursos considerables. Sería deseable emplear principios ágiles y reutilización para producir software que se adapta fácilmente a necesidades cambiantes, a la vez que mejora la calidad y reduce los esfuerzos de desarrollo. Ya que el desarrollo de software es una actividad intensiva en conocimientos, la comprensión desde un punto de vista de la gestión del conocimiento ofrece perspectivas importantes sobre reutilización y métodos de ingeniería de software.

Por otro lado, la creatividad y la innovación son habilidades esenciales en casi cualquier equipo. Tener un equipo que puede resolver los problemas de forma rápida y eficaz con un poco de creatividad es beneficioso para todos. El rendimiento de un equipo no sólo depende de la competencia del propio equipo en hacer su trabajo, sino también en el contexto organizacional.

Las condiciones de la organización en la cual se inserta el equipo son muy importantes también. Si los estudiantes ven que se les alienta en sus ideas y son aceptadas, será más probable que sean creativos, mejorando el potencial de innovación en el lugar de trabajo. La creación de un entorno de trabajo colaborativo fomentará la comunicación entre los estudiantes y recompensará a aquellos que trabajan en conjunto para resolver problemas. Se debe animar a los miembros del equipo a tomar riesgos, lo opuesto de la creatividad es el miedo, entonces es necesario crear un ambiente que esté libre de miedo al fracaso: los fracasos son una herramienta de aprendizaje. La utilización de roles es un aspecto central de un trabajo creativo en equipo. Estos aspectos se pueden organizar de acuerdo con la estructura que adopte el equipo y el rendimiento que caracteriza al equipo.

A pesar de los comentarios anteriores, creemos que en las organizaciones los métodos agiles de desarrollo de software deben tener una referencia más explícita al rol del Provocador que se describe a fondo en la creatividad como factor fundamental para generar innovación. Esto se explica porque, en general, estas metodologías no tienen por objeto, como elemento central, generar software original, pero si efectivo. En segundo lugar, es esencial la distinción y la formalización de las fases creativas para generar etapas de incubación y elección de opciones. En tercer lugar, se requiere una mención más directa a la atmósfera física de trabajo, que en la creatividad es considerada como de gran importancia para mejorar el rendimiento. Estos aspectos deben tener una mayor consideración ya que el desarrollo de software es un caso especial de desarrollo de productos.

**REFERENCIAS**

Apostolou, D. and Mentzas, G. (2003). Experiences from knowledge man-agement implementations in companies of the software sector. Business Process Management Journal, 9(3).
Beck, K. (2000). Extreme programming explained: embrace change. Addi-son-Wesley Longman Publishing Co., USA.
Beck, K., Beedle, M., Bennekum, A. V., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., Kern, J., Mar-ick, B., Martin, R. C., Mellor, S., Schwaber, K., Sutherland, J., and Thomas, D. (2001). Manifesto for agile software development. Available at http://agilemanifesto.org.
Boden, M. (2004). The Creative Mind: Myths and Mechanisms. Routledge, USA.
Brown, T. (2008). Design Thinking. Harvard Business Review.
Bueno, E. (2003). Knowledge management in the emerging strategic busi-ness process. Journal of knowledge Management, 7(3):1–25.
Chau, T. and Maurer, F. (2004). Knowledge sharing in agile software teams. In Lenski, W., editor, Logic versus Approximation: Essays Dedicated to Mi-chael M. Richter on the Occasion of his 65th Birthday, volume 3075 of Lec-ture Notes in Artificial Intelligence, p. 173–183. Springer.
Chau, T., Maurer, F., and Melnik, G. (2003). Knowledge sharing: Agile methods versus tayloristic methods. Twelfth International Workshop on Ena-bling Technologies: Infrastructure for Collaborative Enterprises, WETICE, p. 302–307.
Crawford, B. and Leon de la Barra, C. (2007). Enhancing creativity in agile software teams. Lecture Notes in Computer Science, 4536:161–162.
Crawford, B. and Leon de la Barra, C. (2008). Integrating creativity into ex-treme programming process. In Cordeiro, J. and Filipe, J., editors, ICEIS (3-1), p. 216–219.
Crawford, B., Leon de la Barra, C., and Rubio, J. (2008). Knowledge sharing in traditional and agile software processes. In Cordeiro, J., Shishkov, B., Ranchordas, A., and Helfert, M., editors, ICSOFT (PL/DPS/KE), p. 376–379. INSTICC Press.

Glass, R. (1995). Software creativity. Prentice-Hall, USA.
Gu, M. and Tong, X. (2004). Towards hypotheses on creativity in software development. PROFES, 3009:47–61.
John, M., Maurer, F., and Tessem, B. (2005). Human and social factors of software engineering: workshop summary. ACM SIGSOFT Softw. Eng., Notes, 30:1–6.
Kelley, T. and Littman, J. (2005). The Ten Faces of Innovation: IDEOs Strat-egies for Defeating the Devil's Advocate and Driving Creativity Throughout Your Organization. Doubleday Random House, USA.
Kotler, P. and Trías de Bes, F. (2004). Marketing Lateral. Editorial Pear-son/Prentice Hall, Spain.
Leon de la Barra, C. and Crawford, B. (2007). Fostering creativity thinking in agile software development. Lecture Notes in Computer Science, 4799:415–426.
Leonard, D. and Swap, W. (1999). When Sparks Fly: Igniting Creativity in Groups. Harvard Business School Press, Boston.
Lumsdaine, E. and Lumsdaine, M. (1995). Creative Problem Solving: Think-ing Skills for a Changing World. McGraw-Hill, New York.
Maiden, N. and Gizikis, A. (2001). Where do requirements come from? IEEE Software, 18:10–12.
Maiden, N., Gizikis, A., and Robertson, S. (2004). Provoking creativity: Imag-ine what your requirements could be like. IEEE Software, 21:68–75.
Mentzas, G. (2000). The two faces of knowledge management. International Consultant's Guide, p. 10–11.
Mich, L., Anesi, C., and Berry, D. (2005). Applying a pragmatics-based crea-tivity-fostering technique to requirements elicitation. Requir. Eng., 10:262–275.
Moe, N., Dingsoyr, T., and Dyba, T. (2010). A teamwork model for under-standing an agile team: A case study of a scrum project. Information and Software Technology, 52:480–491.
Nonaka, I. and Takeuchi, H. (1995). The Knowledge Creating Company. Oxford University Press, USA.
Ringstad, M., Dingsoyr, T., and Moe, N. (2011). Agile process improvement: Diagnosis and planning to improve teamwork. Communications in Comput-er and Information Science, 172:167–178.
Robertson, J. (2002). Eureka! why analysts should invent requirements. IEEE Software, 19:20–22.
Robertson, J. (2005). Requirements analysts must also be inventors. IEEE Software, 22:48–50.
Rus, I. and Lindvall, M. (2002). Knowledge management in software engi-neering. IEEE Software, 19(3):26–38.
Sutherland, J. (2001). Agile can scale: Inventing and reinventing scrum in five companies. Cutter IT Journal, 14:5–11.
Takeuchi, H. and Nonaka, I. (1986). The new product development game. Harvard Business Review.
Wallas, G. (1926). The art of thought. Harcourt Brace, New York