IX Jornadas de Ciencia y Tecnología
de Facultades de Ingeniería del NOA
Santiago del Estero, 3 y 4 de Octubre de 2013

TÉCNICAS Y HERRAMIENTAS PARA CAPTURA DE EXPERIENCIAS EN PROYECTOS SOFTWARE

Miriam E. Ríos[1], Gustavo J. López[2], Eve L. Coronel[3], Liliana M. Figueroa[4] & Sebastián I. Scaglione[5]

(1), (2), (5) Departamento de Matemática, Facultad de Ciencias Exactas y Tecnologías, Universidad Nacional de Santiago del Estero.
merios15@yahoo.com.ar, gustavojlopez@gmail.com, scaglionesebastian@yahoo.com

(3) Facultad de Agronomía y Agroindustrias, Universidad Nacional de Santiago del Estero.
ecoronel@unse.edu.ar

(4) Departamento de Informática, Facultad de Ciencias Exactas y Tecnologías, Universidad Nacional de Santiago del Estero
lmvfigueroa@yahoo.com.ar

RESUMEN: En los tiempos actuales, los conocimientos y experiencia que los integrantes de los equipos de proyecto crean y adquieren durante los proyectos software constituyen un valioso activo para las organizaciones que buscan mejorar sus prácticas y procesos software. Además, la experiencia juega también un rol principal en estas actividades relacionadas con el conocimiento, en consecuencia, existe una necesidad de recopilar experiencias y conocimientos de Ingeniería de Software. Es pertinente por tanto el presente trabajo que surge en el marco del proyecto de investigación Optimización de la Calidad del Proceso Software con Gestión del Conocimiento (GC), y tiene como objetivo analizar las herramientas existentes para capturar experiencia que los integrantes de los equipos de proyecto crean y adquieren durante los proyectos software. Para lograr este objetivo se realiza, como primer paso una investigación exploratoria sobre las herramientas de captura de experiencias, y como segundo paso se plantean observaciones, según lo expuesto, a los enfoques y técnicas detalladas en el primer paso. Con estos resultados se procura delimitar una de las bases conceptuales que dará soporte a la mejora de la calidad del software, objetivo último del mencionado proyecto de investigación.

1. INTRODUCCIÓN

En los tiempos actuales, los conocimientos y experiencia que los integrantes de los equipos de proyecto crean y adquieren durante los proyectos software constituyen un valioso activo para las organizaciones que buscan mejorar sus prácticas y procesos software. Fuggetta (2000) define el término proceso software como un conjunto coherente de políticas, estructuras organizacionales, tecnologías, procedimientos y artefactos que son necesarios para concebir, desarrollar, instalar y mantener un producto software. Por lo tanto, se debe prestar atención a la compleja interrelación que se produce en un PS entre los diversos factores organizacionales, culturales, tecnológicos y económicos. Los procesos software han evolucionado desde ser procesos conducidos por datos, pasando por ser procesos conducidos por información hasta los actuales procesos conducidos por conocimientos (Alagarsamy, Justus & Iyakytti, 2008). Además, la mejora de los procesos y los productos software constituyen casos especiales de GC y la experiencia juega también un rol principal en estas actividades relacionadas con el conocimiento, en consecuencia, existe una necesidad de recopilar experiencias y conocimientos de Ingeniería de Software y re-utilizarlos para mejorar los procesos software. Es pertinente por tanto el presente trabajo que surge en el marco del proyecto de investigación Optimización de la Calidad del Proceso Software con Gestión del Conocimiento (GC), y tiene como objetivo analizar las herramientas existentes para capturar experiencia que los integrantes de los equipos de proyecto crean y

adquieren durante los proyectos software. Para lograr este objetivo se realiza, como primer paso una investigación exploratoria sobre las herramientas de captura de experiencias y como segundo paso se plantean observaciones, según lo expuesto, a los enfoques y técnicas detalladas en el primer paso.

En la siguiente sección se desarrolla una breve descripción del enfoque fábrica de experiencia y de bases de experiencia software, y se muestran trabajos relacionados don dicho enfoque y con la utilización de bases o repositorios de experiencias como elementos centrales en la GC. En la sección tres se presentan Técnicas para captura de experiencia a partir de proyectos software y herramientas software y entornos de desarrollo software. En la sección cuatro se exponen observaciones realizadas a los enfoques y técnicas presentadas; y finalmente, en la sección cinco se sintetizan algunas conclusiones del trabajo.

2. FÁBRICA DE EXPERIENCIA Y BASES DE EXPERIENCIAS SOFTWARE

2.1. *Fábrica de experiencia*

El enfoque fábrica de experiencia o Experience Factory es una es una infraestructura física y/o lógica que apoya los proyectos de desarrollo; Apunta esencialmente a la captura, análisis y empaquetado de experiencias de todo tipo adquiridas durante el desarrollo de proyectos software con el objetivo de reutilizar esa experiencia en nuevos proyectos de desarrollo de software (Basili, Caldeira & Rombach, 1994).

Este enfoque divide los esfuerzos de desarrollo de software en dos unidades con responsabilidades separadas de desarrollar proyectos de software y capturar experiencias.

Basili, Lindvall & Costa (2001) explica que la implementación física de una fábrica de experiencia es un Sistema de Administración de Experiencia, compuesto de contenido, estructura, procedimientos y herramientas.

- El contenido (que pueden ser datos, información, conocimientos o experiencias) y
- la estructura (que es la forma en que está organizado el contenido) constituyen lo que se denomina Base de Experiencia.
- Los procedimientos son instrucciones acerca de cómo manejar la base de experiencias, y las herramientas soportan la gestión del contenido y la ejecución de los procedimientos.

En una fábrica de experiencia sus valores centrales son que, para mejorar, los empleados necesitan aprender de experiencias pasadas y, para que los empleados aprendan, la organización necesita crear un ambiente de aprendizaje (Basili, Lindvall & Costa, 2001).

2.2. *Bases de experiencia software*

El cuerpo de conocimientos de una base de experiencia son típicamente de distintos tipos (know-how, know-why, know-what) y utiliza diferentes esquemas de representación, tales como modelos explícitos, experiencias documentadas o lecciones aprendidas, así como conocimientos tácitos y habilidades más o menos estructurados poseídos por las personas (Ruhe & Bomarius, 2000).

Para implementar con éxito una base de experiencia, existen cuatro factores claves (Conradi, Lindvall & Seaman, 2000):

- Cambio cultural: es importante que las personas provean conocimiento a la base de experiencia y que también hagan uso del conocimiento que esté disponible en ella.
- Estabilidad: relacionada con la habilidad para gestionar los cambios de manera controlada.
- Valor para el negocio: la base de experiencia se percibirá como un elemento exitosos, si ésta provee un valor concreto y demostrable para el negocio
- Implementación incremental: si la implementación y la introducción de una base de experiencia se realizan en pequeños incrementos y en estrecha conexión con sus futuros usuarios recibiendo de estos una retroalimentación continua, ambas instancias se consideran exitosas.

2.3. *Trabajos y propuestas relacionados con fábrica de experiencia y bases de experiencia software.*

2.3.1. Software Experience Center

El Software Experience Center (SEC) (Schneider, von Hunnius & Basili, 2002) basado en el concepto de la Experience Factory (Basili, Caldeira & Rombach, 1994) tiene como objetivo operacional proveer a las unidades de negocio de los conceptos de una organización que aprende y un prototipo de una base de experiencia. Proporciona apoyo en todas las actividades, desde la educción de experiencias hasta hacer disponible esas experiencias para la tarea de software entre manos.

El SEC ha mejorado muchos procesos en la organización y a logrado que el aprendizaje por medio de la experiencia se convierta en una parte más natural de la vida diaria de las unidades de negocio (Schneider, von Hunnius & Basili, 2002).

2.3.2. Experience Engine

El software Experience Engine *(*Johansson, Hall & Coquard, 1999*)* se sustenta en el conocimiento tácito a pesar de ser una variante de Experience Factory (Basili, Caldeira & Rombach, 1994) basada en experiencias almacenadas en bases de experiencia.
A partir de la definición de dos nuevos roles en la organización, el comunicador de experiencia "experience communicator" y el agente de experiencias "experience broker", se logró hacer accesible el conocimiento tácito a un grupo mayor de personas. El primero es una persona que posee un conocimiento profundo acerca de uno o más temas, mientras que la misión del agente de experiencias es conectar al comunicador con la o las personas que tienen un problema. El comunicador no resuelve el problema sino que educa y asiste al poseedor del mismo acerca de cómo resolverlo (Johansson, Hall & Coquard, 1999).

2.3.3. Knowledge Dust to Pearls

El enfoque Knowledge Dust to Pearls (Basili & Seaman, 2002) provee mecanismos que facilitan la iniciación rápida de una base de experiencia. Mediante la idea de capturar las partículas de conocimiento "knowledge dust" que los empleados utilizan e intercambian diariamente en sus actividades e, inmediatamente y con mínimas modificaciones, hacerlas disponibles a través de toda la organización. En forma paralela, estas partículas de conocimiento son analizadas, sintetizadas y transformadas en perlas de conocimiento "knowledge pearls" que representan elementos de conocimiento más sofisticados, refinados y valiosos (Basili & Seaman, 2002).

3. TÉCNICAS PARA CAPTURA DE EXPERIENCIA A PARTIR DE PROYECTOS SOFTWARE

3.1. Análisis post mortem de proyectos

La técnica del análisis *post mortem* de proyectos (APM) (Birk, Dingsoyr & Stalhane, 2002) se emplea con el propósito de capturar experiencias y sugerencias de mejora a partir de proyectos completados o cuando en el proyecto se haya alcanzado un hito significativo. Birk, Dingsoyr & Stalhane (2002) delimitan la técnica a proyectos de software y sostienen que en cada proyecto de este tipo, los miembros del equipo ganan nuevo conocimiento y experiencia que pueden ser beneficiosos tanto para futuros proyectos como para el propio desarrollo profesional de los miembros del equipo.
La aplicación de esta técnica consiste de tres fases:
i) Preparación: en esta fase se recorre la historia del proyecto para obtener un mejor entendimiento de lo que ha ocurrido y se revisan todos los documentos disponibles, tales como planes del proyecto, la estructura de desglose de tareas ("work breakdown structure") y los reportes de revisión. En esta fase también se determinan los objetivos específicos para el análisis a realizar.
ii) Recolección de datos: esta fase está dedicada a obtener la experiencia relevante del proyecto consistente no sólo de los problemas o aspectos negativos que deberían haberse evitado sino también de los aspectos exitosos. Una vez identificados los temas importantes, se debe priorizarlos antes de proceder al análisis. El establecimiento de prioridades asegura que se traten en primera instancia los aspectos de mayor significación.
iii) Análisis: en esta fase un moderador conduce una sesión de retroalimentación
("feedback") para discutir e intercambiar ideas y puntos de vista sobre los temas identificados en la fase anterior y finaliza con la elaboración de un informe con las conclusiones a las que se llegó y las recomendaciones que correspondan.

3.2. Sesiones Legacy

El enfoque de sesiones *legacy* (Cooper, Majchrzak & Faraj, 2005) consiste en sesiones de trabajo donde los miembros de un equipo de proyecto identifican innovaciones y mejoras que han realizado en sus proyectos y que tienen un valor potencial para futuros usuarios. En opinión de Cooper, Majchrzak & Faraj (2005) su característica distintiva respecto de otros enfoques similares tales como las *after-action reviews* o las *project lessons learned reviews es que* se enfoca en las oportunidades de reutilización en lugar de sobre el riesgo de repetir errores. Cooper, Majchrzak & Faraj (2005) describen una sesión legacy en las cuatro fases que la constituyen. La primera fase consiste de una sesión de tormenta de ideas ("brainstorming") para identificar potenciales legados ("legacies"). Estos legados representan aprendizajes que tienen el potencial de ser reutilizados por los miembros del equipo de proyecto o por otros miembros de la organización. En la segunda fase, los participantes sintetizan los resultados de la fase anterior combinando elementos similares, quitando elementos disímiles y categorizando los resultados según sean procesos, productos, personas u otros. De la lista final se selecciona

luego un elemento para su posterior discusión. La tercera fase es la discusión detallada del elemento seleccionado para, en la cuarta fase, elaborar un sumario de la discusión, siguiendo una plantilla de estructura predefinida.

3.3. Revisiones post-proyecto

Harrison (2003) considera la realización de revisiones post-proyecto (post-project reviews) como forma de proveer un mecanismo formal para transferir la experiencia de un equipo de proyecto a una memoria corporativa una vez que se ha completado un proyecto y mientras esas experiencias están aún frescas en las mentes de los participantes. La experiencia capturada se vuelca a un repositorio de lecciones aprendidas cuyo propósito es facilitar la organización, mantenimiento y diseminación del conocimiento capturado. El repositorio está basado en tecnología web y dispone de una interfase basada en formularios para que los suministradores de lecciones aprendidas puedan agregar nuevas experiencias al mismo.

3.4. Herramientas software y entornos de desarrollo software que incorporan funcionalidades de GC y de captura de experiencias.

3.4.1. Semantic Reuse System

El sistema Semantic Reuse System (SRS) se basa en tecnologías de la web semántica para la adquisición, gestión y reutilización de conocimientos sobre desarrollo software y apunta a proveer mecanismos eficientes para capturar, almacenar, recuperar y administrar conocimientos.
El SRS provee medios para capturar y almacenar diferentes elementos que resultan de los procesos de desarrollo software, tales como documentos de especificación, diagramas de diseño y código fuente, entre otros. Además, de diversas formas de reutilizar y compartir conocimientos, incluyendo una manera proactiva de sugerir conocimiento relevante al desarrollador software en función del contexto de trabajo del usuario (Antunes, Seco & Gomes, 2007)

3.4.2. RAMALA

RAMALA es una base de conocimientos desarrollada por Ramawi, et al. (2005). Está apoyada por una herramienta software también denominada RAMALA. Los principales alcances y metas del trabajo que dio lugar al desarrollo de RAMALA fueron modelar y desarrollar una base de conocimientos para la mejora del proceso software, apoyada por una herramienta software que posibilitara la definición, la evaluación ("assessment") y la mejora del proceso software de una organización.
RAMALA proporciona varios beneficios a las organizaciones software tras su utilización, por ejemplo: el hecho de que todos los conocimientos de los procesos de desarrollo software pueden ser recopilados, clasificados y asociados con sus correspondientes elementos del proceso; y que la base de datos histórica de activos de procesos puede reutilizarse en futuros proyectos; además de proveer buenos indicadores del grado de institucionalización del proceso software en la organización.

3.4.3. PM-CAKE

El entorno de trabajo PM-CAKE, "Process Management Computer Aided Knowledge Environment", propuesto por Martínez, et al., (2005) es un marco de trabajo ("framework") para la GC de proyectos y procesos en una organización software y para ser utilizado por gerentes de proyectos, ingenieros de software, grupos de gestión de procesos y gerentes de unidades de Tecnología de la Información.
Según sostiene Martínez, et al. (2005) PM-CAKE permite transferir todo el *know-how* sobre ejecución de proyectos software desde las personas hacia la organización y provee un repositorio para gestionar las prácticas de gestión, las que se agrupan en procesos que pueden luego utilizarse para introducir nuevas prácticas o para modificar las existentes. Asimismo, permite definir el proceso software a seguir en un nuevo proyecto, seleccionar valores estimados para el ciclo de vida, los factores de complejidad y características del equipo de desarrollo así como también indicar parámetros estimados de tamaño del proyecto, esfuerzo, duración, costos y calidad. A partir de esta información, un gerente de proyecto usa PM-CAKE para buscar en las experiencias previas almacenadas en el repositorio la información necesaria para definir las actividades del proyecto y su estructura de desglose de tareas ("work breakdown structure"). Una vez finalizado el proyecto, el gerente ingresa al sistema la información del mismo referente, por ejemplo a requisitos, diseño, pruebas de software y código fuentes, así como las estructuras de desglose de tareas y de productos.
PM-CAKE y su repositorio se utilizarán para la extracción de procesos software en forma automática a partir de la experiencia de proyectos y también para la evaluación del desempeño organizacional (Martínez, et al., 2005).

3.4.4. Process Asset Database

Process Asset Database (Ramasubramanian & Jagadeesan, 2003) permite capturar entregables de proyectos tales como planes de proyectos, documentos de diseño y planes de prueba. Los usuarios pueden buscar estos documentos en base a diferentes criterios como son: tipo de proyecto, tecnología involucrada, dominio o área de conocimientos y por nombre del cliente, entre otros.
Esta herramienta ayuda a proveer a los nuevos proyectos de información relativa a proyectos similares ejecutados en el pasado, así como a establecer metas cuantitativas para los mismos.

3.4.5 Wikis, wikis semánticos y ontologías

Un wiki es esencialmente una colección de sitios web conectados por vínculos de hipertexto (Schaffert, S., 2006a). Mientras que, los *wikis* semánticos son la combinación de wikis con tecnologías de la web semántica (Schaffert,Westenthaler & Gruber, 2006b).
Tanto a nivel personal como organizacional, los *wikis* se están transformando en herramientas populares de GC (Oren, et al., 2006) y pueden verse como una plataforma liviana ("lightweight") para intercambiar artefactos reutilizables dentro y entre proyectos software, y entienden que también pueden ser considerados como formas de memorias organizacionales o fábricas de experiencias.
En otro sentido, una ontología define los términos y relaciones básicas que componen el vocabulario de un área tópica, así como las reglas para combinar términos y relaciones para definir extensiones a ese vocabulario (Corcho, O. et al., 2006).

3.4.5.1. Riki

Rech, Bogner & Hass (2007) reportan el desarrollo de un wiki, denominado Riki, que utiliza tecnologías de razonamiento basado en casos ("case-based reasoning") y ontologías para proveer un marco formal y consistente para describir conocimientos y experiencias, y cuyo propósito es implementar un sistema de documentación orientada a la reutilización de conocimientos sobre proyectos software.
La ontología y las plantillas de documentos ("templates") enriquecen el contenido de Riki con semántica que permite a los usuarios aumentar el conocimiento incorporado en Riki con información adicional y experiencias documentadas, así como compartir experiencias y reutilizar conocimiento relativo a proyectos (Rech, Bogner & Haas, 2007).

3.4.5.2. Wikitología

wikitología (Klein, Hoecht & Decker, 2005) es un enfoque aúna técnicamente wikis y ontologías, y cuyo propósito es que el conocimiento sobre IS pueda ser constantemente mantenido y cultivado por los ingenieros de software. Este enfoque, a largo plazo, debería resultar en una calidad superior de los sistemas software debido a una mejor disponibilidad de los conocimientos relativos a IS.

3.4.5.3. MASE

Chau & Maurer (2005) describen la utilización de wikis en una herramienta, denominada MASE, orientada a mejorar los mecanismos de compartición de experiencias entre diferentes equipos de desarrollo software en una organización. Esta herramienta permite a los usuarios registrar información de manera informal y no estructurada, así como también definir las tareas de un proyecto y almacenar información sobre las mismas en base a formatos específicos (estructurados). La herramienta, además, provee capacidades de búsqueda de texto en cualquiera de las páginas wiki y apoya el trabajo colaborativo tanto en forma sincrónica como asincrónica.

4. OBSERVACIONES, SEGÚN LO EXPUESTO, A LOS ENFOQUES Y TÉCNICAS PRESENTADAS

Enfoque y herramientas	Características
Fabrica de experiencias y sus derivaciones que hacen uso de repositorios de experiencias	• Es el enfoque más comprehensivo para la GC y de la experiencia en el ámbito de la IS • su marco de trabajo establece cuales son las actividades de GC que es necesario realizar (educción, análisis, generalización, empaquetado y diseminación) • pero presenta la carencia de no prescribir cómo deben llevarse a cabo esas actividades, así como el estar basados, la mayoría, en la estrategia de codificación de conocimientos (Chau & Maurer, 2005).
Análisis post	• presentan el inconveniente de ser extemporáneos respecto del momento de ocurrencia de las experiencias que se pretenden capturar

mortem y similares	• desafortunadamente, la mayoría de los ingenieros de software no tienen tiempo para "finalizar" un proyecto cuando ya están siendo reasignados a otros (Desouza, Dingsoyr & Awazu, 2005).
Sesiones legacy	• se enfoca en las oportunidades de reutilización en lugar de sobre el riesgo de repetir errores (Cooper, Majchrzak & Faraj, 2005)
Herramientas software y entornos de trabajo	• en general, se orientan a apoyar actividades particulares dentro de la IS, tales como la arquitectura software, la gestión de los proyectos, la instanciación de procesos software y la estimación y recolección de métricas.
Wikis y wikis semánticas	• las mismas apuntan a apoyar el trabajo colaborativo de gestión documental para facilitar la reutilización de los conocimientos plasmados en esos documentos.

Tabla 1. Características y observaciones a fábrica de experiencia, análisis post mortem, sesiones legacy, herramientas software y entornos de trabajo y wikis.

Otras observaciones:

*Con respecto a la fábrica de experiencias y el análisis post mortem ambos se centran en el resultado o producto final de la adquisición de experiencia, esto es, en la experiencia misma adquirida, pero no establecen ningún mecanismo relacionado con el propio proceso de adquisición de esa experiencia; no delimitan, a priori, los tipos de conocimientos y de experiencias a capturar.

*Por su parte las sesiones legacy, en el estudio exploratorio reportado en el artículo referido por Cooper (Majchrzak & Faraj, 2005), de los nueve equipos de proyecto contactados para llevar a cabo estas sesiones, si bien todos estuvieron interesados en participar, sólo cuatro estuvieron finalmente disponibles para hacerlo.

5. CONCLUSIONES

A partir del marco logrado por la investigación y del análisis realizado, se elaboran las siguientes conclusiones:

- Varias herramientas software lograron hacer accesibles tanto bases de experiencias como conocimiento tácito a un grupo mayor de personas; han mejorado muchos procesos en la organización logrando que el aprendizaje por medio de la experiencia se convierta en una parte más natural de la vida diaria de las unidades de negocio; capturar las partículas de conocimiento que los empleados utilizan e intercambian diariamente en sus actividades e, inmediatamente y con mínimas modificaciones, hacerlas disponibles a través de toda la organización.
- Las herramientas y técnicas de captura analizadas permiten, incorporar, preservar, aplicar, diseminar en una organización, la experiencia y el conocimiento adquiridos en proyectos software pasados puede utilizarse para mejorar las prácticas en proyectos futuros. Por lo tanto, conocer los distintos tipos de técnicas, herramientas de captura de experiencias de la GC al desarrollar software da una oportunidad de mejorar su calidad.

*En cuanto a las fabricas de experiencias y al análisis post mortem, en particular, ambos tipos de enfoques presentan ciertas características en común, se centran en el resultado o producto final de la adquisición de experiencia, esto es, en la experiencia misma adquirida, pero no establecen ningún mecanismo relacionado con el propio proceso de adquisición de esa experiencia; no delimitan, a priori, los tipos de conocimientos y de experiencias a capturar.

En función de esto último, se como trabajo futuro se planea investigar sobre herramienta para capturar los conocimientos y los aprendizajes basados en la experiencia que los miembros de los equipos de proyecto adquieren "durante" la realización de sus actividades de proyecto.

6. REFERENCIAS

Alagarsamy, K., S. Justus & K. Iyakutti, *The knowledge based software process improvement program. A rational analysis, Proceedings of the 41th Hawaii International Conference on System Sciences*, 2008.

Antunes, B., N. Seco & P. Gomes, *Knowledge management using semantic web technologies: An application in software development*, Proceedings of the Fourth International Conference on Knowledge Capture, 187-188, 2007.

Basili, V. & C. Seaman, *The experience factory organization*, IEEE Software, 19, 30-31, 2002.

Basili, V., G. Caldeira & H. Rombach, *The experience factory*, en Marciniak, J. (ed.)

Encyclopedia of software engineering, J. Wiley & Sons, New York, 1994.

Basili, V., M. Lindvall & P. Costa, *Implementing the experience factory as a set of experience bases*, International Conference on Software Engineering and Knowledge Engineering (SEKE '01), Buenos Aires, 2001.

Birk, A., T. Dingsoyr & T. Stalhane, *Postmortem: never leave a project without it*, IEEE Software, 19, 43-45, 2002.

Chau, T. & F. Maurer, *A case study of wiki-based experience repository at a medium-sized software company*, University of Calgary, Department of Computer Science, 2005.

Conradi, R., M. Lindvall & C. Seaman, *Success factors for software experience bases: what we need to learn from other disciplines*, ICSE '2000, Limerick, 113-119, 2000.

Cooper, L., A. Majchrzak & S. Faraj, *Learning from project experiences using a legacy-based approach*, Proceedings of the 38th Hawaii International Conference on System Sciences, 253, 2005.

Corcho, O., et al., *Ontological engineering. Principles, methods, tools and languages*, en: Calero, C., Ruiz, F., Piattini, M. (eds.): Ontologies for software engineering and software technology, Springer, Berlin, 1-39, 2006.

Desouza, K., T. Dingsoyr & Y. Awazu, *Experiences with conducting project postmortems*, Proceedings of the 38th Hawaii International Conference on System Sciences, 233, 2005.

Fuggetta, A., Software Process: A Roadmap, *22nd International Conference on Software Engineering (ICSE'2000), Future of Software Engineering Track*, June 4-11, Limerick (Irlanda), ACM, 2000.

Harrison, W, *A software engineering lessons learned repository*, Proceedings of the 27th Annual NASA, 2003.

Johansson, C., P. Hall, & M. Coquard, *Talk to Paula and Peter; they are experienced. The experience engine in a nutshell*, Proceedings of the 11th International Conference on Software Engineering and Knowledge Engineering, Learning Software Organizations, Methodology and Applications, 171-185, 1999.

Klein, B., C. Hoecht & B. Decker, *Beyond capturing and maintaining software engineering knowledge. Wikitology as shared semantics*, Knowledge Engineering and Software Engineering Workshop, 28th German Conference on Artificial Intelligence, Koblenz, Alemania, 2005.

Martinez et al., *Requirements for a knowledge management framework to be used in software intensive organizations*, Proceedings of the International Conference on Information Reuse and Integration, IRI, 554-559, 2005.

Oren, E. et al., *Semantic wikis for personal knowledge management*, Lecture Notes in Computer Science, 4080, 509-518, 2006.

Ramasubramanian, S. & G. Jagadeesan, *Knowledge management at Infosys*, IEEE Software, 20, 53-55, 2003.

Rech, J., C. Bogner & V. Haas, *Using wikis to tackle reuse in software projects*, IEEE Software, 24, 99-104, 2007.

Rimawi, et al., RAMALA: *A knowledge base for software process improvement*, 2nd International Conference on Innovations in nformation Technology, 2005.

Ruhe, G. & F. Bomarius, *Learning software organizations*, Springer, Berlin, 2000.

Schaffert, S., *IkeWiki: a semantic wiki for collaborative knowledge management*, International Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises, (WETICE'06), 388-393, 2006a.

Schaffert, S., R. Westenthaler & A. Gruber, *IkeWiki: A userfriendly semantic wiki*, 3rd European Semantic Web Conference (ESWC'06), 2006b.

Schneider, K., J. von Hunnius & V. Basili, *Experience in implementing a learning software organization*, IEEE Software, 19, 46-49, 2002.