UNIVERSIDAD TECNOLÓGICA DE LA MIXTECA

"PROPUESTA DE UN ENFOQUE COLABORATIVO PARA LA ENSEÑANZA-APRENDIZAJE EN EL DESARROLLO DE INICIATIVAS DE MEJORA DEL PROCESO SOFTWARE EN CURSOS DE POSGRADO"

TESIS

PARA OBTENER EL GRADO DE MAESTRO EN TECNOLOGÍAS DE CÓMPUTO APLICADO

PRESENTA

ING. CARLOS ALEJANDRO PEREZ CRUZ

DIRECTOR DE TESIS

DR. IVÁN ANTONIO GARCÍA PACHECO

HUAJUAPAN DE LEÓN, OAXACA. DICIEMBRE DE 2014

Tesis presentada en diciembre de 2014,
ante los siguientes sinodales:

Dr. Carlos Alberto Fernández y Fernández
Dra. Carla Leninca Pacheco Agüero
Dr. Moisés Homero Sánchez López
Dr. José Figueroa Martínez

Director de Tesis:

Dr. Iván Antonio García Pacheco

Agradecimientos

Mi más sincero agradecimiento a todas las personas que me apoyaron para seguir estudiando durante estos dos últimos años.

A mi padre, por tu apoyo incondicional dado que a pesar de encontrarte lejos siempre has estado en constante comunicación conmigo.

A mi madre, porque se que cuento contigo para cualquier situación que se presente.

A mi hermana, por ser tan dulce como siempre.

A la nueva integrante de mi familia, mi futura esposa, por formar parte de este esfuerzo que hice en la universidad.

A mi director de tesis, el Dr. Iván García, por su apoyo y amistad.

Muy especialmente al Consejo Nacional de Ciencia y Tecnología por la beca otorgada, la cual me permitió realizar los estudios de maestría.

Índice

Lista de tablas

Lista de figuras

Resumen

En los últimos años la demanda de mejores servicios y funcionalidad en los productos software ha crecido; como consecuencia muchos modelos, técnicas y herramientas han sido desarrollados para mejorarlos, tales como CMMI-DEV v1.2, TSP o SCRUM. Sin embargo los productos software aún presentan costos elevados, retardos en las entregas y baja calidad. En este sentido, debido a que la Mejora del Proceso Software está siendo recientemente explorada en la industria mexicana de software, no existe suficiente material educativo que proporcione altos niveles de interacción entre los estudiantes y ésta, aspecto necesario para aprender a adaptar estos modelos, técnicas y herramientas al trabajo diario. Esta tesis describe la definición de un enfoque colaborativo como soporte de un curso a nivel posgrado en cooperación con la industria de software local, para mejorar las habilidades, tanto prácticas como profesionales, de los estudiantes incrementando así su participación y esfuerzo en la conducción de iniciativas de mejora, en comparación con los enfoques tradicionales de enseñanza que están basados regularmente en clases teóricas.

Abstract

In recent years there has been an on-going demand for better services and functionality in software products; as a consequence many models, techniques and tools have been developed such as CMMI-DEV v1.2, TSP, or Scrum. However, software products still suffer from excessive costs, delays in delivery and low quality. Furthermore, there is a lack of educational material providing high levels of interaction between students and the software industry to learn about how enterprises adopt these models, techniques and tools into their daily work. This thesis describes a web-based tool to support a graduate course in collaboration with the local software industry. The main goal of this research is to demonstrate that a Software Engineering course may use the tool to improve students' practical and professional skills, thus increasing their participation and effort in improvement initiatives, in comparison to traditional educational approaches which are only based on theory classes.

1. Introducción

Desde el principio de la década de los 80's, la industria de software ha tratado de incrementar su calidad y productividad mediante la aplicación de nuevos métodos y técnicas. En este sentido, se ha reconocido que desafortunadamente el problema fundamental de muchas empresas de desarrollo de software es la incapacidad de gestionar su proceso de creación [Oktaba & Piatinni, 2008]. Así, los productos de software siguen presentando costos excesivos, retrasos en la entrega y baja calidad en su desarrollo. De acuerdo con las investigaciones de Oktaba [Oktaba, 2006] y Johnson [Johnson, 2006], el proceso de desarrollo de software está lejos de ser un "proceso maduro". Así, tanto las grandes empresas como las pequeñas han realizado importantes esfuerzos para mejorar sus procesos de software. Un indicador de esto es el creciente número de iniciativas internacionales relacionadas con el proceso de mejora, tales como CMMI-DEV v1.2 [CMMI, 2006], ISO/IEC 15504:2004 [ISO/IEC, 2004], SPICE [ISO/IEC, 1998], ISO/IEC 12207:2008 [ISO/IEC, 2008] y MoProSoft [NYCE, 2005a; Oktaba, 2006]. Además del surgimiento de muchos métodos para la evaluación de los procesos de software de las organizaciones, tales como el Método de Evaluación Estándar de CMMI para la Mejora del Proceso (SCAMPI, *Standard CMMI Appraisal Method for Process Improvement*) [SCAMPI, 2006] y el ISO/IEC 15504:2004 [ISO/IEC, 2004], y ciclos de vida para la mejora, tales como IDEAL [McFeeley, 1996] e ISO/IEC 15504:2004 [ISO/IEC, 2004]. Por lo tanto, las iniciativas de mejora de procesos se han convertido en uno de los principales objetivos de las empresas desarrolladoras de software, dado que la calidad de un producto está directamente relacionada con la calidad del proceso empleado para producirlo. Así, la Mejora del Proceso de Software (SPI, *Software Process Improvement*) puede definirse concretamente como: *"un conjunto integrado de iniciativas cuyo objetivo es el mejoramiento de las actividades de desarrollo y/o mantenimiento de productos basados en software"* [Zahran, 1998] (p. 17).

En este sentido, con el objetivo de obtener mejores productos, las compañías requieren mejorar su proceso de desarrollo para incrementar la calidad del software producido y mantener el presupuesto y el calendario bajo control. Por ejemplo, la investigación de Komi-Sirviö [Komi-Sirviö, 2004] expone que *"el propósito de la mejora a menudo se enfoca en optimizar el desarrollo de software con el objetivo de aumentar la calidad del mismo. Por otro lado, el alcanzar esta meta puede acortar el ciclo de entrega, reducir los costos y aumentar la rentabilidad, o fortalecer la posición de la organización en el mercado. También puede existir la necesidad de demostrar la madurez en el desarrollo, lo cual requiere muchos cambios en las actividades actuales para el desarrollo de software"* (p. 54).

Actualmente el interés de las empresas desarrolladoras de software en la mejora del proceso se está extendiendo a las universidades a través de la incorporación de este tópico en sus planes de estudio, con el objetivo de proporcionar a los estudiantes el conocimiento necesario para dirigir este tipo de iniciativas en un entorno real. Sin embargo, las universidades no han puesto suficiente

atención al desarrollar material educativo enfocado a la mejora del proceso, que les ayude a interactuar con la industria de software. Por lo tanto, cualquier intento por desarrollar una iniciativa de mejora se torna completamente caótico dado que los conceptos importantes son desconocidos o no aplicados por los estudiantes. En consecuencia, el comprender cómo implementar exitosamente una iniciativa de mejora de procesos es uno de los problemas más desafiantes en el campo actual de la Ingeniería de Software. A pesar de que la literatura sobre la mejora del proceso aporta muchos casos de estudio de compañías exitosas y descripciones de sus programas de mejora [Dyba, 2005; Unterkalmsteiner et al., 2012], los esfuerzos en investigación sobre su enseñanza son limitados e inconclusos, y sin una justificación adecuada tanto teórica como pedagógica es imposible enseñar a los estudiantes a conducir exitosamente un esfuerzo de mejora.

En el contexto de una iniciativa de mejora, el estado actual del proceso de desarrollo de software es evaluado y se determinan acciones de mejora dependiendo del resultado obtenido. Sin embargo, estas acciones de mejora están basadas en modelos de proceso (o modelos de referencia), como CMMI-DEV v1.2[1] o MoProSoft[2], y son cotejadas a través de entrevistas y análisis de documentos. Así, se genera una buena cantidad de conocimiento por lo que es necesario gestionar los activos del conocimiento, tales como lecciones aprendidas, prácticas fuertes (o fortalezas del proceso actual), prácticas débiles (o debilidades del proceso actual), problemas y soluciones, recomendaciones y planes, aspectos que contribuyen no solamente al aprendizaje individual sino a la iniciativa de mejora en sí. En este contexto, la investigación de [Ardis et al., 2008] clasifica como "suaves" las diferentes habilidades sociales que los estudiantes de la Ingeniería de Software deben desarrollar. Concretamente la observación, la revisión, la presentación oral de información, la escritura, la planificación, la cooperación, la reflexión y el juicio son habilidades sociales que los estudiantes deben desarrollar para conducir iniciativas de mejora en un entorno de desarrollo de software.

En este sentido, la investigación de Viana et al. [Viana et al., 2012] señala que la comprensión sobre cómo lidiar con los aspectos técnicos y humanos involucrados en iniciativas de mejora del proceso es todo un reto. Los aspectos técnicos no son suficientes para garantizar el éxito de las actividades; los factores humanos (e.g. las experiencias, opiniones, y percepciones) impactan en la efectividad de las iniciativas de mejora y los estudiantes de posgrado deben aprenderlo a través de la práctica real. Los estudiantes deben aprender que una iniciativa de mejora es afectada directamente por muchos factores (e.g. recursos financieros, recursos humanos, objetivos de negocios, cultura y visión organizacional), y estos temas no pueden aprenderse únicamente con clases teóricas. En este sentido, la Figura 1.1 muestra que cuando se inicia una iniciativa de mejora, es imprescindible que los estudiantes determinen el alcance apropiado de la misma y planifiquen las actividades, considerando el tamaño de la empresa. Después, es necesario que evalúen el estado actual del

[1] De acuerdo al Instituto de Ingeniería de Software (SEI, *Software Engineering Institute*) de la Universidad de Carnegie Mellon, *"CMMI es un modelo de madurez para la mejora del proceso orientado al desarrollo de productos y servicios. Este modelo está constituido por las mejores prácticas que engloban las actividades para el desarrollo y el mantenimiento y que cubren el ciclo de vida del producto desde su concepción hasta su entrega y mantenimiento"* [CMMI, 2002] (p. 47).

[2] De acuerdo con la Secretaría de Economía, MoProSoft es un modelo para la mejora y evaluación de los procesos de desarrollo y mantenimiento de sistemas y productos de software que fue desarrollado por la Asociación Mexicana para la Calidad en Ingeniería de Software a través de la Facultad de Ciencias de la Universidad Nacional Autónoma de México (UNAM), para obtener una norma mexicana (denominada NMX-059/01-NYCE-2005) que resultara apropiada a las características de tamaño de la gran mayoría de empresas mexicanas de desarrollo y mantenimiento de software.

proceso, "piloteen" las acciones correctivas propuestas y finalmente presenten sus soluciones al personal de la empresa.

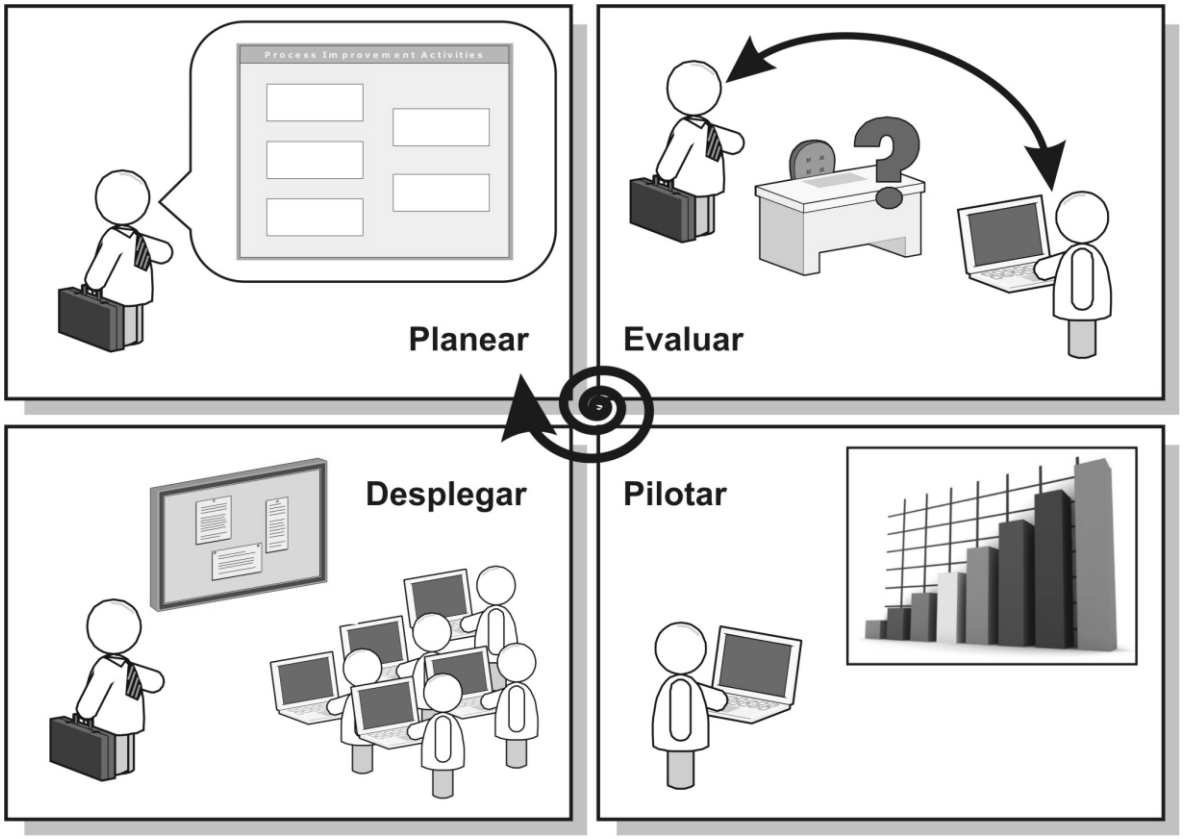


Figura 1.1. Contexto de un ciclo de mejora

Por lo tanto, a través del aprendizaje a lo largo de un ciclo de mejora es posible preparar a los estudiantes en estándares y modelos de proceso, y simultáneamente mejorar el entendimiento y los métodos de enseñanza de los profesores relacionados con esta área. En este sentido, con el objetivo de mejorar la educación de la Ingeniería de Software, una tendencia general se enfoca a enfatizar la experiencia práctica[3] de los estudiantes, relacionándola ya sea con la industria o algún entorno simulado. Sin embargo, hasta ahora, existen muy pocas propuestas para la enseñanza de la mejora de los procesos que sigan esta tendencia de forma eficaz a través de la representación de un contexto auténtico en el cual los procesos software sean diseñados en el aula. En otras palabras, es necesario incorporar enfoques alternativos de enseñanza que apoyen al desarrollo de las actividades de mejora como un objetivo básico de la educación especializada.

1.1. Importancia del problema y necesidad de resolución

Durante la última década la Ingeniería de Software ha cambiado de manera importante, impulsada por una combinación crítica de las fuerzas tecnológicas y económicas, y provocada por la evolución actual de la tecnología y las presiones económicas causadas por la globalización. En este sentido, es necesario replantear muchas de las suposiciones que han proporcionado la base para la

[3] El término para denominar correctamente esta forma de aprendizajes es "*hands on*", que se refiere a la interacción humana, a menudo con la tecnología, para incrementar la participación del estudiante de una forma directa y práctica [Joyce et al., 2013].

educación de la Ingeniería de Software en el pasado, y hacer cambios fundamentales en la manera en que se educa a los ingenieros de software con el fin de: 1) prepararlos para actuar en los entornos cada vez más dinámicos que las empresas de software establecen, y 2) dotarlos de las perspectivas y habilidades que necesitarán para hacer frente a los desafíos, cada vez mayores, que encontrarán en su vida profesional. Por lo tanto, la educación a nivel posgrado, particularmente la que se relaciona con áreas tecnológicas, requiere de metodologías modernas para su efectiva impartición. Regularmente las universidades hablan de la importancia de la adquisición de competencias genéricas o transversales como la capacidad de resolución de problemas reales, trabajo en equipo, habilidad de comunicación oral y escrita, adaptación a situaciones nuevas en entornos altamente demandantes, planificación y organización, etc.; sin embargo, dichas competencias son difíciles de enseñar y habitualmente son adquiridas, si es que llegan a serlo, de forma accidental por los estudiantes.

Al respecto, la Secretaría de Economía de México afirma que es necesario contar con empresas altamente competitivas que a su vez estén soportadas por talento humano competente no sólo en el uso de tecnologías, sino también en las habilidades gerenciales (o de gestión de procesos y de recursos) necesarias para ofrecer productos y servicios de clase mundial. De manera similar, Watts Humphrey[4] afirmó en el 2008 que la educación de la Ingeniería de Software sería uno de los tres principales impulsadores de la economía mexicana, junto con los apoyos gubernamentales y la colaboración de la industria privada [Humphrey, 2008]. Independientemente de las perspectivas, es claro que ambas concuerdan en que el fortalecimiento de la competitividad de los ingenieros de software en México, sobre todo de los recién egresados, podría ayudar a aliviar el problema del crecimiento económico del país. Por lo tanto, se cree que la clave de una mayor competitividad está en la educación. Sin embargo, la educación actual de la Ingeniería de Software no siempre expone a los estudiantes a proyectos grandes, complejos, y del mundo real. Ya que los estudiantes, incluso los de nivel posgrado, suelen trabajar en proyectos académicos más pequeños e hipotéticos en los que los beneficios de la adopción de procesos eficientes o técnicas de Ingeniería de Software no son evidentes. Esta desconexión entre la enseñanza de la Ingeniería de Software y las prácticas socavan la competitividad de los estudiantes de posgrado que posiblemente se integren a la industria en un futuro cercano.

1.1.1. Revisión de literatura

En la actualidad, un problema importante es la forma en que se desarrollan los cursos relacionados con la Ingeniería de Software. De acuerdo con von Wangenheim y Hauck [von Wangenheim & Hauck, 2010], las lecciones en forma de exposición continúan siendo la técnica instruccional dominante en, básicamente, todos los niveles de educación y entrenamiento de la Ingeniería de Software. Mientras estas lecciones son adecuadas en la presentación de conceptos abstractos e información factual, no lo son para alcanzar objetivos cognitivos superiores que dependan de la aplicación y transferencia del conocimiento a situaciones en un entorno real [Cater-Steel et al., 2006]. La investigación de Mead [Mead, 2009], argumenta que el profesionalismo continua siendo un tópico importante en la enseñanza de la Ingeniería de Software, y es aún impredecible si una licencia o una certificación serán la tendencia que domine en el futuro. Más

[4] Watts Humphrey fue uno de los iconos de la Ingeniería del Software y uno de los máximos representantes, al nivel de Barry Boehm, Fred Brooks, y Victor Basili, que han ayudado a definir esta área joven. Humphrey, reconocido como el "padre de la calidad del software", dedicó la mayor parte de su carrera a abordar problemas en el desarrollo de software, incluyendo aumento de costos, problemas en las planificaciones de los proyectos, problemas de rendimiento, y defectos.

importante aún, la discusión que Mead establece en su trabajo afirma que la discusión sobre la calidad del software se convertirá en un tópico cada vez más común. En este contexto, alrededor del mundo se están desarrollando diversas iniciativas de innovación educativa que pretenden modificar las metodologías tradicionales utilizando nuevas técnicas entre las que se encuentra el aprendizaje basado en proyectos, el aprendizaje colaborativo, el enfoque construccionista, etc. De acuerdo con [Dingsøyr et al., 2000], un método eficiente para la enseñanza de la Ingeniería de Software es relacionar al curso ya sea con la industria o con un entorno simulado.

En el contexto de la mejora del proceso, algunos enfoques han sido propuestos siguiendo este método. Por ejemplo, el Instituto Tecnológico de Georgia estableció un curso llamado *Real World Lab* [Moore & Brennan, 1995], donde los estudiantes de pregrado fueron involucrados con proyectos reales de la industria, y con productos y clientes también reales. Además, los estudiantes tomaron parte en el desarrollo de una evaluación basada en el modelo CMM [Paulk et al., 1993] sobre la industria local a través de entrevistas. Otros ejemplos fueron las clases diseñadas para los estudiantes de la Universidad de Texas [Werth, 1994], donde el modelo CMM y el estándar ISO 9000 [Marquardt, 1992] fueron introducidos para enseñar y aplicar algunas ideas sobre la mejora de la calidad; así como el curso de Ingeniería de Software impartido en la Escuela Politécnica de Montreal [Robillard et al., 1994], donde los estudiantes utilizaron una versión simplificada del modelo Trilliium [Coallier, 1995] para evaluar sus proyectos.

La investigación de [Dingsøyr et al., 2000] describió las opciones principales para el diseño de un curso sobre la mejora del proceso. El curso fue organizado alrededor de un caso de estudio de la industria, donde los estudiantes tuvieron contacto con representantes de las empresas locales. Los resultados obtenidos mostraron problemas relacionados con los temas de calidad y procesos, dado que la participación fue demasiado superficial; además, se registraron problemas con la participación de los estudiantes. Cabe mencionar que el curso diseñado no incorporaba el uso de la tecnología y se basó únicamente en el análisis de la información proporcionada para estructurar el caso de estudio.

Hislop discutió en [Hislop, 1999], la experiencia de la Universidad de Drexel en Estados Unidos al utilizar el modelo Proceso Personal de Software (PSP, *Personal Software Process*) [Humphrey, 1996] para enseñar la mejora del proceso en un curso de Ingeniería de Software de nivel pregrado. La investigación describió el contexto en el cual PSP es usado y discutió algunos problemas relacionados con los estudiantes, la estructura del curso, las características de PSP, y la participación del profesor. El trabajo de Hislop incluyó algunos resultados preliminares que proporcionaron una idea sobre el impacto del curso en las actitudes del estudiante hacia la Ingeniería de Software.

Jaccheri [Jaccheri, 2002], describió un curso sobre la calidad del software y la mejora del proceso. El curso de Jaccheri abarca lecturas y trabajos grupales sobre proyectos en colaboración con la industria local de software. La principal contribución de este trabajo está relacionada con el patrón de interacción propuesto para colaborar con la industria. El modelo de interacción se basó en preguntas de investigación, presentaciones de la industria, y presentaciones subsecuentes de preguntas y documentos escritos por los estudiantes.

Calvo-Manzano et al. [Calvo-Manzano et al., 2008] presentan un caso de estudio que describe la enseñanza de la versión estudiantil del Proceso Grupal de Software (TSPi, *Introduction to the Team Software Process*) [Humphrey, 2000] a estudiantes de cuarto año de pregrado, agrupados por equipos, de la Facultad de Informática de la Universidad Politécnica de Madrid. En este trabajo, se analizan y discuten los logros de los equipos después de recibir el entrenamiento y usar TSPi.

Otro ejemplo de investigación en la educación de la mejora del proceso es un curso de pregrado para Ingeniería de Software, propuesto por Hawker en [Hawker, 2009], para enseñar a los

estudiantes los conceptos básicos del área. El diseño del curso está basado en el Metamodelo OMG de Ingeniería para el Proceso Software [Bézivin & Gerbé, 2001], y el estándar de la IEEE para desarrollar un Ciclo de Vida para un Proyecto de Software [IEEE, 2006] como medios para modelar y comparar diseños alternativos de procesos, y proporcionar mecanismos para ensamblar componentes de procesos reutilizables.

Figueiredo et al., [Figueiredo et al., 2010] crearon un curso de pregrado para la Ingeniería de Software en la Universidade de Brasília, que se enfoca en el desarrollo y la calidad del software, y que se basa en modelos para la mejora del proceso, tales como CMMI, el Modelo de Referencia del Programa para la Mejora del Proceso Software Brasileño (MR-MPS, *Reference Model of the Program for the Improvement of Brazilian Software Process*) [Rocha et al., 2005]; y las recomendaciones proporcionadas por documentos especializados, como el Cuerpo de Conocimiento sobre la Educación de la Ingeniería de Software (SEEK, *Software Engineering Education Knowledge*) [Sobel, 2003].

Un enfoque diferente es propuesto por von Wangenheim y Hauck [von Wangenheim & Hauck, 2010], adoptando un enfoque constructivista para la enseñanza de los procesos de evaluación y mejora en un curso de posgrado en combinación con los proyectos finales de un curso de pregrado. Además de proponer y aplicar el diseño del curso, los autores investigaron el impacto del curso sobre el aprendizaje, su adecuación, sus fortalezas y debilidades, mediante la aplicación de cuestionarios antes y después de que el curso fuera aplicado.

El trabajo de Mendoza et al., [Mendoza et al., 2009], enfoca sus esfuerzos en el desarrollo de una herramienta basada en los estándares de *e-learning*: el proyecto HEPALE!. El sistema *e-learning* intenta resolver los problemas repetitivos en la aplicación de un nuevo modelo de mejora de procesos, relacionados con la falta de conocimiento sobre cómo funciona un modelo, su diseminación y las herramientas disponibles que pueden ser usadas como soporte y facilitar así su entendimiento.

Por último, la investigación presentada en [Garcia & Pacheco, 2012] establece un enfoque que integra una metodología de trabajo en equipo (TSPi) y el aprendizaje basado en proyectos (PBL, *Project Based Learning*) para mejorar las habilidades de gestión de proyectos software en estudiantes de pregrado. Como apoyo complementario, se desarrolló una herramienta computacional interactiva, la Plataforma de Trabajo en Equipo, para la enseñanza de la Ingeniería de Software en colaboración con la industria local de software.

En resumen, los estudios mencionados reflejan una clara tendencia por mejorar la enseñanza de la mejora del proceso en base al diseño de cursos y la colaboración activa de la industria del software en el proceso de formación. Sin embargo, en México el contenido sobre la mejora del proceso es regularmente aprendido en cursos de capacitación profesional o cursos para certificaciones profesionales. Por lo tanto, resulta importante ofrecer alternativas diferentes a los estudiantes para que aprendan los conocimientos y habilidades técnicas necesarias mediante el uso de enfoques alternativos como parte de su educación formal [Biberoglu & Haddad, 2002]. Por ejemplo, de acuerdo con Zhu, Wang and Tan [Zhu, 2004] las técnicas tradicionales empleadas en la educación de la Ingeniería de Software deberían ser complementadas por simulaciones para conseguir una experiencia de aprendizaje más eficaz.

Teniendo esta idea en mente, esta tesis pretende diseñar e implementar un enfoque colaborativo para planificar, conducir y "pilotar" iniciativas de mejora de procesos de desarrollo en empresas reales. Con esta propuesta se pretende establecer un enfoque iterativo educacional a nivel

posgrado para la mejora continua a través de proyectos reales en colaboración con la industria local de software.

1.2. Hipótesis de la tesis

La hipótesis del presente trabajo se plantea como una relación causal y se enuncia de la siguiente manera:

"La introducción de un enfoque colaborativo permitirá mejorar las habilidades de los estudiantes de posgrado para desarrollar iniciativas de mejora en un entorno real de trabajo."

1.3. Objetivos de la tesis

El objetivo general de esta tesis es:

1.3.1. Objetivo general

Implantar un enfoque colaborativo en el área de mejora del proceso de software como soporte al proceso de enseñanza/aprendizaje a nivel posgrado a través de la interacción con la industria local de software.

1.3.2. Objetivos específicos

Para alcanzar el objetivo general será necesario conseguir ciertos objetivos específicos. Éstos establecerán las aportaciones esperadas al final de la tesis y se resumen de la siguiente manera:

- *Realizar una investigación sobre la literatura existente relacionada con enfoques educativos desarrollados como soporte a la educación de la Ingeniería de Software, particularmente en el área de Mejora del Proceso de Software.*
- *Establecer una base comparativa con propuestas similares (si existiesen) a manera de benchmarking (comparativo de propuestas), de tal forma que permita identificar aspectos comunes para definir una solución en el contexto analizado.*
- *Establecer una estrategia de desarrollo para construir un soporte computacional que apoye al enfoque educativo propuesto como solución de esta tesis, considerando a MoProSoft como el modelo base de referencia para su aplicación en empresas mexicanas.*
- *Probar el enfoque educativo en, por lo menos, un curso y dos empresas de desarrollo de software del Estado de Oaxaca.*
- *Obtener datos sobre la implantación del enfoque para confirmar, o no, la hipótesis establecida anteriormente.*

1.4. Delimitaciones de la tesis

Esta tesis está enmarcada por los siguientes puntos:

- El enfoque colaborativo se enfocará a proporcionar un entorno interactivo para establecer iniciativas de mejora únicamente en Micro, Pequeñas y Medianas Empresas (MiPyMEs) de desarrollo de software (en el trascurso de la tesis se referenciaran a ellas como "industria local

de software"). En este sentido es importante mencionar que el enfoque únicamente se centra en la formación del estudiante y no de la empresa.

- El enfoque colaborativo apoyará a los estudiantes en la aplicación del modelo de mejora IDEAL [McFeeley, 1996], que es un modelo genérico e independiente del modelo de referencia que se desee utilizar durante la iniciativa. En relación al modelo de evaluación, este enfoque apoyará al estudiante en el uso del modelo EvalProSoft [Oktaba et al., 2004; NYCE, 2005c], por su relación con el modelo mexicano MoProSoft. Por último, el modelo de referencia que soportará el enfoque propuesto es MoProSoft [Oktaba, 2006; NYCE, 2005a] dado que es el modelo creado para México.
- Dada la extensión de los modelos anteriormente mencionados, el enfoque colaborativo cubrirá las áreas de proceso relacionadas con los dos primeros niveles de MoProSoft, que son los idóneos para introducir a los estudiantes y a las empresas en el proceso de adopción.
- La validación del enfoque colaborativo propuesto será realizada en por lo menos un grupo de la Maestría en Tecnologías de Cómputo Aplicado de la Universidad Tecnológica de la Mixteca y en dos MiPyMEs del Estado, por lo que se especificarán detalladamente las características de la experimentación si se desea replicar el estudio en otro contexto y obtener resultados similares.

1.5. Aproximación a la solución

La presente tesis asume que es más fácil aprender a conducir iniciativas de mejora cuando existe un enfoque colaborativo que soporte las actividades resumidas en la Figura 1.1 y que promueva la experimentación con empresas reales. Mejor aún, si una vez que se puede aprender a través de la experimentación se introduce el trabajo colaborativo (es decir, estudiantes apoyándose entre sí para aprender más rápido) se estaría agregando una alta dosis de realidad a los cursos de Ingeniería de Software.

Así, esta investigación plantea el desarrollo de un enfoque diferente para que los estudiantes de posgrado aprendan a conducir iniciativas de mejora a través de un enfoque iterativo para la mejora de procesos. Siguiendo este enfoque, el primer paso sería que los estudiantes entiendan lo que existe en la empresa y determinen qué causa los problemas más importantes. Entonces podrían idear soluciones a través de planes de acción y evaluarlas en proyectos piloto o incluso en experimentos. Por consiguiente, hasta que una solución sea eficaz y eficiente deberá integrarse al proceso existente de la empresa colaboradora. Es obvio que a través del uso de este enfoque y la práctica con una empresa real, la responsabilidad del profesor será guiar a los estudiantes en la realización de las actividades del ciclo de mejora y deberá también evidenciar, con el trabajo práctico, la importancia de los recursos (financieros, humanos, y de infraestructura) y de la cultura y visión organizacional en una mejora de procesos (véase Figura 1.2). Al desarrollar este trabajo se pretende crear una oferta de solución alternativa, que al discutirla con otros estudiantes y profesores, permitirá demostrar que un enfoque colaborativo puede apoyar al proceso de enseñanza/aprendizaje a nivel posgrado. Ahora bien, la idea propuesta en la Figura 1.2 se construirá a través de modelos ya existentes que reducen la complejidad del problema y facilitan el diseño e implementación de una solución práctica.

En este sentido, el enfoque colaborativo recibirá el soporte de una herramienta computacional que es desarrollada de acuerdo a los lineamientos del *modelo incremental* que combina elementos del modelo lineal secuencial (aplicado repetidamente) con la filosofía interactiva de construcción de prototipos, que a partir de un conjunto de requisitos funcionales logra la implementación a través de algún lenguaje de programación. En este contexto, se considera utilizar el modelo incremental para

el desarrollo de software propuesto por Mills et al. [Mills et al., 1980], como una forma de reducir la repetición del trabajo en el proceso de desarrollo de la herramienta y proporcionar oportunidades para concretar las decisiones sobre los requisitos detallados hasta que se adquiera cierta experiencia con su uso. De esta forma, la tesis proporcionará la información detallada sobre el diseño del soporte computacional (requisitos funcionales, diseño de bajo nivel, diseño de alto nivel, manual de usuario) que pueda servir como referencia a otros proyectos similares.

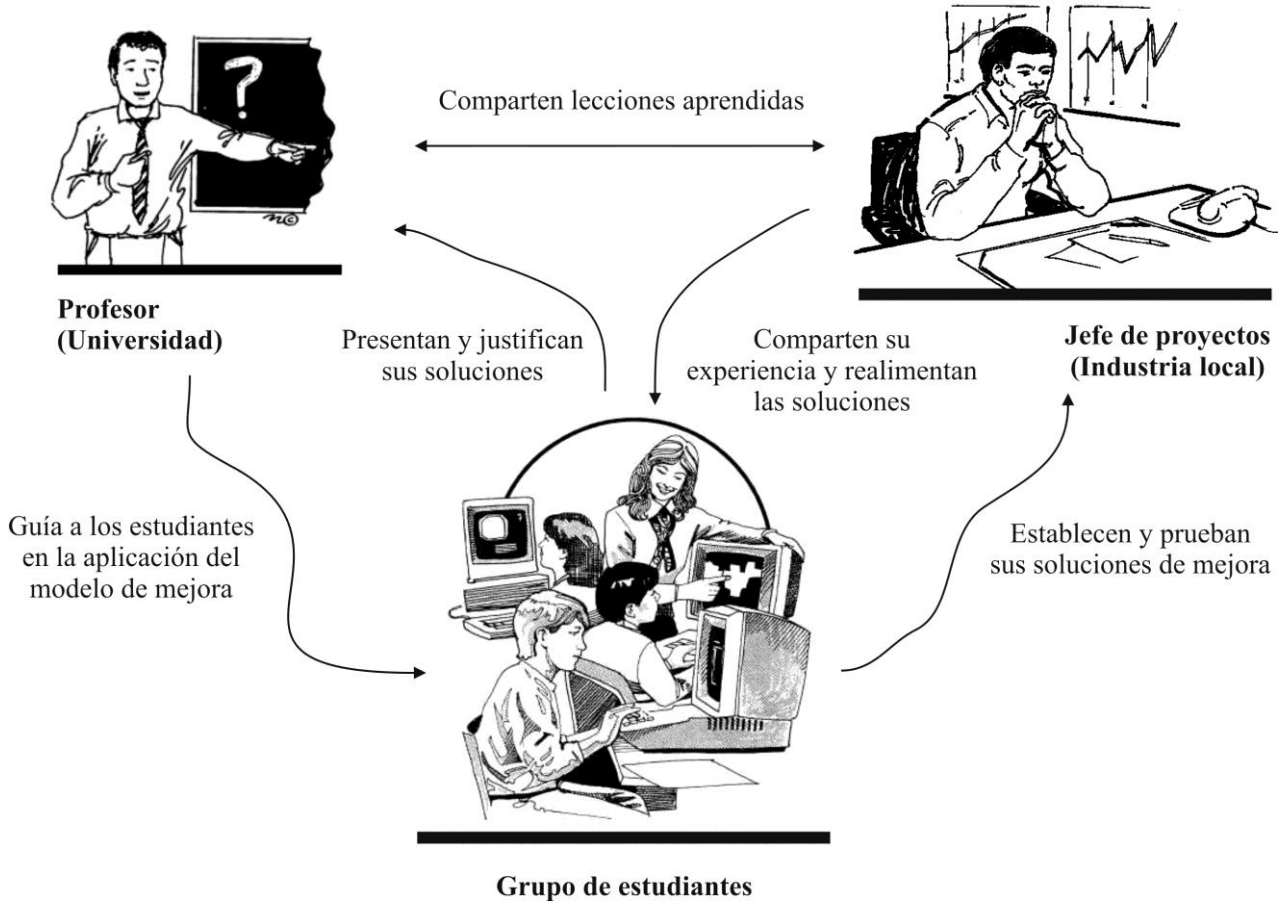


Figura 1.2. Contexto de la construcción de la solución planteada

Dado que se habla de una herramienta con enfoque educativo es necesario contemplar un *modelo instruccional*[5] que permita desarrollar una solución informática bajo un principio pedagógico bien establecido. Por esta razón, a lo largo de este primer capítulo se ha estado hablando reiteradamente del aprendizaje colaborativo, en el sentido en que la enseñanza se basará en el intercambio de ideas, conocimientos y experiencias entre los estudiantes y la industria. De forma adicional, los problemas deberán resolverse de forma colaborativa estableciendo equipos de cuatro estudiantes[6] de acuerdo a la teoría establecida por Göl y Nafalski [Göl & Nafalski, 2007]. En este sentido, el entorno de soporte dispondrá de un área de chat para la discusión e intercambio de ideas durante la resolución de los problemas. Más importante aún, es que se pretende mezclar estudiantes

[5] Un modelo instruccional es en esencia, un conjunto de guías o estrategias en las que se basan los enfoques de aprendizaje utilizados por los profesores [Clinton & Hokanson, 2012].

[6] De acuerdo con Göl y Nafalski [Göl & Nafalski, 2007], los grupos colaborativos son pequeños, normalmente no superan los cinco miembros, y se pretende que sean propicios para lograr un aprendizaje exitoso que puede demostrarse mediante la adquisición de conocimientos y habilidades o por la realización de un conjunto de tareas.

con diferentes capacidades cognitivas o niveles de aprendizaje, esto con la intención de mejorar la interacción social de los estudiantes al verse motivados por otros más adelantados, o incluso por el conocimiento compartido por los jefes de proyectos de las empresas colaboradoras. Por último, dado que se habla de Evaluación y SPI, será necesario considerar EvalProSoft [NYCE, 2005c] para diseñar un mecanismo de evaluación para empresas que deseen establecer iniciativas de mejora con el modelo MoProSoft. En este sentido, estos modelos recomiendan un número importante de instrumentos de recolección de datos que pueden ser utilizados para realizar evaluaciones sobre los procesos software en entornos empresariales: cuestionarios, encuestas, entrevistas y revisión de documentos, cada uno con sus propias ventajas y desventajas. Una de las técnicas más utilizadas es un cuestionario debido a que éstos se pueden aplicar a muchas personas de forma rentable, no invasiva, además de que proporcionan datos cuantitativos y los resultados se pueden analizar rápidamente [Gillham, 2000]. Esta tesis, además de contemplar el uso de cuestionarios, introducirá las técnicas de entrevistas y revisión de documentos para involucrar más al estudiante con el entorno real. Adicionalmente, se considera el modelo establecido en [García et al., 2011a] para establecer un mecanismo para la generación de planes y monitorización de actividades en coordinación con las empresas colaboradoras.

En relación a los aspectos técnicos de la herramienta de soporte al enfoque propuesto, el Capítulo 3 también justificará debidamente el uso de las herramientas tecnológicas utilizadas en su construcción, relacionando cada una de éstas con las características establecidas para la herramienta computacional.

1.5.1. Metodología aplicada

La metodología para construir la solución presentada en la Figura 1.2 establece siete fases que guiarán también el desarrollo de esta tesis (véase Figura 1.3) y que se resumen como:

1. Dado que se pretende crear una alternativa diferente para enseñar/aprender los aspectos relacionados con las iniciativas de SPI, será necesario definir primeramente un curso orientado a un modelo genérico de mejora, en este caso particular el modelo IDEAL. Este modelo es reconocido mundialmente como el punto de partida para aquellos interesados en iniciar una iniciativa de mejora, y es independiente del modelo de referencia que se pretenda implantar (CMMI-DEV v1.2 o MoProSoft, por ejemplo). Es importante mencionar que la definición del curso establecerá temas teóricos y prácticos a realizar a través del enfoque propuesto en la tesis.
2. Una vez que el curso haya sido definido será necesario adaptar el modelo MoProSoft a los contenidos del mismo. Para esto, será necesario hacer una descomposición de ambos modelos en términos de prácticas, guías, roles, responsabilidades, productos de entrada, productos de salida, criterios de medición para cada actividad/proceso, etc. Básicamente tendrá que realizarse una adaptación de los Niveles 1 y 2 de los procesos establecidos en la norma NMX-NYCE-2005 identificando aquellos elementos que deben ser explorados en estos niveles iniciales del modelo para que los estudiantes entiendan cómo funciona en un entorno real.
3. Dado que se pretende establecer un enfoque colaborativo que utilice como principal motor la interacción entre los estudiantes, los profesores, y la industria local, será necesario diseñar los medios, métodos, y procedimientos de un curso que soporte este enfoque y que permita que las iniciativas de mejora desarrolladas a lo largo del curso sean monitoreadas y evaluadas a través de una herramienta computacional.

4. En base al diseño obtenido se construirá la herramienta de soporte de tal forma que sea posible probar que el enfoque propuesto puede ser introducido en un curso de posgrado y potenciado a través del uso de tecnología.
5. Además del contenido propuesto para el curso, de los elementos identificados en MoProSoft para que sean parte activa del enfoque, y de la herramienta de soporte al curso, será necesario implantar las propiedades colaborativas definidas con el diseño del enfoque propuesto. Para esto será necesario establecer un conjunto de funcionalidades derivadas del comparativo empírico de herramientas a desarrollarse en el Capítulo 2 de la tesis y complementarla con otras obtenidas de la literatura actual.

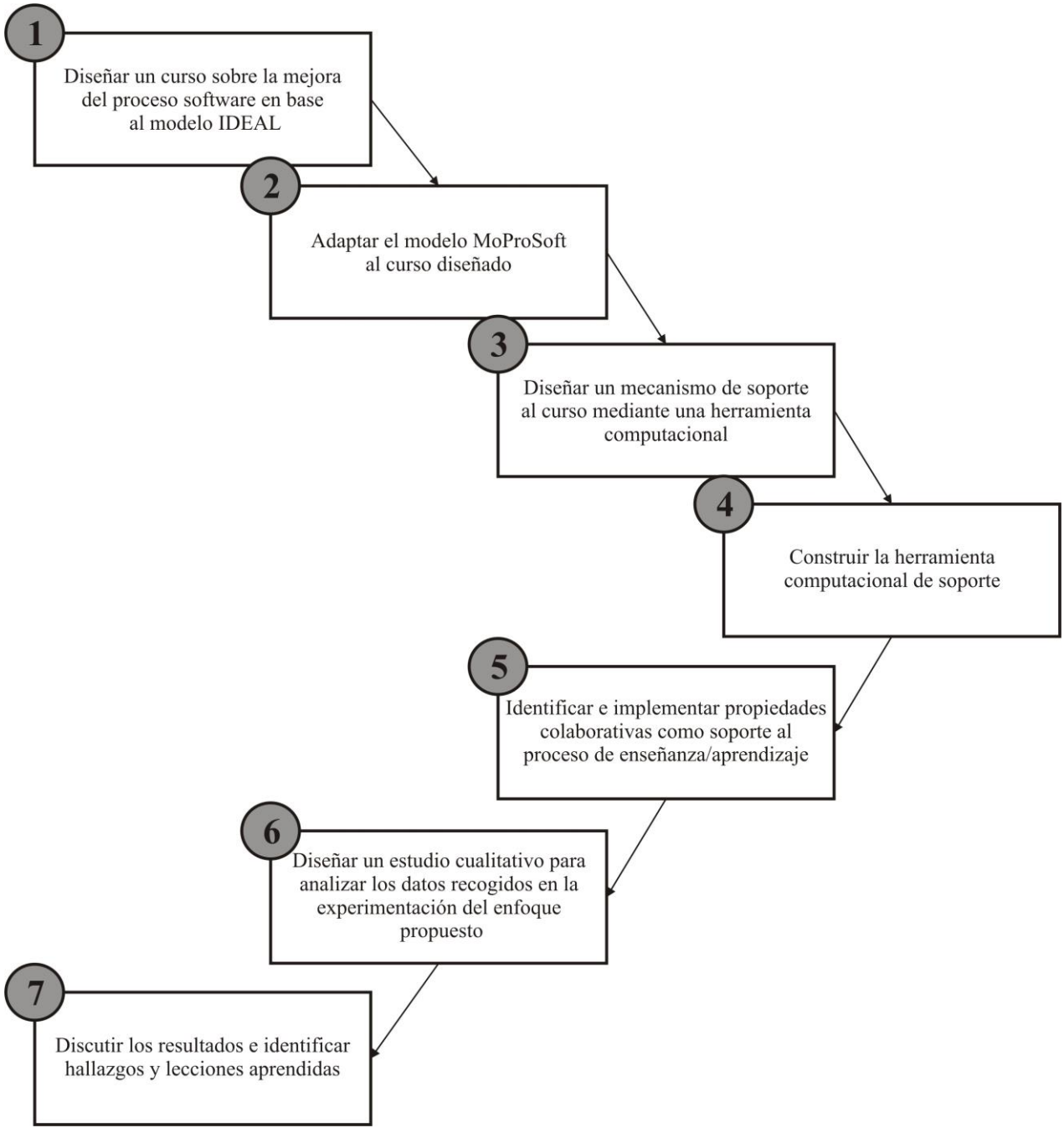


Figura 1.3. Metodología propuesta para alcanzar la solución planteada

6. Se deberá definir un estudio de tipo cualitativo para recabar las percepciones de las dos partes implicadas en este enfoque: estudiantes e industria. Para esto, se definirán cuestionarios de evaluación que deberán ser respondidos al finalizar el caso de estudio establecido y que aportarán información sobre la factibilidad de utilizar el enfoque propuesto.
7. Por último, los resultados obtenidos serán discutidos para su presentación y de identificarán hallazgos y lecciones aprendidas que puedan servir para investigaciones similares.

1.6. Estructura de la tesis

El resto del documento de tesis está organizado de la siguiente manera:

El Capítulo 2 presentará un marco teórico sobre los aspectos relevantes sobre la educación de la Ingeniería de Software. De manera más específica, este capítulo presenta un estudio sobre herramientas/enfoques similares que han sido utilizados principalmente para mejorar la enseñanza de la Mejora al Proceso Software a nivel pregrado y posgrado. Como resultado de este análisis, al final del capítulo se identifica un conjunto de requisitos funcionales que es utilizado más adelante para establecer una solución tecnológica.

El Capítulo 3 presentará el desarrollo de la metodología propuesta en la Figura 1.3 para sustentar el desarrollo del enfoque colaborativo producto de esta tesis. Al final de este capítulo, se agregarán también las consideraciones técnicas del soporte tecnológico propuesto para implementar el curso presentado.

El Capítulo 4 expondrá la validación del enfoque colaborativo mediante su implementación en un curso de la División de Estudios de Posgrado de la Universidad Tecnológica de la Mixteca y empresas del Estado de Oaxaca.

El Capítulo 5 presentará las conclusiones y líneas futuras sobre este trabajo.

Los Anexos mostrarán información complementaria sobre la solución propuesta: el Anexo A resume la lista de acrónimos utilizados en este documento; el Anexo B presentará el modelo computacional utilizado para guiar el diseño detallado del soporte tecnológico que comprende la especificación de los requisitos, el diseño de alto nivel y el diseño de bajo nivel; el Anexo C presentará un manual de usuario, y el Anexo D mostrará evidencia del artículo publicado.

Por último se presentarán las referencias bibliográficas utilizadas en el desarrollo de esta tesis.

1.7. Publicaciones generadas

A continuación se enlistan algunas de las publicaciones que se han generado durante el desarrollo del presente trabajo.

Autores:	García, I., Pacheco, C., Calvo-Manzano, J. & **Cruz, C.**
Título:	"Software engineering education for a graduate course: a web-based tool for conducting process improvement initiatives with local industry collaboration"
Revista:	Computer Applications in Engineering Education Índice JCR/THOMPSON ISI
Año:	2013

2. Marco Teórico

2.1. Aspectos relevantes sobre la educación de la Ingeniería de Software

A lo largo de los años, el desarrollo de proyectos y el trabajo en equipo han sido reconocidos como aspectos importantes en la educación de la Ingeniería de Software. De acuerdo con Chen y Chong [Chen & Chong, 2011], el desarrollo de proyectos dentro de un curso, los cuales algunas veces cuentan con la participación de la industria, cumplen la función de un "proyecto final" en los programas universitarios, ofreciendo una formación integral en el desarrollo colaborativo de software. Considerando las características de los proyectos estudiantiles y los aspectos sociales del desarrollo de software, es necesario incluir en el curso temas instruccionales que guíen a los estudiantes en fomentar el trabajo en equipo, formalizar y optimizar la participación de los interesados, y supervisar su trabajo, así como a mantener su esfuerzo colaborativo durante todo el proyecto. En este sentido, algunas publicaciones recientes (e.g. [van Vliet, 2006; Mead, 2009; Sancho-Thomas et al., 2009; Rooji, 2009]) indican la importancia de los proyectos reales en la educación de la Ingeniería de Software. Como se había mencionado anteriormente, es común que en los cursos de Ingeniería de Software se utilice la entrega de proyectos, denominados proyectos de fin de curso o proyectos finales, para determinar una calificación final puesto que los estudiantes deben trabajar en equipo y aplicar todo lo que aprendieron durante seis meses en su implementación con sistemas basados en computadora. La duración de estos proyectos puede variar de seis meses a un año, dependiendo de la cantidad de cursos recomendados por el plan de estudios para la enseñanza de la Ingeniería de Software, permitiendo así que los estudiantes tengan tiempo suficiente para completar todos los aspectos de su proyecto.

Sin embargo, los estudiantes a menudo carecen de experiencia y conocimientos para abordar en su totalidad proyectos de largo plazo; por lo tanto, los profesores tienen un reto importante al intentar motivar y mantener el trabajo en equipo de los estudiantes durante el desarrollo del proyecto. La investigación de Hassan [Hassan, 2008], por ejemplo, establece que los profesores no deben fomentar únicamente el trabajo en equipo, sino también supervisar el trabajo de los estudiantes contra los "síndromes clásicos" relacionados con el equipo (e.g. la delegación de la responsabilidad al estudiante mejor capacitado, la incompatibilidad de caracteres, diferencias importantes en las habilidades de cada estudiante, etc.). Es verdad que es útil para los profesores mejorar la supervisión de sus estudiantes, sin embargo esto alentaría también al proceso de enseñanza. Con el objetivo de que los estudiantes experimenten a través de problemas reales en lugar de utilizar simples ejercicios académicos, muchas universidades acogen la participación de la industria en los proyectos del curso. En estos casos, los estudiantes desarrollan prototipos para la empresa patrocinadora. Así, durante varios años algunos estudios han promovido la participación de la industria en el desarrollo de proyectos para cursos relacionados con la Ingeniería de Software

[Wohlin & orn Regnell, 1999; Reichlmayr, 2006; Fornaro et al., 2007; García & Pacheco, 2012]. Sin embargo, tal participación, si no es gestionada adecuadamente, puede generar más problemas que soluciones. Algunos otros estudios, [Dawson, 2000; van Vliet, 2006; van der Duim et al., 2007] por ejemplo, han argumentado que los estudiantes inexpertos no están preparados para tratar con las artimañas que se presentan en el mundo industrial y su participación podría tener un efecto contraproducente en el aprendizaje. Por otra parte, el profesor debe adquirir todas las responsabilidades contractuales, puesto que los estudiantes no tienen tanta responsabilidad con respecto a los resultados del proyecto.

Dado que la Ingeniería de Software tiene la reputación de producir graduados que están inadecuadamente preparados para aplicar sus habilidades en escenarios de la vida real, la investigación de [Hainey et al., 2011] sugiere que las técnicas educativas tradicionales tales como la asignación de roles, el aprendizaje mediante casos de estudio y el análisis de artículos científicos son insuficientes y que es necesario incorporar otros enfoques. Durante los últimos 20 años, los académicos y profesionales han dedicado grandes esfuerzos a la mejora y el avance del enfoque práctico y profesional de la Ingeniería de Software [Ardis et al., 2011]. En 1989, por ejemplo, el Instituto de Ingeniería de Software (SEI, *Software Engineering Institute*) de la Universidad de Carnegie Mellon publicó un importante informe sobre la educación de la Ingeniería de Software a nivel posgrado [Ardis & Ford, 1989], desde entonces varias universidades han utilizado sus directrices para establecer sus programas de posgrado. Sin embargo, el desarrollo de software ha cambiado dramáticamente desde ese año y se han multiplicado la escala del software, su complejidad y criticidad. Pero más importante aún es el hecho de que no se ha hecho ningún esfuerzo significativo para revisar y actualizar las directrices originales del SEI.

En relación a lo anterior, la Figura 2.1 muestra la evolución en la enseñanza de la Ingeniería de Software. En este sentido, muchos especialistas citan a la Conferencia de la Organización del Tratado del Atlántico Norte (NATO, North Atlantic Treaty Organization) como el comienzo del debate general sobre el contenido y la naturaleza de la Ingeniería de Software. Sin embargo, los primeros programas de posgrado siguieron los consejos reunidos en los informes de 1976 del Subcomité sobre el Modelo Curricular en Ingeniería de Software de la IEEE Computer Society (IEEE CS) [Freeman et al., 1976; Freeman & Wasserman, 1978]. En 1993, la Association for Computing Machinery (ACM) y la IEEE CS formaron un comité directivo para trabajar sobre la profesionalización de la Ingeniería de Software. En 1998, éste fue reconstituido como el Comité Coordinador de la Ingeniería de Software (SWECC, *Software Engineering Coordinating Committee*), que patrocinó el desarrollo de un cuerpo de conocimientos y un código de ética y conducta profesional para los ingenieros de software. Además, el SWECC patrocinó actividades educativas que condujeron al establecimiento de los criterios de la Junta de Acreditación de Ingeniería Tecnología (ABET, *Accreditation Board of Engineering Technology*) para la Ingeniería de Software [URL-2] y un modelo de plan de estudios para nivel pregrado [URL-3]. Es evidente que la Figura 2.1 presenta un enfoque un tanto centrado en los Estados Unidos de Norteamérica, pero muchas de las actividades representadas aquí tuvieron un alcance internacional. En este sentido, de acuerdo con [Ardis et al., 2011], se reconocen las contribuciones importantes al avance de la educación de la Ingeniería de Software que han ocurrido en todo el mundo, especialmente en el desarrollo y la acreditación de los programas educativos de software en Canadá, México, Europa, Australia, Asia y América del Sur.

El "Plan de Estudios para Posgrados en Ingeniería de Software 2009 (GSwE2009): Lineamientos Curriculares para los Programas de Posgrado en Ingeniería de Software" [Pyster, 2009] es un conjunto de lineamientos curriculares para los programas de maestría que fue desarrollado recientemente. Un grupo de educadores y profesionales en Ingeniería de Software

desarrollaron GSwE2009 como parte del proyecto de Plan de Estudios Integrado de Ingeniería de Software y Sistemas (ISSEC, *Integrated Software & Systems Engineering Curriculum*) en el Instituto Stevens de Tecnología. Un objetivo fundamental establecido en el GSwE2009 es cómo mejorar el estado de la práctica de la Ingeniería de Software y apoyar al mejor entendimiento y acuerdo sobre la naturaleza de los ingenieros de software profesionales.

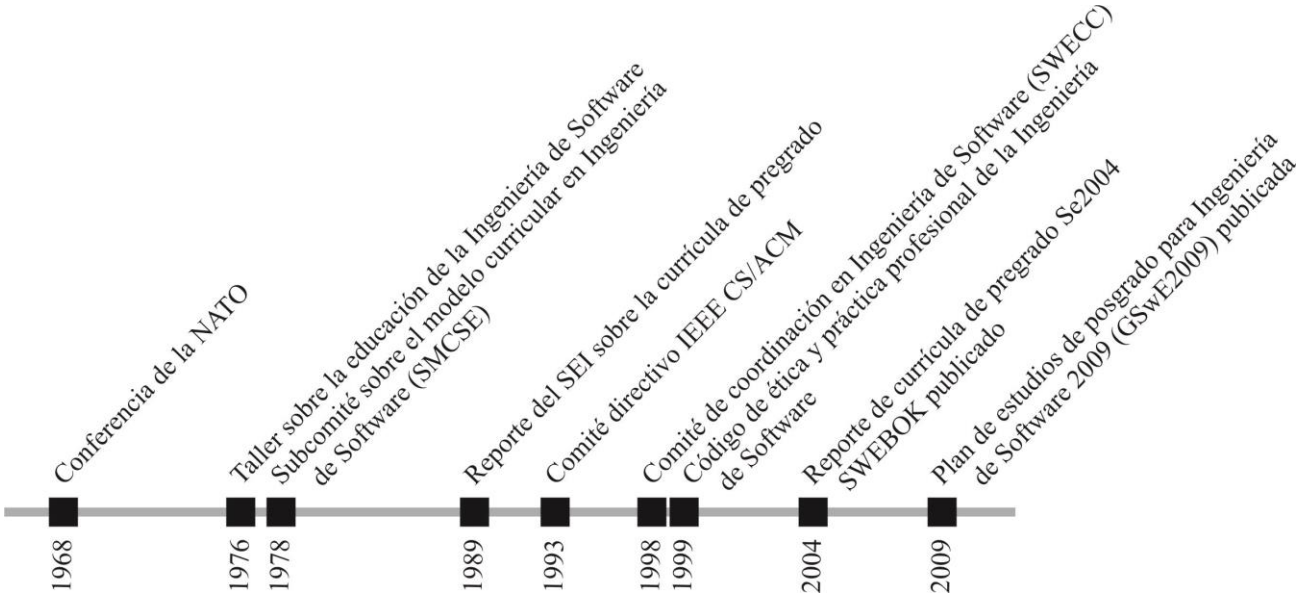


Figura 2.1. Línea de tiempo que muestra la evolución en la educación de la Ingeniería de Software [Ardis et al., 2011]

Aunque los 17 principios rectores del GSwE2009 influencian todos los aspectos de un proyecto, éstos particularmente ayudan a obtener los siguientes resultados de los estudiantes:

- *CBOK* (*Core Body Of the Knowledge*). Dominar el cuerpo central de conocimiento.
- *Dominio*. Dominar la Ingeniería de Software en un dominio de aplicación (e.g. finanzas, médico, transporte, telecomunicaciones) y en un tipo de aplicación (e.g. tiempo real, empotradas, de seguridad crítica, o sistemas altamente distribuidos).
- *Profundidad*. Dominar por lo menos un área o subárea del CBOK al nivel de la síntesis de Bloom [Krathwohl et al., 1964].
- *Ética*. Ser capaz de tomar decisiones éticas profesionales y practicar el comportamiento ético profesional.
- *Ingeniería de Sistemas*. Comprender la relación entre la Ingeniería de Software y la Ingeniería de Sistemas y aplicar los principios y prácticas de la Ingeniería de Sistemas en la Ingeniería de Software.
- *Equipo*. Ser un miembro eficaz de equipo, tanto en equipos multinacionales como en los distribuidos geográficamente; comunicarse eficazmente tanto de forma oral como escrita, y liderar un área de desarrollo de proyectos, tales como la gestión de proyectos, el análisis de los requisitos, el diseño de la arquitectura, la construcción, o el aseguramiento de la calidad.
- *Conciliar*. Ser capaz de conciliar los objetivos conflictivos del proyecto y alcanzar compromisos aceptables dentro de las limitaciones de costo, tiempo, conocimiento, riesgo, sistemas existentes, y de las organizaciones.

- *Perspectiva*. Comprender y apreciar el análisis de viabilidad, la negociación y la buena comunicación con las partes interesadas (o *stakeholders*) en un entorno típico de desarrollo de software y realizar bien dichas tareas. Tener hábitos efectivos de trabajo y ser un líder.
- *Aprender*. Ser capaz de aprender nuevos modelos, técnicas y tecnologías que van surgiendo y apreciar la necesidad de un desarrollo profesional continuo.
- *Tecnología*. Ser capaz de analizar una tecnología actual de software, articular sus puntos fuertes y débiles, compararla con las tecnologías alternativas, y especificar y promover mejoras o ampliaciones de esa tecnología.

La arquitectura de la Figura 2.2 muestra la estructura del plan de estudios GSwE2009. En él se identifican, a través del CBOK, el material mínimo que todos los programas deben incluir y la flexibilidad para que cada institución desarrolle sus propios programas en base al propuesto. La arquitectura del plan de estudios es similar al propuesto por Mark Ardis y Gary Ford en [Ardis & Ford, 1989] y es compatible con los programas de maestría existentes basados en el curso y los datos curriculares descritos por Arthur Pyster y sus colegas [Pyster et al., 2009]. La arquitectura del plan de estudios incluye material de preparación, materiales básicos, materiales específicos de la universidad, materiales opcionales, y la experiencia a través de proyectos. La línea de color negro en la Figura 2.2 representa el conocimiento básico de preparación para los estudiantes en un programa de maestría bajo el GSwE2009.

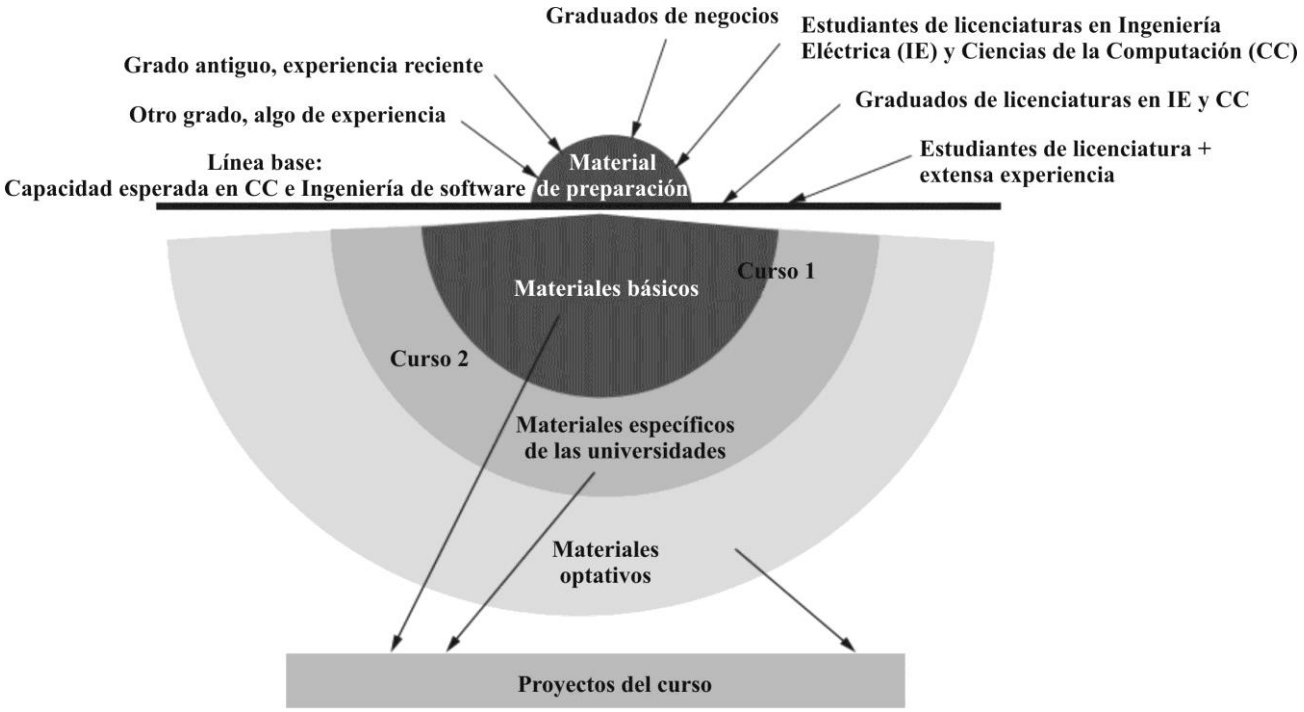


Figura 2.2. Arquitectura del GSwE2009 para un programa de maestría [Pyster, 2009]

Por ejemplo, un estudiante podría cubrir la preparación a través de la obtención de un grado o título de ingeniero (ya sea en las licenciaturas de Ingeniería en Computación, Ciencias de la Computación, Ingeniería Eléctrica o Ingeniería de Software), además de dos años de experiencia en el desarrollo de software. Los estudiantes podrán dominar el material que está debajo de la línea negra solamente después de alcanzar la preparación establecida en la línea base. Los programas individuales determinarán la forma de preparar a los estudiantes cuyos antecedentes no están a la

altura; el elemento de "material de preparación" que está por encima de la línea de las expectativas representa esta preparación adicional.

GSwE2009 recomienda reiteradamente que los estudiantes demuestren sus habilidades y conocimientos acumulados en una "experiencia culminante" que podría ser un proyecto, una práctica o una tesis. Los estudiantes que completen el plan de estudios deben entender y apreciar la importancia de la negociación, los hábitos efectivos de trabajo, el liderazgo y la buena comunicación con las partes interesadas en un entorno típico de desarrollo de software.

El contenido consiste principalmente en el CBOK y sus extensiones, que están fuertemente relacionadas con los resultados resumidos debajo de la Figura 2.1. Por ejemplo, la Figura 2.3 representa la organización del CBOK, designando el porcentaje de contenidos curriculares para cada área básica. Es importante observar que el CBOK ocupa aproximadamente el 50% del plan de estudios, proporcionando flexibilidad y especialización en el diseño curricular y permitiendo su extensión para apoyar los resultados de dominio y profundidad.

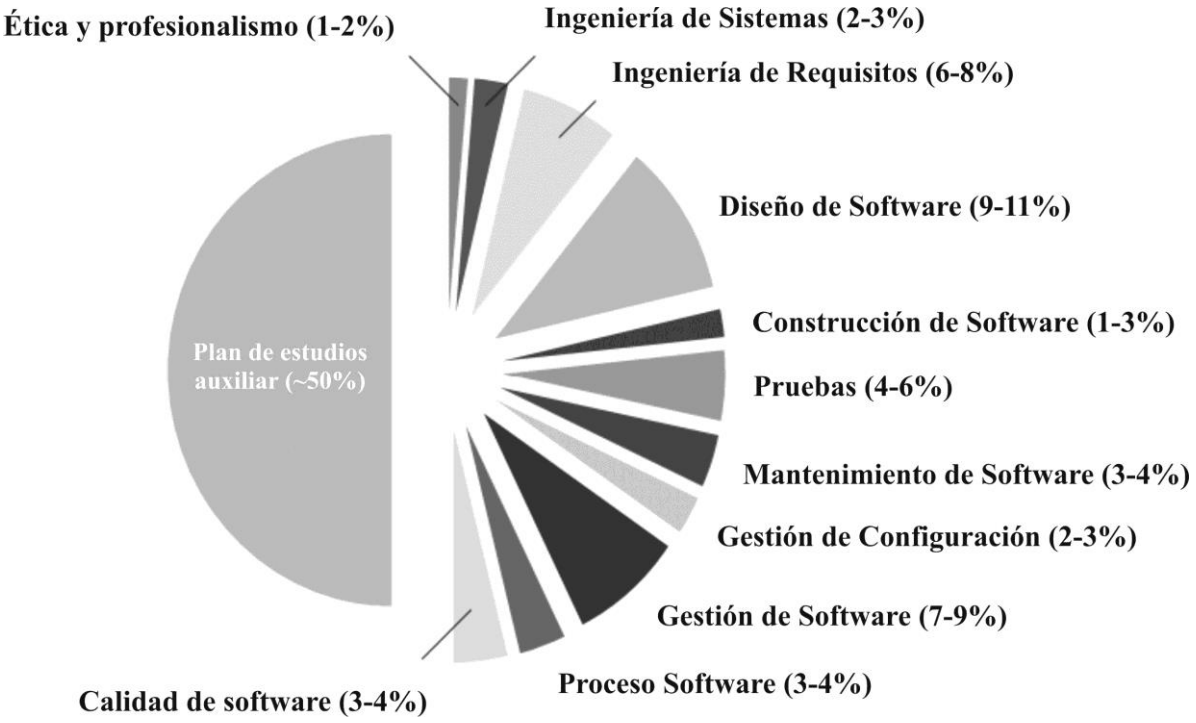


Figura 2.3. Organización del CBOK que muestra el porcentaje de contenido del plan de estudios para un programa de maestría bajo el GSwE2009 [Pyster, 2009]

La fuente principal para el desarrollo del CBOK fue el Cuerpo de Conocimiento de la Ingeniería de Software (SWEBOK, *Software Engineering Body of Knowledge*) [Abran et al., 2001]. El equipo de trabajo también derivó elementos de conocimiento del documento "Ingeniería de Software 2004" [LeBlanc & Sobel, 2004] y otras fuentes. En el estudio y análisis de estas fuentes, el equipo de trabajo decidió también que aunque la organización y contenido del SWEBOK dominara, era necesario realizar cambios en varias áreas y temas para apoyar los resultados esperados del GSwE2009 y dar cabida a las necesidades y opiniones de la academia, la industria y las sociedades profesionales de computación. Por lo tanto, GSwE2009 incluye dos áreas de conocimiento que no se encuentran en la versión actual del SWEBOK: fundamentos de la Ingeniería de Sistemas, y la ética y conducta profesional.

Sin embargo, a pesar de todo este trabajo en México estás guías no son consideradas en los programas de posgrado relacionados con la Ingeniería de Software. En este sentido, es importante que se haga un estudio exploratorio sobre la situación que predomina en nuestro país en términos de educación de pregrado y posgrado.

2.2. Análisis de la educación de la Ingeniería de Software en México

La investigación de [Garcia et al., 2010], iniciada en el 2004, expuso un análisis sobre los cursos relacionados con la Ingeniería de Software ofertados en dicho año académico (véase Tabla 1). Los datos fueron recogidos de los sitios web de cada universidad e instituto que cumplían con dos criterios: incluían carreras tecnológicas (relacionadas con la TI específicamente), y estaban incorporadas a la Secretaría de Educación Pública de México. Estos datos fueron corroborados más adelante a través de entrevistas con los responsables de cada programa. La selección del año 2004 se debió a dos aspectos cruciales en la educación de la Ingeniería de Software Mexicana: en primer lugar, en el 2004 se realizó una reestructuración importante de los planes de estudios universitarios (muchos programas fueron mejorados e incorporados con la sustitución del plan anterior); y en segundo lugar, la aparición del modelo Mexicano MoProSoft en el 2005 generó una reestructuración en los planes de estudio relacionados con la Ingeniería de Software particularmente.

El estudio incluyó una muestra de 210 universidades y centros especializados en educación tecnológica. De esta muestra, 182 universidades ofertaban cursos relacionados con la Ingeniería de Software, de las cuales solamente 8 basaban su educación en módulos (es decir, dividían la formación del alumno en cursos modulares ordenados y relacionados para obtener la "especialización" en Ingeniería de Software). Durante el 2004 se ofertaron un total de 292 cursos relacionados con la Ingeniería de Software. En relación a la información de la Tabla 1, se identificaron dos problemas importantes con la educación de la Ingeniería de Software en México:

- Asumiendo un promedio de 20 inscripciones en cada curso relacionado con la Ingeniería de Software, solamente 5,840 estudiantes (292 cursos multiplicados por 20 inscripciones por curso) tomaron estos cursos en el año académico 2004. Sin embargo, de acuerdo con la Secretaría Mexicana del Trabajo, solamente el 25% de estos estudiantes tenía las habilidades que la industria de software requirió en el 2005 cuando MoProSoft fue introducido en las empresas de software de pequeño tamaño (hasta 50 empleados). Por lo tanto, solamente 1,460 estudiantes representaron la mano de obra disponible para los puestos vacantes de ingenieros de software. La Tabla 1 proporciona un esquema general sobre la formación de estos ingenieros de software jóvenes y es obvio que los conocimientos proporcionados por las universidades eran muy limitados.
- En relación con las habilidades, una falla más importante es que la mayoría de los cursos relacionados con la Ingeniería de Software, 82 de 292, eran demasiado generales, concretamente "Ingeniería de Software", y por lógica estaban más orientados a temas introductorios, 87 cursos se enfocaban a la "Gestión de Proyectos de Software" (sin embargo, 70 cursos se centraban en proyectos generales y solamente 17 demostraron evidencias de centrarse específicamente en proyectos de software), 67 cursos utilizaban el "Modelado y Análisis de Software" como base para el desarrollo de software, y un pequeño número de cursos -36 en concreto- se preocupaba por la "Calidad de Software" tratando conocimiento muy limitado. A través de una cuidadosa revisión de los cursos ofrecidos, se determinó que solamente una pequeña porción de los conocimientos importantes sobre la Ingeniería de Software fueron cubiertos en el 2004, mientras que otros cursos, tales como "Modelos de Proceso", faltaban; lo que explica por qué los egresados relacionados con las

Ciencias de la Computación no estaban calificados para trabajar como ingenieros de software. De hecho, hubo un claro abismo entre los cursos ofertados por las universidades y las necesidades reales de la industria de software en esos años.

Tabla 1. Cursos relacionados con la Ingeniería de Software ofertados en el 2004

Nombre del curso	Universidades	Cursos
Ingeniería de Software	74	82
Arquitectura de Software	6	6
Gestión de Proyectos de Software	87	87
Pruebas de Software	1	1
Modelado y Análisis de Software	67	67
Métricas del Software	1	1
Desarrollo de Proyectos de Software	67	67
Evaluación de Proyectos de Software	10	10
Calidad del Software	36	36
Especificación Formal para el Diseño de Software	1	1
Desarrollo de Requisitos de Software	1	1
Educación basada en un solo curso	**60**	
Educación basada en módulos	**8**	

Por lo tanto, en el 2004 existían problemas relacionados con la cantidad y la calidad de los estudiantes. Sin embargo, es interesante observar que a finales del 2008, Watts Humphrey hizo un llamado urgente para mejorar la educación de la Ingeniería de Software en una Conferencia Internacional especializada organizada en México. En este sentido, en [Humphrey, 2008] se afirmó que la industria Mexicana de software enfrentaba un gran desafío para lograr su crecimiento; básicamente México debía integrar el soporte de gobierno-industria-academia y establecer planes modernos de estudio para crear un cuerpo de ingenieros altamente calificados, y aumentar los cursos relacionados con el conocimiento especializado y la formación basada en habilidades de PSP y entrenamiento basado en el desempeño de TSP. Sin embargo, este llamado de Humphrey fue ignorado en su momento y el recurso definitivo necesario para lograrlo no fue comprometido por el gobierno Mexicano hasta hace poco.

Es cierto que el gobierno intentó apoyar a la industria de software a través de la iniciativa PROSOFT, pero sin los recursos económicos apropiados destinados a la educación (solamente el 0.9% del Producto Interno Bruto (PIB) se destina a la educación de nivel superior), las universidades se veían limitadas para proporcionar educación acorde con los avances en la TI[7]. Por lo tanto, ¿qué sentido tendría promover un modelo propio de referencia, MoProSoft, si los estudiantes mexicanos no aprendían a utilizarlo? Además, un modelo de referencia (o de proceso), como la familia de modelos CMMI y MoProSoft, asume por su propia naturaleza que los ingenieros de software saben, por ejemplo, cómo estimar el costo y el calendario, cómo realizar planes de proyectos, cómo elicitar

[7] Del 5.7% que México destina del PIB para la educación, el 95% se utiliza para el pago de nómina y el 2.5% se utiliza para infraestructura e I+D, el otro 2.5% se emplea en servicios y otros. [Sampedro, 2011].

los requisitos o realizar la gestión de la configuración (control de versiones), pero estos conocimientos no se ofrecen ni se observan en la información resumida en la Tabla 1.

En un esfuerzo para solventar los problemas detectados en este análisis, se identificaron cinco disfunciones en la situación actual de la educación mexicana de la Ingeniería de Software: falta de atención en el proceso de software, falta de conciencia sobre la calidad del proceso de software, poca preocupación para planificar y controlar los proyectos de software, un abismo entre los cursos y la industria de software, y la falta de especialización en habilidades específicas. Estos fallos representan la base para la investigación sobre los temas y cursos que deberían desarrollarse para reforzar los conocimientos en Ingeniería de Software de los nuevos ingenieros de software y así aumentar la tasa de matriculación en cursos relacionados con dicha área.

2.2.1. Falta de atención en el proceso software

Normalmente un curso en México relacionado con la Ingeniería de Software en 2004 se enfocaba en los modelos universales del proceso (es decir, el modelo en cascada, el modelo de espiral, etc.); con esta definición establecida de proceso los estudiantes entendían qué debían hacer, lo que podrían esperar de sus compañeros de trabajo, y qué esperarían ofrecer a cambio. Esto les permitió centrarse en realizar su trabajo. Sin embargo, los estudiantes experimentaron dificultades en aprender a usar/escoger correctamente un modelo de proceso dentro de un proyecto de software complejo y grande. En cambio, con un curso de "Modelos de Referencia de Software" se explicaría la importancia de un proceso complejo de software donde un modelo de proceso (como el modelo de espiral) representa una mínima parte de éste. Bajo estas circunstancias del 2004, los estudiantes aprendieron a trabajar de forma independiente con poca coordinación y obtuvieron una concepción errónea del proceso de software dado que éste fue visto a menudo como una sobrecarga de trabajo, es decir, trabajo excesivo con documentos que originan pocos beneficios para justificar el costo.

Sin embargo, a finales del 2004 quedaría evidenciado en México que dado que los productos de software se vuelven más complejos, el proceso de software es demasiado complejo para ser abordado por los individuos en un tiempo establecido y de manera rentable. Por lo tanto, la atención en el establecimiento de un proceso eficiente de software se hizo latente. Para esto, los modelos reconocidos a nivel internacional debían ser incorporados al plan de estudios de pregrado. En este sentido, alguna metas adicionales fueron, para la industria de software de habla hispana, proporcionar una versión en Español de los modelos más populares[8] para facilitar su aplicación y, para las universidades, proporcionar materiales educativos en Español para apoyar la educación de la Ingeniería de Software. A pesar de la reciente creación del modelo mexicano MoProSoft, los modelos CMMI siguen siendo ampliamente utilizados por la industria mexicana de software; de hecho, mientras que en el 2011 la Agencia Mexicana de Normalización y Certificación (NYCE) reportó 304 certificaciones de MoProSoft (Nivel 0 = 6, Nivel 1 = 223, Nivel 2 = 73 y Nivel 3 = 2), el SEI informó 114 certificaciones para México (Nivel 2 = 56, Nivel 3 = 52, Nivel 4 = 1, Nivel 5 = 5). Una diferencia crucial entre ambos modelos radica en que el de origen mexicano se centra en pequeñas y medianas empresas, mientras que el otro fue creado para grandes empresas, pero existe evidencia documentada de su aplicación en las pequeñas. Mientras que este fallo puede tener muchas causas relacionadas con la educación de la Ingeniería de Software, las universidades no habían prestado suficiente atención a la introducción de cursos orientados al proceso de software.

[8] En el 2009 sale a la venta la traducción oficial al Español del Capability Maturity Model Integration for Development v1.2 (CMMI-DEV v1.2) [Chrissis et al., 2009] y en 2012 la traducción del Capability Maturity Model Integration for Development v1.3 (CMMI-DEV v1.3) [Chrissis et al., 2012]; ambas traducciones son las únicas reconocidas por el SEI.

Para ayudar a cambiar las cosas, cursos como "Modelos de referencia de software", "Calidad del Software", "MoProSoft" y "CMMI" empezaron a ganar atención a finales del 2004.

2.2.2. Falta de conciencia sobre la calidad del proceso

En el 2004 se pudo observar una falta general de interés en los cursos relacionados con la Ingeniería de Software sobre la calidad en los productos de software y en el proceso de desarrollo en general. En un curso de programación (previo a cualquier curso de Ingeniería de Software), los estudiantes aprendían a confundir las pruebas con la depuración. Regularmente, realizaban una gran cantidad de depuración para corroborar que sus códigos eran correctos para ser entregados, pero los programas no eran probados. La mayoría de los programas que los estudiantes escribieron durante este análisis fueron solamente depurados pero no probados, y rara vez fueron ejecutados más de un par de veces. De acuerdo a los planes de estudio del 2004, no había demasiado tiempo para hacer pruebas.

De acuerdo a la evidencia recogida en los cuestionarios, un aspecto importante fue que un número importante de universidades mexicanas incluían cursos de usabilidad para apoyar la enseñanza en temas de calidad del software. De hecho, a través de las entrevistas se determinó un énfasis exagerado en pruebas de usabilidad como una medida de calidad en el software. Incluso, fue común escuchar que algunos estudiantes de aquellas generaciones relacionaban a la "usabilidad" con la calidad en el "proceso de software" (no calidad del software, sino del proceso en sí). De acuerdo a los jefes de proyectos entrevistados, esto pudo representar un gran problema, dado que esta mentalidad fue asimilada por los graduados que entraron al mercado laboral a mediados del 2007. En este sentido, sugieren como buen cambio que cursos como "Mejora al Proceso de Software", "Verificación y Validación", "Pruebas de Software" y "Calidad del Software" recibieran mayor atención en los planes de estudio del 2005.

2.2.3. Poca preocupación para planificar y controlar los proyectos de software

Con el objetivo de indagar más sobre este fallo y para obtener más información acerca de la escena actual, en el 2011 se diseñó un enfoque basado en el modelado para apoyar a la evaluación del proceso de software en pequeñas empresas mexicanas de software [García et al., 2011]. A mediados de 2011 se utilizó este mecanismo de evaluación en 96 empresas de software que utilizaban MoProSoft como modelo de referencia. Para ello, se eligieron cuidadosamente a pequeñas empresas que hubieran contratado recientemente a, por lo menos, cinco graduados en el último año. El objetivo de este análisis se enfocó en obtener una visión instantánea sobre la educación de los graduados en los años 2008, 2009 o 2010 una vez que éstos fueron insertados en el contexto industrial. En la revisión de los resultados obtenidos, fue común encontrar que el 92% de los estudiantes contratados fueron incapaces de producir una descripción de un proceso común para la planificación y el control de los proyectos de software. Pero fue más preocupante observar que el 85% de los jóvenes ingenieros no sabía cómo estimar los costos, generar calendarios y presupuestos, establecer técnicas de medición o las medidas para el seguimiento y control de los proyectos. Todas las prácticas establecidas eran productos del trabajo diario y eran pobremente documentadas. Por lo tanto, ¿cómo podría producirse software de calidad con el modelo mexicano cuando los ingenieros de software no saben cómo realizar todas las prácticas recomendadas? Estos resultados evidenciaron el hallazgo de la Tabla 1 en el sentido de que en el 2004 el curso de "Gestión de Proyectos de Software" se centró más en proyectos generales (80.45% de cursos) y las universidades no estaban preocupadas por enseñar cómo planificar y controlar proyectos de software específicamente. Como un resultado adicional, se detectó que la educación mexicana de la Ingeniería de Software en el 2004 se centró más en temas de programación que en habilidades de gestión/administración de proyectos de software.

2.2.4. El abismo entre los cursos y las necesidades de la industria de software

Al revisar los cursos ofrecidos en el 2004, como se muestra en la Tabla 1, y al entrevistar a los responsables de los programas en las universidades, se determinó que ninguno de los cursos hizo hincapié en la brecha de conocimiento para preparar mejor a los estudiantes para el contexto de una empresa de software en el que se pueden encontrar inmersos como ingenieros de software o profesionistas. En este sentido, uno de los problemas cruciales en la industria mexicana de software (y en general) es que las empresas normalmente deben invertir grandes recursos económicos en la formación de nuevos trabajadores debido a sus deficiencias de conocimiento. En un entorno real industrial, los ingenieros de software deben comunicarse efectivamente con los usuarios, clientes y expertos del dominio con el fin de capturar los requisitos correctos; estimar de manera eficiente los costos del proyecto y calendarizar; formular planes apropiados; supervisar y controlar los planes establecidos; y tomar medidas correctivas cuando sea necesario. Aquellos cursos que proporcionan formación a este respecto pueden también ser útiles para otros estudiantes de informática. Sin embargo, un problema importante del 2004 en la enseñanza de la Ingeniería de Software en México era la brecha que existía entre los cursos ofrecidos por las universidades y las necesidades reales de la industria de software. En este sentido, los planes de estudio tendrían que ser actualizados con mayor regularidad de acuerdo a las demandas de la industria nacional; pero es más importante relacionar los cursos actuales con la industria local de software con el fin de preparar a los estudiantes de pregrado en un entorno real de trabajo. Es muy interesante observar que se hicieron observaciones similares en un estudio reciente a nivel Maestría con programas relacionados con las Ciencias de la Computación [Garcia et al., 2010].

2.2.5. Falta de especialización en habilidades específicas

A través de este estudio fue posible determinar que la mayoría de los cursos relacionados con la Ingeniería de Software están basados en exposiciones que utilizan el soporte de prácticas extracurriculares para complementar el material teórico proporcionado. Esta forma "convencional" de enseñanza promueve el individualismo en el aula y segrega a los estudiantes cualificados del resto. Las clases expositivas de fondo teórico necesitan ser apoyadas por plataformas tecnológicas propuestas para el desarrollo y aplicación de conocimientos específicos de la Ingeniería de Software [Hainey et al., 2011] y deben utilizar técnicas pedagógicas alternativas para promover el aprendizaje activo entre los estudiantes y obtener así mejores resultados. Así, en este sentido los problemas observados con los planes de estudio del 2004 fueron los siguientes: (1) los cursos ofrecidos se centraron en cuestiones generales y habilidades básicas de la Ingeniería de Software, (2) la mayoría de los cursos relacionados con la Ingeniería de Software seguían las clases expositivas de enseñanza técnica, promoviendo el individualismo entre los estudiantes, (3) no se incluyeron habilidades específicas como un tema normal o avanzado dentro de los cursos (habilidades especializadas como PSP/TSP, SixSigma, COCOMO II, MoProSoft o CMMI, por ejemplo); y lo que es más importante, el problema que temas como estimación de costos, métricas de software o desarrollo de requisitos no fueron contemplados en la educación de los alumnos.

2.2.6. Revisión de las acciones realizadas para eliminar los fallos del 2004

Como parte de esta tesis, la investigación realizada en el 2004 fue extendida al 2012 con la intención de establecer una base comparativa y determinar los cambios y acciones realizadas en relación a la educación de la Ingeniería de Software. Así, después de siete años de cambios la industria mexicana de software ha recibido fondos a nivel nacional, a través de la Secretaría de Economía, para promover su internacionalización. Se han creado programas de gobierno para apoyar esta tarea con la colaboración de académicos y especialistas de TI. Por otra parte, las universidades han actualizado sus planes de estudios relacionados con las Ciencias de la

Computación y educación en TI (donde se incluyen cursos relacionados con la Ingeniería de Software) y otras más han creado, en 2006-2007, el programa de "Ingeniería de Software" específico para pregrado. Estos esfuerzos se han realizado para intentar resolver los problemas de escasez y baja calidad de los ingenieros de software y pueden encajar perfectamente como la corrección de las cinco disfunciones identificadas. Este esfuerzo se basa principalmente en el establecimiento y la promoción de un plan educativo moderno para organizar una estrategia viable para la asignación de recursos, personas y actividades y una modernización de los planes de estudio de acuerdo a las necesidades de la industria. A continuación se explicará la relevancia de estos esfuerzos en las disfunciones mencionadas anteriormente.

2.2.7. Establecimiento de una estrategia viable para la asignación de recursos

Como una estrategia primordial para promover el fortalecimiento de las industrias de software y de TI, el gobierno mexicano estableció en su Plan Nacional de Desarrollo 2007-2012 [URL-1] el objetivo de promover la productividad y competitividad de la economía mexicana para lograr un crecimiento económico sostenible y acelerar la creación de empleos para mejorar la calidad de vida de los mexicanos; algo que no fue posible hacer. Este plan consideró la importancia estratégica de establecer condiciones que le permitieran a México lograr la vanguardia tecnológica, considerando que esto ha abierto importantes oportunidades de superación personal a través de un mayor acceso a la información. Para intentar alcanzar el objetivo de aumentar la competitividad del país, la Subsecretaría de la Industria y Comercio de la Secretaría de Economía lanzó en febrero del 2008 diez pautas para aumentar la competitividad 2008-2012. En este contexto, la octava directiva de este programa proponía transformar a México en el eje de distribución de servicios y logística, aprovechando las ventajas geográficas del país, el acceso preferencial a un gran número de mercados y la amplia dotación de la fuente más importante en el sector de servicios: capital humano. En el contexto de la Ingeniería de Software, la iniciativa PROSOFT surgió como la principal estrategia para aumentar la producción de software y servicios y obtener una ventaja competitiva en América Latina. A través de esta iniciativa, el gobierno mexicano proporciona el recurso económico para la certificación de profesionales, empresas de software (organizadas por clústeres) o académicos de una universidad previamente registrada. Este último aspecto ha tenido un impacto directo sobre la educación de los estudiantes dado que empezaron a obtener conocimiento actualizado de sus profesores. Otra estrategia lanzada por el gobierno mexicano en el 2008 fue el Programa para el Desarrollo de la Industria de Medios Interactivos (PROMEDIA) para crear las condiciones necesarias para garantizar el crecimiento y consolidación de la industria de los medios interactivos en México, así como para aumentar su competitividad internacional a través de la TI. PROMEDIA oferta recursos económicos para proyectos en los que la industria y las universidades establezcan un vínculo de trabajo. Además, en 2009, la Secretaría de Economía y el Consejo Nacional de Ciencia y Tecnología (CONACYT) firmaron un convenio de colaboración con el compromiso de construir un fiduciario con recursos concurrentes que se denomina "Fondo de innovación tecnológica" para apoyar proyectos que permitan la mejora de la competitividad de las empresas mexicanas. Particularmente, la "Demanda 4.1: Fortalecimiento de la industria de TI y comunicaciones a través de la certificación PSP/TSP" convocó a las universidades públicas y privadas, centros de investigación, laboratorios especializados y personas físicas y morales para presentar proyectos que pudieran ser compatibles con los recursos. Como apoyo a la mejora de la enseñanza de la Ingeniería de Software, esta llamada argumentó la necesidad de certificar a un miembro de cualquier empresa de software de cualquier clúster y a un académico de cualquier universidad en los modelos PSP y TSP, para convertirse en cuerpos de conocimiento que proporcionen capacitación a estudiantes y brinden consultoría a empresas.

Para concluir, en el 2009 la iniciativa "MexicoFIRST" fue establecida con el apoyo de la industria, la Secretaría de Economía y el Banco Mundial con el objetivo de generar capital humano capacitado en TI de nueva generación. A través de esta iniciativa, 21,395 personas fueron capacitadas y 17,116 fueron certificadas en 2011 (incluyendo académicos, jefes de proyectos, programadores, etc.). Los recursos que recibe esta iniciativa provienen del Banco Mundial y forman parte de la política pública y el programa PROSOFT. En relación al entrenamiento de los académicos, se ha firmado un acuerdo con el Instituto iCarnegie, de la Universidad de Carnegie Mellon, para la capacitación de 600 profesores en más de 30 universidades en México. Este personal será certificado en las metodologías y contenidos que la Universidad de Carnegie Mellon utiliza (CMMI, TPS, PSP, etc.). Del mismo modo, MexicoFIRST ha negociado precios preferenciales con Microsoft para las certificaciones más solicitadas.

2.2.8. Planes educativos modernos para organizar a las personas y a las actividades

En la actualidad, existen más de 5,000 universidades en México que están coordinadas para mantener actualizados los planes de estudio y establecer programas homogeneizados que promuevan una mejor educación en Ciencias de la Computación e Ingeniería de Software. En este esfuerzo, la labor de la Asociación Nacional de Instituciones de Educación en Informática A.C. (ANIEI) ha destacado para asegurar que el perfil de los egresados no solamente alcance los requerimientos empresariales actuales, sino que también les permita explorar y participar en nuevos mercados y líneas de negocio, así como fortalecer los esquemas de innovación de la empresa. Así, la Secretaría de Economía y la ANIEI propusieron la actualización de los planes de estudio, relacionados con las Ciencias de la Computación, de las universidades y han creado seis cursos extracurriculares basados en una plataforma de *e-learning*, con el fin de incorporar las necesidades reales de la industria en los perfiles académicos. Durante 2007, este proyecto fue presentado al comité de e-México[9] con el fin de desarrollar un perfil completo del modelo extracurricular. El contenido de los cursos fue colocado en una plataforma, denominada CapaciNet, para darle mayor alcance, difusión y disponibilidad. También NYCE y el Consejo Nacional de Normalización y Certificación de Competencias Laborales (CONOCER) han comenzado a trabajar para hacer certificable como competencia laboral los perfiles del modelo extracurricular.

En el 2005 se creó la Sociedad Academia-Industria-Gobierno en Tecnologías de la Información (IMPULSA-TI). Esta sociedad tiene el objetivo principal de constituir un espacio institucional que permita la articulación de iniciativas para promover el uso de la TI a través de esfuerzos coordinados entre la academia, la industria y el gobierno. Durante el año 2007, esta sociedad estableció el proyecto para un Sistema Inteligente de Información en Capacidades de la Industria de TI (SIICAP) para permitir la identificación y mapeo de los egresados de Ciencias de la Computación en todos los Estados de la República Mexicana. Otro avance en este sentido, fue la firma de un acuerdo de colaboración con el Foro de Cooperación Económica Asia-Pacífico (APEC) para el intercambio de académicos y su actualización en técnicas de enseñanza. Este acuerdo de colaboración ha establecido las bases para iniciar en México la educación innovadora y la formación de estudiantes a través del "Parque APEC de Edutainment". Los resultados relativos a las

[9] De acuerdo con la Agencia Informativa SNC del Tecnológico de Monterrey, el 12 de marzo de 2001 la Secretaría de Comunicaciones y Transportes convocó a los académicos, investigadores, instituciones públicas y privadas, Cámaras, Asociaciones, trabajadores del Sector Comunicaciones y Transportes, así como al público en general, a participar en el Foro de Consulta Ciudadana para el desarrollo del Sistema Nacional e-México. Para constituir el Sistema Nacional e-México se definieron tres ejes rectores o estrategias principales: Conectividad, Contenidos y Sistemas. También se contemplaron cuatro pilares básicos para el desarrollo de contenidos y servicios digitales, en temas de e-Aprendizaje, e-Salud, e-Economía y e-Gobierno.

habilidades desarrolladas por el capital humano serían observados a mediano plazo una vez que los estudiantes entren al mercado laboral. Sin embargo, es importante seguir trabajando en mejorar el rango de universidades y maximizar la ayuda para proporcionar a los estudiantes las herramientas y habilidades que necesita la industria mexicana de software.

2.2.9. Planes de estudio adaptables a las necesidades de la industria

En el 2010, la mayoría de las universidades mexicanas acordó establecer programas estandarizados para los programas de pregrado relacionados con las áreas de las Ciencias de la Computación. Actualmente, los consejos académicos de las universidades mexicanas han definido programas modulares con cursos básicos y cursos de especialización. Así pues, un programa de Ciencias de la Computación se define como un conjunto de cursos básicos (llamado "tronco común") que representa un conjunto de temas relacionados con los programas comunes de la misma área de conocimiento y que tienen que ser cursados en la etapa básica; y cursos de especialización que se distribuyen en tres áreas principales (Inteligencia Artificial, Redes y Comunicaciones, e Ingeniería de Software) y que los estudiantes deben cursar en una etapa avanzada para adquirir experiencia en un área específica. Este programa de módulos optativos en las universidades mexicanas comprende cursos básicos y avanzados para satisfacer mejor las expectativas de los estudiantes. A través de este programa modular los estudiantes de pregrado pueden elegir un dominio de TI y desarrollar habilidades más específicas. Sin embargo, mientras se desarrolla un plan modular de estudios, los materiales provienen continuamente de referencias modernas y especializadas (por ejemplo, [Bourque et al., 1999; CMMI, 2006; O'Regan, 2010;]); y su principal desventaja es la limitada cobertura de las áreas de conocimiento. En la Tabla 2, los temas relacionados con la Ingeniería de Software están separados por tres categorías: el arte de la práctica, la gestión del proceso, los y métodos de apoyo; cada uno con módulos en tres niveles de complejidad: preparatorio, de transición, e innovador. Estos cursos fueron establecidos en las universidades como un recurso inicial para evitar las cinco disfunciones detectadas en el 2004.

Por ejemplo, un curso de Ingeniería de Software puede ser mejorado agregando contenido nuevo y específico para aquellos estudiantes que quieran adquirir mejores habilidades, mientras que para otros estudiantes de otras áreas de especialización, un curso puede ser suficiente. Otro ejemplo es el curso de Calidad de Software incluido en la categoría de gestión del proceso. Una vez que el estudiante ha adquirido la base sobre la calidad del software a nivel de proceso, en el nivel de transición aprende una disciplina individual para desarrollar sus programas bajo PSP. Al mismo tiempo, otras universidades incluyeron al modelo CMMI ya que aborda problemas de procesos y de calidad a nivel organizacional. Es necesario mencionar que en pocas instituciones se ha incluido un curso sobre MoProSoft, a pesar de que éste se ha convertido en el modelo de referencia para la industria y el gobierno en México.

Para intentar reducir el tiempo de adaptación del estudiante en la industria, en el nivel innovador se incluyeron cursos como la Industria de Software en el Contexto Nacional o Fábricas de Software. Para ofrecer una visión prismática para analizar los dominios de aplicación del mundo real en diferentes niveles, los cursos que cubren abstracciones como el concepto de los objetos, patrones y componentes se mantienen en los planes de estudio, incluyendo al Desarrollo de Software orientado a Objetos, Desarrollo de Software orientado a Componentes y la Ingeniería de Software basada en Patrones. Para elevar la conciencia sobre la calidad, el plan de estudios estándar incluyó cursos como Pruebas de Software y Verificación y Validación de Software, Mejora Continua, Métricas de Software, y Evaluación del Proceso de Software. Finalmente, cursos como Modelado de Procesos de Negocio, COCOMO II, SixSigma, y Gestión de Riesgos se incluyeron como apoyo a

los modelos de referencia (tanto nacionales como internacionales) en la categoría de gestión del proceso en el nivel de transición.

Es importante decir que la gran mayoría de estos cambios fue establecida en el 2010 y algunas universidades aún no tienen resultados de su aplicación; a pesar de ello, los "viejos" planes de estudio están llegando a su fin y los actualizados han comenzado a utilizarse recientemente, los resultados preliminares son prometedores.

Tabla 2. Ejemplo de un plan modular de estudios para la educación de la Ingeniería de Software

El arte de la práctica	La gestión del proceso	Métodos de apoyo
Nivel preparatorio		
• Introducción a la Ingeniería de Software • Desarrollo de Software orientado a Objetos • Desarrollo de Software orientado a Componentes • Diseño de Software basado en Patrones • Pruebas de Software • Verificación y Validación de Software	• Calidad de Software • Integración de Software • Modelos de Referencia de Software (Modelos de Proceso)	• Desarrollo de Requisitos de Software • Modelado de Procesos de Negocio
Nivel de transición		
• Proyectos de Ingeniería de Software • Arquitectura de Software • Desarrollo de Software Seguro • Modelado y Análisis de Software	• PSP • TSP • CMMI • MoProSoft	• Especificación Formal para el Diseño de Software • COCOMO II • SixSigma
Nivel innovador		
• Tópicos Avanzados de la Ingeniería de Software • Desarrollo Ágil de Software • Ingeniería de Software para Sistemas Empotrados • La Industria de Software en el Contexto Nacional	• Gestión de Proyectos de Software • Mejora Continua • Evaluación del Proceso de Software • Métricas de Software • Fábricas de Software	• Comercialización de Software • Gestión de Riesgos para Proyectos de Software

2.2.10. Evaluación del esfuerzo para modernizar la educación de la Ingeniería de Software

Con el fin de evaluar los resultados de los esfuerzos de mejora mencionados anteriormente, en el 2012 se ha realizado un nuevo análisis sobre las mismas 210 universidades [Garcia et al., 2014]. Tal y como lo muestra la Tabla 3, se ha intentado determinar el número de cursos ofrecidos relacionados con la Ingeniería de Software, el número de universidades que los ofrecen y el número de cursos que se imparten como un solo curso. Los datos obtenidos en este análisis provienen del uso de las mismas fuentes de información empleadas para obtener la Tabla 1. La Tabla 3 resume la información comparando los años de 2004 y 2012.

Se puede observar claramente una mejora significativa a través de la medición de los mismos criterios establecidos en el 2004, concretamente:

- El crecimiento del número total de cursos ofrecidos relacionados con la Ingeniería de Software ha aumentado un 204.10% del 2004 al 2012, y existen un total de 596 cursos desde 2012.
- Desde 2004, el número de universidades que ofrecen cursos relacionados con la Ingeniería de Software aumentó un 16%, de 182 en 2004 a 210 en 2012; y ha aumentado en un 30% el número de inscripciones en cada curso relacionado con la Ingeniería de Software, 17,880 estudiantes tomaron estos cursos en el año académico 2012.
- El número de universidades que ofrecen programas modulares para los cursos relacionados con la Ingeniería de Software ha aumentado de 8 en el 2004 a 105 en el 2012, lo que representa un aumento significativo. De manera similar, el número de universidades que ofrecen la educación a través de un único curso se redujo de 60 a 27.

Para poner en perspectiva el impacto de este cambio, se han analizado a profundidad algunas otras acciones que las universidades han incorporado en este esfuerzo de mejora. Por ejemplo, el 12% de las universidades de la muestra han abierto un programa de Ingeniería de Software para la educación de pregrado. Los planes de estudio para estos programas se centran en la resolución de cuatro de las disfunciones detectadas en este estudio, y las universidades en cuestión están trabajando para incorporar estrategias y métodos para reducir el abismo entre los cursos y la industria de software. En este contexto, algunas universidades han empezado a crear fábricas de software como soporte a la educación de la Ingeniería de Software (por ejemplo, [Garcia et al., 2010; Peredo-Valderrama, 2011]) promoviendo la idea de mejorar las capacidades de los estudiantes a través de una combinación de teoría y práctica y enfocándolas en las necesidades de la industria. Este concepto consiste en un esquema que se basa en una plantilla como una instancia de una fábrica de software, para proporcionar una descripción educativa sobre cómo implementar los productos que son producidos en una fábrica real con clientes y demandas reales. La primera fase de este esfuerzo se enfocó en la construcción de parques tecnológicos como parte de las universidades, donde los estudiantes pueden interactuar en grupos y resolver problemas reales para o con las empresas locales.

Otra cuestión importante, es que las universidades han comenzado a reconocer la necesidad de introducir estrategias pedagógicas alternativas que permitan la mejora de las habilidades que los estudiantes adquieren. En este sentido, con el fin de mejorar la educación de la Ingeniería de Software, se está presentando una tendencia general para destacar la experiencia "*hands on*" en los estudiantes, ya sea relacionada a la industria o en un entorno simulado. Así, se determinó que la estrategia pedagógica más implementada es el enfoque de PBL, que es diferente de las técnicas convencionales de aprendizaje porque el estudiante es introducido en el problema, y al trabajar en él crea una solución. Algunos ejemplos de este cambio para mejorar la forma de enseñar la Ingeniería de Software en México son [Noguez & Espinosa, 2004; Polanco et al., 2004; Maxinez et al., 2011; García & Pacheco, 2012].

Además, algunas universidades reportan esfuerzos adicionales centrados en otras estrategias pedagógicas como el Aprendizaje Activo [Ramirez-Hernandez & Leon-Rovira, 2005], el Diseño Social [Cárdenas, 2009], y el Aprendizaje Colaborativo [Alanis-Funes et al., 2011].

Tabla 3. Cursos relacionados con la Ingeniería de Software ofertados en el 2012

	2004		2012	
Nombre del curso	**Universidades**	**Cursos**	**Universidades**	**Cursos**
Ingeniería de Software	74	82	155	260
Arquitectura de Software	6	6	16	18
Gestión de Proyectos de Software	87	87	124	127
Pruebas de Software	1	1	12	12
Integración de Software	0	0	1	1
Desarrollo de Software orientado a Componentes	0	0	1	1
Desarrollo de Software orientado a Objetos	0	0	1	1
Diseño de Software basado en Patrones	0	0	8	8
Modelado y Análisis de Software	67	67	27	27
Gestión de Empresas de Software	0	0	1	1
Métricas del Software	1	1	5	5
Desarrollo de Proyectos de Software	67	67	5	5
Desarrollo de Software Seguro	0	0	1	1
Verificación y Validación de Software	0	0	3	3
Tópicos Avanzados de la Ingeniería de Software	0	0	5	5
Modelado de Procesos de Negocio	0	0	2	2
Modelos de Proceso	0	0	8	8
Desarrollo Ágil de Software	0	0	3	3
Evaluación de Proyectos de Software	10	10	22	22
Calidad del Software	36	36	64	64
Evaluación del Proceso de Software	0	0	1	1
Taller de Productos de Software	0	0	2	2
La Industria de Software en el Contexto Nacional	0	0	3	3
Comercialización de Software	0	0	1	1
Desarrollo Profesional en Ingeniería de Software	0	0	1	1
Normatividad del Software	0	0	2	2
Especificación Formal para el Diseño de Software	1	1	3	3
Mejora Continua	0	0	1	1
Desarrollo de Requisitos de Software	1	1	4	4
PSP	0	0	2	3
TSP	0	0	1	1
SixSigma	0	0	1	1
Gestión de Riesgos en Proyectos de Software	0	0	1	1
Fábricas de Software	0	0	1	1

MoProSoft	0	0	1	1
CMMI	0	0	1	2
Educación basada en un solo curso	**60**		**27**	
Educación basada en módulos	**8**		**105**	

Por último, las universidades mexicanas han comenzado a trabajar con la industria para mejorar la calidad de los productos de software a través de la educación y la certificación. En el 2006, por ejemplo, el SEI (Pittsburgh, USA), el *Next Process Institute* (Kawasaki, Japón) y el Tecnológico de Monterrey (Monterrey, México) firmaron el acuerdo de "Socios estratégicos del SEI" con el objetivo de colocar internacionalmente a la industria mexicana de software a través de la incorporación de los modelos PSP y TSP [Nichols & Salazar, 2009]. Esta iniciativa permitió que México recibiera el conocimiento del SEI sobre los modelos, así como la autorización a un grupo de expertos e investigadores para certificar académicos y empresas mexicanas de software.

En este sentido, actualmente México se ha posicionado como el país número uno a nivel mundial en relación con el número de personas certificadas en el modelo PSP, con el 61% de todas las personas certificadas en este modelo alrededor del mundo. Sin embargo, este logro no es suficiente para consolidar la industria mexicana; es necesario asegurar que los estudiantes aprendan estas habilidades de sus profesores certificados y que las empresas certificadas usan sus habilidades recién adquiridas de manera exitosa en los proyectos de software que desarrollan diariamente, evitando de esta manera la desconexión entre las universidades y las necesidades de la industria [Chen & Li, 2011; Llorens et al., 2013]. Ahora bien, en el contexto de esta modernización sobre la educación de la Ingeniería de Software, la presente tesis se enfoca a presentar un soporte educativo sobre el área de SPI que es cubierta por los cursos que han sido sombreados en la Tabla 3.

2.3. La Mejora del Proceso de Software en el contexto educativo

De acuerdo con [Zahran, 1998], la Mejora del Proceso Software *"es el arte y la ciencia de cambiar el proceso de software de una organización para construir un software mejor"*. En este sentido, la literatura relacionada con esta área de la Ingeniería de Software indica que los estudiantes deben primero aprender a diagnosticar y solucionar problemas para proyectos individuales a través de herramientas, técnicas, y prácticas específicas. Sin embargo, la Mejora del Proceso de Software siempre implica analizar el panorama completo, a nivel de la organización. Es decir, ésta implica no solamente diagnosticar problemas específicos y atacarlos individualmente, sino también analizar proyectos completos e identificar las áreas que pueden ser mejoradas.

La mejora puede ser evolutiva, lo que permite utilizar ciclos iterativos para establecer y probar partes pequeñas de un ciclo de vida dentro de una organización, e implantarlas de forma general cuando se obtengan los resultados deseados. Sin embargo, es importante diferenciar esta mejora del enfoque de diagnóstico y reparación, dado que este último no requiere un cambio en la cultura de trabajo de la organización para mejorar continuamente. Es decir, para que la Mejora del Proceso de Software proporcione resultados eficaces se requiere del apoyo consistente (tanto en acciones como en recursos) de la alta dirección y el consenso entre los ingenieros de software.

Es evidente que en la actualidad existen herramientas computacionales específicas para brindar soporte a las organizaciones. Existen también modelos y certificaciones que ayudan a las organizaciones a evaluar el estado actual de sus procesos y que sirven como marco para mejorarlos. También existen procesos y metodologías que se pueden adoptar y que describen un conjunto

completo de actividades, roles y productos necesarios para desarrollar software. Mediante la aplicación de todas estas herramientas, los ingenieros de software pueden ayudar a su organización a establecer una capacidad para solucionar los problemas antes de que sea imposible hacerlo y que conduzcan al fracaso de los proyectos.

Las siguientes secciones pretenden hacer un análisis sobre estos modelos, describiendo en primera instancia un modelo genérico de mejora y posteriormente dos de los modelos de proceso más explorados en los programas mexicanos de estudio.

2.3.1. El modelo de mejora IDEAL

Con el objetivo de ayudar a las empresas de software a implantar un programa de mejora de sus procesos, y ante la demanda de software que se presentaba en esos años, el SEI desarrolló en 1996 el modelo IDEAL [McFeeley, 1996]. Este modelo está basado en las experiencias tanto gubernamentales, como de clientes e industria relacionados con el SEI. A pesar de que este modelo está enfocado a grandes organizaciones y de estar influenciado por la familia de modelos CMM, existen diversas publicaciones que demuestran que puede ser aplicado en empresas pequeñas. Dado que dicho modelo es demasiado extenso, a continuación se proporciona una síntensis de aquellos elementos que son útiles para el desarrollo de esta tesis.

2.3.1.1. Propósito del modelo

El propósito del modelo es servir como guía de consulta para aquellas organizaciones que se inician en la Mejora del Proceso de Software, así como para aquellas que ya tienen establecido un plan de mejora continua. IDEAL proporciona cinco fases que establecen un ciclo continuo para realizar una iniciativa de mejora; el número de ciclos que se requieren para alcanzar la mejora varía de organización a organización dependiendo la disponibilidad de infraestructura, recursos humanos, etc. Estos factores desempeñan un papel importante puesto que representan el factor principal para medir el éxito de un programa de mejora.

2.3.1.2. Descripción del modelo

Tal y como se mencionó anteriormente, el modelo IDEAL establece cinco fases (véase Figura 2.4) para proporcionar una descripción genérica y una secuencia recomendada para alcanzar la mejora continua: Inicio (*Inititating*), Diagnóstico (*Diagnosing*), Establecimiento (*Establishing*), Ejecución (*Acting*) y Adopción (*Leveraging*). A continuación se proporciona una descripción detallada de cada fase.

2.3.1.2.1. ***Fase de inicio***

En esta fase la alta dirección de la empresa de software debe comprender la necesidad de incorporar un programa de mejora a través de un análisis sobre costos y beneficios. En este sentido, es necesario definir roles y responsabilidades como infraestructura inicial, además de gestionar el compromiso de la alta dirección y los mandos intermedios (o gerenciales) para comunicar los objetivos y metas de la organización, así como para garantizar la disponibilidad de recursos y la priorización del proyecto de mejora. Así, es necesario especificar los objetivos del programa de mejora y el personal involucrado debe comprometerse a estar disponible para que el plan de mejora sea ejecutado. Por consiguiente, se define un Grupo Directivo de Gestión (MSG, *Management Steering Group*) y un Grupo de Procesos de Ingeniería de Software (SEPG, *Software Engineering Process Group*), que deben formarse con personas con ciertas habilidades y conocimientos que les permitirán asumir la responsabilidad en el desarrollo de los procedimientos, planes y calendarios que conduzcan a la organización a través del proceso de mejora.

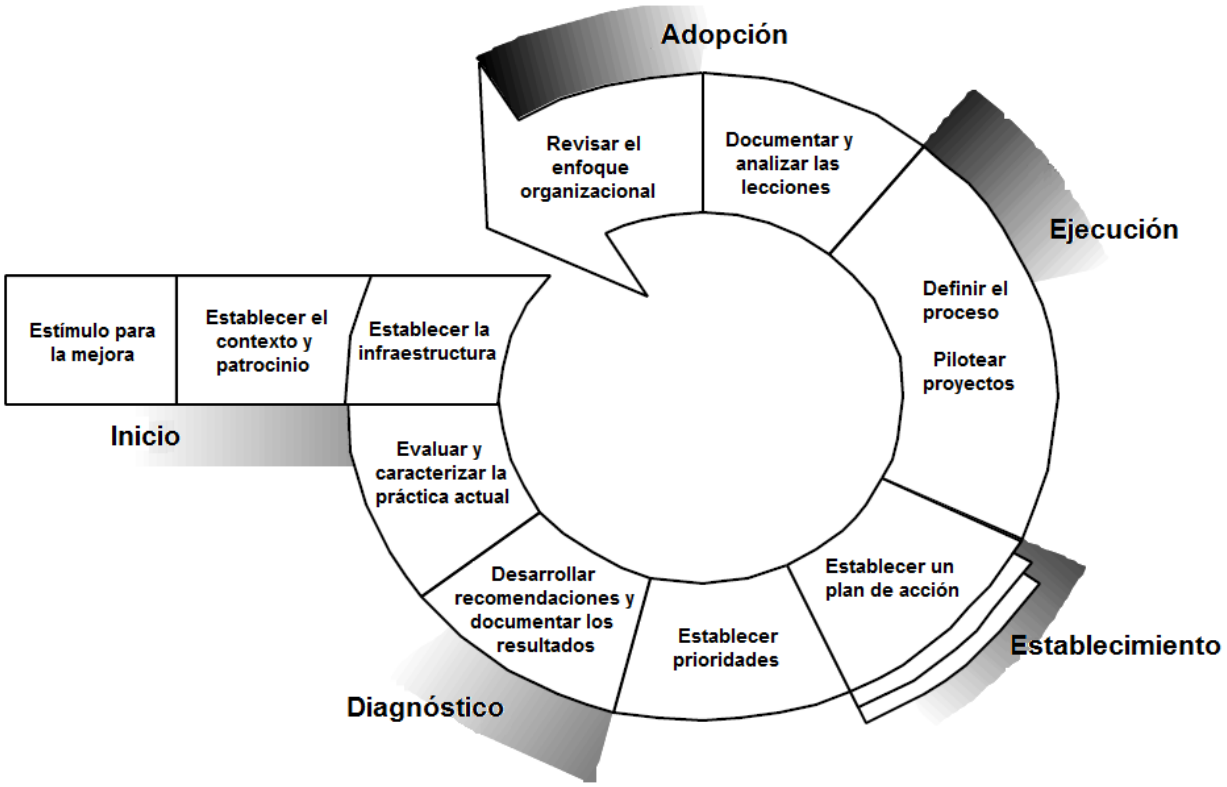


Figura 2.4. Las fases del modelo IDEAL [McFeeley, 1996]

Al final de esta fase inicial y una vez que se han establecido correctamente los criterios mencionados, el SEPG habrá alineado la iniciativa de mejora con cada uno de los objetivos de la empresa, y habrá creado un plan de comunicación que deberá seguirse en las fases posteriores del modelo. De manera más específica, en esta fase se identifican diez actividades que deberán ser completadas de la siguiente forma:

- *Comenzar*: El propósito de esta actividad es organizar a un equipo de exploración para que establezca los propósitos para gestionar la fase inicial del programa de mejora. La información a recoger comprende a las necesidades actuales, políticas organizacionales, regulaciones que puedan afectar al programa, actividades similares de mejora que ya existan dentro de la organización o que estén planeadas a futuro, etc. El objetivo es básicamente identificar a los departamentos que participarán como *stakeholders* evaluando y seleccionando un enfoque que conduzca el programa de mejora. El equipo de exploración seleccionará a un líder que cumpla con las habilidades específicas y que posea conocimientos en el área de planificación, y además será necesario seleccionar al personal representativo de los grupos de *stakeholders* que se involucren en los planes de mejora. Por último, se debe establecer un plan de consulta y entrenamiento que brinde soporte a las propuestas seleccionadas.
- *Identificar las necesidades del negocio y conductores para la mejora*: El propósito de esta actividad es entender, desde una perspectiva gerencial, las necesidades clave del negocio para vincularlas con los requerimientos del programa de mejora. Entre las tareas que se desarrollan en esta actividad se encuentran la revisión de la visión organizacional y las necesidades de mejora en el contexto gerencial, la recogida de documentos que identifiquen las necesidades actuales, la realización de entrevistas con los *stakeholders* de nivel gerencial,

la revisión de las necesidades que pueden ser, completamente o parcialmente, satisfechas a través de la mejora y la definición de cómo el programa de mejora satisfará las necesidades gerenciales.

- *Desarrollar una propuesta de mejora*: El propósito de esta tarea es establecer una propuesta que explique a la alta dirección la importancia sobre la mejora del proceso (por qué debe iniciarse un programa de mejora, cuánto costará realizarlo, cuánto tiempo se llevará hasta visualizar los primeros resultados, etc.), y proporcione una aproximación a los resultados que se obtendrán. Esta tarea se considera un punto decisivo para decidir si el programa continúa o no. El borrador de la propuesta deberá enviarse a los *stakeholders* para su revisión y obtención de comentarios. Adicionalmente, se establecen objetivos y metas para el programa de mejora, asegurando la consistencia con los objetivos de la organización y necesidades críticas previamente identificadas. Se determina también el alcance identificando para los departamentos de la empresa que serán incluidos mediante el establecimiento de roles y responsabilidades a los participantes del programa: alta dirección, grupo de soporte, SEPG, MSG y otras entidades. Por último, se desarrolla un plan de alto nivel que resume las actividades y calendarios a través de la Fase de Establecimiento, y que constituye los requerimientos básicos en cuanto a personal, necesidades de entrenamiento (si es que la hay), equipo, etc.

- *Educar y brindar soporte*: Los propósitos de esta actividad son crear conciencia, establecer expectativas, y dar soporte al programa de mejora dentro de la empresa. El objetivo es informar las necesidades de negocio en relación a la mejora e involucrar a los *stakeholders* claves que formarán parte del programa. Para esto, es necesario establecer sesiones informativas que puedan ser adaptadas a diferentes unidades de negocio de la empresa para explicar el esfuerzo que está por iniciar. Estas sesiones informativas deben orientarse desde la alta dirección y su personal, hasta los líderes de equipo de software, profesionales de software y otras áreas interesadas. En este sentido, es recomendable que durante el programa se establezca directamente el diálogo con el personal de cada área.

- *Obtener la aprobación para la propuesta de mejora y los recursos iniciales*: El propósito es que la propuesta de mejora sea presentada ante la alta dirección para obtener su aprobación y lograr la asignación de tiempo y recursos requeridos para lanzar el programa. Es posible que existan iteraciones entre la actividad *Comenzar* y la aprobación, hasta que se alcance el acuerdo sobre la asignación de recursos para continuar con la iniciativa de mejora o bien se decida abandonarla por la falta de éste. Una vez que se obtiene la aprobación, los recursos iniciales son asignados para iniciar el trabajo y se establece una estrategia de lanzamiento que identifica a las responsabilidades sobre los recursos, un presupuesto para cubrir las necesidades, la distribución de recursos (incluyendo el tiempo requerido de la alta dirección para participar en las actividades), y la actualización del plan de comunicación para la organización.

- *Establecer la infraestructura para la mejora*: Para gestionar el programa de mejora es necesario que se defina una infraestructura que incluya tanto deberes como responsabilidades para asegurar el éxito del programa de mejora. Esta infraestructura puede proveer recursos cuando sea necesario, mantener la visibilidad del programa de mejora, y obtener y mantener las lecciones aprendidas. La infraestructura se proporciona en términos de personal específico, entidades organizativas, documentos, responsabilidades y lugares. En resumen, esta actividad asigna las responsabilidades necesarias que aseguren que la información referente a los avances del programa de mejora se mantenga visible y sea compartida a través

de seis sub-actividades: Establecer al MSG, establecer al SEPG, mantener la visibilidad del programa de mejora, facilitar y fomentar el intercambio de información, conservar las lecciones aprendidas y las mejoras desarrolladas, y proporcionar una red de soporte.

- *Evaluar el clima para la mejora*: El propósito de evaluar el clima es identificar las barreras de la empresa que tendrán impacto en el programa de mejora y desarrollar planes eficaces que garanticen que las mejoras realizadas perduren. Las tareas identificadas en esta actividad hacen uso de diagnósticos para evaluar: (1) el historial de las barreras identificadas en planes similares de mejora, (2) la cultura de la empresa, (3) el patrocinio para la mejora y determinar qué se necesita para mejorarlo y (4) la resistencia actual con el programa de mejora para identificar las barreras relacionadas con el rechazo al cambio.
- *Definir las metas generales de la mejora*: La Mejora del Proceso de Software es una inversión a largo plazo. Se requiere de metas mensurables, claramente definidas, para orientar y ayudar en el desarrollo de tácticas de mejora. Estas metas también permiten la medición objetiva de los resultados de mejora. La creación de buenas metas requiere la comunicación bidireccional entre diferentes grupos de gestión y entre la dirección y los especialistas. Así, es necesario definir qué mediciones serán necesarias para determinar la satisfacción del objetivo, de esta manera el programa de mejora puede ser una estrategia para determinar una visión más clara y vincularla con el plan de negocios.
- *Definir los principios guía del programa de mejora*: El propósito de esta actividad es usar el programa de mejora como un modelo y mecanismo para experimentar con diferentes procesos y comportamientos deseados. Un principio rector típico es utilizar el programa de mejora para experimentar con procesos revisados de gestión, tal como nuevas formas de planificación, monitorización y control, etc. Los nuevos métodos pueden "fracasar" en una tarea del programa de mejora con efectos mucho menos dramáticos que en la organización en sí. El fracaso en este sentido significa que el nuevo proceso no funciona tan eficientemente como se había previsto inicialmente –un defecto común en el pilotaje inicial sobre un proceso nuevo o revisado. Cualquiera de estos principios rectores deben ser documentados para el resto de las personas que deseen utilizarlos como guía en el plan de acción estratégico de mejora.
- *Lanzar el programa*: El propósito de esta actividad es iniciar la parte principal del programa de mejora (el Diagnóstico) y comenzar con el ciclo continuo del programa de mejora de procesos. Para ello se requiere revisar los planes e iniciativas documentadas en los pasos anteriores.

2.3.1.2.2. ***Fase de diagnóstico***

En esta fase el MSG debe de haber comprendido el proceso de software actual de la empresa de tal forma que pueda desarrollar un plan que cubra los cambios especificados en las metas de mejora. Las actividades realizadas en la fase de Diagnóstico proporcionan esta información a través de la planificación de la mejora y el proceso de priorización. El conocimiento sobre las fortalezas y oportunidades de mejora en la empresa es un prerrequisito esencial para identificar y priorizar un programa efectivo de mejora. La salida principal de esta fase resume las conclusiones finales y un reporte de recomendaciones que es producido como resultado del diagnóstico. La información sobre el estado actual de la empresa es usada por los Grupos de Trabajo Técnico (TWG, *Technical Working Group*) durante la fase de Establecimiento para desarrollar soluciones de mejora.

El propósito de esta fase es obtener una imagen de las fortalezas y debilidades actuales de la empresa, es decir obtener una línea base. Estas líneas base proporcionan información sobre cómo y

qué tan bien realiza sus actividades de software en la actualidad. Esta información es útil para iniciar el desarrollo de un plan de acción que proporcione guía y dirección al programa de mejora en los años siguientes. Las actividades de la empresa deben ser auto-verificadas. La credibilidad de éstas depende de la habilidad percibida para extraer la información importante y real de la empresa y presentársela en una forma coherente y viable. Esta fase debe realizarse a través de las seis actividades siguientes:

- *Determinar las líneas base necesarias*: La organización tiene razones importantes para emprender un programa de mejora. El propósito de decidir cuántas líneas base se deben considerar y de qué tipo es para asegurar que el enfoque del programa de mejora esté ligado a las necesidades de negocio de la empresa. La determinación sobre las líneas base dependerá mucho de la misma.
- *Planificar las líneas base*: Para lograr las actividades de la línea base se requiere la coordinación de personal, datos, instalaciones, entrenamiento y servicios de soporte. Esta actividad puede repetir algo del trabajo realizado en la fase de Inicio. En otras ocasiones no es necesaria esta repetición, pero a menudo, el MSG se forma con diferentes miembros a los que crearon el programa de mejora, y tendrán que cubrir algunos de los mismos temas para desarrollar su propio entendimiento y estrategia. Cuando esta actividad es introducida como resultado de un ciclo posterior a través del modelo IDEAL, estos temas son revisados muy poco.
- *Realizar las líneas base*: El propósito de conducir las líneas base es reunir información factible requerida para dar soporte al esfuerzo de mejora. La información recopilada presentará una vista instantánea de las fortalezas y debilidades de la empresa relativas a su proceso de software y a sus prácticas de gestión del desarrollo de software.
- *Presentar los hallazgos*: El equipo de evaluación, al final de la recogida de datos, presenta a los participantes lo que ha encontrado. En esta reunión se describen el método usado, los participantes, las áreas de investigación y las fortalezas y debilidades que se hayan encontrado.
- *Desarrollar las conclusiones finales y el reporte de recomendaciones*: Las conclusiones finales y el reporte de recomendaciones documentan los esfuerzos de la línea base y representa el estado actual de la empresa. El equipo de evaluación desarrollará un conjunto de recomendaciones en base a las conclusiones que fueron descubiertas durante el análisis de la línea base. Regularmente estas actividades permiten identificar inconsistencias y generar recomendaciones basadas en un consenso más amplio. Estos resultados son incorporados en el plan de acción como una estrategia para mejorar las actividades actuales.
- *Comunicar las conclusiones y recomendaciones a la* empresa: El personal se preguntará sobre todas las actividades que ocurrieron durante el diagnóstico. Para aliviar cualquier temor que pueda haber surgido, o que pudiera surgir, se recomienda que los resultados de las actividades del diagnóstico sean comunicados a toda la empresa. La organización puede lograr esto mediante la realización de una serie de reuniones, de tal manera que todos los miembros de la empresa escuchen el mismo mensaje. Esto contribuirá a la construcción del patrocinio y del apoyo al programa de mejora.

2.3.1.2.3. ***Fase de establecimiento***

La creación del plan estratégico de acción para la Mejora del Proceso de Software es una de las actividades más críticas en la iniciativa de mejora –y una de las más frecuentemente rechazadas.

En esta fase es donde el equipo de gestión desarrolla o actualiza el plan de acción basándose en la visión, plan de negocios y esfuerzos anteriores de mejora de la empresa, junto con los hallazgos obtenidos en la fase de Diagnóstico.

Esta fase se repetirá cuantas veces sea necesario; usualmente se inicia por la falta de un plan de acción para una empresa que se encuentra en el primer ciclo del modelo IDEAL. Para aquellas organizaciones que ya estén en un ciclo posterior, esta fase puede iniciarse por la necesidad de actualizar planes, objetivos o directrices previas. El MSG tiene la responsabilidad de crear el plan estratégico de acción. En cierto sentido, se trata de la creación de una línea base de gestión, similar a la línea base orientada a los procesos o técnicas evaluadas en la fase de Diagnóstico. Esta fase identifica catorce actividades que deben realizar de la siguiente manera:

- *Seleccionar y obtener entrenamiento sobre el proceso de planificación estratégica*: El propósito de esta actividad es escoger un enfoque consistente para planificar el programa de mejora mediante la revisión de métodos actuales para la planificación estratégica y de las necesidades de planificación para el programa de mejora.
- *Revisar la visión de la empresa*: El propósito de esta actividad es vincular claramente la estrategia de mejora con la visión y dirección de la empresa, de modo que la orientación al programa de mejora sea consistente con la orientación de otras actividades dentro de la misma. Esta actividad puede repetir algo del trabajo realizado en la fase de Inicio. En otras ocasiones no es necesaria esta repetición, pero a menudo, el MSG se forma con diferentes miembros a los que crearon el programa de mejora, y tendrán que cubrir algunos de los mismos temas para desarrollar su propio entendimiento y estrategia. Cuando esta actividad es introducida como resultado de un ciclo posterior a través del modelo IDEAL, estos temas son revisados muy poco.
- *Revisar el plan de negocios de la empresa*: El propósito de esta actividad es vincular claramente la estrategia de mejora con el plan de negocio de la empresa, de modo que la orientación al programa de mejora puede ser compatible con la orientación a otras actividades dentro de la misma. Si bien no todas las actividades de la mejora de procesos pueden ser fácilmente vinculadas a un plan o metas de negocio, eso no significa que no sean necesarias. Algunas cosas deben hacerse porque hacen que el negocio funcione mejor, pero éstas tal vez no contribuyan directamente a la línea de fondo. Esta actividad puede repetir algo del trabajo realizado en la fase de Inicio. En otras ocasiones no es necesaria esta repetición, pero a menudo, el MSG se forma con diferentes miembros a los que crearon el programa de mejora, y tendrán que cubrir algunos de los mismos temas para desarrollar su propio entendimiento y estrategia. Cuando esta actividad es introducida como resultado de un ciclo posterior a través del modelo IDEAL, estos temas también son revisados muy poco.
- *Determinar problemas claves del negocio*: Las necesidades clave del negocio deben ser claramente definidas, medibles, y entendidas para ofrecer una visión común a los equipos de mejora. Como se describió anteriormente, no todas las actividades de la mejora de procesos pueden vincularse fácilmente a los problemas claves del negocio; sin embargo, los problemas detectados deben usarse para priorizar los proyectos de mejora. Esta actividad puede repetir algo del trabajo realizado en la fase de Inicio. En otras ocasiones no es necesaria esta repetición, pero a menudo, el MSG se forma con diferentes miembros a los que crearon el programa de mejora, y tendrán que cubrir algunos de los mismos temas para desarrollar su propio entendimiento y estrategia. Cuando esta actividad es introducida como resultado de un ciclo posterior a través del modelo IDEAL, estos temas son revisados muy poco.

- *Revisar esfuerzos anteriores de mejora*: La gente suele repetir comportamientos pasados, incluyendo aquellos que conducen al éxito y aquellos que no. La empresa debe asegurarse que no se repitan aquellos errores que pudieron haber causado el fracaso de iniciativas similares en el pasado. La información recopilada en la fase de Diagnóstico es revisada y analizada, identificando cambios anteriores o proyectos de mejora y evaluando qué tan buenos o malos fueron y por qué.
- *Describir las motivaciones para mejorar*: La gente debe entender por qué las empresas invierten tanto esfuerzo y tiempo en un programa de mejora, dado que a medida que su entendimiento crece también lo hará su apoyo. Por lo general, las motivaciones exitosas eliminan el "dolor" de la situación actual, a diferencia de promesas sobre un estado deseado. Estas motivaciones deben documentarse en el plan de acción estratégico. Además, el plan de comunicación debe ser actualizado para asegurar que la motivación por mejorar es extendida a toda la empresa.
- *Identificar esfuerzos de mejora (planificados) actuales y anteriores*: Es recomendable identificar otras iniciativas que la empresa puede tener en marcha. Esto se realiza durante la revisión de la visión, plan de negocios, y esfuerzos anteriores de mejora de las organizaciones. Por lo general, la mayoría de las empresas tienen muchos diferentes esfuerzos de mejora en curso. A menudo, estas iniciativas son descoordinadas y compiten entre sí por los escasos recursos. Si una empresa desea maximizar la eficacia de su inversión en la Mejora del Proceso de Software, debe evaluar todas las iniciativas en curso y determinar cuánto se está invirtiendo en cada una y en total.
- *Finalizar los roles y responsabilidades de las varias entidades de infraestructura*: Es posible que los planes iniciales, junto con la definición de roles y responsabilidades de la infraestructura, estén desactualizados. Aquellas empresas que realizan el primer ciclo de IDEAL deberían tener ahora mucho más conocimiento sobre lo que es la mejora y lo que se necesita para lograr el éxito. Sería beneficioso que estas empresas revisen los roles y responsabilidades que definieron inicialmente para su infraestructura y hagan los ajustes necesarios. Aquellas empresas que están entrando de nuevo a la fase de Establecimiento desde un ciclo anterior, deben contar con lecciones aprendidas que pueden aplicar a los roles y responsabilidades definidos para la infraestructura.
- *Priorizar las actividades y desarrollar la agenda de mejora*: Las líneas base, en particular la que se refiere a la madurez del proceso, por lo general identifican problemas y proporcionan recomendaciones basadas en un consenso mucho más amplio que quizás no estaba disponible antes. Esta información sirve para proporcionar cierta guía y, a menudo, ayuda en la priorización de las acciones.
- *Reconciliar los esfuerzos existentes/planificados de mejora con los hallazgos y recomendaciones del diagnóstico*: Los resultados de las líneas base deben ser incorporados en el plan de acción estratégico de mejora y reconciliados con todos los esfuerzos de mejora existentes y/o previstos. Esto dará lugar a una estrategia única que tratará todas las acciones de mejora y a los esfuerzos de mejora relacionados que afectan a los mismos grupos de personas.
- *Transformar las metas generales de Mejora del Proceso de Software a metas específicas mensurables*: Ahora que los resultados de las actividades iniciales se han reconciliado, existen suficientes datos para tomar los objetivos generales a largo y corto plazo, desarrollados en la fase de Inicio, y hacerlos más específicos. Esto se realiza mediante la

incorporación de la medición de la situación actual de estos objetivos y la definición de una mejora agresiva pero alcanzable sobre esas medidas.

- *Crear/actualizar el plan estratégico de mejora*: Ahora que todas las secciones del plan de acción estratégico de mejora están listas, el plan se ha reconciliado con los resultados del diagnóstico, y los objetivos se han transformado, el plan debe ser elaborado, editado y terminado.

- *Desarrollar el consenso, revisar y aprobar el plan estratégico de mejora y comprometer los recursos para su ejecución*: El plan de acción será inútil si es construido en el vacío y solamente un pequeño grupo cree en él. Para que sea útil, tiene que ser "vendido" y aceptado mediante el consenso. El plan de acción que se desarrolla debe ser comunicado a la empresa. Al inicio se establecieron los componentes y el personal de la organización para que este plan funcione, por lo tanto es justo que se comunice qué hay en él y lo que se espera de las personas involucradas.

- *Formar al TWG*: El equipo debe estar compuesto por voluntarios de la misma empresa que serán afectados directamente por el proceso de mejora y que están interesados en trabajar sobre la mejora. Este grupo puede ser identificado desde las fases tempranas del ciclo.

2.3.1.2.4. ***Fase de ejecución***

En esta fase las mejoras son desarrolladas, puestas en práctica, y desplegadas en la empresa. Muchas de las mejoras que los grupos de trabajo han desarrollado están completas y su valor será "probado" en la empresa a través del "pilotaje". El MSG y el SEPG gestionarán y darán soporte al desarrollo, "pilotaje" y despliegue de las mejoras. En este sentido, la fase de implementación vincula la misión del programa para mejorar los procesos y la misión de la organización para generar productos. Es importante estar atento a los efectos producidos por los cambios introducidos en cada área, puesto que este es el momento en que se le pide a la empresa que cambie ciertas cosas que hacía previamente. Para esto, el TWG desarrolla mejoras específicas para procesos específicos a través de dos enfoques básicos: (1) el TWG se enfoca en problemas específicos y desarrolla una solución usando proyectos piloto para validar y refinar la solución, o (2) el TWG se enfoca en un proceso en particular y desarrolla refinamientos incrementales con proyectos piloto para probarlos. Esta fase se compone de las siguientes diez actividades:

- *Completar el plan táctico para el TWG*: El propósito de esta actividad es completar un plan táctico a partir de una plantilla que el MSG proporciona al TWG. El plan completado deberá ser aprobado por el MSG. Los esfuerzos iniciales del equipo deben centrarse en reducir el alcance de la mejora específica sobre la que van a trabajar.

- *Desarrollar las soluciones*: En esta actividad se desarrollan soluciones a los problemas encontrados durante la fase de Diagnóstico. El propósito de esta tarea es crear soluciones a los problemas o procesos que la empresa determina que son necesarios para satisfacer las necesidades de negocio. La solución seleccionada debe ser compatible con la cultura de la organización de modo que sea fácilmente aceptada e institucionalizada.

- *Refinar el proceso (enfoque centrado en el proceso)*: El enfoque centrado en el proceso aborda la comprensión de un proceso clave específico, identificado en la fase de Diagnóstico, y aplica mejoras incrementales al mismo. Este enfoque es útil para alcanzar mejoras a largo plazo en el proceso. Sin embargo, debido a las presiones inmediatas e incertidumbres típicas de las organizaciones con niveles bajos de madurez, es difícil mantener este enfoque. El mantener un enfoque centrado en el proceso requiere del

compromiso sólido para la gestión y el impulso y entusiasmo organizacional. A pesar de la necesidad de este compromiso, se recomienda utilizar el enfoque centrado en el problema en programas de mejora que inician por primera vez.

- *Analizar y corregir el problema (enfoque centrado en el problema*): El enfoque centrado en el problema difiere del enfoque centrado en el proceso dado que el primero es más útil para identificar fácilmente problemas y puede proporcionar resultados más rápidamente. Sin embargo, cuando los problemas se vuelven complejos o las soluciones son difíciles de manejar, los resultados del enfoque centrado en el problema a menudo son alcanzados por otros problemas que surgen cuando los primeros apenas se están resolviendo. En este sentido, el enfoque centrado en el proceso es más útil para alcanzar resultados a largo plazo.

- *Pilotear las soluciones potenciales*: Los proyectos piloto se utilizan para probar las soluciones tanto en el enfoque centrado en el proceso como en el centrado en el problema. Las soluciones requieren de adaptación y refinamiento para que encajen en los proyectos de la empresa, y precisamente los proyectos piloto ayudan a determinar las necesidades de adaptación y las directrices para el resto de la organización. En este sentido, es posible realizar varios pilotos para una solución y pueden existir varias iteraciones entre el desarrollo de la solución y los pasos del "pilotaje" para tener lista la solución para su implementación en toda la organización.

- *Escoger a los proveedores de soluciones*: Es posible que existan varias fuentes de soporte para la solución de mejora de procesos, y que algunas compitan o sean complementarias. Los proveedores de soluciones pueden ser internos o externos a la empresa, y podrían ser el mismo TWG o algún subconjunto de éste. Teniendo en cuenta las diversas necesidades de la empresa, el TWG debe determinar la mejor fuente para proporcionar una solución. Durante esta fase, el TWG debe trabajar en estrecha colaboración con el SEPG para utilizar a los proveedores de soluciones establecidas y examinadas. Este paso puede realizarse en paralelo con la creación de soluciones.

- *Determinar necesidades de soporte a largo plazo*: Las soluciones a largo plazo requieren del soporte a largo plazo. A medida que la solución es implementada en otras partes de la empresa, el personal tendrá que ser entrenado, surgirán problemas, y puede requerirse de adaptación adicional. Esta actividad identifica los requerimientos del soporte a largo plazo en términos de conocimientos y habilidades necesarias. La mejora debe ser planificada para que dure a través de los años (posiblemente como parte de un esfuerzo más grande de mejora). El soporte continuo para cualquier herramienta, método, clase, material, etc., debe ser planificado en paralelo con la etapa de desarrollo de soluciones.

- *Desarrollar la estrategia de despliegue y la plantilla del plan*: Una vez que la solución desarrollada se ha puesto a prueba y que las necesidades de soporte a corto y largo plazo se han abordado, la solución está lista para desplegarse en la organización. El TWG debe crear un plan de despliegue que oriente a los proyectos que utilizarán la mejora de procesos. El plan debe incluir: ¿qué entrenamiento es necesario?, ¿qué herramientas y métodos es necesario adquirir?, ¿qué pasos de instalación deben seguirse?, ¿qué información debe utilizarse para obtener soporte?, etc.

- *Empaquetar la mejora y entregarla al SEPG*: Durante el desarrollo de la solución es probable que el TWG desarrollara varios productos y artefactos intermedios. Éstos deben ser recogidos en un paquete que se entrega al SEPG para el mantenimiento y soporte a largo plazo. (Esta tarea es mucho más sencilla si el TWG la realiza a medida que avanza).

- *Disolver el TWG*: Como tarea final, el TWG debe presentar un reporte sobre las lecciones aprendidas para ayudar al SEPG y MSG a mejorar el proceso de gestión durante el desarrollo de soluciones. El otorgar un tipo de recompensa al equipo que realizó el trabajo es regularmente una buena muestra de patrocinio de la mejora y, al mismo tiempo, un elemento que mejora la comunicación con el resto de la empresa.
- *Desplegar la solución*: El propósito de esta actividad es instalar en toda la empresa la solución probada. La solución ha sido desarrollada y probada a través de las pruebas piloto sobre un proyecto. Así, la solución debe ser institucionalizada en toda la empresa. Las tareas de institucionalización podrían ir desde crear nuevas herramientas de software, instalaciones, o información de sistema, hasta documentar todo lo relativo al nuevo proceso.
- *Completar la transición de soporte a largo plazo*: La mejora del proceso no requiere una vigilancia constante; y si lo hace, ésta debe ser reajustada. El equipo de desarrollo debe ser capaz de continuar sin mucha orientación y soporte, pero debe ser capaz de recurrir al asesoramiento cuando sea necesario. Cuando la empresa demuestra que puede ejecutar repetidamente el nuevo proceso, la participación del SEPG asume el rol de apoyo de guardia y el grupo de soporte a largo plazo se hace cargo de la asesoría.

2.3.1.2.5. ***Fase de adopción***

Antes de comenzar con el siguiente ciclo del modelo IDEAL es necesario revisar qué ha pasado durante el ciclo y preparase para el siguiente antes de comenzar otra vez. Esta fase consiste en revisar y analizar las lecciones aprendidas en las fases anteriores, incorporar mejoras en los procesos, revisar la información obtenida, revisar y evaluar las metas, evaluar el patrocinio y generar un plan continuo de mejora que sirva de guía para el programa de mejora. Ciertos aspectos deben ser comunicados, como los resultados del primer ciclo a través del modelo, los objetivos de negocio y las metas alcanzadas hasta ahora, los cambios que pueden ocurrir en la infraestructura, etc. Las siguientes actividades enmarcan la realización de esta fase:

- *Reunir las lecciones aprendidas*: El propósito de esta actividad es asegurar que toda la información sobre las lecciones aprendidas esté disponible para su revisión antes de comenzar el siguiente ciclo de IDEAL. Sin esta información será difícil recordar las actividades que se realizaron durante las fases anteriores. Con suerte, el equipo ha reunido estas lecciones aprendidas a lo largo de todas las actividades previas y residen en la base de datos de procesos de la empresa; si no fuera así, las lecciones deben reunirse de donde sea que puedan estar.
- *Analizar las lecciones aprendidas*: El propósito de esta actividad es asegurar que el proceso que se está usando para conducir la mejora es el mejor que se puede utilizar. Ahora que se cuenta con toda la información del ciclo previo es necesario reflexionar sobre los procesos que se siguieron o no se siguieron. En este sentido, es necesario aprender de los errores u omisiones que se pudieron realizar en el ciclo anterior y modificar y cambiar el enfoque para no repetirlos.
- *Revisar el enfoque organizacional*: El propósito de esta actividad es hacer que el siguiente ciclo sea más efectivo y eficiente. Cualquier mejora que se haga al proceso permitirá realizar cambios con mayor eficacia, reducirá la resistencia al cambio y permitirá que la mejora del proceso proceda a un ritmo más rápido.
- *Revisar el patrocinio y compromiso*: Se ha podido observar que durante el ciclo anterior, el patrocinio y el compromiso son fundamentales para el éxito del programa de mejora. Al

igual que como se realizó la primera vez a través de la fase de Inicio, es necesario asegurarse de que se tiene tanto el patrocinio y compromiso adecuados para apoyar al programa en el nuevo ciclo.

- *Establecer metas de alto nivel*: Al igual que en la fase de Inicio, es necesario establecer metas generales de alto nivel. Estas metas se volverán más específicas durante la actividad de planificación de la fase de Establecimiento. Metas medibles, claramente definidas, son necesarias para orientar y ayudar en el desarrollo de tácticas de mejora. También permiten la medición objetiva de los resultados de mejora.
- *Desarrollar o revisar la propuesta de mejora*: Hasta que el plan estratégico de acción sea actualizado o recreado, el programa de mejora necesita una guía inicial. El propósito de esta actividad es crear un plan que guíe al programa hasta los planes de acción posteriores.
- *Continuar con la mejora*: El propósito de esta actividad es moverse hacia la fase principal del programa de mejora (el diagnóstico) y comenzar el ciclo continuo del mismo. Para esto es necesario revisar y/o actualizar los enfoques documentados y aprobados sobre la mejora, revisar las metas y objetivos del programa, y obtener la aprobación para continuar con la mejora.

2.3.2. Los modelos de proceso CMMI-DEV v1.2 y MoProSoft

Una vez que se ha analizado la naturaleza de un modelo genérico de mejora, se procede a analizar los dos modelos recientemente introducidos en los planes de estudio de las universidades mexicanas y que se relacionan con la educación en el área de Mejora del Proceso de Software. Cabe recordar que estos dos modelos son tomados del análisis presentado previamente y que permitió determinar la evolución de la educación de la Ingeniería de Software (véase Tabla 3). De igual manera que con el análisis del modelo IDEAL, dado que estos dos modelos son demasiado extensos, a continuación se proporciona una síntensis de aquellos componentes que son útiles para el desarrollo de esta tesis.

2.3.2.1. Capability Maturity Model Integration for Development v1.2 (CMMI-DEV v1.2)

El Modelo de Madurez y Capacidad Integrada (CMMI, *Capability Maturity Model Integration*) es un modelo de madurez de mejora de los procesos para el desarrollo de productos y servicios. Este modelo consiste en las mejores prácticas que tratan las actividades de desarrollo y de mantenimiento y que cubren el ciclo de vida del producto, desde su concepción hasta su entrega y mantenimiento. Las últimas versiones del modelo, integran los cuerpos de conocimiento que son esenciales para el desarrollo y el mantenimiento de software, pero que se habían tratado por separado en el pasado, tales como la Ingeniería de Software, la Ingeniería de Sistemas, la Ingeniería del Hardware y de Diseño, los aspectos no funcionales y la Adquisición. En este sentido, las denominaciones CMMI para la Ingeniería de Sistemas y la Ingeniería de Software (CMMI-SE/SW) fueron reemplazadas por el título "CMMI para el Desarrollo", reflejando así realmente la integración completa de estos cuerpos de conocimiento y la aplicación del modelo en el seno de una organización. CMMI para el Desarrollo (o CMMI-DEV) propone una solución integrada y completa para las actividades de desarrollo y de mantenimiento aplicadas a los productos y a los servicios de software.

CMMI-DEV v1.2 [CMMI, 2006] corresponde a la tercera versión entregable del modelo CMMI, posterior a las versiones 1.02 (primera versión, año 2000) y 1.1 (versión del año 2002). Las versiones previas sirvieron como realimentación para que los propios usuarios, evaluadores y

organizaciones evaluadas hicieran acotaciones sobre posibles mejoras, las cuales fueron estudiadas, refinadas e incluidas en la versión 1.2.

2.3.2.1.1. ***Propósito del modelo***

El propósito del CMMI-DEV v1.2 es ayudar a las empresas a mejorar sus procesos de desarrollo y de mantenimiento, tanto para los productos como para los servicios de software. Para esto, las 'constelaciones' de modelos CMMI representan una colección de componentes que incluyen un modelo, sus materiales de formación y los documentos de evaluación concernientes a un dominio de interés.

2.3.2.1.2. ***Descripción del modelo***

Ante la necesidad de desarrollar y mantener productos y servicios de calidad, el SEI ha identificado varias dimensiones sobre las que una organización puede enfocarse para mejorar sus actividades. Las tres dimensiones críticas sobre las cuales típicamente se concentran las organizaciones son las personas, los métodos y procedimientos, y las herramientas y el equipamiento. En este sentido, diferentes organizaciones de diversos tipos de industrias (incluyendo la aeroespacial, los bancos, la construcción de computadoras, el software, la defensa, la fabricación de automóviles y las telecomunicaciones) utilizan el CMMI-DEV. Los modelos de la constelación de CMMI-DEV contienen prácticas sobre la Gestión de Proyectos, la Gestión de Procesos, la Ingeniería de Sistemas, la Ingeniería de Hardware, la Ingeniería de Software y otros procesos de soporte utilizados para el desarrollo y el mantenimiento de software.

CMMI-DEV es considerado para introducir en las empresas los elementos esenciales de los procesos eficaces para una o más disciplinas y describir una trayectoria evolutiva de mejora, permitiendo así la transformación de procesos *ad hoc* e inmaduros a procesos disciplinados y maduros (con calidad y eficacia mejorada). El CMMI-DEV proporciona un enfoque en la mejora de procesos a través de dos representaciones diferentes: la *representación continua* y la *representación por etapas*. CMMI-DEV v1.2 está estructurado principalmente por diversos tipos de componentes y por áreas de proceso. Los componentes son los mismos independientemente de la representación elegida, por lo que en esta tesis éstos serán definidos de acuerdo al esquema propuesto por la representación por etapas.

En este sentido, la Figura 2.5 muestra que en CMMI-DEV v1.2 un *área de proceso* está asociada a un nivel de madurez, posee además un conjunto de *objetivos específicos*; los cuales describen características únicas que deben estar presentes para satisfacer el área de proceso y que representan componentes requeridos del modelo, y uno o varios *objetivos genéricos* asociados; los cuales se denominan así porque la misma declaración se aplica en múltiples áreas de proceso para describir las características que deben estar presentes para institucionalizar los procesos que son implementados. Dependiendo del nivel de madurez al cual pertenece el área de proceso, los objetivos específicos y genéricos cuentan con un conjunto de *prácticas específicas* y *prácticas genéricas* respectivamente.

Una práctica específica es la descripción de una actividad que se considera importante para alcanzar la meta específica asociada. Ciertas prácticas específicas cuentan con subprácticas que proporcionan una descripción detallada como guía para interpretarlas e implantarlas. Por otro lado, las prácticas genéricas se denominan así porque la misma práctica se aplica a múltiples áreas de proceso y se consideran importantes para el logro de la meta genérica asociada. Dentro de los componentes representados en el modelo se pueden encontrar además las *declaraciones de propósitos* que describen la finalidad del área de proceso y son un componente de tipo informativo.

La sección de *notas introductorias* del área de proceso describe los conceptos principales cubiertos por el área de proceso y que también representan un componente informativo. Se describe además una lista de *áreas de proceso relacionadas* para una mejor adaptación dentro de la organización.

Así, el área de proceso de la Figura 2.5, por ejemplo, representa un conjunto de prácticas relacionadas que cuando son implementadas colectivamente, satisfacen un conjunto de objetivos considerados importantes para mejorarla.

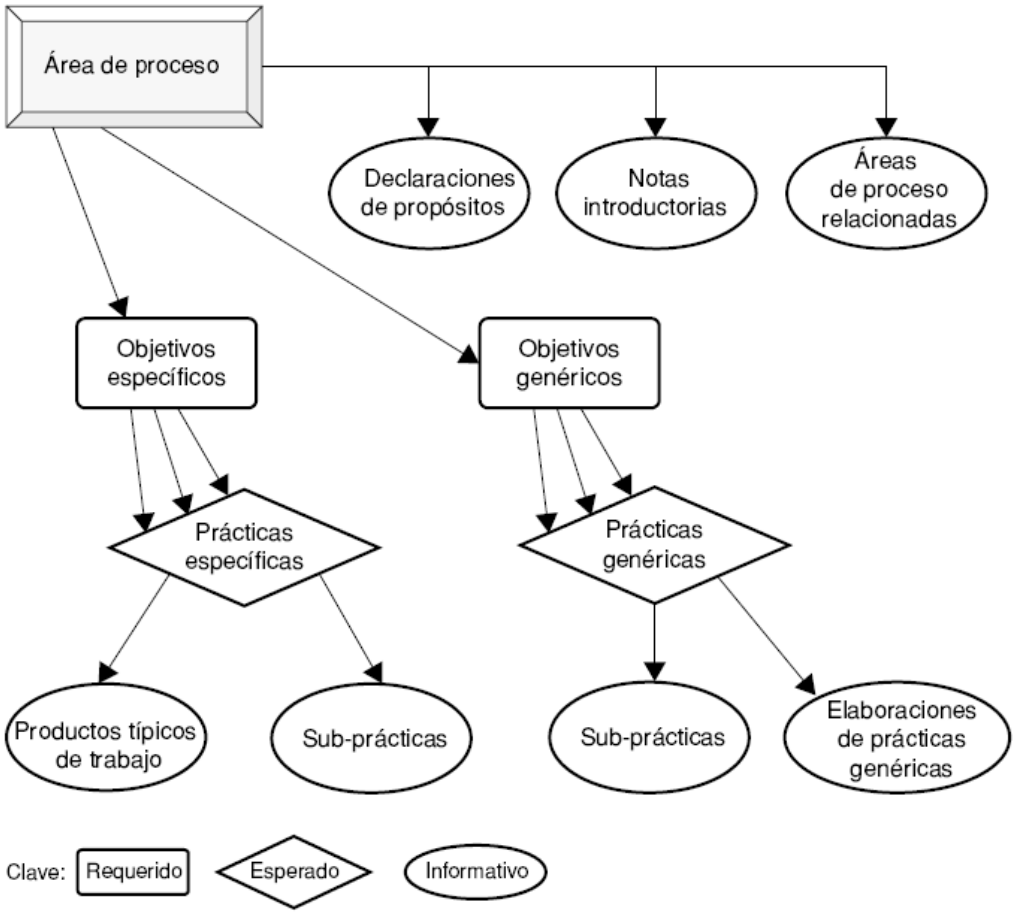


Figura 2.5. Componentes del CMMI-DEV v1.2 – representación por etapas [CMMI, 2006]

CMMI-DEV v1.2 se compone de 22 áreas de proceso, cada una de éstas es implementada para alcanzar el nivel de madurez correspondiente y son agrupadas de acuerdo a las siguientes cuatro categorías:

- *Gestión de Procesos*: Las áreas de proceso de la Gestión de Procesos contienen las actividades transversales a los proyectos relacionadas con la definición, planificación, despliegue, implementación, monitorización, control, evaluación, medición y mejora de los procesos.
- *Gestión de Proyectos*: Las áreas de proceso de la Gestión de Proyectos cubren las actividades relacionadas con la planificación, monitorización y control de proyectos.
- *Ingeniería*: Las áreas de proceso de Ingeniería cubren las actividades de desarrollo y de mantenimiento que se comparten entre las disciplinas de ingeniería. Las áreas de proceso de

Ingeniería fueron escritas usando terminología general de ingeniería de tal forma que cualquier disciplina técnica implicada en el proceso de desarrollo del producto (e.g. Ingeniería de Software o Ingeniería Mecánica) pueda usarlas para la mejora de procesos. Las áreas de proceso de Ingeniería también integran los procesos asociados con diferentes disciplinas de ingeniería en un único proceso de desarrollo de producto, dando soporte a una estrategia de mejora de procesos orientada al producto. Esta estrategia apunta a los objetivos de negocio esenciales más que a disciplinas técnicas específicas.

- *Soporte*: Las áreas de proceso de Soporte cubren las actividades que dan soporte al desarrollo y al mantenimiento del producto. Las áreas de proceso de Soporte tratan los procesos que se usan en el contexto de la ejecución de otros procesos. En general, las áreas de proceso de Soporte tratan los procesos que están orientados al proyecto y pueden tratar procesos que se aplican de manera más general a la organización.

CMMI-DEV v1.2 utiliza el concepto de "representación" para referirse a una guía que indica cómo efectuar las actividades de mejora de los procesos, además que es utilizada en el método de evaluación. Como se había mencionado anteriormente, el modelo establece dos formas de mejorar: la primera se enfoca en la mejora de un proceso específico, o un conjunto de procesos, a través de la representación continua, y la segunda se enfoca en la mejora completa de la organización de acuerdo a los procesos definidos a través de la representación escalonada o por etapas. En este sentido, la representación continua permite a una organización seleccionar un área de proceso (o un grupo de áreas de proceso) y mejorar los procesos relacionados con ésta. Esta representación utiliza *niveles de capacidad* para caracterizar la mejora concerniente a un área de proceso individual (véase Figura 2.6). Esta representación ofrece máxima flexibilidad cuando se utiliza un modelo CMMI para la mejora de procesos. La organización puede decidir si desea mejorar un solo proceso o varios que estén alineados a sus objetivos estratégicos, esta representación es recomendable para aquellas empresas que conocen bien todas su áreas.

La representación por etapas, por lo contrario, utiliza conjuntos predefinidos de áreas de proceso para definir un camino de mejora para una empresa. Este camino de mejora se caracteriza por diversos *niveles de madurez*. Cada nivel de madurez proporciona un conjunto de áreas de proceso que caracterizan diferentes comportamientos organizativos (véase Figura 2.6). Esta representación ofrece una manera sistemática y estructurada de aproximarse a la mejora de procesos basada en el modelo etapa por etapa. En el caso de que una organización no sepa por donde comenzar ni qué procesos elegir para mejorar, la representación por etapas es la opción designada. Esta ofrece un conjunto específico de procesos para mejorar en cada etapa.

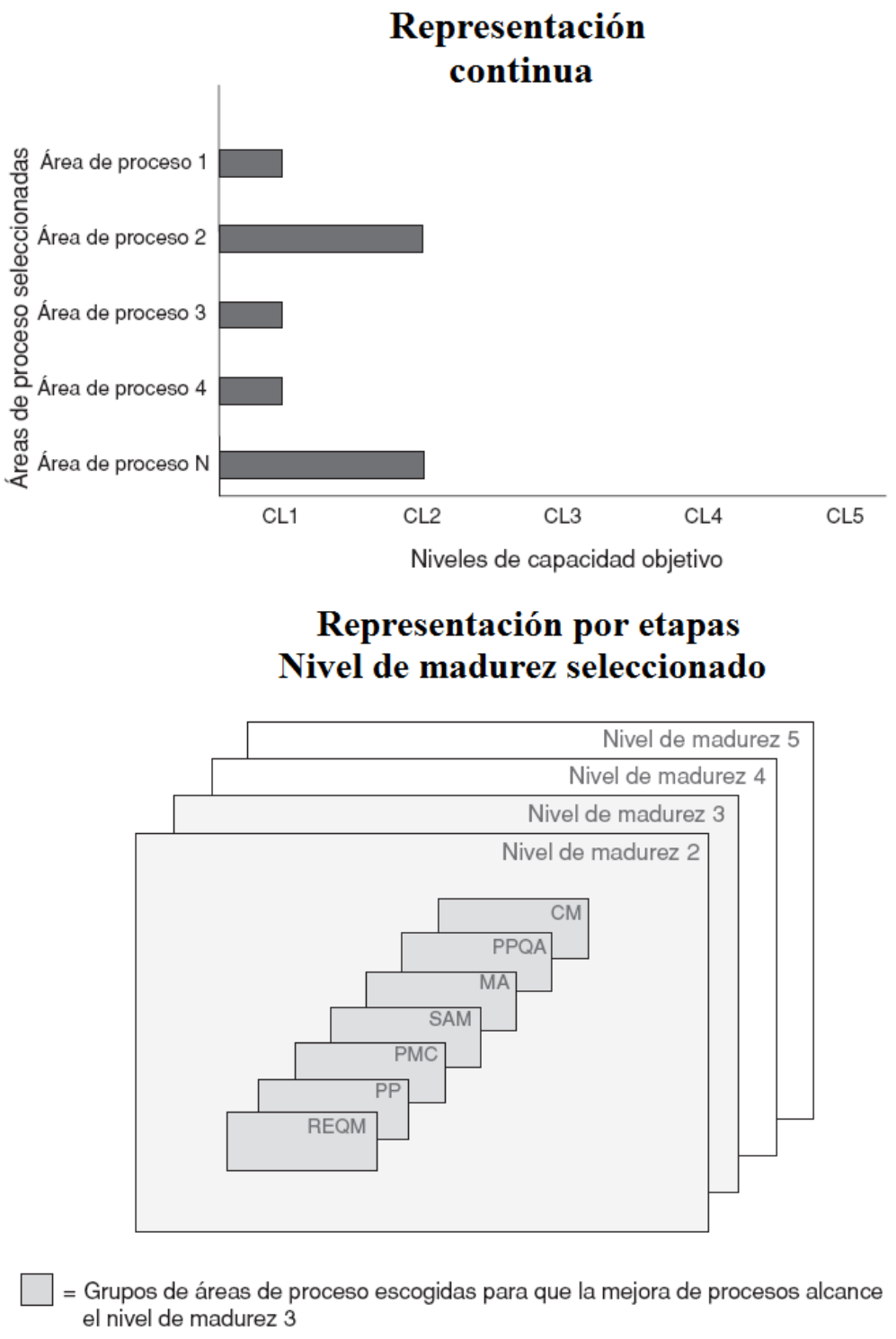


Figura 2.6. Alcance sobre las áreas de proceso en las representaciones continua y por etapas [CMMI, 2006]

La Tabla 4 compara las ventajas de cada representación y puede dar una idea sobre la representación que le conviene a cada empresa.

Tabla 4. Diferencias entre la representación continua y la representación por etapas

Representación continua	Representación por etapas
Concede la libertad explícita para seleccionar el orden de mejora que mejor satisface los objetivos de negocio de la organización y atenúa sus áreas de riesgo.	Permite a las organizaciones tener una trayectoria predefinida y probada de mejora.
Permite visibilidad incrementada de la capacidad alcanzada en cada área de proceso individual.	Se centra en un conjunto de procesos que proveen a una organización con una capacidad específica que está caracterizada por cada nivel de madurez.
Permite que las mejoras de diversos procesos sean realizadas en diversos valores.	Resume resultados de la mejora de procesos en un simple número de nivel de madurez.
Refleja una aproximación nueva, que todavía no tiene los datos para demostrar sus relaciones con el retorno de la inversión.	Se construye sobre una historia relativamente larga del uso, que incluye casos de estudio y datos que demuestran el retorno de la inversión.

La Tabla 5 resume las áreas de proceso del CMMI.DEV v1.2, además de sus categorías y niveles de madurez asociados. De acuerdo a la Figura 2.5, los *componentes requeridos* deben ser satisfechos e implementados obligatoriamente para poder cumplir con un área de proceso. Un componente requerido es, por lo tanto, usado en las evaluaciones para ayudar a determinar si un área de proceso es satisfecha o no.

Tabla 5. Áreas de proceso y sus categorías y niveles de madurez asociados

Área de proceso	Categoría	Nivel de madurez
Análisis causal y resolución - CAR	Soporte	5
Innovación y despliegue en la organización – OID	Gestión de proyectos	5
Gestión cuantitativa de proyecto – QPM	Gestión de proyectos	4
Rendimiento de procesos de la organización – OPF	Gestión de proyectos	4
Análisis de decisiones y resolución – DAR	Soporte	3
Definición de procesos de la organización + IPPD – OPD + IPPD	Gestión de proyectos	3
Desarrollo de requerimientos – RD	Ingeniería	3
Enfoque en proceso de la organización – OPF	Gestión de proyectos	3
Formación organizativa – OT	Gestión de proyectos	3
Gestión integrada de proyecto + IPPD – IPM + IPPD	Gestión de proyectos	3
Gestión de riesgos – RSKM	Gestión de proyectos	3
Integración de producto – PI	Ingeniería	3
Solución técnica – TS	Ingeniería	3
Validación – VAL	Ingeniería	3
Verificación – VER	Ingeniería	3
Aseguramiento de la calidad del proceso y de producto – PPQA	Soporte	2
Gestión de acuerdos con proveedores – SAM	Gestión de proyectos	2
Gestión de configuración – CM	Soporte	2
Gestión de requerimientos – REQM	Ingeniería	2

Medición y análisis – MA	Soporte	2
Monitorización y control de proyecto – PMC	Gestión de proyectos	2
Planificación de proyecto – PP	Gestión de proyectos	2

Por lo tanto, los niveles que utiliza CMMI-DEV describen un camino evolutivo recomendado para una empresa que quiera mejorar los procesos que utiliza para desarrollar y mantener sus productos y servicios. Los niveles pueden también ser el resultado (calificación) de las evaluaciones. Independientemente de la representación que se seleccione, el concepto de nivel es el mismo puesto que caracterizan a la mejora desde un estado mal definido hasta un estado que utiliza información cuantitativa para determinar y gestionar las mejoras que se necesitan para satisfacer los objetivos de negocio de una empresa. Para alcanzar un nivel particular, una organización debe satisfacer todas las metas apropiadas del área o conjunto de áreas de proceso que son objeto de mejora, independientemente de si es un nivel de capacidad o de madurez. En este sentido, el modelo establece seis niveles de capacidad y cada nivel es construido sobre el nivel anterior, es decir para que un proceso alcance un nivel de capacidad necesariamente debe haber alcanzado el nivel anterior; y cinco niveles de madurez que están compuestos por áreas de proceso cuyos objetivos deben ser cumplidos para que la organización pueda certificarse en cierto nivel de madurez (véase Figura 2.7).

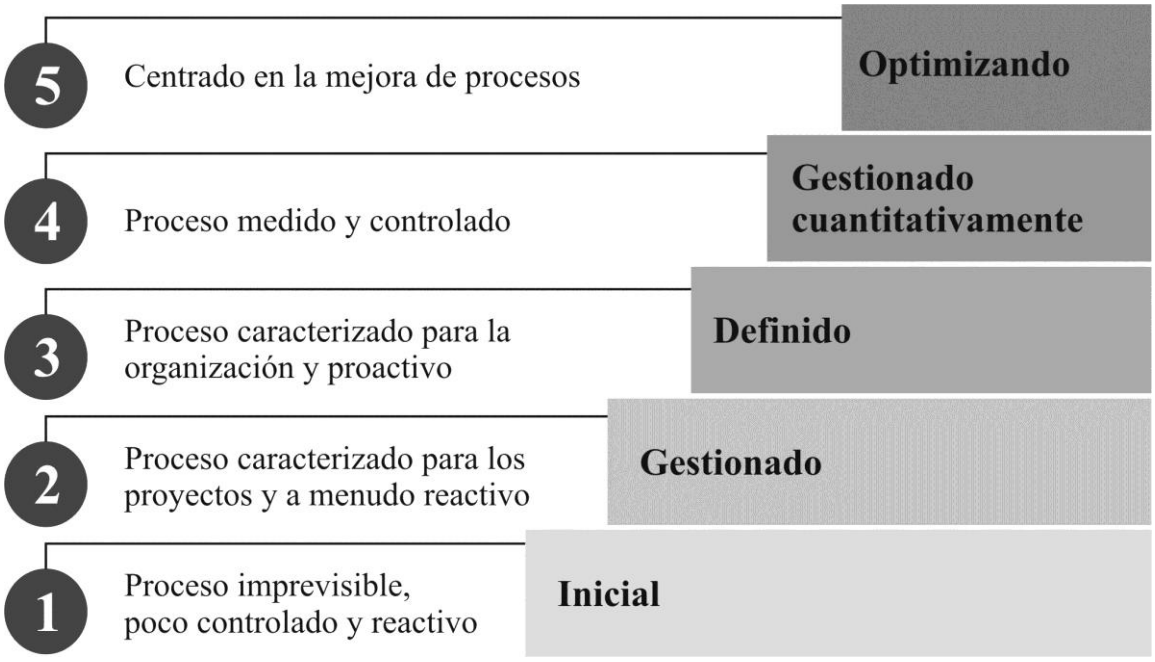


Figura 2.7. Niveles de madurez implantados por CMMI-DEV v1.2 [CMMI, 2006]

Por último, la Tabla 6 muestra los niveles de capacidad/madurez para los dos tipos de representaciones existentes en CMMI-DEV v1.2.

Tabla 6. Descripción de los niveles de capacidad/madurez definidos en CMMI-DEV v1.2

Niveles	Descripción general	
	Niveles de capacidad	**Niveles de madurez**
Nivel 0	Incompleto: Un proceso es denominado "incompleto" cuando uno o más objetivos específicos del área de proceso no son satisfechos.	-
Nivel 1	Realizado: Un proceso es denominado "realizado" cuando satisface todos los objetivos específicos del área de proceso. Además, soporta y permite el trabajo necesario para producir artefactos.	Inicial: En este nivel la mayoría de los procesos son *ad-hoc* y caóticos. Los éxitos en estas organizaciones se deben principalmente a la competencia y esfuerzos heroicos del personal y no al uso de procesos probados. A pesar de este caos, las organizaciones con frecuencia producen productos y servicios que funcionan; sin embargo, éstas frecuentemente exceden sus presupuestos y no cumplen sus planes.
Nivel 2	Gestionado: Un proceso es denominado "gestionado" cuando tiene la infraestructura base para apoyar el proceso. El proceso es planeado y ejecutado en concordancia con una política, emplea gente calificada que tiene los recursos adecuados para producir salidas controladas; involucra a las partes interesadas; es monitoreado, controlado y revisado; y es evaluado según la descripción del proceso.	Gestionado: En el nivel de madurez 2 se ordena el caos dado que las organizaciones se enfocan en aquellas tareas cotidianas referentes a la gestión. Cada proyecto de la organización cuenta con una serie de procesos para llevarlo a cabo, los cuales son planeados y ejecutados de acuerdo con políticas establecidas; los proyectos utilizan gente capacitada quienes disponen de recursos para producir salidas controladas; se involucra a las partes interesadas; son monitoreados, controlados y revisados; y son evaluados según la descripción del proceso.
Nivel 3	Definido: Un proceso denominado "definido" es adaptado desde el conjunto estándar de procesos de la organización de acuerdo a las guías de adaptación de la organización, y aporta artefactos, medidas, y otra información de mejora a los activos organizacionales.	Definido: En el nivel de madurez 3, los procesos son caracterizados y entendidos de buena forma, y son descritos en estándares, procedimientos, herramientas, y métodos. El conjunto estándar de procesos de la organización, el cual representa la base para el nivel de madurez 3, es establecido y mejorado continuamente.
Nivel 4	Gestionado cuantitativamente: Un proceso denominado "gestionado cuantitativamente" es controlado usando técnicas estadísticas y otras técnicas cuantitativas. Se establecen y usan objetivos cuantitativos para la calidad y realización del proceso como criterios para gestionar el proceso.	Gestionado cuantitativamente: En el nivel de madurez 4, la organización y los proyectos establecen objetivos cuantitativos para medir la calidad y realización de los procesos y los usa como criterios para su gestión. Los objetivos cuantitativos son definidos en base a las necesidades de los clientes, los usuarios finales, la organización, y los actores de los procesos. La calidad y realización de los procesos son entendidos en términos estadísticos y son gestionados durante todo el ciclo de vida del proceso.
Nivel 5	Optimizando: Un proceso denominado "en optimización" es mejorado en base al entendimiento de causas comunes de variación del proceso. Un proceso en optimización se enfoca en la mejora continua del proceso realizado a través de mejoras incrementales y usando la innovación tecnológica.	Optimizando: En el nivel de madurez 5 una organización mejora continuamente sus procesos basándose en el conocimiento de las causas comunes de variación inherente en los procesos. El nivel de madurez 5 se enfoca en la mejora continua de los procesos a través de las mejoras continuas, incrementales y tecnológicas.

2.3.2.1.3. ***Método de evaluación empleado***

Muchas organizaciones dan valor a la medición de su progreso mediante la realización de una evaluación para obtener una calificación de nivel de madurez o un perfil de logro de nivel de capacidad. Estas evaluaciones se realizan normalmente por una de las siguientes razones:

- Para determinar hasta qué punto los procesos de la organización se ajustan con las mejores prácticas del CMMI-DEV v1.2 e identificar áreas donde realizar mejoras.
- Para informar a los clientes y proveedores externos hasta qué punto los procesos de la organización se ajustan a las mejores prácticas del CMMI-DEV v1.2.
- Para cumplir con los requerimientos contractuales de uno o más clientes.

Las evaluaciones de las organizaciones que usan el modelo CMMI-DEV v1.2 deben ajustarse a los requerimientos definidos en el documento de Requerimientos de Evaluación para CMMI (ARC, *Appraisal Requirements for CMMI*). El documento ARC describe los requerimientos para diferentes tipos de evaluaciones. Una clase de evaluación comparativa total se define como una evaluación de Clase A. Otros métodos menos formales se definen como métodos de Clase B o de Clase C. En este sentido, SCAMPI [SCAMPI, 2006] es generalmente el método más aceptado para realizar evaluaciones utilizando modelos como CMMI-DEV v1.2.

SCAMPI define las reglas para asegurar la consistencia de las calificaciones de la evaluación. Para realizar una comparativa frente a otras organizaciones, las evaluaciones deben de asegurar calificaciones consistentes. El logro de un nivel de madurez específico o la satisfacción de un área de proceso debe significar lo mismo para las diferentes organizaciones evaluadas. SCAMPI incluye los métodos de evaluación de Clase A, B y C. En este sentido, las evaluaciones de Clase A corresponden a un método más riguroso que es el único que otorga una calificación oficial. SCAMPI B proporciona opciones para ajustar el alcance del modelo, pero la caracterización de las prácticas está fijada en una escala y se realiza sobre prácticas ya implementadas. SCAMPI C ofrece un amplio rango de opciones, incluyendo la caracterización de enfoques planificados para la implementación de procesos de acuerdo a una escala definida por el usuario.

Para realizar evaluaciones de Clase C, que son las menos formales dentro de la suite de métodos de evaluación del SEI, no se requiere de un equipo de evaluadores, es suficiente con una persona que asuma el rol de líder o *lead appraisal*. Así, estas evaluaciones son más cortas, flexibles y menos costosas que las realizadas por las Clases A y B. La Tabla 7 resume una comparativa de las principales características de las tres clases de evaluaciones permitidas en SCAMPI.

En el contexto de un curso a nivel posgrado, las evaluaciones Clase C se realizan a menudo por las siguientes razones:

- Proporcionan un análisis rápido sobre las deficiencias del proceso de una organización en relación con algún modelo CMMI.
- Evalúan la idoneidad de un nuevo proceso antes de su implantación.
- Supervisan la ejecución de un proceso.
- Determinan la preparación de una organización para una evaluación de Clase A.

Los resultados de una evaluación de Clase C permiten describir las fortalezas y debilidades de los procesos evaluados. Dependiendo del alcance y de la estrategia de evaluación, los resultados se

pueden asignar a los componentes relevantes de CMMI y determinar acciones de mejora para los procesos evaluados.

Tabla 7. Comparativa general de las clases de SCAMPI

Característica	Evaluaciones		
	Clase A	**Clase B**	**Clase C**
Importancia de las pruebas objetivas	Alto	Medio	Bajo
Valores generados	Si	No	No
Necesidad de recursos	Alto	Medio	Bajo
Tamaño del equipo	Grande	Medio	Pequeño
Fuente de datos (instrumentos, entrevistas y documentos)	Requiere las tres fuentes de datos	Requiere dos de las tres fuentes de datos (uno deben ser las entrevistas)	Requiere únicamente una fuente de datos
Requisitos de un líder de evaluación	Autorizado por el SEI	Autorizado por el SEI o personal entrenado o con experiencia	Personal entrenado o con experiencia

Las evaluaciones de Clase C son definidas por tres fases principales (Planificación y preparación de la evaluación, Realización de la evaluación, y Preparación del reporte de resultados) que contienen un conjunto de procesos que pueden ser adaptados.

2.3.2.1.3.1 Planificación y preparación de la evaluación

La preparación de los participantes de la evaluación requiere la consideración de los roles a desempeñar; en este sentido, un plan de evaluación debe incluir las actividades para comunicar a los participantes sus funciones y responsabilidades. En esta fase se especifican los requisitos mínimos para planificar los procesos y crear artefactos a través de los siguientes procesos:

- Análisis de requisitos. La evaluación inicia sobre la base para la planificación y la realización de las actividades de evaluación de tal manera que se maximice el logro de los objetivos del 'patrocinador' (regularmente el patrocinador es una persona de la organización evaluada que desempeña un puesto en la alta dirección). Las necesidades y restricciones de la evaluación deben determinarse con estas personas, y se debe crear un documento que recoja los parámetros de planificación. Toda la información de entrada (borradores de planes de evaluación, necesidades, etc.) debe ser migrada al plan de evaluación. El evaluador debe conocer al patrocinador; buscar, clarificar y verificar los requisitos de la evaluación; documentar y mantener la entrada de la evaluación y obtener la aprobación del patrocinador para realizar la evaluación. El documento elaborado debe identificar lo siguiente:
    - La identidad del patrocinador y la relación con la organización evaluada.
    - El propósito de la evaluación, incluyendo la alineación con los objetivos de negocio.
    - El alcance de la evaluación.
    - La unidad organizativa que es objeto de la evaluación.
    - El contexto del proceso.
    - Las limitaciones de la evaluación.

  - Los modelos CMMI usados, incluyendo la versión, disciplina y representación (continua o por etapas).
  - Una declaración escrita del evaluador que indique que conoce los criterios mínimos especificados por el SEI para realizar evaluaciones SCAMPI C.
  - La identidad y afiliación del evaluador, incluyendo sus responsabilidades en la evaluación.
  - Cualquier información que se obtenga durante la evaluación con el fin de apoyar el logro de sus objetivos.
  - Una descripción de los resultados de la evaluación.
  - Actividades de seguimiento previstas (e.g. informes, acciones de evaluación, planes, re-evaluación).

- Desarrollar un plan de evaluación. Un plan de evaluación establece las bases para un compromiso real y permite monitorear y controlar el proceso de evaluación. Por lo tanto ambos usos del plan (establecer un compromiso y monitorear el proceso) son incluidos en los requisitos para desarrollar el plan de evaluación. El evaluador debe documentar y dar mantenimiento al plan de evaluación, así como obtener la aprobación del patrocinador. El plan debe identificar la entrada de la evaluación (o alguna referencia si se desea), las actividades que serán desarrolladas, la logística que será empleada, y los riesgos que pueden presentarse. Todos estos elementos deben ser incluidos en la versión final del plan de evaluación.
- Seleccionar y preparar al equipo. Para el uso de SCAMPI C solamente es recomendado el uso de equipos en contextos donde exista suficiente carga de trabajo y soporte para que se justifique su existencia.
- Preparar a los participantes y obtener una evidencia objetiva inicial. La eficiencia de la evaluación es mucho mayor si se identifica evidencia objetiva de antemano y después se asocia con los elementos de la organización, así como con los componentes del modelo. La disponibilidad de esta información conforma el plan de recopilación de datos, y se discute normalmente con el personal que está apoyando las actividades de evaluación en el lugar donde se está llevando a cabo. La formalidad de este proceso debe adaptarse a la naturaleza de la recopilación de datos previstos para la evaluación. Por último, la preparación de los miembros de la organización que participarán en la evaluación tiende a aumentar el éxito de los esfuerzos encaminados a conseguir evidencia objetiva. Los mecanismos de comunicación utilizados para la preparación de los participantes van desde presentaciones formales a breves correos electrónicos o mensajes de correo de voz. Entre las actividades que debe desarrollar el evaluador se encuentran las siguientes:
  - Establecer una estrategia de comunicación para preparar a los participantes de la evaluación en la provisión de la información necesitada.
  - Establecer la comunicación entre los participantes de la evaluación para crear expectativas y responder preguntas.
  - Obtener un conjunto inicial de evidencia objetiva.
  - Inventariar la evidencia objetiva. La estrategia para preparar a los participantes de la evaluación puede incluir comunicados escritos, presentaciones formales e interacción directa con participantes individuales. Los comunicados usados para preparar a los

participantes deben incluir: el propósito, alcance, y enfoque de la evaluación; los roles y responsabilidades de los participantes; y la calendarización de las actividades a realizar.

Cada participante debe tener la oportunidad de entender la misma información básica acerca de la evaluación y debe tener una manera de comunicarse directamente con el evaluador.

- Preparar un conjunto de evidencia objetiva. La realización de una revisión provee mitigación de riesgos en las situaciones cuando la finalización de la evaluación depende de la suficiencia de los datos examinados. Por ejemplo, una técnica más general puede incluir exámenes de preparación para evaluar la satisfacción de los criterios de entrada para cualquier actividad. Los requisitos mínimos que se deben considerar para conducir una revisión de la preparación centrada en la evidencia objetiva son los siguientes:
    - Criterios para juzgar la suficiencia de datos.
    - La cantidad y tipos de datos requeridos.
    - Los datos actuales disponibles para la evaluación.
    - Resultados documentados de la revisión.
    - Comunicación de los resultados de la revisión al patrocinador.

2.3.2.1.3.2 Realización de la evaluación

El objetivo fundamental de esta fase es recoger información relevante acerca de la organización y relacionarla con las prácticas genéricas y específicas del modelo de referencia. Los datos recogidos deben ser consistentemente guardados y mantenidos para asegurar su uso y validar los resultados. Esta fase consiste de los siguientes procesos:

- Examinar la evidencia objetiva. Las actividades que el evaluador debe seguir son las siguientes:
    - Buscar y revisar información y artefactos que se relacionen con el enfoque, despliegue e institucionalización de las prácticas.
    - Identificar los productos de evaluación que identifiquen la información o artefactos revisados.
    - Anotar los productos de evaluación para indicar la parte de CMMI-DEV v1.2 a la que se aplica la información.
    - Anotar los productos de evaluación para identificar la parte de la unidad organizativa a la que se aplica la información.
- Documentar la evidencia objetiva. El registro consistente de los datos asegura que la información importante no sea perdida y los datos sean tratados apropiadamente durante el proceso de evaluación. El evaluador debe realizar lo siguiente:
    - Escribir notas para registrar la información que sirva como evidencia objetiva o ayuda para validar la interpretación de la misma.
    - Escribir notas para documentar la ausencia de evidencia objetiva necesaria para verificar el enfoque, despliegue, o institucionalización de las prácticas planeadas en la organización.

- Documentar problemas o deficiencias en el enfoque, despliegue o institucionalización de las prácticas que apoyan al alcance de las metas del CMMI-DEV v1.2.
- Comparar, combinar y consolidar periódicamente la información documentada.
- Inventariar periódicamente la evidencia objetiva revisada para identificar la información ya recolectada.
- Establecer acuerdos de confidencialidad durante la fase de planeación.

- Verificar la evidencia objetiva. Los datos usados para formular las salidas de la evaluación deben ser verificados para asegurar que los resultados conducen a las salidas apropiadas. El evaluador deberá hacer lo siguiente:
  - Verificar los artefactos directos, indirectos, afirmaciones, y otra información relacionada con el modelo de referencia.
  - Verificar que el criterio para la suficiencia de datos es satisfecho.
  - Verificar que cada búsqueda preliminar es apoyada por la evidencia objetiva.
- Validar las salidas preliminares de la evaluación. El uso de una "presentación preliminar de resultados" es una estrategia de valor agregado que ha funcionado durante muchos años en el contexto de las evaluaciones de procesos. En este sentido, los resultados preliminares deben ser precisos dado que se derivan de lo que se ha visto o escuchado durante la recolección de datos. Así, el evaluador debe realizar lo siguiente:
  - Asegurar que las versiones preliminares de los resultados son validadas por los participantes definidos en el plan de evaluación (si es que fueron definidos).
  - Considerar la realimentación obtenida durante la validación de los resultados preliminares.
- Generar los resultados de la evaluación. Los resultados de la evaluación, en forma de conclusiones (declaraciones de fortalezas y debilidades), están destinados a apoyar las necesidades de los patrocinadores de la evaluación. En este sentido, con frecuencia se emplean escalas de caracterización para ayudar a ilustrar las tendencias y patrones de los datos de la evaluación. En las practicas identificadas el evaluador debe:
  - Documentar las declaraciones de fortalezas, debilidades, y otras expresiones escritas obtenidas como resultado de la evaluación.
  - Documentar detalladamente el alcance del modelo de evaluación.

2.3.2.1.3.3 Preparación del reporte de resultados

Los resultados de la evaluación deben ser entregados a los receptores designados, incluyendo al patrocinador y participantes identificados en el plan de evaluación. Esta fase consiste de los siguientes procesos:

- Entregar los resultados de la evaluación. Los resultados son presentados o entregados a grupos o personas individuales designados por el patrocinador y convenidos en la fase de planeación de la evaluación. El evaluador debe realizar lo siguiente:
  - Presentar, entregar o comunicar los resultados de la evaluación a los participantes identificados en el plan de evaluación.

- Fortalecer, a través de un escrito y/o la comunicación oral, las disposiciones vigentes para la confidencialidad de los resultados de la evaluación. Es común organizar un foro de discusión para que el patrocinador de la evaluación pueda discutir los resultados en una junta cerrada. El propósito de esta junta ejecutiva es permitir al patrocinador (y otras entidades designadas por él) la formulación de preguntas para buscar y clarificar cuestiones que no es necesario comentar en público. Por eso es importante enfatizar la confidencialidad que se tendrá durante estas actividades.

- Empaquetar y archivar las evaluaciones. Al patrocinador de la evaluación le pertenecen los datos de la evaluación, y él controla la distribución y diseminación de la información contenida en los datos. Los datos de la evaluación deben contener lo siguiente:
  - Datos de la evaluación.
  - Entrada de la evaluación.
  - Identificación del método usado en la evaluación.
  - Resultados, incluyendo declaración de fortalezas y/o debilidades.
  - Caracterizaciones generadas.
  - Otras caracterizaciones de datos o atributos de prácticas o proyectos generados durante la evaluación (si los hay).
  - Declaración de la evaluación.

  Una vez que toda la información de la evaluación es entregada al patrocinador (en papel o en formato electrónico) los participantes no tienen la posibilidad de modificar su contenido puesto que el proceso de evaluación ha terminado.

2.3.2.2. Modelo de Procesos para la Industria del Software (MoProSoft)

MoProSoft es el modelo mexicano que fomenta la estandarización de su operación a través de la incorporación de mejores prácticas de gestión e Ingeniería de Software [NYCE, 2005]. La adopción de este modelo permite elevar la capacidad de las organizaciones para ofrecer servicios con calidad y alcanzar niveles internacionales de competitividad.

2.3.2.2.1. ***Propósito del modelo***

MoProSoft proporciona a la industria de software en México, que en su gran mayoría está formada por pequeñas y medianas empresas, un modelo basado en las mejores prácticas internacionales con las siguientes características:

- Fácil de entender y fácil de aplicar.
- No costoso en su adopción.
- Es la base para alcanzar evaluaciones exitosas con otros modelos o normas, tales como ISO 9000:2000 o CMMI.

2.3.2.2.2. ***Descripción del modelo***

Para la elaboración del modelo de procesos (MoProSoft), se aplicaron criterios para la generación de una estructura de los procesos que fuera acorde con la estructura administrativa de las organizaciones de la industria de software: Alta Dirección, Gerencia y Operación (véase Figura 2.8).

Cabe resaltar que MoProSoft se encuentra contenido en la Norma Mexicana NMX-I-059/NYCE-2005, al igual que el modelo de evaluación EvalProSoft.

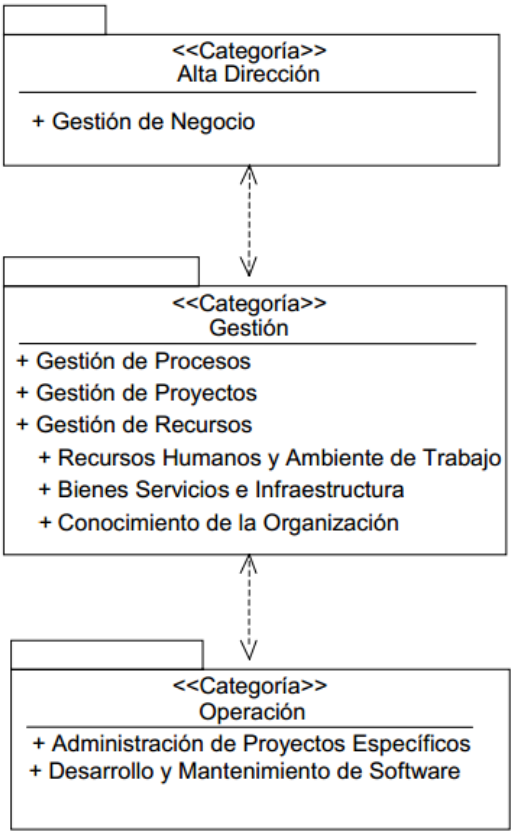


Figura 2.8. Diagrama de categoría de procesos en MoProSoft [NYCE, 2005a]

La categoría de Alta Dirección contiene el proceso de Gestión de Negocio. Esta categoría de procesos aborda las prácticas de la Alta Dirección, relacionadas con la gestión del negocio. Proporciona también los lineamientos para los procesos de la categoría de Gestión y se realimenta de la información generada por éstos. La Figura 2.9 muestra la relación entre los procesos de las tres categorías.

La categoría de Gestión está integrada por los procesos de Gestión de Procesos, Gestión de Proyectos y Gestión de Recursos. Este último proceso está constituido por los subprocesos de Recursos Humanos y Ambiente de Trabajo; Bienes, Servicios e Infraestructura; y Conocimiento de la Organización. Esta categoría de procesos aborda las prácticas de Gestión de Procesos, Gestión de Proyectos, y Gestión de Recursos en función de los lineamientos establecidos en la categoría de Alta Dirección. La principal labor de esta categoría es proporcionar los elementos para el funcionamiento de los procesos de la Categoría de Operación, recibir y evaluar la información generada por éstos y comunicar los resultados a la Categoría de Alta Dirección.

La categoría de Operación está integrada por los procesos de Administración de Proyectos Específicos y Desarrollo y Mantenimiento de Software. Esta categoría de procesos aborda las prácticas de los proyectos relacionados con el desarrollo y mantenimiento de software. Esta categoría realiza las actividades de acuerdo a los elementos proporcionados por la Categoría de Gestión y le devuelve a ésta toda la información y productos generados.

Los productos generados en los procesos se clasifican en productos de software, planes, reportes, registros, lecciones aprendidas y demás. Los productos de software se clasifican de manera

general como Especificación de Requisitos, Análisis y Diseño, Software, Prueba, Registro de Rastreo y Manuales.

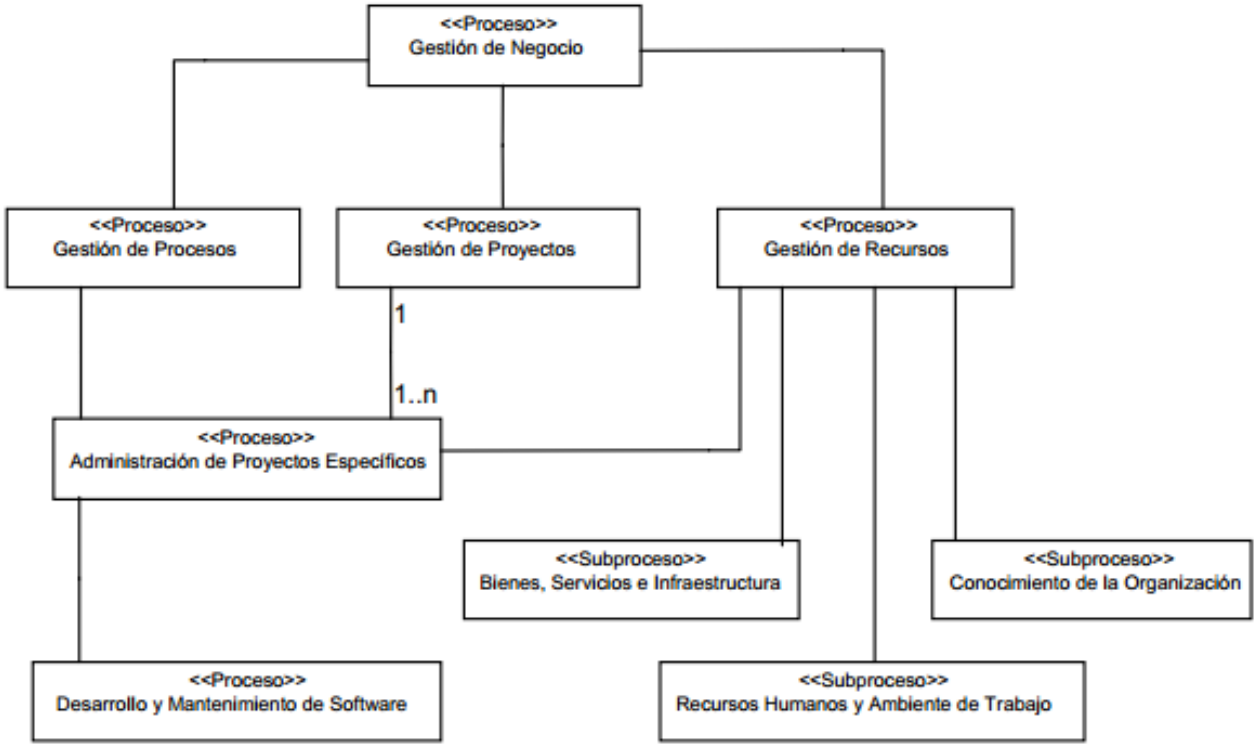


Figura 2.9. Relación entre los procesos [NYCE, 2005a]

Dentro del modelo, un rol es responsable de llevar a cabo un conjunto de actividades de uno o más procesos. La clasificación general de los roles se presenta en la Figura 2.10.

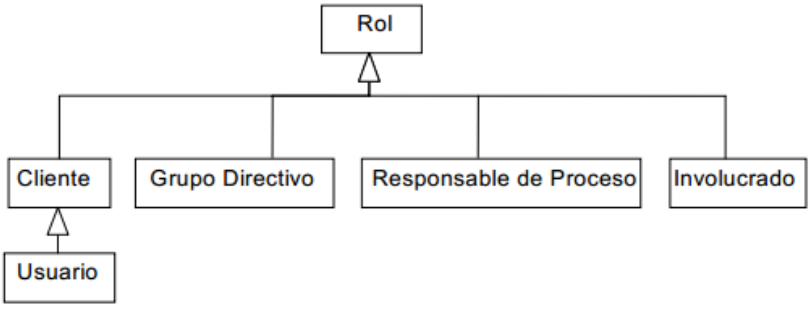


Figura 2.10. Clasificación general de roles [NYCE, 2005a]

El rol del Cliente, por ejemplo, es el que solicita un producto de software y financia el proyecto para su desarrollo o mantenimiento. El Usuario es el que utilizará el producto de software. El Grupo Directivo es el que dirige a una organización y es responsable de su funcionamiento exitoso. El Responsable de Proceso es el encargado de la realización de las prácticas de un proceso y del cumplimiento de sus objetivos. Por último, el Involucrado representa a otros roles con habilidades requeridas para la ejecución de actividades o tareas específicas (e.g. analista, programador, revisor, entre otros).

De manera similar al CMMI-DEV v1.2, la dimensión de capacidad de MoProSoft está formada por seis niveles de capacidad y nueve atributos de proceso (véase Tabla 8). Un nivel de capacidad está formado por uno o varios atributos que conjuntamente proporcionan una mejora importante en la capacidad para realizar un proceso. Los niveles proporcionan una manera racional de progresar en la mejora de la capacidad de cualquier proceso, la Figura 2.11 muestra la dimensión de la capacidad de proceso, indicando los Atributos de Proceso (AP) de cada nivel de capacidad.

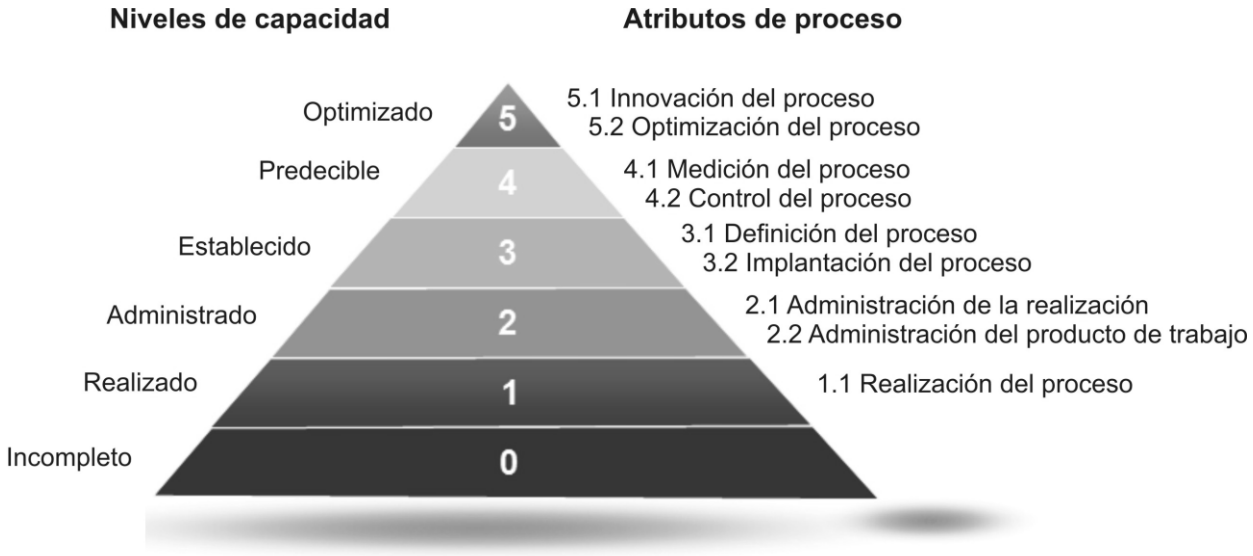


Figura 2.11. Dimensión de la capacidad de proceso [NYCE, 2005a]

La Tabla 7 muestra los niveles de capacidad para MoProSoft.

2.3.2.2.3. ***Método de evaluación empleado***

El método de evaluación EvalProSoft [NYCE, 2005c] examina los procesos empleados por una organización para determinar si éstos son efectivos para alcanzar sus objetivos. La evaluación describe la práctica actual de una organización con respecto a la capacidad de los procesos seleccionados. Los resultados se pueden emplear para realizar actividades de mejora del proceso o para determinar la capacidad del proceso, analizando los resultados conforme a las necesidades del negocio en la organización, identificando fortalezas, debilidades y registros propios de los procesos. Un proceso de evaluación documentado debe contener, por lo menos, las fases de Planificación de la evaluación, Recopilación de información, Validación de datos, Calificación de atributos del proceso, y Generación de reportes.

2.3.2.2.3.1 Planificación de la evaluación

En esta fase es necesario desarrollar y documentar un plan para la evaluación que incluya como mínimo:

- Las entradas requeridas y especificadas en el modelo de referencia.
- Las actividades que se realizan para conducir la evaluación.
- Los recursos y el calendario asignados a estas actividades.
- La identidad y responsabilidades definidas para los participantes en la evaluación.

Tabla 8. Descripción de los niveles de capacidad definidos en MoProSoft

Niveles	**Descripción general**	
	Niveles de capacidad	**Atributos del proceso**
Nivel 0	Incompleto: Existen grandes fallas que limitan o incluso impiden el cumplimiento de los objetivos y propósitos del proceso.	-
Nivel 1	Realizado: El propósito del proceso es generalmente alcanzado a pesar de que éste no es planificado o controlado con regularidad. Los individuos dentro de la organización reconocen que se debe llevar a cabo una acción que se ejecuta cuando es requerida.	AP 1.1: Realización del proceso: El proceso emplea un conjunto de prácticas, que son iniciadas por productos identificables y genera productos reconocibles, que satisfacen el propósito del proceso.
Nivel 2	Administrado: El proceso genera productos capaces de ser liberados en tiempo y bajo planes controlables. Estos productos generados están alineados con determinados estándares y requerimientos.	AP 2.1: Administración de la realización: La ejecución del proceso se gestiona para producir productos dentro de un plazo de tiempo y con requisitos preestablecidos. AP 2.2: Administración del producto de trabajo: La ejecución del proceso se gestiona para generar productos que se documentan y se controlan satisfaciendo los requisitos de acuerdo con los objetivos de calidad establecidos.
Nivel 3	Establecido: El proceso es realizado y administrado usando un procedimiento fundamentado en principios de la Ingeniería de Software.	AP 3.1: Definición del proceso: La ejecución del proceso utiliza una definición de proceso basada en un proceso estándar. AP 3.2: Recursos del proceso: La ejecución del proceso utiliza eficazmente recursos humanos con las habilidades e infraestructura adecuadas.
Nivel 4	Predecible: El proceso es ejecutado bajo controles y límites de tal forma que se asegure la consecución de sus objetivos. Se recolectan y analizan datos y mediciones del proceso.	AP 4.1: Medición del proceso: La ejecución del proceso se soporta en los objetivos y mediciones que son utilizadas para asegurar que la implementación del proceso contribuye a la consecución de los objetivos. AP 4.2: Control del proceso: La ejecución del proceso se controla a través de la recopilación y análisis de mediciones para controlar y corregir, donde sea necesario, el rendimiento del proceso.
Nivel 5	Optimizado: La ejecución de los procesos esta optimizada para asegurar el logro de los objetivos de negocio actuales y futuros. El proceso optimizado requiere el manejo constante de nuevas ideas y tecnologías así como la capacidad de cambiar aquellos procesos que sean necesarios para la consecución de los objetivos. La principal diferencia con el proceso predecible, es que ahora además de ser predecible, el proceso tiene capacidad de adaptación y cambios de acuerdo a las necesidades de los objetivos de negocio actuales y futuros.	AP 5.1: Innovación del proceso: Los cambios a la definición, gestión y rendimiento del proceso son controlados para conseguir los objetivos de negocio de la organización. AP 5.2: Optimización del proceso: Los cambios a los procesos se identifican y se implementan para asegurar la mejora continua en el cumplimiento de los objetivos de negocio definidos de la organización.

2.3.2.2.3.2 Recopilación de datos.

Los datos requeridos para evaluar los procesos que se encuentran dentro del alcance de la evaluación deben utilizarse sistemáticamente en esta fase aplicando, como mínimo, lo siguiente:

- La estrategia y técnicas para la selección, recopilación, análisis de los datos y la justificación deben ser explícitamente identificables y demostrables.
- Se debe establecer la correspondencia entre los procesos de la unidad de la organización, especificados en el alcance de la evaluación, y los elementos del modelo de evaluación del proceso.
- Cada proceso identificado dentro del alcance de la evaluación debe ser evaluado con base en la evidencia objetiva.
- La evidencia objetiva obtenida para cada atributo de cada proceso evaluado debe ser suficiente para lograr el propósito y alcance de la evaluación.
- La identificación de la evidencia objetiva obtenida debe ser registrada y mantenida para proporcionar la base para la verificación de las calificaciones.

2.3.2.2.3.3 Validación de datos.

Los datos recopilados deben ser validados en esta fase para alcanzar lo siguiente:

- Confirmar que la evidencia recopilada es objetiva.
- Asegurar que la evidencia objetiva es suficiente y representativa para cubrir el alcance y el propósito de la evaluación.
- Asegurar que los datos son consistentes en su conjunto.

2.3.2.2.3.4 Calificación de atributos del proceso.

En esta fase es necesario asignar una calificación para cada atributo del proceso evaluado. Así, considerando los datos validados, será posible considerar lo siguiente:

- El conjunto de calificaciones de atributos del proceso se debe registrar como perfil del proceso para la unidad de la organización.
- Durante la evaluación se debe emplear el conjunto definido de indicadores de evaluación del modelo de evaluación del proceso, el cual debe apoyar el juicio de los evaluadores en la calificación de los atributos del proceso, con el fin de proporcionar la base para la repetición entre las evaluaciones.
- Se debe registrar el proceso de toma de decisiones que se debe emplear para justificar la calificación obtenida.
- Se debe mantener el rastreo entre la calificación de un atributo y la evidencia objetiva que se empleó para obtener dicha calificación.
- Para cada atributo del proceso calificado se debe registrar la relación entre los indicadores y la evidencia objetiva.

2.3.2.2.3.5 Generación de reportes.

Los resultados de la evaluación (incluyendo fecha, información de entrada, identificación de evidencia objetiva, proceso de evaluación documentado, perfil de cada proceso evaluado) se deben documentar y reportar al promotor de la evaluación o a su representante.

La escala de calificación ordinal que se define a continuación debe ser empleada para expresar los niveles de cumplimiento de los atributos del proceso:

- *No logrado*. Existe poca o nula evidencia del cumplimiento del atributo definido del proceso evaluado.
- *Parcialmente logrado*. Existe evidencia de una aproximación, o se ha alcanzado parcialmente el atributo definido del proceso evaluado. Algunos aspectos del cumplimiento del atributo pueden ser impredecibles.
- *Considerablemente logrado*. Existe evidencia de una aproximación sistemática, o ya se ha alcanzado hasta cierto grado el atributo definido para el proceso evaluado. Podrían existir algunas debilidades relacionadas con este atributo del proceso evaluado.
- *Completamente logrado*. Existe evidencia de una aproximación sistemática y completa de que se ha alcanzado el atributo del proceso evaluado. No existen debilidades significativas relacionadas con éste.

Los puntos ordinales definidos anteriormente se deben de entender en términos de una escala que representa el grado de cumplimiento. Los valores correspondientes se muestran en la Tabla 9.

Tabla 9. Escala de calificación usada por EvalProSoft

Calificación	Porcentaje de cobertura
No logrado	0 al 15%
Parcialmente logrado	> 15% a 50%
Considerablemente logrado	> 50% a 85%
Completamente logrado	> 85%

Siguiendo estos modelos o lineamientos de mejora, muchas propuestas educativas han sido desarrolladas para apoyar la educación en los niveles de pregrado y posgrado. En este contexto, a continuación serán analizadas algunas propuestas relacionadas con el tema de investigación de esta tesis, concretamente apoyar el proceso de enseñanza/aprendizaje de la Mejora del Proceso de Software. Posteriormente, se presenta un análisis comparativo empírico que permite sentar las bases funcionales del soporte tecnológico propuesto.

2.4. Estado del Arte

La Ingeniería de Software plantea una complejidad importante tanto para los estudiantes en cuanto a su aplicación en un dominio específico, como para los profesores en el sentido de encontrar muchas dificultades para su enseñanza. Es bien sabido que la educación en esta área está relacionada en gran medida con conceptos y modelos teóricos, y que su aplicación en contextos prácticos regularmente no está bien estructurada, o bien se carece de experiencia para hacerlo. En otras palabras, la enseñanza de la Ingeniería de Software debe estar fuertemente apoyada por el mundo

real y la participación activa de los estudiantes en éste parece ser un método adecuado de instrucción. Sin embargo, en la mayoría de los métodos de enseñanza se simula este mundo real y se introduce a los estudiantes en el desarrollo de proyectos ficticios en un entorno seguro de clase. Esta situación puede ser beneficiosa para ciertas áreas de la Ingeniería de Software (e.g. análisis y modelado de sistemas, documentación de código, verificación y validación, etc.), pero no para el caso de la Mejora del Proceso de Software que, como ya se explicó anteriormente, tiene una relación importante con el análisis de un contexto organizacional real.

En este sentido y con el objetivo de analizar las características de propuestas similares desarrolladas a lo largo de los años, las siguientes secciones de este capítulo revisarán la forma en que la enseñanza sobre la Mejora del Proceso de Software ha ido cambiando, empezando con el diseño de cursos y formas de trabajo hasta el desarrollo de herramientas computacionales de soporte.

2.4.1. Real World Lab: Un acercamiento entre estudiantes y la industria a nivel de proyectos, productos y clientes reales

The Real World Lab (RWL) [Moore & Brennan, 1995] fue un curso práctico para estudiantes de pregrado de Ingeniería de Software que los involucró en el desarrollo de proyectos y productos reales con la industria, y el manejo de las relaciones con los clientes. RWL fue un curso que cumplió con los requerimientos del último año de la carrera, que indicaban que los estudiantes debían completar su especialización en el área de la Ingeniería de Software a través de la experiencia. RWL tenía como objetivo proporcionar a los estudiantes esta experiencia real en cuanto a la mejora de los procesos basada en el modelo CMM [Biberoglu & Haddad, 2002], emulando las acciones realizadas dentro de una compañía real. Para esto, se requirió contar con la definición de un proceso maduro con el que los estudiantes se relacionaran para realizar un proyecto de mejora, y observaran los beneficios de establecer un proceso más eficiente. Además, los estudiantes podían identificar problemas en la organización, recomendar e implementar soluciones y observar los resultados de sus propias ideas sobre la mejora de procesos.

2.4.1.1. Propósito del curso

RWL fue creado en el Instituto Tecnológico de Georgia de los Estados Unidos de Norteamérica y pretendía demostrar que un enfoque orientado a la evaluación y mejora de los procesos software podía ser efectivo en un entorno estudiantil adecuado. La visión del curso se enfocó a involucrar a una organización en este entorno estudiantil para alcanzar la optimización de sus procesos a través de los cinco niveles de evaluación propuestos por el CMM.

2.4.1.2. Desarrollo del curso

La creación de RWL se enfocó a dos objetivos principales que se resumen como:

- La mejora de procesos. Con el objetivo de medir la mejora de procesos en una organización, es necesario establecer una línea base. Después de implementar las recomendaciones, de acuerdo a los resultados de las evaluaciones, se realizan evaluaciones repetitivas y periódicas para evaluar la efectividad del programa de mejora desarrollado por los estudiantes. Naturalmente se deseaba que RWL estableciera procesos efectivos en organizaciones de desarrollo de software que satisficieran las necesidades de sus clientes, pero también que estableciera un modelo de trabajo para los estudiantes de la Ingeniería de Software.
- La educación. La perspicacia de los estudiantes generada a través de la propia experiencia resulta más significativa que aprender simplemente con lecturas o libros. La evaluación conducida en RWL proporcionó a los estudiantes la oportunidad de aprender sobre aquellos

factores que influencian el proceso software y de recomendar mejoras basadas en su propia experiencia.

El diseño de RWL consideró características importantes del trabajo práctico, al abordar cuestiones importantes sobre los procesos de software. Por ejemplo, algunas consideraciones son las siguientes:

- Aunque usualmente los estudiantes utilizaban RWL por un año, se creó una mano de obra capacitada capaz de compararse con la empleada por la industria en esos años. Los estudiantes tenían la capacidad para encajar y gestionar los roles demandados por la industria.
- Los proyectos eran continuos pero la mano de obra estaba limitada por el calendario escolar. El trabajo práctico tuvo que ser divido en cuatro trimestres y los estudiantes debieron crear espacios de tiempo entre cada uno de estos.
- Los estudiantes cursaban materias relacionadas con las Ciencias de la Computación, como cursos sobre lenguajes de programación, hardware y diseño de sistemas. Sin embargo fueron requeridos otros conocimientos sobre el dominio (e.g. Ingeniería Aeroespacial y Contabilidad) para entender los requisitos de los clientes con los que se trabajaba.
- Las herramientas usadas en el laboratorio debían tener un enfoque al aprendizaje activo. Es verdad que muchas Herramientas asistidas por la Computadora para la Ingeniería de Software (CASE, por sus siglas en inglés), entre otras, eran extremadamente poderosas, pero los estudiantes debieron disponer de un trimestre entero para aprender a usarlas correctamente.

En los cursos donde RWL fue utilizado consecutivamente durante dos años, por ejemplo, los mismos estudiantes eran capaces de realimentar los procesos, identificar problemas y proporcionar soluciones. Esta experiencia permitió definir lo siguiente:

- Cuadernos de proyectos. Algunos estudiantes notaron que se perdía cierto conocimiento cuando los compañeros de años anteriores no dejaban documentación sobre los proyectos desarrollados. En base a lo anterior, se comenzó a guardar toda la documentación generada en los proyectos con el objetivo de contar con un historial que incluyera documentos, información de contactos, reportes de estado, decisiones importantes, etc. Este documento fue formalizado a través de un estándar que se utilizaba para cada proyecto nuevo.
- Gestión de configuración. Cuando un estudiante iba ganando experiencia a través de los equipos en los que participaba, la necesidad de compartir código y gestionar el control de los cambios se convirtió en una característica crucial de trabajo. En este sentido, los estudiantes desarrollaron un conjunto de *scripts* que les permitieron compartir código y controlar las versiones sin conflictos al final de cada trimestre y/o proyecto. De esta forma todo el código y los documentos eran respaldados para cursos posteriores.
- Estándares de código. Después de revisar código C entre los miembros del equipo y observar problemas de entendimiento para leer y escribir el código, los estudiantes propusieron una notación estándar para todos los proyectos.
- Revisiones. Como parte de los requisitos curriculares, cada estudiante debía participar en una revisión con el cliente al final de cada trimestre. Dado que se determinó que estas revisiones

debían ser más frecuentes para descubrir problemas tempranamente, muchos de los equipos realizaron revisiones semanales con sus patrocinadores, enviando reportes de estado, y participando en reuniones con ellos.

- Roles ejecutivos. En adición a los miembros de los equipos y líderes de proyectos, se identificó que otros roles eran necesarios para que RWL continuara mejorando. El rol de tipo ejecutivo fue creado para introducir una figura de liderazgo. Entre otros roles importantes se encontraba el gestor de evaluación de la calidad del software, que era el responsable de evaluar, recomendar e implementar mejoras a los procesos en el laboratorio; el responsable del laboratorio que se enfocó en los recursos de hardware y software en la organización; el jefe científico que se aseguraba de la consistencia técnica y calidad a través de los proyectos; y "gurús" que eran expertos en una herramienta o tecnología en particular y brindaban soporte en todas las etapas de los proyectos. De esta manera los estudiantes ganaron mucha más experiencia al relacionarse con roles de tipo líder.

2.4.1.3. Resultados alcanzados

RWL se utilizó considerando el modelo CMM para evaluar la situación actual de una organización y desarrollar un plan de acción futuro. Un estudiante fue elegido como gestor de evaluación de calidad del software para supervisar la evaluación sobre los procesos y gestionar la cooperación de los compañeros en la recogida y análisis de los datos. Siguiendo la guía de CMM se desarrolló la evaluación usando un cuestionario de madurez, realizando entrevistas con cada uno de los jefes de proyectos, y discutiendo de forma grupal con los representantes de cada área. Así, los datos fueron recopilados y analizados por los estudiantes para determinar la madurez de la organización.

A través de la evaluación fue posible obtener un Perfil de Área de Proceso que relacionaba las prácticas actuales de la organización con los niveles usados para establecer un valor de cobertura para cada nivel de CMM (por ejemplo, Completamente Satisfecho, Parcialmente Satisfecho, o No satisfecho). En este sentido, las áreas de proceso como Gestión de Configuración, Medición de Calidad, Evaluación de Proyectos, Planificación de Proyectos, y Gestión de Requisitos fueron parcialmente satisfechas. Esto mostró que RWL podía establecer de forma natural un proceso de institucionalización a medida que se avanzaba en la mejora.

Las debilidades del proceso fueron encontradas en el Nivel 3 de modelo, en las áreas de proceso de Enfoque en el Proceso, Definición de Procesos, Gestión Integrada, Ingeniería de Productos, y Coordinación Intergrupal, dado que éstas fueron no satisfactorias.

Después de comparar las entrevistas con los resultados de la evaluación, los estudiantes determinaron que era necesario mejorar algunas de las áreas, por lo que se propuso un plan de mejora en el cual se institucionalizara el entrenamiento de los jefes de proyectos, en particular sus habilidades para la calendarización, la estimación de costos, la motivación del personal, y el establecimiento de los objetivos de procesos. A través de esta experimentación con una empresa real, los estudiantes propusieron las siguientes actividades como base de la mejora:

- Desarrollar un plan de evaluación de calidad para todas las fases del desarrollo de software. Adicionalmente, fue necesario establecer métricas y estándares para vigilar la calidad en las actividades, así como un sistema de monitoreo para detectar posibles desviaciones de los umbrales definidos.
- Incorporar un proceso formal de revisión que incluyera a los requisitos, al diseño, las revisiones de código, las pruebas de usabilidad para prototipos, las revisiones de clientes, y las pruebas sobre los productos desarrollados.

Así, a través de la obtención de un perfil de madurez, los procesos de RWL contribuyeron a mejorar el proceso educativo a nivel de pregrado en el área de la Ingeniería de Software. Dado que el enfoque del curso se basó en la evaluación del personal a través de cuestionarios sobre el modelo CMM, los estudiantes se relacionaron rápidamente con la forma de trabajo de una empresa real y comprendieron la importancia del proceso de mejora a través de la experiencia que adquirieron en el laboratorio.

2.4.2. El curso de Jaccheri sobre la calidad del software y la mejora del proceso de software basado en la interacción con la industria local de software

La investigación de Jaccheri [Jaccheri, 2002] describe el rediseño de un curso enfocado a la calidad del software y a la Mejora del Proceso de Software. Además de que el curso incluye clases teóricas y proyectos de clase, su mayor contribución fue el diseño de un patrón de colaboración con la industria local de software. Dicho modelo de interacción está basado en la investigación, en presentaciones de la industria y en evaluaciones subsecuentes y documentos escritos por los estudiantes. En este proyecto se rediseñó el modelo expuesto en [Dingsøyr et al., 2000] y que fue aplicado en la Universidad de Noruega, donde hasta entonces se habían basado solamente en modelos revisados por libros estadounidenses y algunos europeos.

2.4.2.1. Propósito del curso

La idea de Jaccheri fue basarse en el curso SQPI, creando un nuevo curso que estuviera relacionado con la industria local de software. De esta manera los estudiantes y los profesores aprenderían a través de ejemplos reales sobre temas relacionados con la calidad de los procesos. Este enfoque rediseñado se basó en una investigación previa de Lethbridge para establecer una secuencia de contenidos que permitiera explorar la importancia de temas específicos dentro de la Ingeniería de Software [Lethbridge, 1998] como la realización de pruebas para el aseguramiento de la calidad, la gestión de la configuración, y el uso de estándares para la definición de procesos. Esta investigación previa identificó que tanto en la industria local como en la internacional, estos temas eran importantes y era necesario crear iniciativas dentro de las universidades para implementar cursos especializados que los abordara.

2.4.2.2. Desarrollo del curso

Para crear o adaptar un modelo que lograra la interacción con la industria, Jaccheri planteó dos objetivos para el curso: el primero se relacionaba con la mejora de la calidad educativa en términos de la actualización constante de los módulos de aprendizaje enfocados a la calidad y mejora de los procesos, y que habían sido utilizados en cursos anteriores; y el segundo que establecía que era necesario aprender de la industria local a través de la experiencia con proyectos reales.

Entre otras alternativas, el curso se basó en un modelo de aprendizaje donde los estudiantes trabajaban en forma colaborativa con la industria en determinados niveles de calidad y con diferentes procesos. Bajo este enfoque, los estudiantes debían desempeñar el rol, o al menos intentarlo y observar al experto de la industria, de gestor de calidad y procesos. Este método parecía ser ideal puesto que los estudiantes colaboraban con una organización real y podían experimentar y obtener experiencia sobre problemas reales y cumplir con los dos objetivos del curso. Sin embargo, una realidad sobre la industria es que muchas empresas no estaban interesadas en compartir con las universidades sus problemas sobre calidad y procesos. Este problema, de acuerdo con Jaccheri, se debía a que la industria está conformada en su mayoría por compañías pequeñas que no cuentan con un departamento de calidad o mejora, y suelen resolver este tipo de problemas sin alguna metodología formal. Por otro lado, las empresas grandes no aceptaban que un grupo de estudiantes observara sus manuales de calidad, o las descripciones de sus estándares y procesos.

En base al problema anterior, la investigación de Jaccheri presentó tres alternativas para trabajar colaborativamente con la industria:

- La primera planteaba que los estudiantes trabajasen dentro de las empresas en un departamento de calidad y procesos, establecido específicamente para el curso.
- La segunda establecía que los estudiantes visitaran a las compañías y realizaran entrevistas estructuradas, aplicaran cuestionarios, o que se les diera acceso a cierta documentación para aprender sobre sus iniciativas de calidad.
- La última alternativa definía que algunos representantes de las compañías visitaran la universidad para compartir su experiencia con los estudiantes a través de la presentación de casos de estudio. Dadas las condiciones de las compañías participantes, se decidió implementar esta última alternativa.

Así, Jaccheri diseñó un modelo en el que los estudiantes participaran activamente y produjeran cierta documentación y presentaciones que reflejaran su nivel de comprensión en cuanto a las iniciativas de mejora realizadas en las compañías. Considerando como base el curso SQPI [Dingsøyr, 2000], en relación a la formulación de preguntas fundamentales que son evaluadas a lo largo del curso, al análisis de código C++, y a la aplicación de una evaluación final, se propusieron los siguientes cambios:

A. Introducir las presentaciones de las compañías sobre sus iniciativas de mejora. Sin embargo, solamente Ericsson Norway participó en esta actividad a través de tres representantes: el gestor de calidad, el líder del grupo de procesos, y un jefe de proyectos.
B. Introducir las presentaciones de los estudiantes para evaluar el desarrollo de los cuatro ejercicios establecidos para el curso.
C. Modelar los procesos de las compañías participantes no solamente utilizando el lenguaje E3, sino también con IDEF0.
D. Introducir cuatro preguntas clave para que los estudiantes las respondieran a lo largo del curso mediante el trabajo con los representantes de las compañías. Estas preguntas se resumen de la siguiente manera:
    a. Describe un sistema software incluyendo sus atributos de calidad.
    b. ¿Qué procesos existen alrededor de este sistema?
    c. ¿Cuáles son las iniciativas de mejora que se han realizado sobre estos procesos?
    d. ¿Qué tan importante es el proceso software y las iniciativas de mejora relacionadas con él en el contexto general de la compañía?

Los cambios A y B fueron clasificados como cambios correctivos, mientras C y D como cambios de mejora. El modelo que representa un proceso real fue más motivador que el que representaba un proceso ficticio, incluso para los profesores.

2.4.2.3. Resultados alcanzados

De acuerdo con Jaccheri, la nueva formulación de las preguntas clave fue mejor que aquellas que se diseñaron en el modelo presentado en [Dingsøyr, 2000] puesto que éstas se relacionaban con un proceso software en concreto y los documentos generados por los estudiantes tuvieron mejor

calidad, mostrando su comprensión sobre la calidad e iniciativas de procesos, y resaltando las discrepancias encontradas durante las presentaciones de las respuestas a los diferentes representantes de las compañías.

Para contrastar la información, en el examen escrito se requirió que los estudiantes contestaran las cuatro preguntas clave establecidas.

2.4.3. Aplicación del enfoque constructivista a la enseñanza de la evaluación y mejora del proceso de software

La investigación de von Wangengeim y Hauck establece un enfoque constructivista para un curso de Ingeniería de Software de nivel maestría en la universidad UNIVALI de Brasil, el cual es apoyado por la relación existente entre academia e industria de software [von Wangenheim & Hauck, 2010]. Este curso se centra específicamente en involucrar a los estudiantes en proyectos reales de mejora de procesos de software. En base a los problemas que se presentan con empresas renuentes a compartir sus procesos o experiencias sobre calidad, y que además no tienen la disponibilidad para asignar personal para trabajar en un conjunto de actividades con los estudiantes, se diseñó un curso ligado con la industria para aprender los conceptos de mejora. Actualmente este modelo se ha aplicado en otras áreas de estudios y se han obtenido resultados positivos. De acuerdo a sus creadores, el curso diseñado tiene la ventaja de ofrecer un entorno estimulante para enseñar habilidades como la comunicación y hacer más motivadora la experiencia de trabajar con la Ingeniería de Software.

2.4.3.1. Propósito del curso

El propósito de esta investigación se basó en diseñar un curso que permitiera enseñar a estudiantes de posgrado a definir y documentar los procesos software, evaluar la capacidad y/o madurez de los procesos software y áreas específicas de la Ingeniería de Software (como la gestión de proyectos), entender y aplicar sus habilidades de comunicación, trabajo en equipo, liderazgo y resolución de problemas. Para alcanzar estos objetivos, el curso diseñado propone un enfoque educativo basado en la teoría constructivista a través de la integración de un entorno simulado donde se desarrolla un proyecto que se realiza en paralelo al curso impartido en la universidad.

2.4.3.2. Desarrollo del curso

El curso pretende que los estudiantes comprendan los conceptos referentes a la Mejora y Evaluación del Proceso de Software a través de su aplicación práctica. Así, éste está diseñado tanto teórica como prácticamente, y se ve apoyado con exposiciones introductorias y discusiones sobre lecturas relacionadas con diversos temas, como conceptos básicos sobre los modelos de proceso y de evaluación como IDEAL [McFeeley,1996], ISO/IEC 15504:2004 [ISO/IEC, 2004], CMMI-DEV v1.2 [CMMI, 2006], ISO/IEC 12207 [ISO/IEC, 2008], MPS.BR [Rocha et al., 2005], SCAMPI [SCAMPI, 2006], y es complementado con exámenes y trabajos en equipo.

A través del enfoque constructivista, el curso proporciona experiencia a los estudiantes para ayudarles a comprender y aplicar los conceptos y enfoques teóricos en la práctica. La exposición de lecturas se reduce al mínimo, puesto que se considera que éstas son esenciales para establecer únicamente un marco introductorio de los conceptos que se aprenderán y serán aplicados en los proyectos durante el curso. De esta forma, los proyectos son desarrollados por equipos de estudiantes quienes deben analizar entornos simulados de mejora, y seguir las instrucciones de sus profesores para estudiar las características de la organización que será evaluada (para contextualizarse), decidir sobre el modelo a utilizar, etc. Este enfoque hace necesario que para que los estudiantes aprendan y se vean involucrados en una situación altamente organizativa y en la

definición de un nuevo proceso de software, el profesor deba mantener un entorno ideal para el aprendizaje activo, responder a todas las preguntas de los estudiantes, proporcionar material adicional, fomentar la reflexión entre los equipos, y promover la discusión entre los estudiantes sobre sus resultados.

El curso se divide básicamente en tres partes que se resumen a continuación:

- La primer parte se enfoca en la evaluación de los procesos. Este trabajo práctico tiene una orientación administrativa para relacionarse con los diferentes conceptos y enfoques relacionados con la gestión de un proceso de software. En este sentido, el curso utiliza una versión reducida de SCAMPI para establecer un proceso de evaluación que incluye plantillas y predefine una descripción del proceso. Durante esta actividad los estudiantes analizan las actividades de la gestión de proyectos enfocándose en el Nivel de Capacidad 2 del área de proceso de *Planificación del Proyecto* que establece el CMMI-DEV 1.2. Así, los estudiantes realizan la evaluación durante el curso considerando las características, competencias y roles de los proyectos de una organización participante. Basándose en la información recogida, los estudiantes identifican y documentan todas las evidencias directas e indirectas del proceso evaluado. Estas evidencias son documentadas en una hoja Excel que resume los resultados obtenidos para cada práctica genérica y específica del área de proceso evaluada. Más tarde esta información es analizada y presentada en un reporte de resultados de la evaluación, indicando las mejoras recomendadas por los estudiantes. El curso está diseñado también para que un grupo de estudiantes, con ayuda del profesor, caracterice las actividades, competencias y roles de una organización. Esto en caso de que en un momento determinado no esté disponible alguna organización y se retrasé la práctica.
- La segunda parte se enfoca en la definición de un proceso de software en base a los resultados obtenidos durante la evaluación. Cada equipo de mejora define un proceso de software para la organización participante ajustando las prácticas requeridas en el Nivel de Capacidad 2 establecido por el CMMI-DEV 1.2 para el área de *Planificación del Proyecto*. Después de analizar los procesos de la organización, considerando las prácticas que establece el modelo, se documenta un nuevo proceso describiendo los objetivos, actividades, métodos, técnicas y herramientas que deben ser usadas durante la mejora, incluyendo plantillas y productos de trabajo que tendrán que generarse en la práctica, además de la identificación de roles y responsabilidades. Todos los procesos son modelados con Enterprise Architect [URL-4] y se ajustan a las practicas específicas que define el CMMI-DEV v1.2.
- La última parte del curso se enfoca a que los estudiantes aprendan a gestionar las fases del desarrollo software utilizando las plantillas definidas en la segunda parte. Durante las fases definidas, los estudiantes recopilan datos sobre el tiempo y esfuerzo invertido en las actividades con el fin de realimentar su conocimiento.

Los objetivos establecidos durante la evaluación del curso se resumen como:

1. Analizar si existe un efecto positivo de aprendizaje, al recordar, entender y aplicar las habilidades adquiridas en el curso.
2. Analizar si el diseño del curso es apropiado en términos del método de enseñanza, el trabajo desarrollado y la integración con la industria.
3. Determinar las fortalezas y debilidades del curso.

Las preguntas de investigación analizadas se basan en el modelo de evaluación Kirkpatrick [Kirkpatrick & Kirkpatrick, 2006] que comprende cuatro niveles.

2.4.3.3. Resultados alcanzados

En general se obtuvo una realimentación positiva de los estudiantes de maestría con respecto al curso. En cuanto al objetivo central de su diseño, los resultados obtenidos reflejaron que los estudiantes creen que el curso les ayudó a obtener habilidades importantes como el trabajo en equipo, la resolución de problemas y la comunicación. La evaluación del segundo objetivo se basó en una escala de aceptación tipo Likert del 1 al 7, todas las respuestas se concentraron en los niveles más altos asociados a las respuestas "Totalmente de acuerdo" y "De acuerdo". Las fortalezas identificadas en el tercer objetivo fueron las siguientes:

- Las clases teóricas están totalmente relacionadas con el trabajo práctico.
- Se hace fuerte énfasis en el trabajo práctico.
- Es motivador realizar el trabajo práctico en combinación e interacción con la industria.
- Es gratificante presentar y discutir los resultados en clase.

Algunas debilidades que se detectaron son las siguientes:

- Se cuenta con grupos reducidos de estudiantes en cada curso.
- Escases de ejemplos reales cuando las empresas no tienen tiempo de participar.
- Uso inadecuado del tiempo reservado para las prácticas en clase.

2.4.4. HEPALE!: Proyecto e-learning para enseñar la mejora de procesos de software en entornos industriales pequeños

Esta propuesta difiere un poco de las analizadas anteriormente en el sentido de que ésta determina que existe también la necesidad de que las empresas pequeñas y los entornos pequeños de desarrollo de software comprendan mejor las características de los modelos de proceso que están siendo adaptados a sus necesidades. Estos modelos, que teóricamente simplifican de manera considerable las prácticas y tareas recomendadas para cualquier organización pequeña, requieren de tiempo para ser institucionalizados en estas organizaciones. En este sentido, para facilitar la adaptación de estos modelos, se ha creado un paquete de implementación llamado HEPALE! [Mendoza et al., 2009], el cual está diseñado para adaptarse a modelos como COMPETISOFT [Oktaba et al., 2007]. Los elementos que conforman a esta herramienta son los siguientes:

- Descripción de procesos (actividades, tareas, roles y productos).
- Plantillas.
- *Checklists*.
- Herramientas de soporte.
- Relaciones entre modelos y estándares.

2.4.4.1. Propósito del curso

Para facilitar la adopción de las prácticas eficientes y mejorar la distribución de conocimiento sobre los estándares y conceptos, la herramienta HEPALE! utiliza paquetes de implementación que

se adaptan a un modelo de referencia enfocado en la mejora de procesos. Así, ante la falta de plantillas que hagan más fácil la implementación de los procesos de Administración de Proyectos y Desarrollo y Mantenimiento de Software, se han desarrollado una serie de plantillas basadas en RUP [Barnes, 2007] y PMBoK [PMI, 2009] que se pueden adaptar a cualquier modelo de referencia. Para hacer la adaptación de los modelos referencia a los pequeños entornos, se pretende utilizar los paquetes de implementación adoptando el estándar IMS para *e-learning* [URL-5]. Los autores están convencidos que al adaptar métodos como *e-learning* y los Objetos de Aprendizaje (LO, *Learning Objects*) [Beck, 2008] definen una excelente estrategia para el acercamiento instruccional a los modelos de referencia. De acuerdo con [Nichols, 2008], el *e-learning* combina el aprendizaje a distancia, el acceso a contenido, la flexibilidad de calendarios y la reducción de costos para mejorar la enseñanza, mientras que los LO ofrecen una nueva perspectiva de aprendizaje y enseñanza, proveyendo información reusable e interoperabilidad de los diferentes Sistemas de Gestión del Aprendizaje (LMS, *Learning Management Systems*) que ayudan a gestionar las fases de entrenamiento. HEPALE! tiene el propósito de difundir y extender los modelos de proceso mediante la combinación de todos estos conceptos, además de introducir un repositorio y un LMS propio que aborda estándares internacionales.

2.4.4.2. Desarrollo del curso

Las características más importantes que guiaron el diseño de un curso bajo el enfoque de HEPALE! se pueden resumir como:

- *Consideraciones generales*. Las primeras consideraciones que se presentaron se relacionaron con la estructura del repositorio de datos y como se guardarían sus metadatos, así los paquetes consistirían de archivos XML que por su estructura son fáciles de mantener y permiten describir cualquier tipo de contenido para que sea leído desde cualquier sistema operativo. Los datos correspondientes a los LO pueden ser archivos de texto, páginas HTML o incluso imágenes que corresponden a descripciones, explicaciones, definiciones, etc., de cada modelo de proceso. Concretamente, la creación de los LO se realizó mediante la herramienta *OpenSource* llamada eXe, la cual permite crear contenido multimedia basado en web y está diseñada para crear contenido académico a través de la creación de guías, recursos y LO. De esta forma, el contenido multimedia fue creado de acuerdo al estándar ISO/IEC 29110-5-1 (véase Figura 2.12).
- *Descripción de las plantillas.* Una cuestión importante, identificada en el contenido de los paquetes de implementación, fue la necesidad de proveer varias plantillas que faciliten la adopción de algún modelo de proceso. De acuerdo a las actividades que recomiendan los modelos COMPETISOFT e ISO/IEC 29110-5-1, se propusieron plantillas como una forma alternativa de generar un producto asociado con una actividad, y a la vez se decidió que éstas estuviesen basadas en las metodologías RUP y PMBoK. La principal contribución en este sentido es la modularidad que ofrecen las plantillas puesto que facilitan la adaptación del paquete de implementación. Cada plantilla contiene una descripción de cómo rellenar su contenido y de cómo crear el producto que va a ajustarse a los requerimientos de la actividad. La Figura 2.13 muestra que los productos pueden ser creados siguiendo el orden de las actividades presentadas en el modelo de referencia. Así, el diagrama asociado a cada plantilla muestra, por ejemplo, los roles involucrados a través de las columnas y la actividad asociada con el formato *A.2.#*.

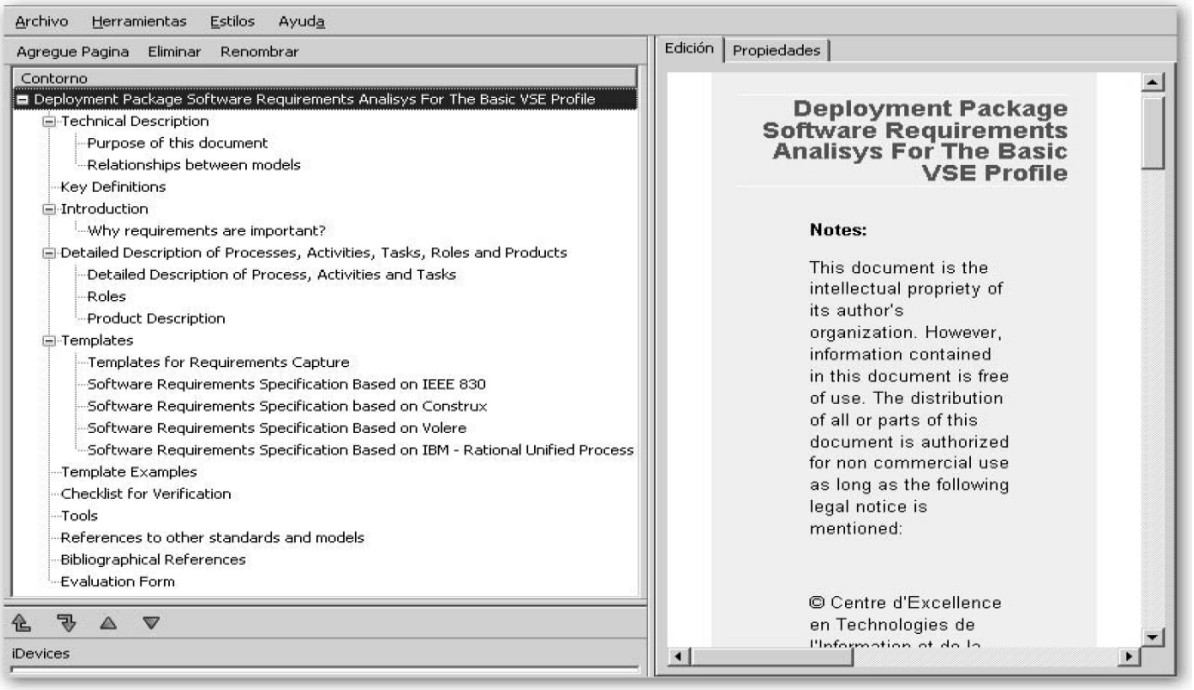


Figura 2.12. Ejemplo de index.html en eXe [Mendoza et al., 2009]

- *Arquitectura*. Una vez establecidos los paquetes definidos para el proyecto se procede a establecer una arquitectura, en la Figura 2.14 se muestra el diagrama de despliegue donde se observan todos los componentes involucrados en el sistema. Este diagrama evidencia el uso del protocolo SOAP y de *Web Services* (WS) que representan un paquete que almacena todas las especificaciones en formato XML y HTML. Es bien sabido que una de las ventajas del uso de WS a través del protocolo SOAP es que facilita la interacción con sistemas externos, independientemente de la plataforma en la que corran y el lenguaje de programación con el que hayan sido creados, permitiendo así el intercambio de información. La arquitectura incluye también una base de datos y un módulo que permitirá administrar y visualizar los contenidos de la herramienta a través de un navegador web.
- *Descripción del repositorio*. El repositorio puede ser utilizado por tres tipos diferentes de clientes: usuarios, implementadores del modelo, y aprendices de sistemas de gestión. El implementador del modelo es el perfil al que se le permite agregar, modificar y borrar contenido de los paquetes, los cuales deberán estar visibles para búsquedas futuras de contenido. Dichas búsquedas están filtradas primeramente por el contenido del paquete y posteriormente por autor, título o nombre del contenido. La diferencia con un perfil de usuario y uno de aprendiz es la forma en que los WS son utilizados, es decir, estos usuarios solamente necesitan ejecutar los servicios y que los datos les sean presentados en un formato estándar, mientras que el otro tipo de usuario tienen acceso directo a los WS.
- *Construcción del LMS*. El LMS es el componente con el que los empleados interactuarán durante su experiencia de aprendizaje, en él se concentran las características que facilitarán el proceso para ayudar tanto a empleados como a implementadores a adquirir el conocimiento sobre los modelos de proceso. HEPALE! incluye un módulo que permite la búsqueda de contenidos de aprendizaje y la conexión con diferentes sistemas haciendo más eficientes las búsquedas. Además, proporciona información útil desde dos puntos de vista,

(1) un análisis cualitativo basado en el estándar SCORM (*Sharable Content Object Reference Model*) [URL-6] y que se relaciona con la experiencia de aprendizaje gracias a su compatibilidad con LMS; y (2) un cuestionario que ayuda a determinar el nivel de aceptación de los empleados sobre el componente utilizado. Además de lo anterior, se genera una evaluación cuantitativa a través de Google Analytics [URL-7] que provee información estadística sobre los accesos a la web y proporciona un perfil de los usuarios y su interés sobre los diferentes módulos del sistema.

- *Tecnología*. Por último, la lista de herramientas utilizadas para la creación del HEPALE!, incluye, en su mayoría, software *OpenSource*: Grails 1.1, Development Kit Java J2EE 5SDK, JQuery 1.3, Postgresql 8.3, Google Analytics, Metro 1.3, SAML-Security Assertion Markup Language, Sun Glassfish App Server 2.1, Netbeans 6.5.1, Enterprise Architect 6.5.806.

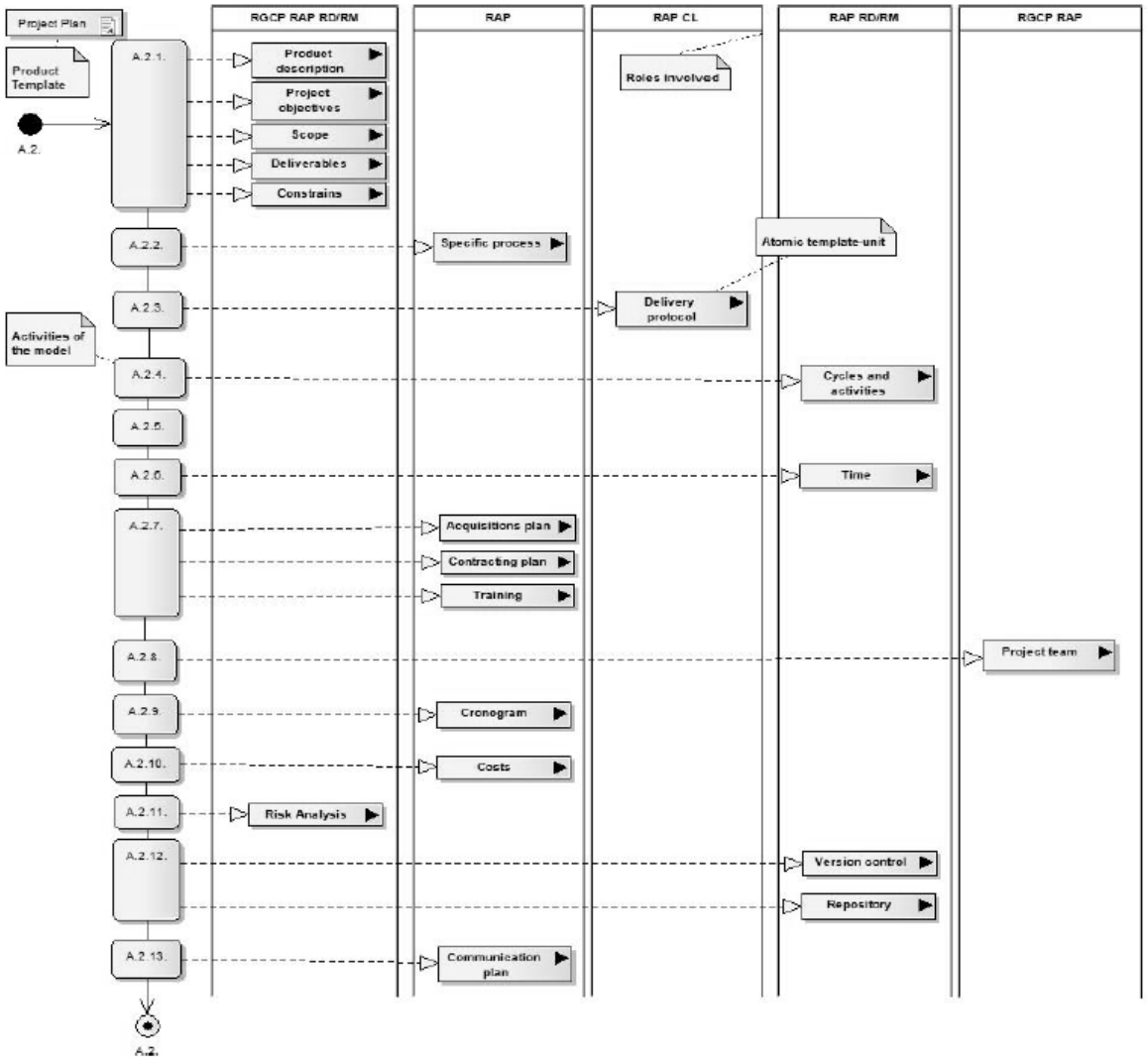


Figura 2.13. Diagrama asociado con la plantilla de Plan de Proyecto [Mendoza et al., 2009]

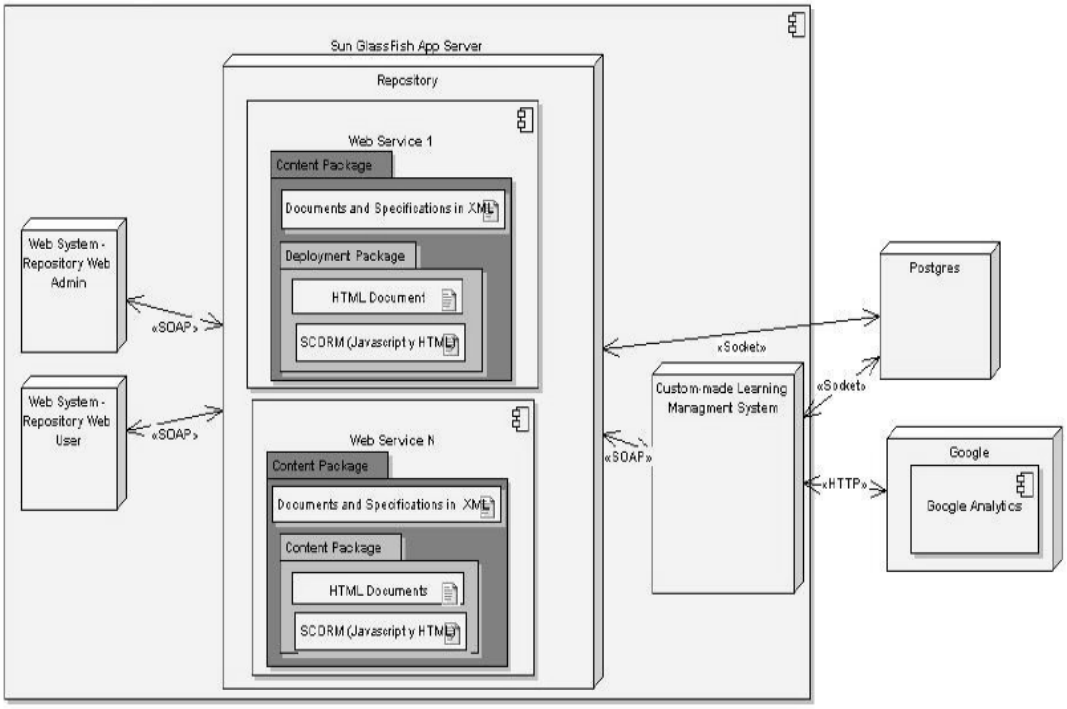


Figura 2.14. Arquitectura de HEPALE! [Mendoza et al., 2009]

Así, mediante eXe se estructura el contenido de los paquetes en base a los estándares mencionados anteriormente, se indexa y almacena en el repositorio para su posterior consulta por los empleados (véase Figura 2.15). De esta manera es que los empleados pueden recibir información sobre la mejora de procesos (véase Figura 2.16).

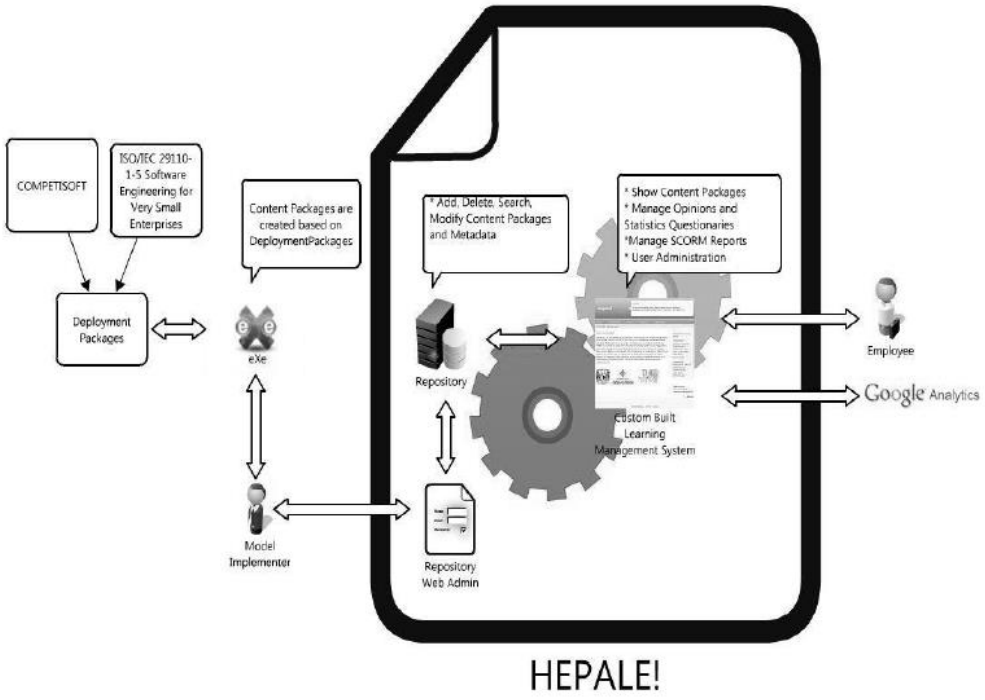


Figura 2.15. Flujo del paquete de contenido con HEPALE! [Mendoza et al., 2009]

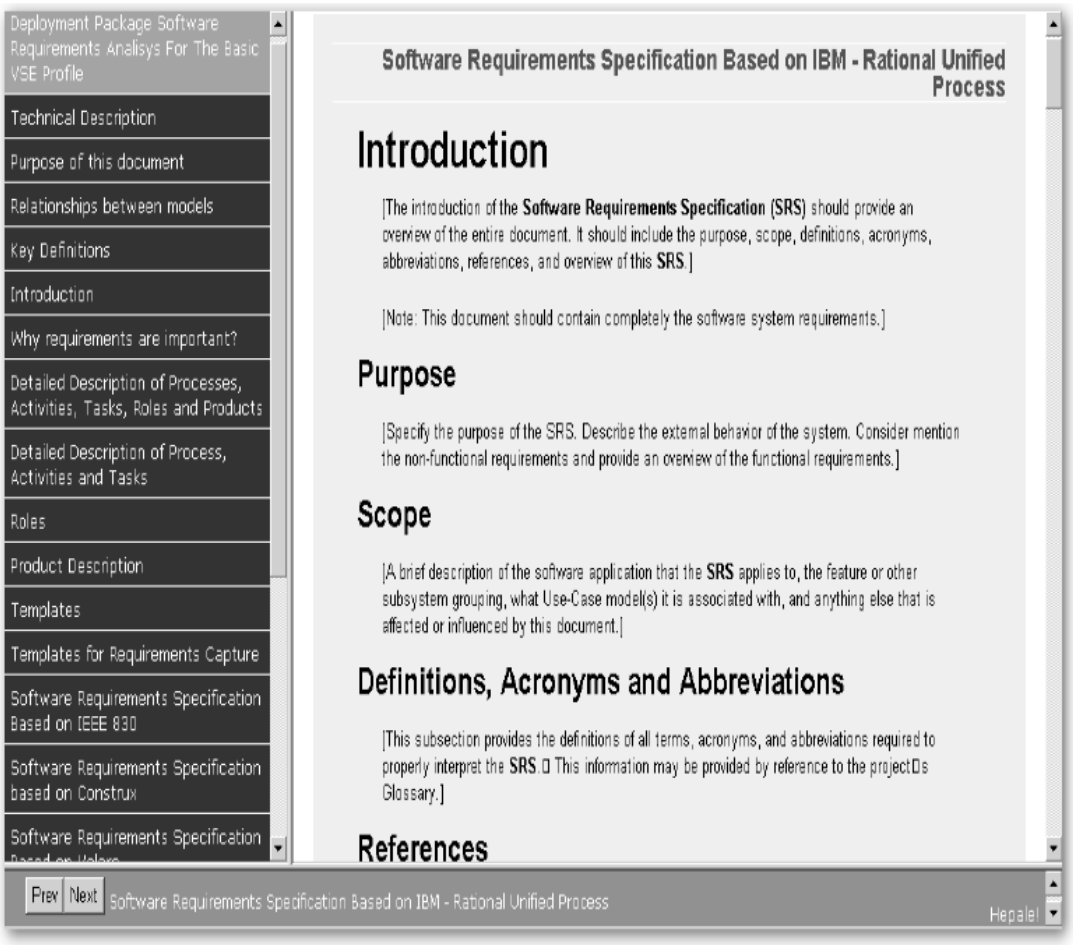


Figura 2.16. Ejemplo de presentación de información al empleado con HEPALE! [Mendoza et al., 2009]

2.4.4.3. Resultados alcanzados

De acuerdo con la investigación de Mendoza et al., la formación sobre temas relacionados con la Mejora del Proceso de Software, en el contexto de las pequeñas empresas, es un tópico bastante abandonado por los investigadores relacionados con la Ingeniería de Software. En este sentido, el desarrollo de HEPALE! promueve el desarrollo de habilidades entre los empleados de cualquier organización, definiendo nuevas prácticas para su trabajo diario y recomendando herramientas de soporte para mejorar su rendimiento desde un enfoque de aprendizaje y participación. En el caso de la herramienta presentada, los autores consideran que se ha cumplido con estos aspectos y, que además, la incorporación del concepto de *e-learning* introduce una alternativa para distribuir conocimiento sobre los diferentes modelos de proceso existentes de una manera fácil y práctica, y al mismo tiempo representa una contribución importante para el área de Mejora del Proceso de Software. Otros resultados derivados de esta investigación mencionan lo siguiente:

- Se ha podido demostrar que cualquier modelo de proceso puede ser descrito, adaptado y difundido en los entornos pequeños de desarrollo.
- La estructura basada en módulos de plantillas permite una fácil adaptación de cualquier modelo (particularmente los usados en esta investigación, COMPETISOFT e ISO/IEC 29110).

- La creación y aplicación de guías y plantillas reduce considerablemente el tiempo invertido en la comprensión de cada fase de un proceso.

2.4.5. Usando TSPi y PBL como apoyo a la educación de la Ingeniería de Software en un curso universitario

De acuerdo al trabajo de [García & Pacheco, 2012] la necesidad de educar al ingeniero de software reside en la importancia de enseñarle a diseñar un sistema, crear componentes de código, probar sus componentes y además, supervisar a los programadores, *testers*, evaluadores de calidad y especialistas en gestión de configuración, que son los que desarrollan el producto final. En base a esto, es necesario que el estudiante identifique y se relacione con los roles que están involucrados dentro de un proyecto de software, y que desempeñé al mismo tiempo un rol en la gestión de proyectos.

2.4.5.1. Propósito del curso

A través de los años, la literatura especializada ha identificado la necesidad de que el alumno adquiera experiencia técnica durante su fase de aprendizaje en la universidad. En este sentido, los profesores deben proveer un conjunto de técnicas, herramientas y conocimientos que garanticen el desarrollo de las habilidades prácticas en el desarrollo y gestión de un proyecto de software. Sin embargo, los estudiantes ganan poca experiencia puesto que los cursos de Ingeniería de Software están enfocados en la teoría, lo que muchas veces ocasiona que los conceptos aprendidos en las aulas sean difíciles de aplicar en ejercicios reales.

Continuando con la revisión de propuestas relacionadas con el soporte al proceso de enseñanza/aprendizaje de la Mejora del Proceso de Software, la propuesta de [García & Pacheco, 2012] comparte la idea de que para que los estudiantes desarrollen experiencia y habilidades prácticas, es necesario vincular el curso con la industria de software o desarrollarlo en un entorno simulado. Así, el principal objetivo de esta propuesta es proporcionar un enfoque que combine una metodología de trabajo en equipo que fortalezca las habilidades de los estudiantes para gestionar proyectos de software (TSPi) y un método de enseñanza que fortalezca las habilidades de los estudiantes a través de problemas reales (PBL). Como herramienta de soporte a este enfoque integrado TSPi/PBL se propone el desarrollo de un entorno computacional de trabajo, la Plataforma de Trabajo en Equipo para la Educación de la Ingeniería de Software, que promueve la práctica a través de la interacción entre los conceptos teóricos y experiencias con la industria local de software.

2.4.5.2. Desarrollo del curso

Para que los estudiantes puedan trabajar eficientemente en equipos deben aprender primero que la disciplina individual es el requisito esencial para completar las diferentes actividades que se presenten. De acuerdo a los aspectos humanos de la Ingeniería de Software [Tomayco & Hazzan, 2004], una manera de enseñar la importancia de la disciplina a los ingenieros jóvenes es involucrarlos en experimentos o proyectos reales en colaboración con un entorno profesional o en entornos académicos. En entornos reales los alumnos reconocen sus malos hábitos y deficiencias al trabajar en equipo, y éstos pueden ser eliminados tempranamente si el curso incorpora revisiones de proyectos entre los estudiantes dentro de un entorno colaborativo, además de que se potencian las habilidades sociales como la comunicación.

En este sentido, TSPi [Humphrey, 2000] es un marco de trabajo definido por el SEI orientado a establecer prácticas eficientes para el trabajo en equipo en cursos universitarios sobre la Ingeniería de Software. TSPi es una versión reducida con enfoque educativo del TSP y está orientado a establecer un proceso y principios de trabajo dentro de un entorno de equipo, al mismo tiempo

proporciona las bases para la adquisición de la experiencia relacionada con la planificación y gestión de los proyectos de software. De acuerdo con Watts Humphrey, TSPi se basa en los siguientes principios:

- El aprendizaje es más efectivo cuando los estudiantes siguen un proceso definido y obtienen rápidamente realimentación sobre su trabajo. Las guías y formatos de TSPi proveen un marco de trabajo definido, medible y repetitivo para los equipos que lo utilicen. Además, se obtiene una rápida realimentación dado que introduce ciclos de desarrollo que evalúan y reportan los resultados obtenidos.
- Un equipo productivo de trabajo requiere la combinación de metas específicas, un entorno propicio de trabajo, liderazgo y entrenadores capaces. TSPi provee un entorno propicio donde un estudiante juega el rol de líder y el profesor proporciona el entrenamiento necesario.
- Cuando los estudiantes identifican problemas en el proyecto y son guiados para proporcionar soluciones efectivas, ellos aprecian los beneficios de las prácticas sólidas de desarrollo. En este sentido, sin la guía precisa de TSPi los estudiantes pueden gastar tiempo importante definiendo sus propias prácticas, roles y métodos.
- La instrucción es más eficaz cuando se basa en los conocimientos previos. Existe una buena cantidad de experiencia resumida sobre la formación de equipos de software y el establecimiento de cursos para éstos. TSPi se fundamenta sobre esta base.

A partir de estos cuatro principios anteriores, el diseño de TSPi involucra siete opciones:

1. Proveer un marco simple de trabajo que se basa en los fundamentos del PSP [Humprey, 1996].
2. Desarrollar los productos en varios ciclos.
3. Establecer medidas estándar para la calidad y rendimiento del equipo/proyecto.
4. Proporcionar medidas precisas para equipos y estudiantes.
5. Usar las evaluaciones de roles y equipos.
6. Implantar la disciplina del proceso.
7. Proporcionar orientación sobre problemas del trabajo en equipo.

En este sentido, a pesar de que el trabajo en grupo puede tener ventajas importantes [Aguinis & Kraiger, 2009; Dubisnky et al., 2010; Akgün et al., 2011], suele presentar también problemas [Moe & Šmite, 2008; Cataldo et al., 2008]. Muchos estudios empíricos han sido desarrollados durante los últimos años (e.g. [Carver et al., 2003; Razmov & Anderson, 2006; Collins et al., 2008; Liu et al., 2009; Mahmood, 2011]), para demostrar que los problemas más comunes que enfrentan los equipos de estudiantes se relacionan con liderazgo, cooperación, participación, dilación, calidad, desviación de uso, y evaluación. Así, el criterio para que un equipo sea exitoso es que el equipo esté claramente identificado, sus tareas sean claras y únicas, los estudiantes tengan control de sus tareas, y exista una necesidad real del equipo. El entorno debe apoyar también al trabajo en equipo, fomentar a los estudiantes a planificar su trabajo, y esperar que éstos mantengan la disciplina individual.

Algunas de las propuestas analizadas en este capítulo de la tesis evidencian una marcada tendencia en el diseño formal de un curso, la incorporación de entornos reales o simulados donde los estudiantes colaboren activamente, y la adopción de principios pedagógicos para lograr un

aprendizaje efectivo. En este sentido, PBL ha sido aplicado exitosamente en muchas investigaciones bajo un contexto educativo [Cavalcanti et al., 2008; Richardson et al., 2011; Chen & Teng, 2011; Debnath & Pandey, 2011], dado que a través de la asignación de un proyecto los estudiantes pueden adquirir experiencia en tiempo real como ingenieros de software. PBL incorpora diferentes técnicas de aprendizaje puesto que primero introduce al estudiante al problema, éste desarrolla el problema y alcanza una solución; así los estudiantes son contextualizados en situaciones del mundo real y se trabaja de forma colaborativa. Algunas de las características de PBL son:

- El aprendizaje es impulsado por problemas desafiantes, con proyectos abiertos, bien definidos y estructurados.
- Los estudiantes generalmente trabajan en equipos.
- Los profesores asumen el papel de "facilitador de aprendizaje".

Así, en la investigación de [García & Pacheco, 2012] la definición de PBL fue contextualizada como: *"un paradigma constructivista de aprendizaje, donde pequeños grupos de estudiantes participan en el aprendizaje cooperativo y la resolución colaborativa de problemas para gestionar, desarrollar y supervisar proyectos complejos y auténticos de software. Estos proyectos persiguen resultados específicos de aprendizaje que están alineados con el proceso definido de TSPi y los objetivos del curso de pregrado"*. De manera concreta, esta propuesta se enfocó en establecer un enfoque integrado para mejorar las habilidades de los estudiantes a través de (1) tomar PBL como referencia para implementar proyectos reales y prácticos en el curso y (2) usar TSPi como un proceso definido para planificar y gestionar tales proyectos. Siento el principal objetivo fortalecer la interacción entre los equipos para obtener mejores resultados usando TSPi.

Para incluir estos elementos en un curso práctico se desarrolló como soporte una herramienta de software, llamada Plataforma de Trabajo en Equipo para la Educación de la Ingeniería de Software, en código ActionScript y MXML de Adobe Flex. Se usó además PHP como enlace con la base de datos y HTML con archivos SWF para que la aplicación estuviera disponible en la web. Esta plataforma proporciona un foro de discusión y módulos de chat para facilitar la comunicación entre cada equipo. También se incorporan módulos para resolver y debatir los conflictos, promover el liderazgo y la autodirección, e incorporar el aprendizaje dirigido. La Figura 2.17 muestra que esta plataforma colaborativa permite el desarrollo de estas habilidades enfocándose en el contenido técnico de la Ingeniería de Software (particularmente la gestión de los proyectos). Los estudiantes pueden iniciar un curso a través del uso de TSPi, promoviendo las aptitudes para la gestión de tiempos y las demás características mencionadas anteriormente. Para la asignación de estudiantes a los equipos, la herramienta dispone de un módulo que ayuda en la tarea a través de criterios como el balanceo de fortalezas y debilidades de los alumnos en cuanto a su desempeño académico evaluado por un examen previo, la participación y habilidades de liderazgo que el profesor haya identificado, etc.

Esta herramienta también posee características que garantizan la interacción entre los miembros de los equipos a través de los roles que identifica TSPi: líder de equipo, gestor de desarrollo, gestor de planificación, gestor de calidad y gestor de apoyo. En la Figura 2.18 se muestra la interacción del líder de equipo, el gestor de calidad y el gestor de apoyo a través del módulo chat para actualizar el Plan de Calidad.

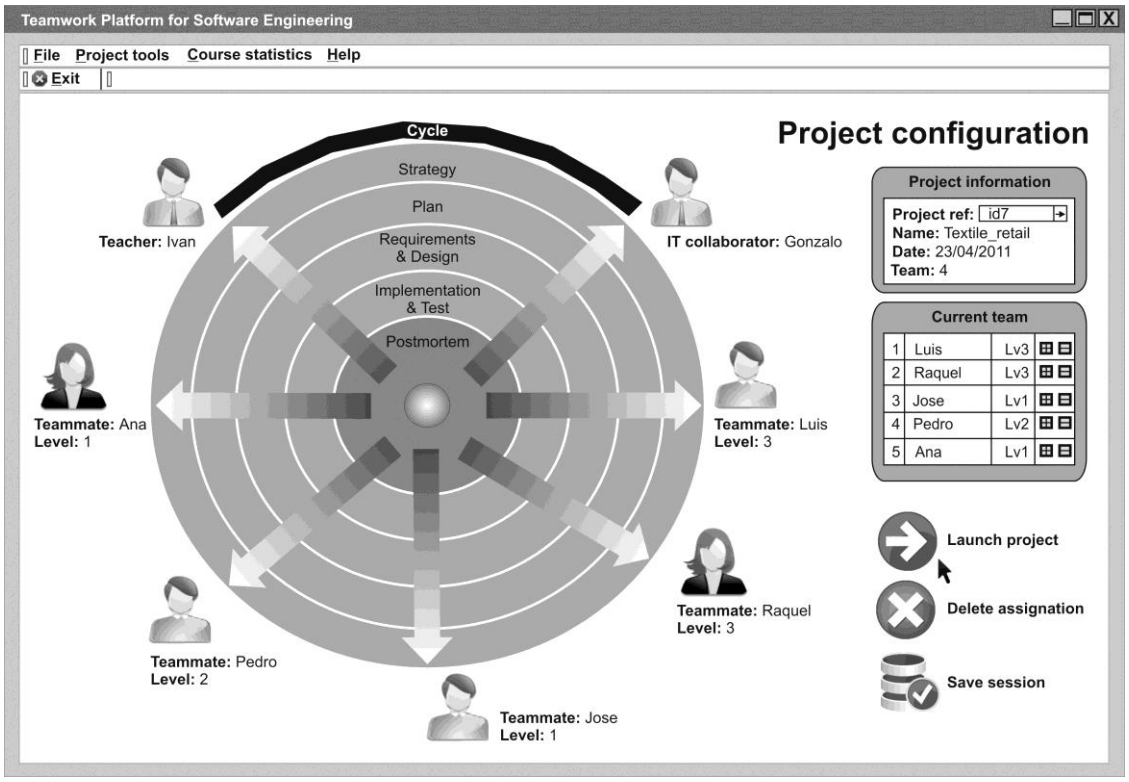


Figura 2.17. Configuración del ambiente de trabajo en equipo [García & Pacheco, 2012]

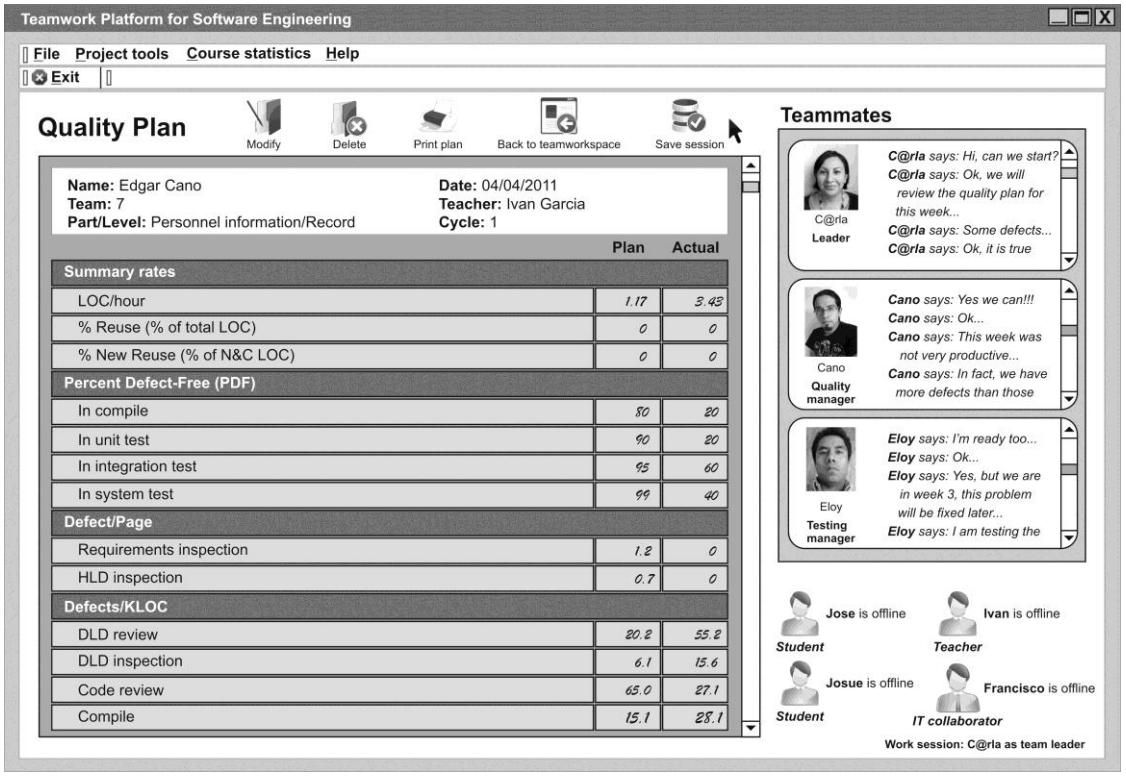


Figura 2.18. Interacción de los estudiantes para alcanzar el plan de calidad [García & Pacheco, 2012]

La herramienta bajo el contexto PBL, en el sentido de la alta interacción de los estudiantes como principal motivador del aprendizaje, tiene como política que una tarea no puede ser completada sin la participación de los miembros del equipo. El desarrollo de los proyectos a través de la herramienta se enfoca a cubrir las fases recomendadas por TSPi (Estrategia, Planeación, Requisitos y Diseño, Implementación y Test, y Postmortem) para resolver los ejercicios planteados a los equipos de trabajo en dos o tres semanas a lo más. De esta forma se pretende realizar un total de cuatro proyectos reales para que el profesor evalúe el desempeño de los estudiantes. En cada uno de estos proyectos, los estudiantes desarrollarán actividades propias de la Ingeniería de Software como selección de un ciclo de vida, estimación, calendarización, elicitación de requisitos, *testing* y más.

2.4.5.3. Resultados alcanzados

La herramienta dispone de un repositorio de problemas reales los cuales son provistos por un conjunto de siete empresas pequeñas de software (de 3 a 25 personas como personal software); todos estos proyectos son dirigidos por un gestor de proyectos que asesora y da realimentación a los estudiantes. Los problemas que son propuestos son modulares y, una vez que los equipos de estudiantes han sido formados, el profesor asignará uno diferente a cada equipo. De esta manera, los estudiantes obtienen habilidades sobre el desarrollo y gestión de proyectos con TSPi y la industria local de software se beneficia con los proyectos realizados y al mismo tiempo colabora en el entrenamiento de personal potencial a través de la especialización en diferentes áreas de la Ingeniería de Software. Finalmente, esta plataforma fue implementada en dos generaciones de estudiantes de Ingeniería de Software quienes desarrollaron un total de 47 proyectos modulares, y en los cuales se midió el esfuerzo, productividad y densidad de defectos que marca TSPi. Adicionalmente se evaluó el desempeño académico de cada estudiante, en la Tabla 10 se pueden observar los resultados.

Tabla 10. Resultados obtenidos para las mediciones de productividad, esfuerzo y densidad de defectos [García & Pacheco, 2012]

Métrica	**2010/2011 (promedio)**			**2011/2012 (promedio)**		
	Ciclo I	**Ciclo II**	**Ciclo III**	**Ciclo I**	**Ciclo II**	**Ciclo III**
Exactitud de la estimación de esfuerzo (%)	-15.26	-3.62	7.58	-36.21	-12.96	2.69
Densidad de defectos (defecto/KLOC)	125.24	72.12	36.0	102.54	54.69	21.36
Productividad (KLOC/h)	10.27	17.54	24.89	7.48	16.53	21.47

2.4.6. Soportando reuniones virtuales de Scrum con equipos distribuidos

En el contexto de los proyectos pequeños, los cuales regularmente sufren muchos cambios, y ante la necesidad de modelos que se adapten a las necesidades de éstos que exigen la reducción del tiempo de desarrollo sin perder de vista la calidad, emergen las metodologías ágiles. Por estar especialmente orientadas para proyectos pequeños, las metodologías ágiles constituyen una solución a medida, aportando una elevada simplificación que, a pesar de ello, no renuncian a las prácticas esenciales para asegurar la calidad del producto. Particularmente, la metodología ágil que más aceptación ha tenido en el mercado es Scrum [Schwaber, 2007] por estar principalmente enfocada en la gestión y seguimiento de los proyectos pequeños. En este escenario, las reuniones juegan un rol fundamental en la comunicación entre los miembros del equipo para despejar impedimentos y conocer el estado de avance, no sólo del proyecto en general, sino de cada desarrollador en

particular. Además, los artefactos utilizados y generados en ellas poseen gran valor para fijar y recordar cuestiones analizadas.

No obstante, uno de los principales problemas de la aplicación de Scrum en los proyectos actuales es la dificultad de realizar reuniones diarias o periódicas. Esto se debe a que últimamente existe una creciente tendencia a que los miembros del equipo de desarrollo de un proyecto de software se encuentren ubicados en diferentes locaciones físicas [Almeida et al., 2012]. Estas condiciones hacen que las reuniones diarias se conviertan en un gasto importante de tiempo y recursos llevando en algunos casos a que su concreción se vuelva inviable.

En este sentido, las interfaces 3D han sido una base popular para el software colaborativo y la visualización de información. De hecho, la metáfora de las salas de reunión ha sido exitosa en varias herramientas, las cuales generalmente representan a los participantes humanos por medio de *avatares*. Así, mediante una simulación de este tipo, se permite que las personas se sientan involucradas en este ambiente e interactúen a través de él. La propuesta de [Rodríguez et al., 2012] plantea un enfoque virtual para dar soporte a las reuniones, logrando simular una oficina de trabajo. De esta manera, y no importando su ubicación geográfica, diferentes miembros de equipos de trabajo se pueden reunir para compartir ideas y opiniones como si estuvieran presentes en un lugar físico.

2.4.6.1. Propósito del curso

La investigación de [Rodríguez et al., 2012] propone una herramienta virtual llamada *Virtual Scrum*, que se desarrolla para dar soporte a reuniones y discusiones en cada parte de un ciclo de vida de Scrum. Se pretende que este enfoque de soporte a la parte de planificación de proyectos, gestión de fallos, calendarización de eventos, visor de documentos para realizar presentaciones, navegador web dentro del mundo virtual y soporte *Daily Meeting* (o las reuniones diarias que promueve Scrum). El objetivo del curso es simular un contexto profesional de desarrollo de software en el cual los estudiantes deben organizarse en grupos de desarrollo y llevar a cabo un proyecto real asignado por la cátedra. Los requisitos para tomar el curso son buenas prácticas de programación, programación orientada a objetos, diseño de software y uso de metodologías ágiles (particularmente Scrum, puesto que los estudiantes desempeñarán los roles de *Team* y *Scrum Master*, mientras que el profesor será el *Product Owner*).

2.4.6.2. Desarrollo del curso

Como apoyo al curso, los autores desarrollaron el *groupware* denominado *Virtual Scrum* para dar soporte a reuniones virtuales necesarias en diferentes etapas del ciclo de vida de Scrum. Como se dijo anteriormente, las reuniones son un aspecto central en el desarrollo de software, y en las reuniones *face-to-face* que se realizan en Scrum, que frecuentemente hacen uso de *post-it notes* sobre un pizarrón, y bocetos en rotafolios hechos con marcadores o escritos en pizarras blancas. Sin embargo, en las reuniones con personal distribuido geográficamente estas herramientas quedan obsoletas. Al igual que las propuestas revisadas anteriormente, Rodríguez et al., plantean la necesidad de utilizar herramientas de alta fidelidad que posean la capacidad de realizar el trabajo en forma colaborativa y distribuida. La Figura 2.19 muestra el entorno programado en *Virtual Scrum*, donde se muestra un escenario que permite realizar reuniones virtuales en un contexto de Scrum. Los cuatro participantes de la figura son miembros de un equipo de Scrum con ubicaciones geográficas diferentes y que están vinculados a través del desarrollo de un proyecto software. Es claro observar que cada desarrollador está representado a través de su *avatar*.

Figura 2.19. Vista al interior de Virtual Scrum [Rodríguez et al., 2012]

El proceso de Scrum inicia con la carga y priorización de *user stories* en el *Product Backlog* (véase Figura 2.20) representado por el *Virtual Product Backlog*. Para ello, es necesario realizar una reunión donde se discutan temas con el cliente para despejar dudas y tener más información del contexto. El color de la *user story* variará entre rojo, indicando que la tarea no ha comenzado (TO DO), amarillo, que indica que la tarea se está realizando (DOING), y verde, indicando que la tarea está lista (DONE).

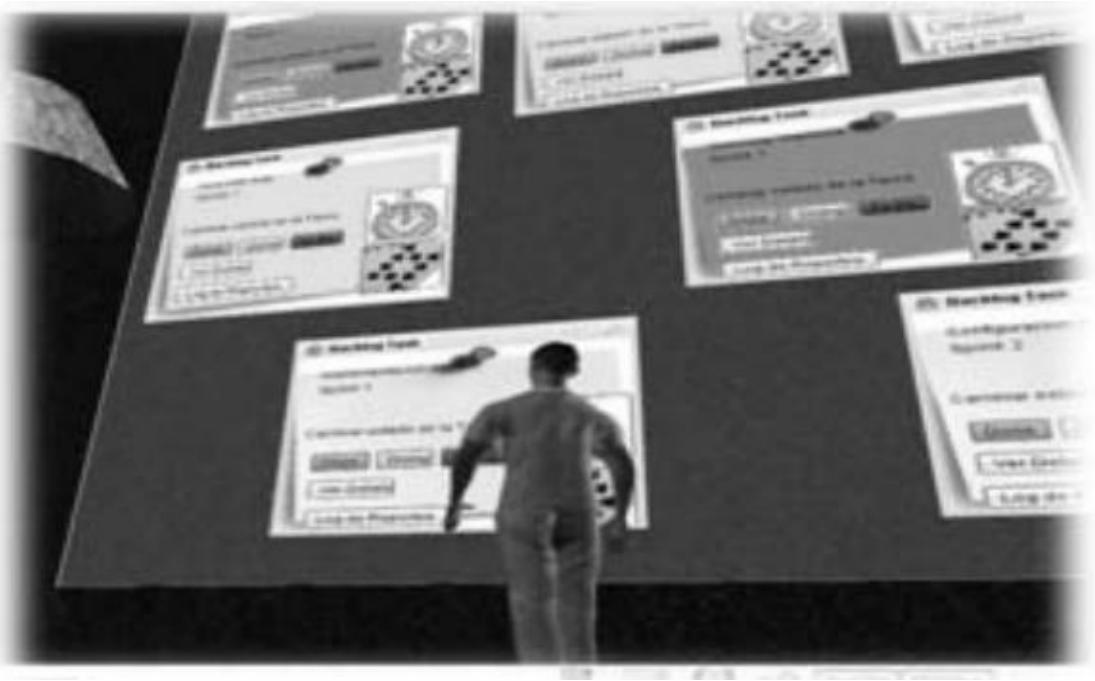


Figura 2.20. Virtual Product Backlog [Rodríguez et al., 2012]

De manera similar, el *Virtual Sprint Backlog* expone las *user stories* inherentes al *sprint* que está ejecutando. Una vez que las *user stories* han sido cargadas, se procede a realizar su estimación. Para realizar las estimaciones, los desarrolladores cuentan con una herramienta basada en la técnica de *Planning Poker* [Cohn, 2005], denominada *Virtual Poker Planning*. En la Figura 2.21 se puede observar la implementación de la técnica de estimación en la cual todos los miembros del equipo votan y ven los resultados en la pantalla compartida. Esta técnica se basa en estimar el valor de esfuerzo o tiempo de realización de cada *user story* mediante la utilización de la serie de *Fibonacci*.

De manera paralela, diariamente los miembros del equipo de Scrum llevan a cabo las *Daily Meetings*. Para dar soporte a esta práctica, este enfoque cuenta con el módulo *Virtual Daily Meeting*. En este sentido, los *Scrum Masters* de cada grupo son los responsables de iniciar las *Daily Meetings* con el soporte de *time-boxing* (15minutos).

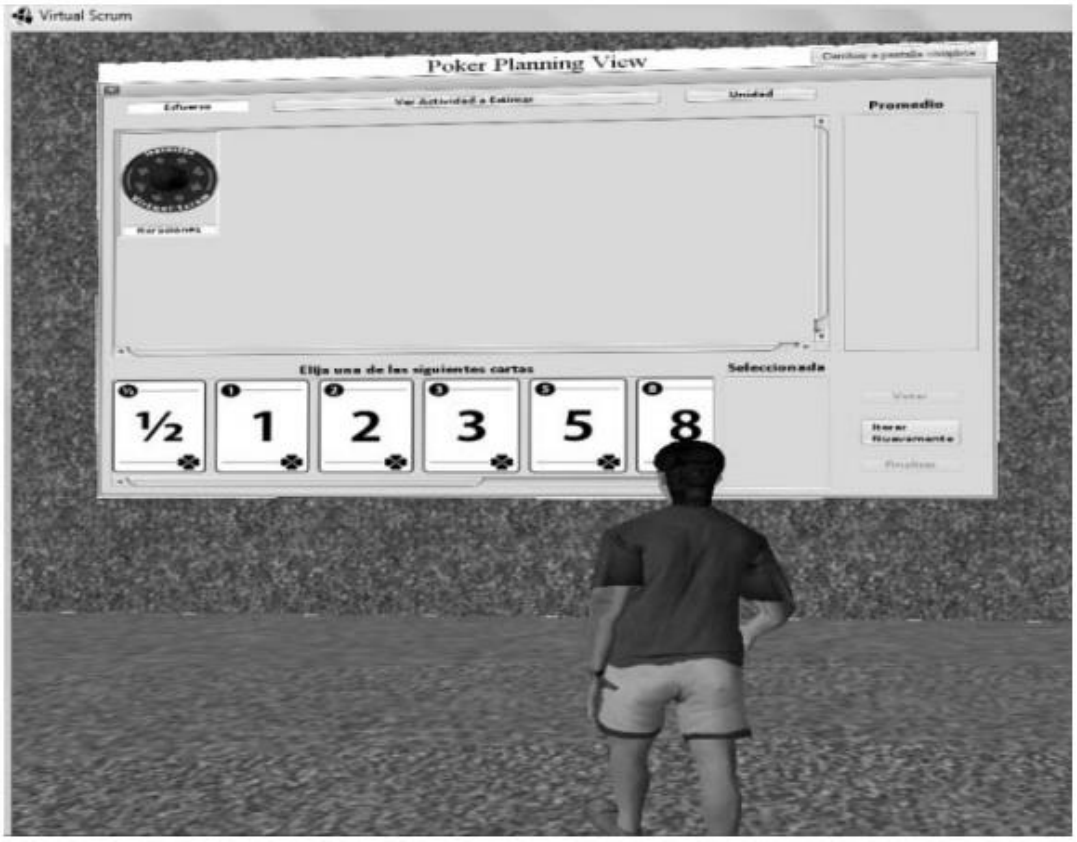


Figura 2.21. Virtual Poker Planning [Rodríguez et al., 2012]

La Figura 2.22 permite apreciar cómo un desarrollador está participando en una *Daily Meeting*. Para esto, existe un temporizador de 15 minutos para contestar a tres preguntas clásicas: ¿Qué hiciste ayer? ¿Qué vas hacer hoy? y ¿Qué ayuda necesitas? Las reuniones y demás eventos pueden ser programados en el *Virtual Calendar* y ser recordados a través de la *Virtual Agenda*. El objetivo es que el marco en el cual se llevan a cabo las reuniones, sea lo más creíble posible para que sea viable el reemplazar las reuniones cara a cara entre los miembros de un equipo, lo cual es bastante costoso para la empresa que busca concretarlas dada la distribución geográfica de los desarrolladores. En una reunión soportada por *Virtual Scrum* es posible utilizar artefactos para dar soporte a las reuniones y hacerlas más interactivas, ya sea para presentar temas o conceptos con el *Virtual PDF Viewer* o realizar búsquedas y consultas en la web mediante el *Virtual Browser*.

Figura 2.22. Virtual Daily Meeting [Rodríguez, 2012]

De acuerdo a los autores, esta herramienta satisface las propiedades presentes en las cuatro categorías que clasifican a las herramientas colaborativas para el desarrollo de software de la siguiente manera:

- Basada en un modelo. Posibilita la visualización de la representación de software (Visualización del *Product Backlog*, *Sprint Backlog* y *Poker Planning*).
- Soporte al proceso. Permite a los desarrolladores visualizar el seguimiento de un proceso (Vinculación y sincronización entre los artefactos de software pertenecientes al ciclo de vida de Scrum).
- Herramienta de conciencia. Incorpora el tener conciencia sobre lo que realizan los demás miembros del grupo a fin de no tener problemas (Visualización de todos los miembros por medio de *avatares*, actualización y comunicación de las acciones realizadas).
- Infraestructura de colaboración. Permite mejorar la interoperabilidad entre las herramientas de colaboración, especialmente la integración de datos (Almacenamiento y registro de acciones realizadas en la herramienta y vinculación de artefactos a proyectos y tareas de un equipo).

2.4.6.3. Resultados alcanzados

Los experimentos se realizaron con 45 estudiantes del curso 2011 de Ingeniería de Software en la carrera de Ingeniería de Sistemas de la Facultad de Ciencias Exactas de la UNICEN en Argentina. El curso tuvo por objetivo simular un contexto profesional de desarrollo de software en el cual los estudiantes debieron organizarse en grupos de desarrollo y llevar a cabo un proyecto real asignado

por la cátedra. En total, la cátedra asignó 54 *user stories* de complejidad similar distribuidas equitativamente entre 6 equipos y que debían ser desarrolladas durante 3 *sprints* de 30 días cada uno. Después de completar las actividades los alumnos rellenaron una encuesta para saber sus opiniones a fin de determinar si la herramienta resultó útil para dar soporte a las *Daily Meetings*. Antes del inicio de los experimentos, los estudiantes participaron de una sesión de entrenamiento donde se identificaron los distintos elementos dentro de *Virtual Scrum*. Luego, los grupos recibieron una lista de requisitos y se utilizaron indicadores usados para análisis del rendimiento del proceso software. Los indicadores son los siguientes:

- Valor ganado. Se refiere al radio entre el decremento de trabajo remanente y la cantidad de trabajo realizado. El valor ideal de este indicador es menor a 1 dado que valores muy superiores a este valor hablan de una planificación pobre.
- Índice de Rendimiento de Calendario. Se refiere al radio entre el valor ganado (de las tareas completadas) y el valor de planificación (la estimación inicial). El valor objetivo de este indicador es mayor o igual a 1. Valores muy superiores a 1 indican que el proyecto terminó antes de lo previsto.
- Índice de Rendimiento de Costo (costos laborales). Se refiere al radio entre el valor ganado y los costos actuales. El valor objetivo es mayor o igual a 1, indicando que el costo de completar el trabajo es justo lo planeado o menos de lo previsto.

Los resultados obtenidos en la experimentación mostraron datos por debajo de 1 para el valor ganado, y datos por encima de 1 para los índices de rendimiento de calendario y costos. Se argumenta que esta situación se presentó puesto que los alumnos aún no tenían conocimiento sobre planificación dada su falta de experiencia y conocimiento en cuanto al uso de ciertas tecnologías. Para recoger las opiniones de los estudiantes sobre *Virtual Scrum*, el estudio siguió el enfoque de Likert usado comúnmente para construir encuestas. Los ítems eran afirmaciones con las que los estudiantes podían estar Totalmente de acuerdo (6), De acuerdo (5), Algo de acuerdo (4), Algo en desacuerdo (3), En desacuerdo (2) o Totalmente desacuerdo (1). En este sentido, se decidió emplear una escala numérica par para capturar mejor las opiniones de los estudiantes, sin tener puntos medios neutrales. La Tabla 11 resume los ítems utilizados en la encuesta con sus resultados. La columna "Promedio" denota el promedio del puntaje Likert obtenido por los 45 estudiantes para el ítem seleccionado. De la misma manera ocurre con la columna "Desviación estándar". La columna (*t*-test *) indica el valor de *t* luego de aplicar el test de *Student* sobre la muestra con un $p < 0.05$. Positivamente, para todos los ítems, los valores de la columna "Promedio" superan el valor 3 ("Algo en desacuerdo"). La última columna de la tabla muestra los resultados del *test de Student* para cada ítem. Este test ha sido usado para determinar cuánto se desvían los estudiantes de la hipótesis nula, la cual expresaba que la actitud de los estudiantes hacia Scrum fue neutral mostrando un valor aritmético promedio del puntaje de Likert de todos los estudiantes igual a 3. Los resultados muestran que 5 de 10 hipótesis fueron rechazadas (valores de *t* resaltados con negrilla). En estos casos, los puntajes de los estudiantes fueron mayores a 3 (con $p < 0.05$); en efecto, es posible aceptar la hipótesis alternativa que los estudiantes tenían una opinión positiva acerca del uso de *Virtual Scrum* en aquellos ítems en donde se rechazó la hipótesis nula. Como consecuencia, fue posible demostrar que la hipótesis que este enfoque revisado es viable para dar soporte a reuniones de software entre grupos de Scrum distribuidos geográficamente.

Finalmente, los estudiantes mostraron cierta inconformidad con los mecanismos de comunicación, puesto que mencionaron que para hacerla más efectiva se debían incorporar módulos de voz por IP y videochat para optimizar la interacción entre los miembros. Definitivamente *Virtual*

Scrum representa una ayuda importante para que los estudiantes aprendan a utilizar esta metodología ágil cubriendo todas las etapas que establece para un ciclo de vida.

Tabla 11. Ítems de la encuestada aplicada a los estudiantes sobre el uso de Virtual Scrum [Rodríguez et al., 2012]

Ítem	Promedio	Desviación estándar	t-test (p)*
1. *Virtual Scrum* es interactivo y permite a los usuarios comunicarse eficientemente.	3.02	1.2	0.12
2. El proceso de desarrollo en *Virtual Scrum* fue coordinado.	3.76	1.3	**3.96**
3. El proceso de toma de decisiones en *Virtual Scrum* fue efectivo.	4.02	1.04	**6.64**
4. *Virtual Scrum* fue efectivo para la discusión de grupo.	4	1.09	**6.19**
5. El entorno de trabajo en *Virtual Scrum* fue cómodo y completo.	3.19	1.02	1.29
6. *Virtual Scrum* es útil para celebrar reuniones de trabajo.	3.13	1.4	0.62
7. *Virtual Scrum* ayudó a que Ud. se sienta más cómodo.	3.3	1.05	1.96
8. Se sintió satisfecho con la calidad del trabajo en *Virtual Scrum*.	3.56	1.22	**3.13**
9. La solución final del *Virtual Product Backlog* satisface la validación.	4.02	1.14	**6.05**
10. La solución alcanzada mediante *Virtual Scrum* satisface la verificación.	3.14	1.2	2.32

2.5. Comparativa empírica sobre las propuestas analizadas

Como principal resultado del análisis entre propuestas similares a la planteada en esta tesis, es posible identificar que se han establecido dos tipos de aportaciones para fortalecer la educación de la Mejora del Proceso de Software: la primera, relacionada con las contribuciones de hace varios años y que proponen la reestructuración completa del plan de estudios para apoyar a las clases teóricas con la realización de ejercicios prácticos vinculados con algunos entornos reales de trabajo; y la segunda, relacionada con las contribuciones más actúales y que se enfocan en la reducción del tiempo destinado a las clases teóricas a través de la introducción de herramientas computacionales para crear entornos virtuales o mejorar la interacción con los entornos reales de trabajo. Así, una comparativa empírica consistiría en analizar la información recogida con los siguientes criterios o características comunes y establecer una línea base con un conjunto básico de propiedades/funcionalidades.

2.5.1. Identificación de características comunes

La Figura 2.23 muestra las características compartidas por RWL [Moore & Brennan, 1995], el Curso de Jaccheri [Jaccheri, 2002], el Curso con enfoque Constructivista [von Wangenheim & Hauck, 2010], HEPALE! [Mendoza et al., 2009], la Plataforma de TSPi & PBL [García & Pacheco, 2012] y Virtual Scrum [Rodríguez et al., 2012], que han sido analizadas anteriormente.

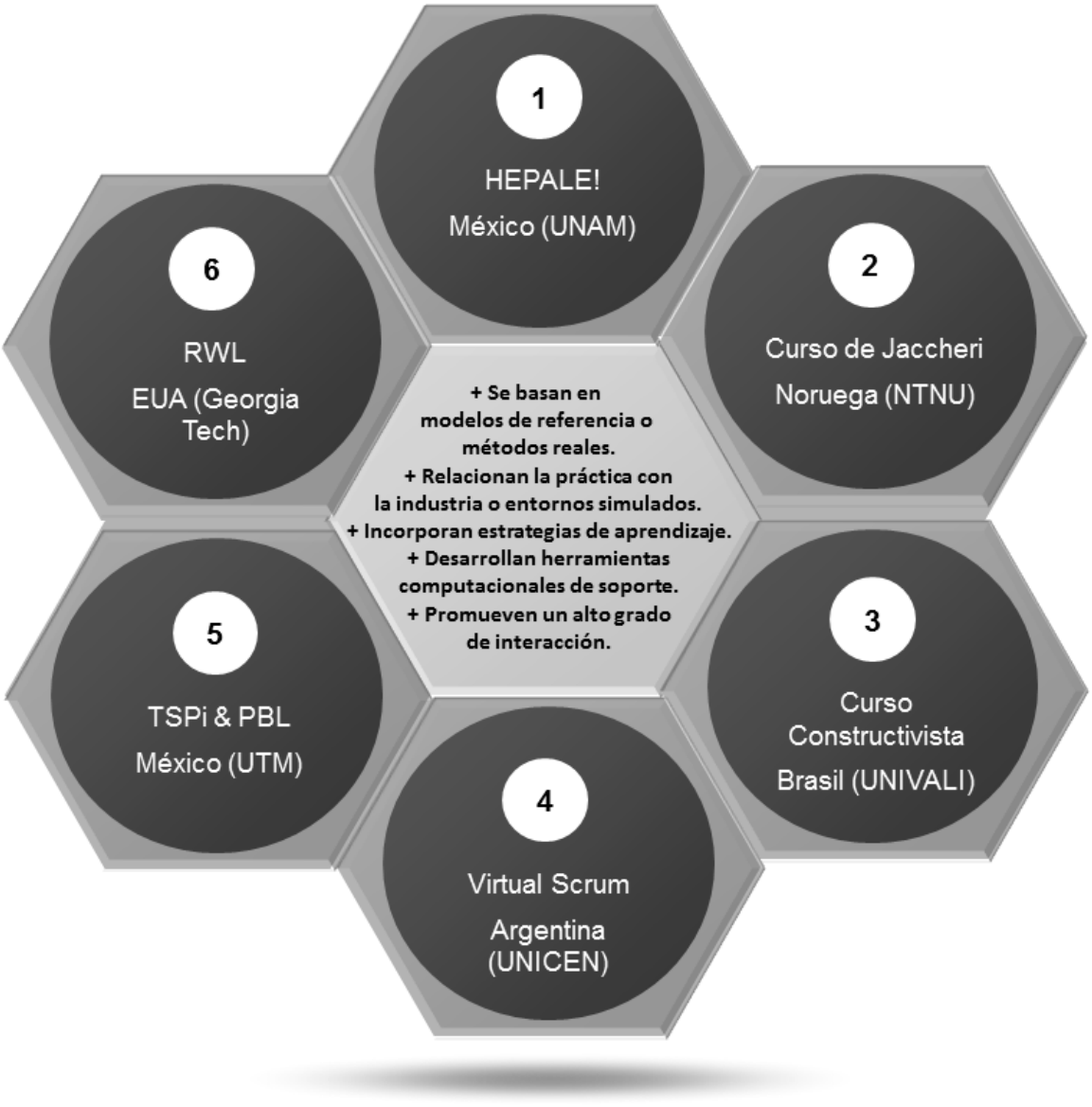


Figura 2.23. Características comunes de las propuestas analizadas

Además de los aspectos resaltados en la Figura 2.23, es importante definir claramente los criterios que permitirán comparar las propuestas analizadas y que se definen como:

- Incorporación de modelos de referencia internacionales al curso teórico. Es importante evaluar la incorporación de estos modelos en el contexto educativo puesto que son de gran importancia dado su nivel de uso y aceptación en diferentes países. La incorporación de estos modelos a un curso relacionado con la Mejora del Proceso de Software contextualiza la información teórica de los libros con la práctica real de la industria local.

- Incorporación de modelos de evaluación al curso teórico. Es imprescindible evaluar la incorporación de modelos de evaluación que en un contexto organizacional, permitirían que los estudiantes determinen las debilidades y fortalezas de las organizaciones en relación a la forma en que desarrollan y mantienen el software.
- Factores para promover la motivación y discusión de los estudiantes. Es necesario evaluar si el enfoque educativo utilizado en cada propuesta introduce tópicos que promueven la motivación para garantizar la participación activa de los estudiantes en la práctica, y la discusión para la generación de su propio conocimiento a través de la experimentación y el debate.
- Fortalecimiento del desarrollo de habilidades "suaves". La necesidad de evaluar cuáles habilidades "suaves", o sociales, de los estudiantes son exploradas a través de las propuestas educativas (habilidades tales como observación, revisión, discusión, presentación oral, presentación escrita, planificación, cooperación, reflexión y el juicio), aporta información en el sentido de cuáles son las características deseadas para que el estudiante se desenvuelva efectivamente en un entorno real de desarrollo de software.
- Introducción de herramientas computacionales que apoyan el diseño y evaluación del curso. Dado que estas herramientas son diseñadas como soporte principal de un curso, puesto que son útiles para mejorar la comunicación entre las personas involucradas en el proyecto o evaluar el desempeño académico de los estudiantes, es necesario determinar qué tipo de herramientas se utilizan y cuál es su aportación más importante a los cursos.
- Incorporación de estrategias pedagógicas. Dado que en el contexto de la tesis se habla de un enfoque educativo, es necesario determinar cuál es la estrategia pedagógica empleada por cada propuesta analizada y determinar su efectividad de acuerdo a los resultados mostrados por sus autores. Es claro que una determinante en el análisis comparativo será prestar mayor atención a aquellas estrategias que apoyen la colaboración entre los estudiantes.
- Definición de propiedades colaborativas. Este criterio, dado el objetivo general establecido para esta tesis de desarrollar un enfoque colaborativo, permitirá evaluar cómo se realiza la comunicación entre los diferentes participantes (profesores, alumnos y personal de la industria local de software), qué tipo de interfaz de comunicación utilizan, qué herramientas utilizan para promover la comunicación, etc.
- Participación de la industria local. A lo largo de los años el enfoque de enseñanza para la Ingeniería de Software ha cambiado de clases presenciales (puramente teóricas) a combinar éstas con experimentos reales, ya sea en vinculación con la industria o a través de entornos simulados. En este sentido, es importante determinar si en el contexto de la educación en Mejora del Proceso de Software esta característica está presente en las propuestas analizadas.

A manera de resumen, la Tabla 12 muestra el análisis comparativo empírico que utiliza los criterios previamente definidos como base de la comparación.

Tabla 12. Comparación de propuestas analizadas en base a los criterios definidos

Propuesta	Modelo de Referencia o Proceso	Modelo de Evaluación	Factores de motivación	Habilidades suaves	Herramienta computacional	Estrategia pedagógica	Propiedades colaborativas	Interacción con la industria
RWL [Moore & Brennan, 1995]	CMM	SCAMPI	Desempeñar roles tipo líder.	Discusión, cooperación, juicio,	Ninguna (propuesta enfocada al diseño de un curso)	Constructivismo.	Trabajo en equipo.	Sí
Curso de Jaccheri [Jaccheri, 2002]	CMM, ISO 9000	SCAMPI	Relacionar la teoría con proyectos reales de la industria.	Observación, revisión, presentación oral y escrita	Ninguna (propuesta enfocada al diseño de un curso)	Estructuración de un modelo propio basado en investigaciones anteriores.	Trabajo en equipo, discusión en clase.	Sí
Curso constructivista [von Wangengeim & Hauck, 2010]	CMMI, ISO/IEC 12207, MPS.BR	SCAMPI	Relacionar la teoría con proyectos reales de la industria.	Presentación oral y escrita, cooperación, reflexión.	Ninguna (propuesta enfocada al diseño de un curso)	Constructivismo.	Trabajo en equipo, discusión en clase.	Sí
HEPALE! [Mendoza et al., 2009]	ISO/IEC 29110-5-1, COMPETISOFT	EvalProSoft	Una forma alternativa de aprendizaje. Incrementar la interacción entre los empleados.	Observación, revisión, presentación oral y escrita, reflexión y juicio.	Sí, basada en web. Sirve para gestionar los contenidos del curso.	e-learning.	Foro de discusión, repositorio compartido, uso de e-mail.	No (enfoque a la empresa)
TSPi & PBL [García & Pacheco, 2012]]	TSPi	Ninguno	Relacionar la teoría con proyectos reales de la industria.	Revisión, discusión, presentación oral y escrita, planificación, cooperación, reflexión y juicio.	Sí, basada en web. Sirve como herramienta de gestión de proyectos.	PBL.	Gestión de comunicación, control de cambios en tiempo real, trabajo en equipo, foro de discusión.	Sí
Virtual Scrum [Rodríguez et al., 2012]	SCRUM (Proceso o marco de trabajo)	Ninguno	Relacionar la teoría con proyectos simulados en un entorno.	Revisión, discusión, planificación, cooperación.	Sí, basada en web. Crea un entorno virtual para gestionar proyectos.	PBL.	Foro de discusión, avatares en tiempo real.	Sí

Por último, como principal resultado del análisis de herramientas y propuestas, es posible determinar un conjunto básico de funcionalidades que sirvan de línea base para la solución propuesta en esta tesis.

2.5.2. Conjunto básico de funcionalidades

La Figura 2.24 muestra un diagrama de contexto general para los principales casos de uso que deben incluirse en un enfoque educativo que busque apoyar la enseñanza en el área de Mejora del Proceso de Software. Los principales roles identificados son: el profesor, que también juega el rol de administrador; los estudiantes, que realizarán experimentos con las soluciones que evalúen y propongan; y el jefe de proyecto, que representa a la industria y que participa en los experimentos y contribuye con su conocimiento práctico. Considerando el modelo IDEAL analizado en la sección 2.3.1 de este capítulo, las propiedades determinadas en la Tabla 12 son complementadas con las actividades de dicho modelo para crear los siguientes casos de uso:

- Inicializar proyecto de mejora. El administrador (o profesor) es el único rol que tendrá privilegios de crear e inicializar un proyecto de mejora una vez que algún jefe de proyecto por parte de la industria local haya aceptado participar.
- Evaluar proceso de la organización. El jefe de proyecto es quien actualizará con información detallada el estado de los procesos y áreas identificadas en su empresa. De acuerdo al modelo de evaluación EvalProSoft, los alumnos evaluarán, analizarán y establecerán una calificación sobre los procesos identificados.
- Establecer plan de mejora. Una vez que la información de la evaluación es analizada y cotejada con documentación objetiva de la empresa, los estudiantes crearán un plan de mejora basándose en las prácticas que establece el modelo de referencia MoProSoft, siempre con la supervisión continua del profesor. Los jefes de proyecto por su parte recibirán el plan de mejora y una vez que acepten las soluciones, los estudiantes procederán a implantar este plan en los procesos a través de proyectos piloto.
- Revisar lecciones. Los alumnos se encargarán de documentar correctamente las lecciones aprendidas durante todo el proceso de mejora a fin de que sirva para futuros cursos. El profesor se encargará de revisar que esta documentación sea relevante y útil.
- Gestionar proyectos. Es necesario administrar los proyectos y las relaciones que existen entre éstos y los estudiantes, roles, empresas participantes, estado, nivel de completitud, empresas locales, fechas, etc.
- Gestionar estudiantes. Dentro del curso se debe gestionar a los estudiantes, así como a los proyectos en los que ha participado, sus roles desempeñados, nivel de participación, nivel de aprovechamiento, nivel de aceptación del curso, etc.
- Gestionar cursos. Es necesario introducir también la administración de los cursos vinculados con la industria local, por lo que será necesario controlar datos como la relación de alumnos del curso, el periodo de fechas, la empresa vinculada, etc.
- Gestionar acceso. De manera general se debe gestionar también la identificación de un usuario y garantizar su acceso a los diferentes módulos propuestos.

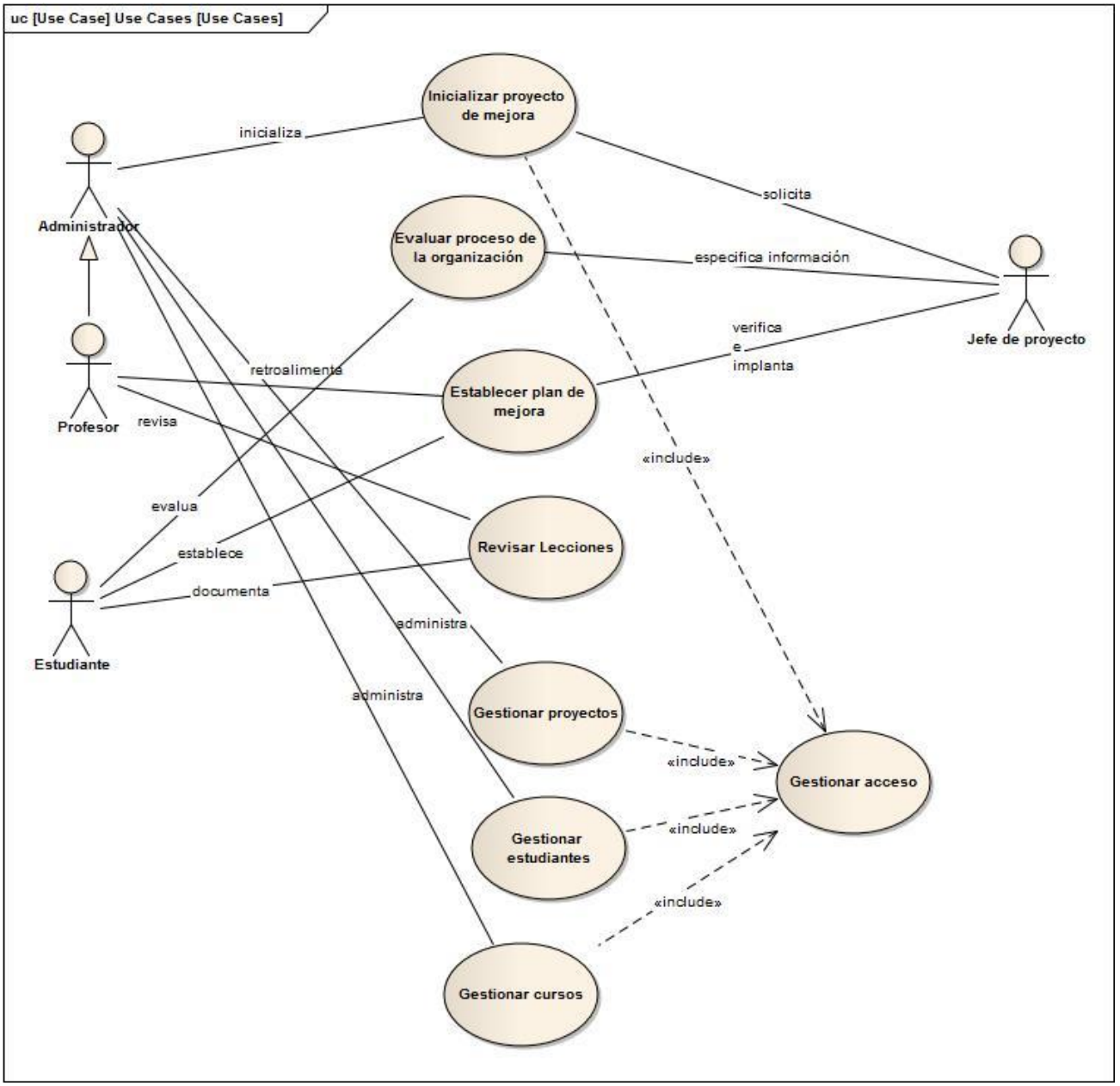


Figura 2.24. Diagrama de casos de uso de contexto general

2.6. Consideraciones finales sobre el capítulo

La realización del estudio del arte presentado en este Capítulo 2 ha permitido alcanzar las siguientes conclusiones:

- El análisis sobre la situación actual de la educación de la Ingeniería de Software en México evidencia un claro abismo entre los planes actuales de estudio de las universidades y las necesidades reales de la industria. Lo anterior se hace evidente cuando a pesar del surgimiento de iniciativas como PROSOFT y COMPETISOFT en nuestro país, los planes de estudio no incluyen aún materias detalladas sobre modelos de referencia promovidos actualmente a nivel nacional (particularmente MoProSoft). Es verdad que al estudiante se le enseña la forma de construir software, pero no se le explica que la industria trabaja bajo

estándares y modelos que representan una pequeña parte de un proceso más complejo. En este sentido, fue posible observar que mediante las iniciativas de apoyo a la industria de TI de índole Federal (MexicoFIRST, por ejemplo) y Estatal (Convocatorias de Fondo Mixto de CONACYT, por ejemplo), las universidades han comenzado a certificar a sus profesores para la impartición de materias cada vez más especializadas que promueven los factores que influyen en la calidad de software, la mejora continua, y el desarrollo de habilidades sociales.

- En relación a la forma en que la industria trabaja actualmente, el análisis sobre los dos modelos de referencia más utilizados en México (CMMI-DEV v1.2 y MoProSoft) ha permitido identificar que la tesis debe enfocarse en este último por las siguientes razones: (1) es el estándar creado para la industria Mexicana, (2) la industria local está conformada por MiPyMEs y obviamente resulta más fácil probar el enfoque propuesto en esta tesis, (3) MoProSoft es un modelo menos extenso que el CMMI-DEV v1.2, dado que a diferencia de éste cuenta con nueve áreas de proceso (el CMMI-DEV v1.2 define veintidós áreas), y (4) incluye a EvalProSoft como modelo de evaluación adaptado para MiPyMEs mexicanas.
- Por otro lado, y continuando con la forma en que se le enseñará al estudiante cómo realizar la implantación de MoProSoft, el uso del modelo de mejora IDEAL resulta importante. Después de analizar la guía propuesta por el SEI para IDEAL, fue posible determinar que a pesar de que este modelo se enfoca a la familia de modelos CMMI, éste puede ser utilizado con diferentes modelos de referencia y existe evidencia documentada de que puede ser adaptado para ser aplicado en compañías pequeñas de software.
- Por último, después de revisar las propuestas internacionales relacionadas con el objetivo de esta tesis, se han identificado y definido características comunes que han servido para establecer un conjunto inicial de funcionalidades básicas que deberá incluir una solución tecnológica que brinde soporte a la propuesta formulada en esta tesis. Se ha podido identificar que dos de los principales soportes del enfoque educativo son: (1) aumentar la motivación de los estudiantes en relación al diseño y desarrollo de iniciativas de mejora, y (2) promover la vinculación con industrias locales de software que proporcionen el apoyo para que los estudiantes desarrollen habilidades que no aprenden con regularidad en sus clases cotidianas.

3. Enfoque Colaborativo para aprender a controlar iniciativas de Mejora del Proceso de Software

Recientemente las universidades mexicanas han comenzado a ofrecer cursos sobre temas relacionados con la mejora y calidad del proceso. Tradicionalmente, estos temas habían sido cubiertos parcialmente por otros cursos, como aquellos orientados a temas de calidad dentro de la Ingeniería de Software en general, o peor aún habían sido ignorados por las actualizaciones curriculares. En la actualidad, la necesidad de este tipo de cursos se deriva principalmente de las demandas de la industria de software, y los beneficios de incorporarlos a nivel de profesionalización están bien documentados en la literatura especializada. En el contexto de esta tesis, se busca concretamente establecer un curso que cubra dos objetivos importantes:

O1. Mejorar la calidad de la educación sobre la Mejora del Proceso de Software en términos de una mayor colaboración y participación de los estudiantes, y

O2. Aprender sobre los problemas de la industria local de software.

Sin embargo, los principales inconvenientes al intentar establecer un curso que se base en la interacción con las empresas de software y que permita que los estudiantes aprendan a identificar y tratar los problemas de calidad y del proceso son la disponibilidad y la confianza mutuas. Es verdad que la enseñanza de la Ingeniería de Software ha presentado una serie de desafíos, que han sido abordados a fondo en la literatura a través de los años, pero son muy pocos los planes de estudio que involucran la participación activa de los estudiantes en proyectos con calendarios y presupuestos reales. No obstante, el diseño y la ejecución de proyectos en colaboración con la industria ha sido tema de exploración por más de 10 años en diferentes universidades del mundo. En este sentido, la literatura existente aborda tres alternativas posibles para diseñar un curso con alto grado de interacción con la industria:

1. La primera es la "alternativa nula", de acuerdo con la cual el profesor propone uno o más proyectos para que sean desarrollados por los alumnos en entornos controlados sin interacción o relación con alguna empresa de software. Los proyectos pueden estar basados en propósitos reales (a través de casos de estudio o experiencias personales), pero no existe dirección o realimentación por parte de la industria.
2. La segunda alternativa se basa en el modelo clásico de colaboración de proyectos (por ejemplo, como se describe en [Favela & Peña-Mora, 2001; Chao, 2007; Chen & Teng, 2011; Kamthan, 2013]), según la cual los estudiantes trabajan para crear un producto para un cliente real de una empresa real. En este modelo, los estudiantes desempeñan los roles estándar de analistas, diseñadores, programadores y *testers*. Además, para cubrir los problemas de calidad y del proceso los estudiantes desempeñan los roles de responsable de

procesos y de la calidad a través de la gestión de la configuración, el control de la calidad y la gestión de los proyectos.

3. La última alternativa es un modelo en el que los estudiantes trabajan en colaboración con las organizaciones a nivel del proceso y la calidad. En este modelo, los estudiantes tienen que jugar el papel, o por lo menos tienen que observar el trabajo de, los responsables de la calidad y de los procesos.

En nuestro contexto universitario la alternativa 1 no contribuye a cumplir con ninguno de los objetivos planteados al inicio de este capítulo. En este sentido, O1 no puede ser cubierto dado que una de las principales críticas a esta alternativa es que los ejemplos no son lo suficientemente concretos, y por lo tanto, no son motivadores. O2 puede ser cumplida con la alternativa 1 si el profesor ya cuenta con buenos casos de estudio sobre la industria de software local y se las arregla para diseñar el trabajo práctico para que los estudiantes emulen el funcionamiento de ésta.

Por otro lado, la literatura relacionada con los cursos enfocados en la construcción de un producto de software para un cliente real [Daniels et al., 2010; Lu & Declue, 2011; García & Pacheco, 2012] indica que la alternativa 2 incrementa la participación estudiantil. Sin embargo, la mayor desventaja de esta alternativa es que los problemas relacionados con el proceso y la calidad a menudo requieren grandes proyectos para que sean entendidos completamente. La gestión de la configuración, por ejemplo, no es un problema real cuando cuatro estudiantes trabajan en equipo durante tres meses para entregar un software, que nunca recibe mantenimiento. Además, los estudiantes tienden a centrarse en la tecnología más que en la correcta lectura y escritura de documentos. El riesgo más importante es que los estudiantes no aprecian la diferencia entre este tipo de curso (enfocado a los problemas relacionados con el proceso y la calidad) y un curso tradicional (enfocado a aspectos generales del desarrollo de software) por lo que O1 y O2 no pueden cubrirse.

Por último, la alternativa 3 es la ideal (*que los estudiantes trabajen en colaboración con las organizaciones a nivel del proceso y la calidad*) puesto que si los estudiantes son capaces de trabajar de manera cercana con empresas reales, serán capaces de identificar y resolver problemas reales relacionados con la calidad y el proceso. De esta forma se puede mejorar de forma directa la calidad de la educación sobre la Mejora del Proceso de Software a través de la colaboración de los estudiantes (O1), quienes junto con los profesores aprenderán sobre la industria local de software (O2). Es verdad que no es fácil convencer a las empresas a compartir con los estudiantes los problemas con sus productos y procesos básicamente por dos razones: las pequeñas tienen procesos *ad hoc* que en realidad no están documentados y no cuentan con los roles específicos para dedicarse concretamente a los problemas de calidad y del proceso. Las grandes, por otro lado, no permiten fácilmente que los estudiantes revisen sus manuales de calidad, normas de proceso, etc., puesto que cuentan con personal especializado en la conducción de iniciativas de mejora. En este sentido, se plantean de nuevo tres alternativas de posible interacción con las empresas de software:

1. Los estudiantes trabajan en el interior de las empresas en los niveles de calidad y del proceso.
2. Los estudiantes visitan a las empresas para aplicar métodos de análisis (entrevistas estructuradas, cuestionarios, revisión de documentación pertinente, etc.) y aprender sobre las iniciativas que inciden en la calidad y el proceso.
3. Los representantes de las empresas (jefes de proyecto) visitan a la universidad y comparten experiencias con los estudiantes a través de exposiciones.

Así, el diseño del curso considera la segunda alternativa (*que los estudiantes visiten a las empresas para aplicar métodos de análisis y aprender sobre las iniciativas que inciden en la*

calidad y el proceso), con una modificación de fondo: incluir una herramienta web que facilite la interacción entre el curso y la industria local de software y que evite incomodar al personal y a la empresa con visitas continuas. Esta decisión está fundamentada en las siguientes observaciones:

- Ya que como se ha mencionado anteriormente, las organizaciones no tienen la voluntad ni disponibilidad de permitir que muchos estudiantes trabajen a nivel de proceso y calidad, se propone diseñar una estrategia que permita establecer gradualmente un nivel de confianza basado en la presentación oportuna de resultados de mejora.
- Por otro lado, es necesario que el responsable de la mejora obtenga el conocimiento sobre métodos de investigación (cualitativos y cuantitativos), por lo que se propone enseñar estos métodos en el curso a través de la práctica real para no perder el enfoque de colaboración.

En otras palabras, se han elegido la alternativa 1 (*los estudiantes trabajan en colaboración con la industria local de software a nivel de proceso y de la calidad*) para diseñar un curso con un alto grado de interacción, y la alternativa 2 con una modificación (*que los estudiantes trabajen con los responsables de las empresas a través de una herramienta web para aplicar métodos de análisis y aprender sobre las iniciativas que inciden en la calidad y el proceso*) para interactuar de forma eficiente con las empresas de software y compartir experiencias y conocimientos prácticos de primera mano. Ahora bien, con el fin de que los alumnos participen activamente en las clases teóricas, se plantea que sean ellos quienes presenten los informes generados por la herramienta para propiciar la discusión y para determinar su comprensión sobre las iniciativas de mejora de la empresa.

3.1. Orientación colaborativa y experimental del enfoque

Considerando la base del plan de estudios GSwE2009, tratado en el Capítulo 2 de esta tesis, el cual tiene una arquitectura similar al propuesto por Mark Ardis y Gary Ford en [Ardis & Ford, 1989] y es compatible con los datos curriculares descritos por Arthur Pyster y sus colegas [Pyster et al., 2009], el enfoque propuesto en esta tesis incluye material básico, material avanzado, material operativo, y material optativo y la interacción activa con una red de trabajo a través de proyectos reales de mejora (véase Figura 3.1). De manera similar que en el GSwE2009, la línea de color negro en la Figura 3.1 representa al conocimiento básico que deben tener los estudiantes que se inscriban a la materia relacionada con la Mejora del Proceso de Software. En este sentido, la estructura de este enfoque está establecida por los siguientes elementos:

- Material básico. El material básico tiene el fin de establecer las bases sólidas sobre el resto del contenido del curso. Es decir, se establecen tópicos que proporcionen un panorama actual de la industria de software en el país. Por lo tanto, este tipo de material se proporciona principalmente con clases teóricas que son impartidas utilizando diapositivas, y que son apoyadas por la lectura, comprensión y discusión de artículos específicos.
- Material avanzado. El material avanzado tiene el fin de proporcionar conocimiento moderno sobre el área de Mejora del Proceso de Software y depende en gran medida del material básico impartido. Este tipo de material también se proporciona a través de clases teóricas que son impartidas utilizando diapositivas, y que son apoyadas por la lectura, análisis y presentación de artículos específicos y casos de estudio.
- Material operativo. El material operativo tiene el objetivo de proporcionar conocimiento práctico sobre el área de Mejora del Proceso de Software y depende de la participación de los

estudiantes en las iniciativas reales establecidas en la red de trabajo. Este tipo de material se proporciona de forma integral combinando las clases teóricas con las sesiones de trabajo colaborativas con las empresas participantes.

- Material optativo. El material optativo tiene el objetivo de proporcionar conocimiento adicional sobre la forma actual de trabajo de la industria de software. Este tipo de material es proporcionado por los representantes de las empresas registradas en la red de trabajo y cubre específicamente la demanda que los estudiantes puedan tener al participar en una iniciativa real de mejora.
- Herramienta de soporte. De acuerdo a la Figura 3.1 esta herramienta representa el pilar del enfoque propuesto dado que, como se mencionó anteriormente, la estrategia de colaboración busca no incomodar a las empresas recibiendo a los estudiantes para realizar las actividades propias de la iniciativa de mejora. De esta forma, una herramienta computacional proporciona el soporte necesario para que el material operativo pueda llevarse a la práctica.

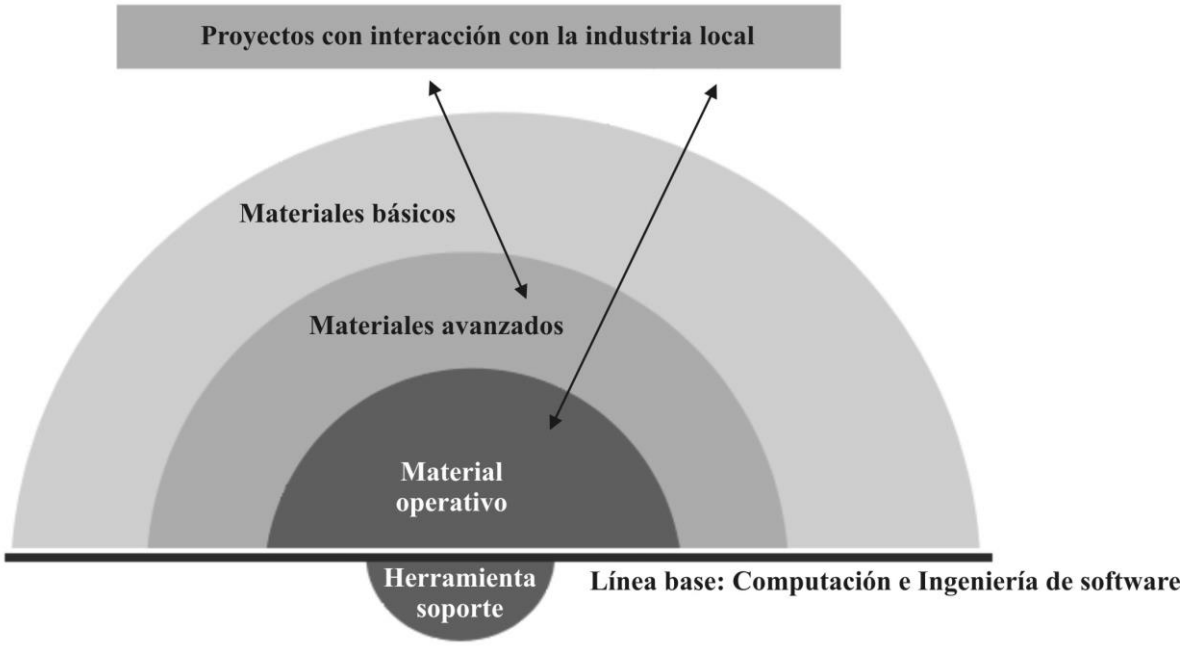


Figura 3.1. Esquema del curso colaborativo para el programa de maestría

En este sentido, un estudiante podría cubrir la preparación a través de la obtención de un grado o título de ingeniero (ya sea en las licenciaturas de Ingeniería en Computación, Ciencias de la Computación, Ingeniería Electrónica o Ingeniería de Software), además de contar con dos años de experiencia en el desarrollo de software. Los estudiantes podrán dominar el material que está sobre la línea negra de la Figura 3.1, solamente después de alcanzar la preparación establecida por la línea base. Las materias del tronco común del programa determinarán la forma de preparar a los estudiantes cuyos antecedentes no están a la altura; el elemento de "material de preparación" que está por encima de la línea de las expectativas representa esta preparación adicional.

Así, es fácil observar que este enfoque requiere de una alta dosis de experimentación utilizando el contexto actual de la organización. En este sentido, de acuerdo con Kitchenham, con el fin de evaluar, analizar e interpretar los resultados de los experimentos realizados en la Ingeniería de Software es necesario formular preguntas de investigación, identificar áreas temáticas o un

fenómeno de interés [Kitchenham, 2004]. Por lo tanto, el curso propuesto en esta tesis utiliza las fases del modelo IDEAL (Inicio, Diagnóstico, Establecimiento, Ejecución, y Adopción) [McFeeley, 1996], el modelo de mejora más utilizado en todo el mundo y que fue descrito en la sección 2.3.1 de esta tesis, para formular las siguientes preguntas de investigación (PI) dentro de un contexto educativo:

- **PI 1**: ¿Qué actividades debe realizar una empresa antes de comenzar con una iniciativa de mejora?
- **PI 2**: ¿Cómo se determinan las fortalezas y debilidades del proceso de software en una empresa?
- **PI 3**: ¿Qué tan buena es la comprensión de un plan de mejora en el contexto general de la empresa?
- **PI 4**: ¿Cómo puede una empresa lograr un mayor rendimiento en su proceso de software con la iniciativa de mejora?
- **PI 5**: ¿Existe una correlación entre la cantidad de esfuerzo que se requiere para entender la iniciativa de mejora y el promedio ganado?

La Figura 3.2 proporciona una visión general de cómo estas cinco preguntas de investigación proporcionan una visión integral del enfoque propuesto en esta tesis abarcando cuatro de las cinco fases de IDEAL (o las actividades de Planear, Evaluar y Pilotar de la Figura 1.1).

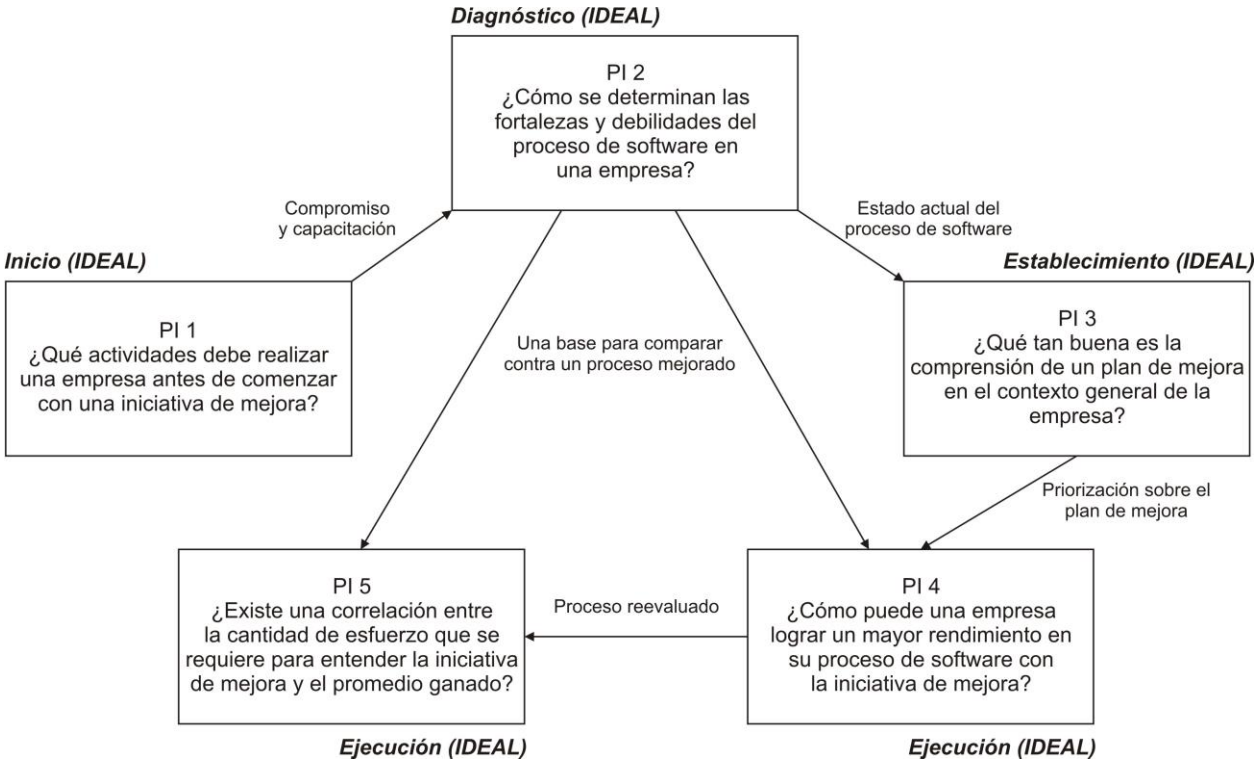


Figura 3.2. Relación entre las cinco preguntas de investigación establecidas

A través del análisis de las cinco preguntas de investigación establecidas, el objetivo del curso se enfoca en obtener un panorama amplio de las actividades que los estudiantes realizan a través de

la herramienta computacional y la práctica e interacción con la industria. A través de este enfoque, es posible recopilar información sobre las actividades que los estudiantes realizan en la empresa antes de que una iniciativa de mejora de procesos comience (PI 1), sobre cuáles son los puntos fuertes y debilidades en el proceso de software de cada empresa (PI 2); sobre qué tan buena es la comprensión de un plan de mejora (PI 3); sobre cómo las empresas pueden lograr un mayor nivel de rendimiento en sus procesos de software después de la iniciativa de mejora de procesos (PI 4); y sobre la correlación, si es que existe, entre la cantidad de esfuerzo necesaria para comprender la mejora de procesos y el promedio obtenido de mejora (PI 5). En este sentido, los estudiantes deben responder a las preguntas de investigación a través de la interacción con las empresas que se encuentran en la red de colaboración. Actualmente, el enfoque propuesto en esta tesis está soportado por cinco empresas que han colaborado activamente con los estudiantes en cursos anteriores y que son descritas por la Figura 3.3.

ticdes

TICDES S.A. de C.V. es una empresa establecida en la Ciudad de Oaxaca de Juárez dedicada al desarrollo de soluciones informáticas y consultoría para empresas mexicanas. La empresa, que fue creada en el 2005, actualmente se encuentra legalmente establecida mediante una Sociedad Anónima de Capital Variable y es socia activa de la Cámara Nacional de la Transformación (CANACINTRA), desde entonces ha ido creciendo en actividad y desarrollo empresarial.

ragasoft

RAGASOFT S.A. de C.V. es una empresa oaxaqueña de desarrollo de software, enfocada a proporcionar a las PyMEs mexicanas las tecnologías de información que les permitan reducir costos operativos y mejorar su ventaja competitiva mediante el análisis, desarrollo y mantenimiento de sistemas de información hechos a la medida de sus necesidades.

SOCIEPRO

SOCIEPRO S.A.P.I de C.V. es una empresa del grupo VEUREKA, establecida en la Ciudad de Huajuapan de León, que se ha constituido con la misión de mejorar la calidad de vida de la sociedad innovando su modo de proceder con las prácticas de gestión de proyectos, procesos y productividad ejecutiva.

OBX Dynamic Software Engineering

OBX Dynamic Software Engineering S.A de C.V es una empresa de Oaxaca formada por profesionistas multidisciplinarios, que crean software en base a las nuevas tecnologías que demanda el mercado, utilizando la investigación y la experiencia en metodologías para el desarrollo de los sistemas, para ofrecer a las micro y medianas empresas de Latinoamérica soluciones móviles de bajo costo.

OASYS OAXACA SOLUCIONES EN SISTEMAS

OASYS OAXACA SOLUCIONES EN SISTEMAS R.L de C.V. es una empresa de Oaxaca, ubicada en el Distrito Federal. En OASYS la principal actividad es el desarrollo de software a la medida y tiene un marcado interés en actividades de I+D que involucre el desarrollo y experimentación de solución tecnológicas de bajo costo.

Figura 3.3. Empresas colaboradoras de la red de la industria local

A través de esta red de colaboración se realizan todas las actividades necesarias para responder las preguntas de investigación definidas. Por último, antes de establecer una secuencia de tópicos para el enfoque propuesto, es necesario definir una serie de propiedades que permitan darle una orientación colaborativa. Dichas propiedades, descritas en la Figura 3.4, aportan ventajas importantes a la Ingeniería de Software cuando se requiere de equipos numerosos ubicados en diferentes zonas geográficas y que contribuyen a fortalecer el enfoque propuesto en esta tesis, de acuerdo con [Favela & Peña-Mora, 2001], [Richardson et al., 2010] y [Cramer et al., 2010].

Motivación

Esta propiedad busca realizar ensayos mentales preparatorios de las acciones que originan una iniciativa de mejora para animar a los estudiantes a realizarlas con interés y diligencia. Los contenidos relacionados con esta propiedad se enfocan a temas que propician el interés de los estudiantes por mejorar la situación actual de una empresa de software.

Discusión

Esta propiedad promueve entre los estudiantes el análisis o comparación de los resultados de una investigación, presentación, caso de estudio, o artículo a la luz de otros existentes o posibles. Los contenidos relacionados con esta propiedad se enfocan a temas que buscan la comunicación y realimentación de los comentarios de los estudiantes.

Colaboración

Esta propiedad promueve que los estudiantes trabajen con otra u otras personas en la realización de una o más actividades dentro de la iniciativa de mejora. Los contenidos relacionados con esta propiedad se enfocan a temas que requieren la participación activa entre los estudiantes y con la red de empresas participantes.

Interacción

Esta propiedad establece un conjunto de acciones que los estudiantes deben realizar a través de la herramienta computacional. Los contenidos relacionados con esta propiedad se enfocan a temas que requieren que tanto los estudiantes como los representantes de cada empresa realicen actividades simultáneas haciendo uso la herramienta computacional de soporte.

Figura 3.4. Propiedades incorporadas a los tópicos del curso

Así, la solución propuesta en esta tesis establece dos objetivos que deben ser cubiertos por los tópicos establecidos para el curso, cinco preguntas de investigación que cubren las fases del modelo IDEAL, cuatro tipos de material que definen el contenido a aprender/enseñar dentro de la iniciativa de mejora, una red de colaboración formada por cinco empresas que participan activamente en la preparación de los diferentes niveles de MoProSoft, cuatro propiedades que fortalecen el enfoque colaborativo a la propuesta, y una herramienta computacional que soporta al enfoque a través de la interacción web con la red de colaboración.

La conjunción de estos elementos se observa en el temario propuesto en la Tabla 13. Cabe mencionar que la mayoría de los temas corresponde al plan actual de la materia optativa “Modelos y Métodos de Evaluación y Mejora” del Programa de la Maestría en Tecnologías de Cómputo Aplicado, línea Metodologías de Desarrollo. La mayor actualización reside en que este plan es asociado con los elementos definidos para el enfoque colaborativo, incluida la herramienta computacional que servirá de soporte.

Tabla 13. Temario para un curso con enfoque colaborativo

Nombre del curso:		Gestión y control de iniciativas de mejora del proceso software			
Semestre:	Segundo	**Total de horas:**	85	**Área de conocimiento:**	Metodologías de desarrollo

Objetivos generales del curso:
Los estudiantes de maestría aprenderán a gestionar y controlar iniciativas de mejora del proceso de software a través del conocimiento práctico de los modelos de mejora, los métodos para la evaluación de la madurez y de la capacidad de una empresa, y los modelos de referencia actuales. En este sentido, el conocimiento práctico es adquirido a través de la interacción continua con la industria local de software de tal forma que sea posible establecer una guía puntual para que ésta mejore de forma sistemática sus procesos. Se pretende que durante el curso, el alumno aborde proyectos de evaluación y mejora de procesos a través del uso continuo de una herramienta computacional que le permitirá considerar todos los factores que inciden en una iniciativa de mejora. De esta forma, el alumno de maestría podrá comprender los modelos de referencia adecuados para nuestro país, teniendo en cuenta factores como la tecnología, la organización y el propio negocio.

Temas y subtemas	**Pregunta de investigación**	**Tipo de material**	**Herramienta de soporte**	**Propiedad**
1. La industria de software en México				
1.1 Desarrollo de una industria competitiva	--	Básico	Clase Teórica	
1.2 Oportunidades en el mercado actual	--	Básico	Clase Teórica	
1.3 Estrategias internacionales para impulsar la industria de software	--	Básico	Clase Teórica	
1.4 El contexto actual de la industria de software en México	--	Básico	Clase Teórica	
1.5 La iniciativa PROSOFT y el programa COMPETISOFT	--	Básico	Clase Teórica	
1.6 El futuro de la industria de software	--	Básico		
2. La formalidad del proceso de software				
2.1 Definición del proceso	--	Básico	Clase Teórica	
2.2 Problemática actual y la crisis del software	--	Básico	Clase Teórica	
2.3 Procesos maduros vs. Procesos inmaduros	--	Básico	Clase Teórica	
2.4 Soluciones tradicionales e implicaciones en la industria de software	--	Básico	Clase Teórica	
2.5 Modelos de referencia: CMMI-DEV v1.3 y MoProSoft	--	Básico	Clase Teórica	
2.6 Revisión de estándares para gestionar el proceso de software	--	Básico	Clase Teórica	
2.7 Estándares vs. Prácticas efectivas	--	Básico	Clase Teórica	

3. Características de la mejora del proceso de software				
3.1 Calidad frente a cantidad: un enfoque sistemático	--	Básico	Clase Teórica	
3.2 Surgimiento de la mejora del proceso de software	--	Básico	Clase Teórica	
3.3 Revisión del enfoque de mejora del proceso de software en el mundo	--	Básico	Clase Teórica	
3.4 Beneficios y consecuencias de la mejora del proceso de software	--	Básico	Clase Teórica	
3.5 Definición de pequeños entornos, pequeños equipos y pequeñas empresas	--	Básico	Clase Teórica	
3.6 Beneficios y problemas de los modelos de buenas prácticas	--	Básico	Clase Teórica	
3.7 Modelos de evaluación (SCAMPI, EvalProSoft) y de mejora (IDEAL, ISO/IEC 15504:2004, AFIM)	PI 1	Avanzado	Clase Teórica	
4. Los modelos de referencia en la industria mexicana				
4.1 CMMI-DEV v1.3	PI 1	Avanzado	Clase Teórica	
4.1.1 Representaciones	PI 1	Avanzado	Clase Teórica	
4.1.2 Categorías de procesos	PI 1	Avanzado	Clase Teórica	
4.1.3 Áreas de proceso	PI 1	Avanzado	Clase Teórica	
4.2 MoProSoft	PI 1	Avanzado	Clase Teórica	
4.2.1 Representaciones	PI 1	Avanzado	Clase Teórica	
4.2.2 Categorías de procesos	PI 1	Avanzado	Clase Teórica	
4.2.3 Áreas de proceso	PI 1	Avanzado	Clase Teórica	
4.3 Análisis comparativo de modelos	PI 1	Avanzado	Clase Teórica	
5. El modelo IDEAL				
5.1 Antecedentes y actualidad del modelo	PI 1	Avanzado	Clase Teórica	
5.2 Fases del modelo	PI 1	Avanzado	Clase Teórica	
5.3 Análisis de actividades internas	PI 1	Avanzado	Clase Teórica	
5.4 Implementación práctica del modelo	PI 1	Avanzado	Clase Teórica	
6. Planeación de la iniciativa de mejora				
6.1 Análisis de objetivos y misión	PI 1	Operativo	Establecimiento de compromisos en línea y configuración de la herramienta web	
6.2 Definición de roles y compromiso	PI 1	Operativo		
6.3 Formulación del plan de trabajo	PI 1	Operativo		
6.4 Formulación del plan de capacitación	PI 1	Operativo		
6.5 Selección de procesos y establecimiento de indicadores de mejora	PI 1	Operativo		

7. Evaluación del proceso de software			Modelado de procesos y evaluación basada en cuestionarios a través de la herramienta web	
7.1 Evaluaciones SCE y SPA	PI 2	Operativo		
7.2 SCAMPI y EvalProSoft	PI 2	Operativo		
7.3 Modelación del proceso actual	PI 2	Operativo		
7.4 Recogida de datos, clasificación e informe	PI 2	Operativo		
7.5 Caracterización de la implementación de la práctica	PI 2	Operativo		
7.6 Análisis sobre niveles de cobertura	PI2	Operativo		
7.7 Reportes de resultados	PI2	Operativo		
8. Generación del plan de mejora			Análisis de respuestas y preparación de la guía preliminar de mejora a través de la herramienta web	
8.1 Análisis de fortalezas y debilidades	PI 2	Operativo		
8.2 Guía preliminar del plan de acción	PI 3	Operativo		
8.3 Priorización de problemas	PI 3	Operativo		
8.4 Revisión de compromiso	PI 3	Operativo		
8.5 Rediseño del proceso	PI 3	Operativo		
8.5 Planes de acción y plan de mejora	PI 3	Operativo		
9. Pilotaje de soluciones			Generación del plan de mejora, y definición y control de proyectos piloto a través de la herramienta web	
9.1 Definición del proyecto piloto	PI 4	Operativo		
9.2 Establecimiento de objetivos de mejora	PI 4	Operativo		
9.3 Comparación con indicadores establecidos	PI 4	Operativo		
9.4 Medición y control del proyecto piloto	PI 4	Operativo		
9.5 Lecciones aprendidas	PI 4	Operativo		
10. El comienzo del cambio			Control de indicadores de mejora a través de la herramienta web	
10.1 Gestión del cambio	PI4	Operativo		
10.2 Medición y análisis en la iniciativa de mejora	PI 5	Operativo		
10.3 Recogida de lecciones aprendidas	PI 5	Operativo		

A continuación se muestra cómo cada una de las cinco preguntas de investigación debe ser abordada a lo largo del curso.

3.2. Proceso para responder las preguntas de investigación establecidas para el curso

Los estudiantes deben trabajar en colaboración con las empresas registradas en la "red de colaboración" a nivel del proceso de desarrollo. Dentro de esta red, los estudiantes tienen que colaborar con, o al menos aprender de, los jefes de proyecto de cada empresa. A través de este enfoque, los estudiantes y los profesores aprenden sobre la industria local de software. La Figura 3.5 muestra una visión general del proceso que los estudiantes realizan durante un curso para responder las preguntas de investigación, en colaboración con la red de empresas y a través del soporte de la herramienta computacional. La experimentación comienza con la definición del alcance relacionado con los procesos que se incluirán en la iniciativa de mejora de procesos (por ejemplo, Administración de Proyectos Específicos (APE) y Desarrollo y Mantenimiento de Software (DMS)). Esto permite que los estudiantes identifiquen puntos débiles en los procesos seleccionados y generen

planes de mejora en colaboración con los responsables de las empresas. Una vez que se genera el plan de mejora, las empresas reciben la orientación necesaria para completar las mejoras recomendadas y darles seguimiento.

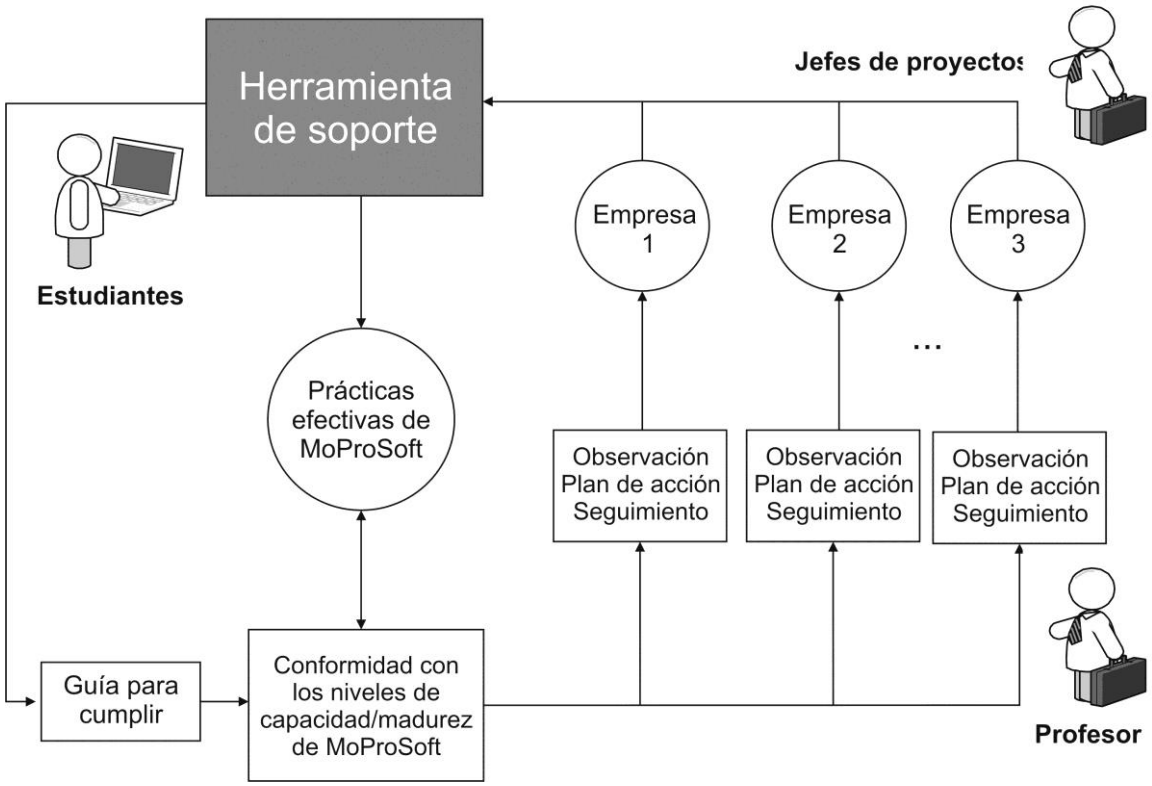


Figura 3.5. Esquema de colaboración entre industria y academia para responder las preguntas de investigación

Bajo este esquema, los estudiantes y profesores forman un grupo de asesoramiento que ofrece orientación para las empresas de acuerdo a su visión de negocio. Este grupo incluye a un equipo de evaluación compuesto por un grupo de estudiantes y profesores y, por lo menos, dos jefes de proyecto por cada empresa para evaluar los procesos y los proyectos seleccionados. Estos jefes de proyecto son miembros profesionales que conocen la cultura de trabajo de su empresa y la forma en que se llevan a cabo los proyectos. Los jefes de proyecto reciben la formación básica sobre la norma NMX-I-059/NYCE-2005, particularmente en los Niveles 1, 2, y 3[10]. Así, una explicación detallada de cómo los estudiantes responden a las preguntas de investigación es proporcionada a continuación.

[10] Esta tesis se centra en abordar los tres primeros niveles de MoProSoft por dos razones: (1) todas las empresas incorporadas a la red se encuentran en la categoría de pequeñas empresas que no tienen experiencia en iniciativas de mejora de procesos de software y están intentado prepararse para evaluaciones oficiales, y (2) los estudiantes de posgrado requieren aprender cómo funciona un modelo de referencia, y la base elemental para esto y cualquier esfuerzo de mejora de procesos se encuentra en los primeros niveles de estos modelos. Los Niveles 1 y 2 garantizan que las organizaciones de software son capaces de planificar y gestionar un proceso estándar para el desarrollo de software. El Nivel 3 implica que la empresa es capaz de adaptar estos procesos estándar a proyectos de diferentes características. Así, los niveles más altos son abordados cuando los niveles de madurez de la organización se han logrado a través de la experiencia.

3.2.1. PI 1: ¿Qué actividades debe realizar una empresa antes de comenzar con una iniciativa de mejora?

La *fase de inicio* de la iniciativa de mejora es cubierta y apoyada por los representantes de las empresas y la participación de los estudiantes. Por lo tanto, no es necesario realizar ningún trabajo adicional para obtener apoyo y compromiso de la alta dirección.

Inicialmente, tanto los representantes empresariales en la red de colaboración como los estudiantes son introducidos (a través de talleres) al área de mejora de procesos y a cómo utilizar la herramienta de soporte. Así, la herramienta computacional permite a la alta dirección el establecer las características iniciales de la mejora: selección del equipo de evaluación, de los jefes de proyecto, de los procesos y proyectos a ser evaluados, y del líder de la mejora. El enfoque de utilizar al profesor y a un alumno como líderes del esfuerzo de mejora ha sido implementado y está proporcionando buenos resultados, teniendo en cuenta que ambos dedican menos esfuerzo a la mejora de los procesos debido a la automatización de las actividades a través de la herramienta computacional. Debido a que las empresas probablemente estén comenzando a aprender sobre un proceso de mejora y a cómo poner en marcha un programa de este tipo, el primer paso requiere de educación y capacitación a todas las partes involucradas.

En este sentido, los profesores programan y proporcionan talleres y clases expositivas para la alta dirección de las empresas y los jefes de proyectos antes de comenzar con el trabajo de la iniciativa de mejora. Los temas abordados son: la planificación de proyectos, el desarrollo de equipos, la gestión del cambio tecnológico, los beneficios de la Mejora del Proceso de Software, el proceso de consultoría, MoProSoft y los modelos de mejora. La herramienta computacional incorpora un tutor que satisface la adquisición de conocimiento de acuerdo a las necesidades de la red de colaboración y los estudiantes. Este tutorial gestiona un conjunto de lecciones/tópicos y promueve la inclusión de nuevos de acuerdo al progreso de los proyectos de mejora. Si el desempeño del jefe del proyecto o algún estudiante es bajo durante la evaluación, el tutorial recomienda temas y alertas con el fin de vigilar y controlarlos a ambos.

El seguimiento continuo es realizado por los profesores con el objetivo de evaluar el progreso de la iniciativa y de los estudiantes, y el desempeño de los jefes de proyectos; como resultado la herramienta computacional sugiere medidas correctivas a través de un archivo PDF que es generado para los profesores con el fin de aconsejarlos a resolver el problema. Por lo tanto, esta herramienta establece un módulo de configuración para automatizar la fase de planificación a través de las siguientes actividades:

- Después del taller inicial, el administrador de la herramienta comienza la iniciativa de mejora agregando los nombres y roles de la alta dirección. La alta dirección también es provista de una idea general sobre la cantidad de trabajo, esfuerzo e inversión (en términos de que el personal además de realizar su trabajo diario, debe participar en las actividades propias de la mejora) necesaria para implementar la iniciativa. El correo electrónico es utilizado como medio de comunicación para enviar una solicitud de compromiso a cada miembro de la alta dirección que participa en la iniciativa, quien puede escoger un enlace para aceptar o rechazar el compromiso.
- Una vez aceptado el compromiso, los profesores integran un equipo de evaluación con los estudiantes y jefes de proyectos para evaluar tanto los procesos como los proyectos de cada empresa.
- Es necesario también que la alta dirección establezca las áreas de proceso relacionadas con la iniciativa de mejora, así como el nivel objetivo para delimitarla.

- Por último, el equipo de evaluación es capacitado y algunos tópicos son incluidos en el administrador del tutorial. La herramienta computacional gestiona el progreso de capacitación y alerta al líder de la iniciativa cuando éste ha terminado.

3.2.2. PI 2: ¿Cómo se determinan las fortalezas y debilidades del proceso de software en una empresa?

Con el fin de caracterizar el estado inicial de los procesos de software de la empresa, así como de evaluar las mejoras realizadas a través del ciclo de mejora, un mecanismo de evaluación [García et al., 2007] es implementado en la herramienta computacional para que los estudiantes evalúen los esfuerzos sobre la mejora de los procesos (*fase de diagnóstico*). Una vez que se realiza la evaluación, la herramienta computacional es capaz de identificar las debilidades y fortalezas del proceso con el objetivo de determinar la situación actual de las empresas y proponer un plan adecuado de mejora. Después de este plan, los esfuerzos se concentran en mejorar la calidad del software producido. Los jefes de proyecto deben responder, por ejemplo, los cuestionarios de APE y DMS para obtener el nivel de cobertura por área. En consecuencia, la herramienta computacional analiza los resultados de la evaluación dando un peso específico a cada tipo de respuesta e identificado las prácticas que están institucionalizadas dentro de las empresas, y cuáles no se realizan en absoluto (véase Tabla 14).

Tabla 14. Clasificación de nivel de rendimiento de la herramienta computacional [Garcia et al., 2007]

Posible respuesta	Nivel de rendimiento	Descripción
Siempre	4	La actividad está documentada y establecida en la empresa. Siempre se lleva a cabo, entre el 76% y 100% de las veces, en sus proyectos de software.
Usualmente	3	La actividad está establecida en la empresa, pero rara vez es documentada. Se realiza por lo general, entre el 51% y 75% de las veces, en sus proyectos de software.
Algunas veces	2	La actividad está débilmente establecida en la empresa. Se realiza a veces, entre el 26% y 50% de las veces, en sus proyectos de software.
Rara vez	1	La actividad se realiza con poca frecuencia en la empresa. Rara vez se lleva a cabo, entre el 1% y el 25% de las veces, en sus proyectos de software.
Nunca	0	La actividad no se lleva a cabo en la organización (0%). Ninguna persona o grupo realiza la actividad de la organización.
No lo sé	--	La persona no está segura de cómo responder a la pregunta.
No aplica	--	La pregunta no se aplica a la empresa.

El mecanismo de evaluación utilizado se basa en dos tipos de actividades: prácticas específicas (relacionadas con los jefes de proyectos) y prácticas genéricas (relacionadas con la alta dirección de la empresa). A modo de ejemplo, la Figura 3.6 muestra la técnica utilizada para identificar fortalezas y debilidades en un proceso de MoProSoft. En el contexto de la Mejora del Proceso de Software, suele elegirse una técnica basada en cuestionarios ya que proporciona soluciones rápidas dentro de una metodología de investigación dado que es el investigador quien determina las preguntas a formular y la gama de respuestas que se pueden dar. Esto hace que el resultado obtenido sea más preciso y fácil de analizar desde el punto de vista del investigador

[Young et al., 2006; Habra et al., 2008; Pino et al., 2010]. La técnica utiliza un promedio ponderado para calcular el nivel de cobertura para cada pregunta (número 1 en la Figura 3.6). Por lo tanto, el análisis se traslada de un punto de vista cualitativo (gama de respuestas Siempre (S), Usualmente (U), Algunas veces (A), Rara vez (R), y Nunca (N)) a una visión cuantitativa (nivel de cobertura por pregunta) (número 2 en la Figura 3.6). Además, la desviación estándar también es considerada mediante el cálculo del nivel de cobertura para cada proceso (número 3 en la Figura 3.6).

Administración de Proyectos Específicos	Jefes de proyecto		1					2		
Preguntas	1	2	#S	#U	#A	#R	#N	Ncp	M	D
1. ¿Se identifican hitos importantes para el proyecto?	S	N	1 50%	0	0	0	1 50%	**50%**	2	2.00
2. ¿Se identifican suposiciones sobre el calendario del proyecto?	A	R	0	0	1 50%	1 50%	0	**37%**	1.5	0.50
3. ¿Se identifican las limitaciones del proyecto?	A	A	0	0	2 100%	0	0	**50%**	2	0
4. ¿Se identifican las dependencias de las tareas del proyecto?	U	N	0	1 50%	0	0	1 50%	**37%**	1.5	1.50
5. ¿Se definen y mantienen el presupuesto y calendario del proyecto?	S	A	1 50%	0	1 50%	0	0	**75%**	3	1
6. ¿Se establecen acciones correctivas para el proyecto?	R	A	0	0	1 50%	1 50%	0	**37%**	1.5	0.50
7. ¿Se identifican los riesgos del proyecto?	A	A	0	0	2 100%	0	0	**50%**	2	0
8. ¿Se documentan los riesgos?	R	R	0	0	0	2 100%	0	**25%**	1	0
9. ¿Se evalúan los riesgos cuando es necesario?	R	R	0	0	0	2 100%	0	**25%**	1	0
10. Do you establish requirements and procedures to ensure privacy and security of the project's data?	U	R	0	1 50%	0	1 50%	0	**50%**	2	1
11. ¿Se establecen mecanismos para archivar los datos y acceder a los datos archivados?	U	U	0	2 100%	0	0	0	**75%**	3	0
12. ¿Se determinan los datos del proyecto que serán identificados, recogidos y distribuidos?	U	U	0	2 100%	0	0	0	**75%**	3	0
13. ¿Se determinan los requisitos de instalaciones y equipo?	R	R	0	0	0	2 100%	0	**25%**	1	0
14. ¿Se determinan los requisitos de personal?	R	R	0	0	0	2 100%	0	**25%**	1	0
15. ¿Se identifican el conocimiento y las habilidades necesarias para desarrollar el proyecto?	U	U	0	2 100%	0	0	0	**75%**	3	0
TOTALES			2 6.7%	10 33.3%	6 20%	12 40%	0	**47%**	3	

Figura 3.6. Ejemplo de análisis de respuestas realizado por los estudiantes

El porcentaje de nivel de cobertura para cada pregunta (NCp) se calcula considerando las respuestas de todos los jefes de proyecto de la siguiente manera:

$$NCp_{ij} = Cr_{iS} * 1 + Cr_{iU} * 0.75 + Cr_{iA} * 0.50 + Cr_{iR} * 0.25 + Cr_{iN} * 0 \quad (1)$$

donde *Cr* es la cobertura de la respuesta, *i* es el número de pregunta, *j* es el número de jefes de proyectos, y los números representan los porcentajes definidos en la Tabla 14 para cada tipo de respuesta. La razón de esta ponderación es porque se considera que las afirmaciones más fuertes deben tener más importancia que las débiles, dado que, por ejemplo, algo que siempre se hace no puede influir de la misma manera que algo que se hace raramente. Sin esta ponderación todas las preguntas tendrían siempre una cobertura del 100%, lo cual no aclararía nada. Además, para cada pregunta, la media matemática y la desviación estándar son calculadas de la siguiente manera:

$$M_i = \frac{\#A * 4 + \#U * 3 + \#S * 2 + \#R * 1 + \#N * 0}{jp} \quad (2)$$

$$Desv_i = \sqrt{\frac{\left[\#jp * \left(\#A * 4^2 + \#U * 3^2 + \#S * 2^2 + \#R * 1^2\right) - \#jp^2 * M_i^2\right]}{jp^2}} \quad (3)$$

donde *i* es el número de pregunta, *jp* es el número de jefes de proyectos que participan en la evaluación de los procesos y las letras con # son los niveles de desempeño definidos en la Tabla 14. Los valores de 1, 0.75, 0.5, 0.25 y 0 también podrían ser utilizados y el valor máximo de la media matemática en lugar de 4 sería 1. Sin embargo, es recomendable utilizar los valores más altos de ponderación (4, 3, 2, 1, y 0) mediante la aplicación de un factor, por lo tanto las diferencias existentes son más visibles. Es importante tener en cuenta que existe una correspondencia biunívoca o el mismo significado entre el porcentaje de cobertura de la pregunta y su media matemática (la única diferencia es que los factores de escala son diferentes), de tal forma que un alto porcentaje de cobertura también reflejará una media matemática alta y en la misma proporción que el porcentaje. Por último, el nivel de cobertura total (NCT) del proceso se calcula de la siguiente manera:

$$NCT(p) = \frac{\sum_{i=1}^{n} Cp_i}{n} \quad (4)$$

donde *p* es el proceso evaluado, *i* es el número de pregunta, y *n* es el total de preguntas. De esta manera, los alumnos determinan el nivel de cobertura en cada empresa y el promedio de cobertura por proceso para identificar las fortalezas y debilidades de los procesos seleccionados. En este sentido, para analizar las fortalezas y debilidades (o problemas), los alumnos tienen que usar las respuestas de los jefes de proyectos obtenidas en las evaluaciones. Aquellas preguntas con una cobertura inferior al 85% (número 4 en la Figura 3.6) implican que la actividad del modelo de referencia correspondiente (número 5 en la Figura 3.6) no está bien implementada en la empresa y tiene que ser considerada como una cuestión a mejorar (o una debilidad) en el proceso. Por otro lado, las preguntas con cobertura igual o mayor a 85% deben ser analizadas por los estudiantes a través de la desviación estándar de la siguiente manera:

- Si la desviación estándar es inferior a 0.8, la actividad del modelo de referencia relacionada con la pregunta representa una fortaleza del proceso evaluado (número 6 en la Figura 3.6).
- Si la desviación estándar es igual o superior a 0.8 (número 7 en la Figura 3.6), la actividad del modelo de referencia relacionada con esa pregunta debe ser profundamente explorada por los estudiantes a través de entrevistas para aclarar si se trata de una fortaleza o de un aspecto

a mejorar. Este caso significa que ha habido fuertes discrepancias en las respuestas dadas por los jefes de proyectos al completar los cuestionarios (las respuestas de los jefes de proyectos son muy divergentes y, por lo tanto, deben ser exploradas a mayor profundidad).

Así, la Figura 3.6 resume el proceso que los estudiantes tienen que realizar para tomar una decisión y formular recomendaciones de mejora tomando en cuenta todas las deficiencias detectadas. Al final, la información obtenida en esta fase debe ser cotejada con la documentación proporcionada por las empresas (planes de proyecto, plantillas, proyectos, etc.) y mapeada a través de entrevistas entre los alumnos y los jefes de proyecto como parte de la evidencia objetiva. Los resultados obtenidos en las empresas son mostrados con gráficos de barras, que resumen el porcentaje de prácticas efectivas que es cubierto por los procesos evaluados en la iniciativa de mejora, y gráficas de Keviat, que muestran el nivel de madurez/capacidad de todos los procesos evaluados en cada empresa. Esta información es analizada con la alta dirección para (1) reforzar el compromiso establecido en PI 1 y (2) recomendar mejoras que deben ser priorizadas en función de los objetivos de negocio.

3.2.3. PI 3: ¿Qué tan buena es la comprensión de un plan de mejora en el contexto general de la empresa?

La respuesta de esta pregunta da inicio a la mejora del proceso a través de la *fase de establecimiento*. El estudiante y el profesor ayudan a la alta dirección a establecer prioridades de acuerdo a sus objetivos actuales de negocio, desarrollar una estrategia de mejora con objetivos medibles (cuantificables) de éxito (por ejemplo, reducir en un 15% el tiempo de calendario, reducir el tiempo para la detección de defectos en un 30%, etc.) y, finalmente, generar planes de acción para mejorar el proceso actual con los datos obtenidos en la fase anterior. El resultado de estas actividades es un plan detallado que se compone de once elementos: acciones específicas, calendario, hitos relevantes, entregables, puntos de decisión, recursos, responsabilidades, medidas objetivo, mecanismos de seguimiento, y estrategias de gestión y mitigación de riesgos.

Un plan de acción es similar a una lista de comprobación que proporciona una herramienta de verificación para registrar cuando una actividad de mejora sea realizada. En este sentido, cómo se trata de gestionar diferente información, la herramienta computacional que de soporte al curso debe ayudar a los estudiantes a generar planes detallados de una manera fácil y sencilla. Sin embargo, dada la nula experiencia de los estudiantes al utilizar modelos de referencia en un ámbito organizacional, es natural considerar el conocimiento del profesor y la experiencia del jefe de proyectos para proporcionar apoyo en la planificación, el seguimiento y el control de las actividades de mejora. Así, un plan de mejora puede integrar diferentes planes de acción (dependiendo la cantidad de procesos que se busque mejorar) organizando la información o conocimiento obtenido durante la fase de evaluación y utilizando la base de datos creada. La conversión de los niveles de rendimiento identifica aspectos importantes del ámbito de conocimiento, por ejemplo, los productos, los atributos, las actividades y las asociaciones con otras áreas de proceso que identifican nociones similares y estos son conglomerados en base a sus atributos. Es así que en el contexto del curso, las actividades registradas por cada empresa son representadas en forma de once áreas de conocimiento y son almacenadas en un modelo relacional de datos de los procesos [Syazwan et al., 2002], con el objetivo de facilitar la comprensión de la red de colaboración.

3.2.4. PI 4: ¿Cómo puede una empresa lograr un mayor rendimiento en su proceso de software con la iniciativa de mejora?

Al inicio del trabajo práctico, los estudiantes realizan una evaluación inicial (antes de la aplicación de los planes de acción) para establecer la capacidad y madurez iniciales de los procesos de la empresa de acuerdo a MoProSoft. Posteriormente, una segunda evaluación (después de la aplicación de los planes de acción o *fase de ejecución*) es realizada para determinar el grado de mejora. En este sentido, las empresas deben lograr un incremento medio en los niveles de capacidad de sus procesos que se verá reflejado en el nivel de cobertura obtenido en PI 2. Por ejemplo, una empresa puede tener una cobertura inicial de 54% en el proceso de APE cuando se realiza la primera evaluación. Una re-evaluación tiene que mostrar un aumento de *k* durante cada ciclo de mejora en el proceso, de tal forma que se obtenga un mejor nivel de cobertura representado por de 54% + 54%*k. El sesgo del personal es controlado mediante verificaciones de la documentación y entrevistas con la alta dirección y los grupos de personas, como en las evaluaciones. Así, los estudiantes utilizan los niveles de cobertura y los objetivos medibles establecidos en PI 3 para lograr un mayor rendimiento del proceso relacionado con la iniciativa de mejora.

3.2.5. PI 5: ¿Existe una correlación entre la cantidad de esfuerzo que se requiere para entender la iniciativa de mejora y el promedio ganado?

Los estudiantes responden a esta pregunta registrando durante el programa de mejora el número de empleados, el esfuerzo total de mejora (en horas) y el esfuerzo por persona para cada empresa. En este sentido, es interesante que los estudiantes tomen nota de la relación que existe entre el esfuerzo dedicado por persona y la mejora promedio del proceso en el contexto de una iniciativa de mejora, puesto que esto les permitirá detectar problemas en iniciativas reales. Normalmente, la experiencia indica que las empresas que invierten la mayor cantidad de horas por persona logran un mayor incremento de mejora en sus procesos, pero en el contexto de las pequeñas empresas nada es seguro.

El compromiso, la iniciativa y la responsabilidad también son factores que contribuyen de manera importante al entendimiento de los datos analizados por los estudiantes en el contexto de mejora. Así pues, la herramienta computacional de soporte debe proporcionar un mecanismo de comunicación que asegure entre los participantes la importancia de la iniciativa de mejora, no solamente para el líder, sino a todo el equipo involucrado en la misma. En este mismo contexto, los estudiantes aprenden que el nivel de compromiso y entendimiento necesarios para lograr una mejor educación en esta área, son cruciales para reportar experiencias exitosas.

Así pues, la aplicación práctica y colaborativa de las iniciativas de mejora permite obtener una combinación de experiencias valiosas, un mayor conocimiento sobre la forma de realizarlas (*know how*) y la observación del comportamiento de las pequeñas empresas durante el proceso de mejora dentro de los grupos de estudiantes. A continuación, siguiendo las pautas establecidas anteriormente para el curso con enfoque colaborativo, se establece una estrategia de diseño para construir el soporte computacional necesario para desarrollar las preguntas de investigación propuestas.

3.3. Diseño del soporte computacional para el curso colaborativo

En la educación de las Ciencias de la Computación, las tareas prácticas son tan importantes como las teóricas. De acuerdo con Gillet et al., *"en el espíritu del aprendizaje flexible, los estudiantes tienen la posibilidad de experimentar en cualquier momento y desde cualquier lugar, beneficiándose así de una experiencia cognitiva más eficaz"* [Gillet et al., 2003]. Sin embargo, la

educación universitaria en general ha sufrido del marcado énfasis en la adquisición individual de conocimientos y habilidades en el entorno académico, mientras que la sociedad y la industria claman por resultados que a menudo sólo pueden lograrse en auténticos contextos colaborativos de aprendizaje [Kirschner, 2004]. Pero, ¿por qué los contextos o entornos de colaboración?, la naturaleza de las áreas de las Ciencias de la Computación implica la colaboración como un imperativo lógico. Normalmente, el conocimiento o habilidad necesarios para realizar una tarea no reside en la mente de una sola persona, o bien la tarea es demasiado grande para que sea realizada por una sola persona. En este sentido, Kreijns et al., afirman que el trabajar con una computadora dentro de un entorno colaborativo tiene consecuencias positivas tanto en términos del aprendizaje (dimensión educativa) y desempeño social (dimensión psicológica/social), además de la expectativa de mejoras en la satisfacción de los mismos estudiantes [Kreijns et al., 2007]. En este contexto, Roschelle y Teasley afirmaron que: *"La cooperación se lleva a cabo por la división del trabajo entre los participantes, como una actividad donde cada persona es responsable de una parte de la solución de un problema…"*, mientras que el aprendizaje colaborativo involucra *"el mutuo compromiso de los participantes en un esfuerzo coordinado para resolver juntos un problema"* [Roschelle & Teasley, 1995]. Por lo tanto, el aprendizaje colaborativo ha sido cada vez más utilizado y el advenimiento de las tecnologías de la información ha hecho posible la colaboración a través de la computadora. Concretamente, en el aprendizaje colaborativo el éxito de cualquier estudiante ayuda a los otros a terminar un trabajo en equipo para llevar a cabo un objetivo común, en lugar de las clases tradicionales donde los estudiantes compiten por el éxito. Así, las investigaciones realizadas en [Bustos & Nussbaum, 2009; Casas et al., 2010; Regueras et al., 2011] establecen que un entorno de colaboración reúne un conjunto de características que lo distinguen de los entornos convencionales, por ejemplo:

- Interdependencia positiva: cada estudiante debe percibir su necesidad de la colaboración de otros para completar las actividades del grupo.
- Responsabilidad individual: la calidad y la cantidad de la contribución de cada miembro del grupo deben ser evaluadas y los resultados deben reportarse tanto al grupo como al individuo.
- Interacción promotora cara-a-cara: cada estudiante depende de los otros miembros del grupo y por lo tanto tiene que ayudar, estimular y apoyar sus esfuerzos. Los maestros también deben alentar a los estudiantes a ayudarse unos a otros.
- Habilidades sociales: para trabajar con eficacia, los estudiantes deben desarrollar y utilizar habilidades sociales como el liderazgo, la creación de confianza, la toma de decisiones, la comunicación y la gestión de conflictos.
- Procesamiento de grupo: para mejorar el procesamiento de grupo, los estudiantes deben evaluar periódicamente lo que están haciendo como equipo e identificar qué cambios son necesarios con el objetivo de trabajar de manera más eficaz en el futuro.

Como parte de un equipo, los estudiantes de las Ciencias de la Computación deben aprender a cooperar para resolver problemas de ingeniería y a cómo manejarse en discusiones sobre temas profesionales, y argumentar y explicar sus puntos de vista en términos científicos. La premisa detrás de esto es que la presentación de un argumento es una manera eficaz y activa[11] de aprender. Así,

[11] De acuerdo con Bonwell y Sutherland, un aprendizaje eficaz y activo "implica que los estudiantes hagan bien las cosas y piensen sobre las cosas que están haciendo." Es decir, los términos "efectivo y activo" están relacionados con

bajo un entorno de colaboración los estudiantes pueden ser apoyados por un entorno activo donde las ideas se producen continuamente para ayudar a los demás a comprender los complejos conceptos de la Mejora del Proceso de Software. En el contexto de la educación superior, Resta y Laferrière [Resta & Laferrière, 2007] han identificado los factores críticos en el uso de la tecnología, como apoyo a entornos de colaboración y el avance del conocimiento, que se pueden aplicar a esta propuesta de tesis: el compromiso del estudiante; el escalamiento del profesor para el desarrollo de una orientación hacia la explicación en el discurso de los estudiantes; las estrategias pedagógicas para transformar una aula tradicional en una comunidad para la construcción del conocimiento; y la revisión entre pares.

A efectos de esta tesis, han sido adoptados los cinco elementos principales postulados por Johnson y Johnson [Johnson & Johnson, 1994] para introducir el aprendizaje colaborativo, que se resumen a continuación:

- La *interdependencia positiva* alude a la creencia de que el grupo de estudiantes puede tener éxito si todos los miembros del grupo combinan sus esfuerzos hacia el logro de objetivos comunes relacionados con un área en particular.
- *Fomentar la interacción* asegura que los estudiantes animan y ayudan a otros a entender los temas básicos y avanzados de un área en particular.
- Con la *responsabilidad individual* cada estudiante es responsable y un elemento clave para alcanzar los resultados de aprendizaje del grupo que hayan sido definidos por el profesor.
- El uso de *habilidades interpersonales*, evidentemente, es de especial importancia para el éxito del grupo, puesto que los estudiantes tienen que confiar y respetarse entre ellos, comunicarse bien y manejar los conflictos.
- La *monitorización del progreso* genera una verdadera sensación de logro en base a la experimentación realista con los estudiantes, que son controlados por sus maestros.

Estos elementos promueven la idea de que cuando el aprendizaje colaborativo es adoptado, la competencia entre los estudiantes es eliminada y es sustituida por la colaboración. Por lo tanto, la idea principal es que los alumnos que obtienen un mayor conocimiento sobre la Mejora del Proceso de Software ayudan al resto y prosperan para ser un líder e influenciar a sus compañeros de equipo. Por otra parte, en relación a la aplicación de la tecnología en la educación de pregrado y posgrado, es común referirse a los siete principios de Chickering y Ehrmann [Chickering & Ehrmann, 1987] dado que han sido ampliamente adoptados. Dos de los principios de Chickering que se relacionan directamente con la propuesta del enfoque colaborativo propuesto en esta tesis son: *"La práctica desarrolla la reciprocidad y la cooperación entre los estudiantes"* y *"Las buenas prácticas emplean técnicas de aprendizaje activo"*. En este sentido, es necesario crear un entorno computacional colaborativo que introduzca el uso de la experimentación a distancia para apoyar a este último principio.

Así, una vez que el enfoque de colaboración ha sido definido para cubrir los objetivos educativos de la Mejora del Proceso de Software, es necesario definir la estrategia de diseño que permita crear la herramienta computacional de soporte. Dadas las características de esta herramienta, es recomendable introducir el modelo incremental propuesto por Mills et al. [Mills et al., 1980]. El

la genuina participación de los estudiantes en el proceso de aprendizaje a través de las actividades relacionadas con el debate y la resolución de problemas, así como la dinámica de cooperación [Bonwell & Sutherland, 1996].

modelo surgió como una estrategia para reducir la repetición del trabajo durante el proceso de desarrollo y dar la oportunidad a los desarrolladores de retrasar la toma de decisiones en relación a los requisitos funcionales hasta adquirir experiencia con el sistema.

En este sentido, el modelo incremental combina elementos del modelo lineal secuencial (aplicado repetidamente) con la filosofía iterativa de construir prototipos rápidos. La diferencia básica con el modelo lineal, reside en que el modelo incremental aplica secuencias lineales de forma escalonada mientras progresa en el tiempo del calendario planeado. Así, cada secuencia lineal produce un incremento del software. Para el desarrollo de esta herramienta, bajo el paradigma incremental, se identifican las fases genéricas del proceso de desarrollo de software que son: análisis, diseño, codificación y pruebas (véase Figura 3.7). La simplificación de funcionalidades y su elección para cada incremento, corresponde con la naturaleza del modelo incremental. Cuando éste es utilizado, el primer incremento a menudo es un producto esencial, es decir, aquel que representa el punto de partida para comenzar a utilizar el software desarrollado. De esta forma, el desarrollador afronta requisitos funcionales básicos al inicio, pero muchas funcionalidades suplementarias (algunas conocidas, otras no) quedan sin ser extraídas. Por lo tanto, como resultado de la utilización y/o evaluación de este primer incremento, se desarrolla un producto utilizable para alcanzar el siguiente. Dicho plan encara la modificación del producto central a fin de cumplir las necesidades de los usuarios, y la entrega de funcionalidades y características adicionales. Este proceso se repite siguiendo la entrega de cada incremento, hasta que se elabora la herramienta completa.

A diferencia de otros modelos, el modelo incremental se centra en la entrega de un producto operacional al término de cada incremento. Es por esto que se menciona que los primeros incrementos son versiones "incompletas" de la herramienta final, pero proporcionan al usuario la funcionalidad que precisa y también una plataforma para la evaluación. Es por esto que el desarrollo incremental es especialmente útil para el desarrollo de la solución propuesta en esta tesis.

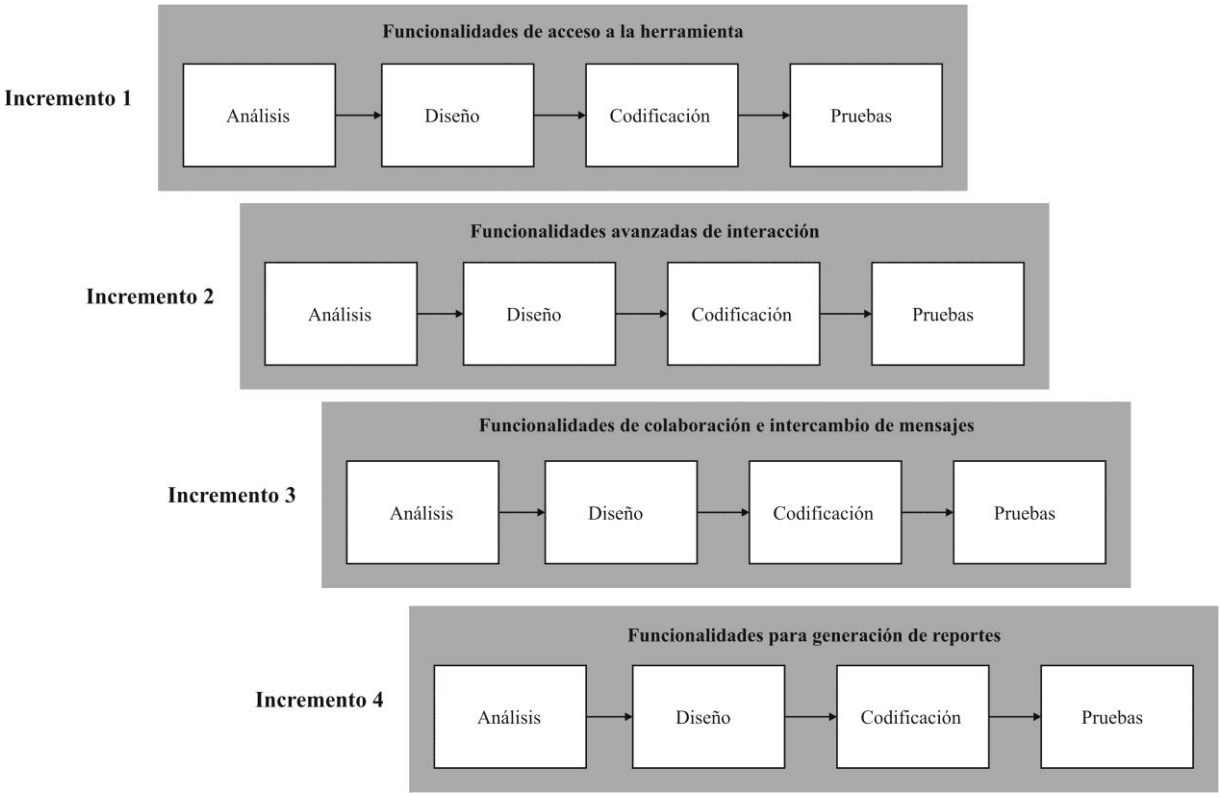


Figura 3.7. Estrategia incremental de desarrollo y división de funcionalidades

Con el objetivo de presentar el diseño implementado para la construcción de la herramienta software, denominada HESESPI (**HE**rramienta de **S**oporte para la **E**nseñanza de **SPI**), el Anexo B se organiza de la siguiente forma:

- Se expone la solución tecnológica planteada para la construcción de la herramienta computacional con el objetivo de justificar las herramientas tecnológicas utilizadas para su construcción.
- Se continúa presentando la primera fase del modelo iterativo mediante el análisis del sistema, lo que permite establecer los requisitos de la herramienta. Para esto, el Anexo B presenta los requisitos funcionales y los casos de uso de acuerdo al estándar IEEE 830-1998 [IEEE, 1998].
- El mismo Anexo B presenta el diseño de alto nivel, a través del modelo entidad-relación para la base de datos y el detalle de los casos de uso, y el diseño de bajo nivel a través de la descripción de tablas y atributos.

El diseño de la herramienta se basa en la arquitectura cliente-servidor como se muestra en la Figura 3.8. Cabe mencionar que todas las tecnologías utilizadas son OpenSource y pueden ser fácilmente descargadas e instaladas en cualquier sistema operativo. El servidor debe alojar a la herramienta computacional para que sea accedida desde cualquier computadora conectada a una red local o a Internet. En relación al hardware que se requiere instalar en el servidor para el correcto funcionamiento de la aplicación, éste al menos debe cubrir los requerimientos mínimos (al menos 100 Mb de espacio libre en disco, procesador Dual Core 1.8 Ghz o superior, 1 Gb Memoria RAM). En cuanto al sistema operativo existe un alto grado de portabilidad dado que la tecnología utilizada para construir la aplicación funciona con Windows, Linux y MacOSX. Por último, el servidor debe tener instalado PHP 5.2.X, Apache 2.X y MySQL 5.X. En cuanto a los *frameworks* de desarrollo, se utilizaron las últimas versiones más estables de Extjs 4.2.1 y CodeIgniter 2.1.4.

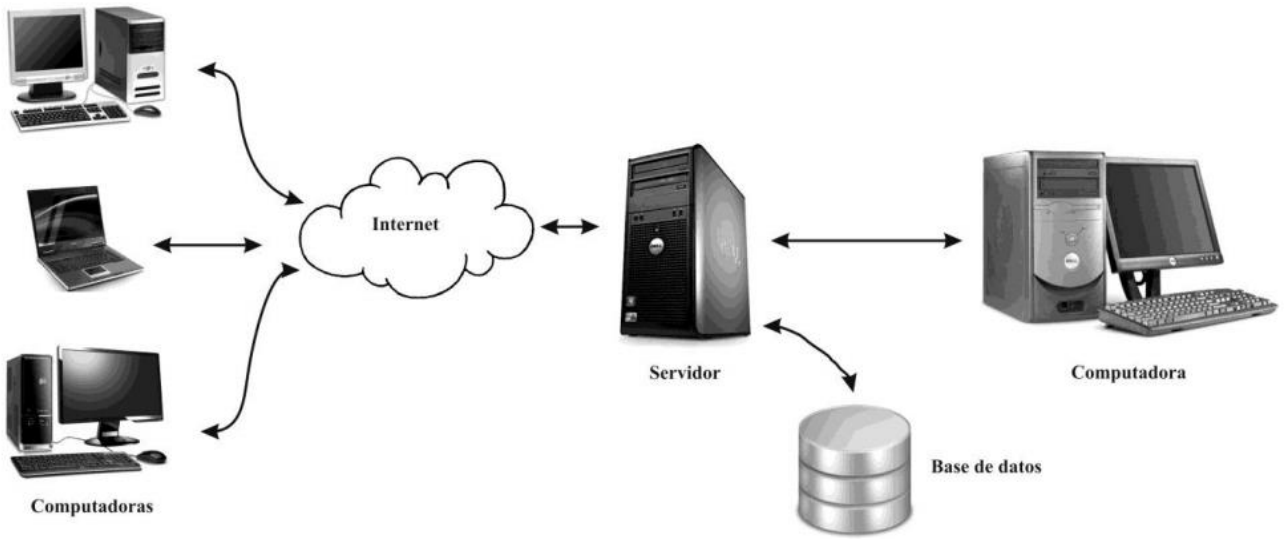


Figura 3.8. Arquitectura propuesta para soportar la estrategia incremental de desarrollo

4. Resultados de la experimentación

Entender cómo implementar exitosamente una iniciativa de mejora es, sin duda alguna, uno de los problemas más difíciles que enfrenta actualmente la educación en el campo de la Ingeniería de Software. Es verdad que la literatura relacionada con la Mejora del Proceso Software resume muchos casos reales de estudio y proporciona información sobre programas exitosos de mejora. Sin embargo, han habido pocos intentos sistemáticos por sintetizar y organizar un curso colaborativo con soporte computacional para enseñar el "cómo" a través de lecciones que incluyen experimentos reales. Con este objetivo en mente se ha realizado una evaluación sobre el enfoque propuesto, con la intención de validar el nivel de ayuda en la enseñanza sobre la mejora de procesos, en colaboración con la industria local. Esta evaluación incluyó la aplicación de cuestionarios y entrevistas de seguimiento. Los cuestionarios consistieron de catorce preguntas y se incluyeron instrucciones para que los estudiantes las comprendieran y respondieran adecuadamente. El diseño de este cuestionario se basó en las preguntas de investigación planteadas en el capítulo anterior, las cuales fueron consideradas en el desarrollo del enfoque colaborativo para motivar a los estudiantes y mantener la colaboración con las empresas en todas las iniciativas de mejora. Los detalles de esta experimentación son proporcionados a continuación.

4.1. Participantes

En la experimentación participaron un total de 48 estudiantes distribuidos de la siguiente manera: de nivel maestría participaron 16 estudiantes, 4 de ellos adscritos al Programa de Maestría en Tecnologías de Cómputo Aplicado de la Universidad Tecnológica de la Mixteca y 12 al Programa de Maestría en Sistemas Computacionales de la Universidad Autónoma de Chiapas; de nivel doctorado participaron 28 estudiantes de la Universidad del Sur en Tuxtla Gutiérrez, Chiapas; y de nivel licenciatura participaron 4 estudiantes de noveno semestre de la Licenciatura en Informática de Nova Universitas, incorporada también al Sistema de Universidades del Estado de Oaxaca (SUNEO). El curso propuesto en la tesis se realizó durante un semestre y en un principio se concentró en temas teóricos adoptando un método educativo clásico mediante exposiciones en aula, discusiones y revisión de casos reales de estudio. Como resultado, los estudiantes adquirieron un conocimiento básico sobre temas generales relacionados con la mejora de procesos, pero tuvieron problemas al intentar aplicarlos en ejercicios dentro del aula y al establecer los lineamientos para iniciar, conducir y cerrar un proyecto de mejora de procesos. Así, siguiendo un diseño cuasi-experimental y dado el número de estudiantes que participaron en la evaluación del enfoque propuesto, se formaron 12 equipos con 4 estudiantes en cada equipo de acuerdo a los lineamientos establecidos por Göl y Nafalski [Göl & Nafalski, 2007]. Los equipos de estudiantes desarrollaron teóricamente ideas sobre la mejora de procesos para implementar diversas iniciativas en la red de colaboración de la industria (véase Figura 4.1). Como se mencionó anteriormente, esta red se

compone de cinco empresas medianas y cada una proporcionó dos jefes de proyectos para participar en las iniciativas de mejora e información de dos proyectos pequeños[12] para ser evaluados.

Las características de estas empresas son:

- El promedio de número de personal de las empresas es de 20 trabajadores dedicados al desarrollo de software.
- Las empresas se enfocan actualmente al desarrollo de software a la medida y portales web con el fin de proporcionar soluciones de software y de consultoría.
- En cuanto a la experiencia previa sobre la mejora de procesos, las empresas tienen un conocimiento limitado de los modelos CMMI-DEV y MoProSoft, y se ha intentado seguir (no implementar) algunas prácticas relacionadas con la planificación y control de los proyectos de software en sus actividades actuales. Todas estas prácticas utilizan solamente un repositorio de proyectos con datos del esfuerzo destinado por el personal para gestionar el proceso.

Figura 4.1. Análisis de ideas con un equipo de estudiantes

De hecho, los principales criterios de selección de las empresas incorporadas a la red se relacionan con su interés en colaborar, su tamaño, y su antigüedad (al menos 5 años de experiencia). Para mantener este interés y el compromiso de estos colaboradores de la industria local, los profesores de la red proporcionaron y continúan brindando capacitación continua a las empresas y apoyo en el campo de la mejora de procesos. Como valor agregado, las empresas obtuvieron los conocimientos necesarios para realizar autoevaluaciones de sus procesos y, si lo desean, son capacitadas de forma gratuita para una evaluación formal o certificación oficial. Por último, seis

[12] En el contexto de las pequeñas y medianas empresas desarrolladoras de software, un proyecto pequeño tiende a ser más viable por su capacidad de realizarse dentro del calendario, presupuesto, y productividad establecidos. Sin embargo, no existe evidencia de su calidad. De acuerdo con el Instituto de Ingeniería de Software *"los proyectos pequeños son identificados como los proyectos que tienen 50,000-100,000 LOC, un plazo de entre 6 a 12 meses, y cuentan con diez o menos programadores"*. [CMMI, 2010].

profesores participaron en el estudio, cuatro con doctorado en Ingeniería de Software y dos con grado de maestría y entre 6 y 8 años de experiencia en la enseñanza en el área.

4.2. Instrumentos

La experimentación incorporó un mecanismo de evaluación para determinar la situación actual del proceso de software en las MiPyMEs incorporadas a la red [Garcia et al., 2007]. Los profesores seleccionaron al azar una empresa para cada equipo de estudiantes y los jefes de proyectos entregaron toda la información relacionada con los proyectos de software acordados, con el objetivo de contar con evidencia objetiva (planes del proyecto, cronogramas, presupuesto, artefactos de análisis y diseño, entregables, etc.) como soporte a la evaluación. Como parte de la experimentación, los profesores realizaron pruebas escritas durante el semestre escolar y se solicitaron reportes de trabajo para determinar el grado de mejora de los estudiantes. En este sentido, se decidió adoptar el método de auto-cuestionamiento metacognitivo IMPROVE [Kramarski & Mizrachi, 2006], que representa el acrónimo de todos los pasos realizados para la enseñanza en el aula: **I**ntroducción de nuevos conceptos; cuestionamiento **M**etacognitivo; **P**ráctica con grupos pequeños; **R**evisión; **O**btención de experiencia; **V**erificación; y el **E**nriquecimiento a través del conocimiento. Este cuestionamiento metacognitivo animó a los estudiantes a participar activamente en la auto-regulación de su aprendizaje mediante el uso de cuatro tipos de preguntas: de comprensión (e.g., "¿Cuál es el problema observado en la empresa?"); de conexión (e.g., "¿Cuál es la diferencia entre estas dos representaciones (la caótica o situación actual y la propuesta de mejora)?"); de estrategia (e.g., "¿Cuál es la estrategia para resolver el problema?") y de reflexión (e.g., "¿Tiene sentido mi solución?"; "¿Puede mi solución ser presentada de otra manera? "). Del mismo modo, se distribuyó un cuestionario entre los estudiantes para explorar las actitudes hacia el uso de la herramienta computacional creada como soporte del curso colaborativo, y evaluar cuatro categorías: la conducción del trabajo en equipo, la creación de un entorno positivo en el equipo, la racionalización de la participación de la industria, y el fortalecimiento del entorno de colaboración. Este instrumento de evaluación fue diseñado para interpretar el aprendizaje de los estudiantes al ser expuestos a problemas reales de las empresas como medio para obtener experiencia práctica. Las calificaciones siguieron una escala Likert de cuatro puntos [Likert, 1932] con "Totalmente en desacuerdo", "En desacuerdo", "De acuerdo", y "Totalmente de acuerdo" como respuestas. Esta escala de 4 puntos fue seleccionada intencionalmente para eliminar el factor de indecisión entre los estudiantes utilizando "elementos socialmente deseables" como una medida semi-obligatoria para permitir al estudiante elegir un lado (positivo o negativo) [Weisberg, 2009].

4.3. Hallazgos

Dentro de la experimentación se establecieron cuatro categorías de preguntas específicas a las características de las iniciativas de mejora del proceso y del trabajo en equipo. Las cuatro categorías fueron las siguientes:

1. Conducción del trabajo en equipo.
2. Creación de un entorno positivo en el equipo.
3. Fortalecimiento de la participación de la industria.
4. Fortalecimiento del entorno de colaboración.

En este sentido, se pidió a los equipos de estudiantes que respondieran al cuestionario de 4 puntos de escala Likert con el objetivo de evaluar cada categoría. Después de la encuesta, se

realizaron entrevistas de seguimiento para revisar y debatir en profundidad los resultados obtenidos. Es importante mencionar que esto se consideró necesario dado que la encuesta fue diseñada para determinar el grado de aceptación del enfoque propuesto en esta tesis en base a las preguntas; por lo tanto, las entrevistas fueron llevadas a cabo para explorar más a fondo cómo el curso colaborativo ayudó a los estudiantes a contestar las preguntas de investigación. El procedimiento llevado a cabo para analizar los datos obtenidos a partir de las preguntas del cuestionario representa la gama de respuestas en términos del porcentaje de encuestados quienes expresaron estar de acuerdo, en desacuerdo, etc., para cada categoría definida dentro de la encuesta. Es importante mencionar que, hasta ahora, el enfoque propuesto no ha sido utilizado en algún experimento práctico que haya sido realizado en cursos de dos años escolares consecutivos. En este sentido, los niveles 1 y 2 de MoProSoft se componen de diferentes áreas de proceso que es necesario evaluar en una empresa y, como cada empresa es diferente, todas las iniciativas de mejora (y, como consecuencia, todos los experimentos) son diferentes. Las preguntas de investigación establecidas permiten que los estudiantes obtengan información específica acerca de la empresa asignada y la situación actual de sus procesos, y proporcionen recomendaciones personalizadas a ésta. Así, es posible que como trabajo futuro se introduzcan preguntas de un modelo de referencia diferente para aumentar el número de empresas en la red. Las siguientes secciones resumen las preguntas y los resultados obtenidos con la encuesta.

4.3.1. Conducción del trabajo en equipo

Como lo indica la Tabla 15, la mayor parte de los estudiantes (aproximadamente el 90%) indicó estar “De acuerdo” y “Totalmente de acuerdo” en que el curso propuesto fue beneficioso al guiar la participación de su equipo de trabajo en los proyectos de mejora. Durante el debate, las preguntas 1, 3 y 4 obtuvieron una respuesta positiva. Estas preguntas están relacionadas con el aprendizaje del estudiante sobre cómo diseñar y realizar iniciativas de mejora en equipo. A medida que los estudiantes avanzaron en el curso, la visualización constante de la iniciativa de mejora les ayudó a entender mejor la situación actual del proceso de las empresas de software (pregunta 2), que, de hecho, han fortificado actividades *ad hoc* y tareas informales del trabajo en equipo que habitualmente se realizan a través de reuniones formales [Hamann, 2006].

Tabla 15. Resultados obtenidos para la categoría relacionada a la conducción del trabajo en equipo

Elementos de evaluación	Totalmente de acuerdo		De acuerdo		En desacuerdo		Totalmente en desacuerdo	
	#	%	#	%	#	%	#	%
1. Este curso me ayudó a aprender cómo diseñar e implementar una iniciativa de mejora de procesos.	35	73.0	8	17.0	2	4.0	3	6.0
2. HESESPI me ayudó a visualizar el progreso de la iniciativa de mejora de procesos.	38	79.0	4	8.0	5	11.0	1	2.0
3. HESESPI estableció un entorno interactivo para el trabajo colaborativo con la red de empresas.	30	63.0	12	25.0	3	6.0	3	6.0
4. El enfoque utilizado en el curso me ayudó a aprender mejor cómo y qué hacen los miembros de un equipo durante una iniciativa real de mejora.	28	58.0	15	32.0	3	6.0	2	4.0
5. HESESPI ayudó directamente a mi equipo de trabajo guiándonos paso a paso a través de las sesiones del tutorial.	36	75.0	10	21.0	2	4.0	0	0.0

Es importante también mencionar que aproximadamente el 12% de los estudiantes no coincidió con la idea de que el enfoque propuesto y su soporte computacional fueran beneficiosos para su aprendizaje. En este sentido, se debe reconocer que los principales obstáculos a los que se enfrentó este enfoque colaborativo fueron la resistencia al cambio en los paradigmas de trabajo en equipo por parte de los estudiantes, y el buen diseño de la interfaz de HESESPI para el trabajo mismo. De manera similar, se debe recalcar que no todas las personas aprenden o generan su conocimiento de la misma forma, es por eso que se buscó ofrecer un recurso que permitiera, al estudiante, comunicarse con el resto de su equipo. Sin embargo, es importante integrar otros mecanismos, además del chat y la librería electrónica, que faciliten la correcta integración dentro de los equipos de trabajo y las tareas especificas asignadas a cada miembro.

Finalmente las sesiones de tutoría a través de HESESPI aseguraron que las colaboraciones entre estudiantes, profesores y jefes de proyectos fueran claramente compatibles dentro de un proyecto de mejora. Este soporte es establecido desde el inicio de la fase de planificación de la iniciativa, donde se visualiza mejor el estado del compromiso entre los individuos, lo que ayuda a explorar y reducir el rechazo cultural que sufren las empresas cuando son evaluadas. Esto es particularmente confirmado por las respuestas a las preguntas 2 y 5, que fueron diseñadas para determinar si la herramienta computacional ayuda o no a resolver los problemas de comunicación/entendimiento entre los estudiantes y la industria en relación a su proceso de software.

4.3.2. Creación de un entorno positivo en el equipo

Para determinar si los estudiantes mostraron entusiasmo al trabajar como equipo en un proyecto diferente, se estableció la pregunta 1 de la Tabla 16 con el objetivo de comprobar si HESESPI había ayudado o no a fomentar la colaboración y la creación de un entorno positivo de equipo. También se utilizó la pregunta 2 para verificar el factor común de confusión sobre que los buenos amigos hacen buenos equipos, y evitar así el sesgo en la evaluación [Wichstraum, 1995]; en este sentido, la pregunta tiene por objeto determinar la forma en que HESESPI ayuda a crear un entorno de trabajo en equipo. Hoy en día, en los cursos relacionados con la Ingeniería de Software es cada vez más común que los estudiantes resuelvan problemas comunicándose y trabajando con otros [Budimac et al., 2011; Chen & Chong, 2011]. Con el objetivo de fomentar una mayor participación y práctica en temas de calidad de la empresa y del proceso, los estudiantes jugaron roles que facilitan su visión tales como líder de equipo, líder de la iniciativa de mejora de procesos y miembro del equipo de evaluación. El objeto de este estímulo es ayudar a los estudiantes a adquirir el conocimiento y la experiencia de sus compañeros [Fuller & Moreno, 2004]. La pregunta 3 fue utilizada para investigar cómo el individualismo de los estudiantes y el desarrollo del equipo, junto con el proceso y las variables contextuales, estaban relacionados con la eficacia de los equipos. En particular, la relación positiva entre los estudiantes y la motivación de colaborar en equipo se incrementó a medida que los equipos trabajaron en condiciones más realistas.

Los resultados obtenidos demostraron que la relación ente la motivación en el trabajo y los comportamientos del equipo (ayudar, compartir, contribuir, e innovar) fue más positiva en equipos más maduros. Estos resultados son soportados por la investigación de [Janz et al., 1997], a través de la corroboración de que los comportamientos de los miembros del equipo se relacionan positivamente con la eficacia, pero esta relación se hace más positiva en presencia de ciertos factores contextuales (e.g., metas realísticas establecidas por la red de la industria local y una comunicación eficaz), y menos positiva en la ausencia de la colaboración.

Tabla 16. Resultados obtenidos para la categoría relacionada a la creación de un entorno positivo en el equipo

Elementos de evaluación	Totalmente de acuerdo		De acuerdo		En desacuerdo		Totalmente en desacuerdo	
	#	%	#	%	#	%	#	%
1. Me gustaría trabajar con mis compañeros de equipo otra vez.	26	55.0	18	37.0	4	8.0	0	0.0
2. Conocía bien a mis compañeros de equipo desde antes que el proyecto de mejora comenzara.	12	25.0	22	46.0	10	21.0	4	8.0
3. Me sentí en plena confianza de compartir mis ideas con el equipo.	36	75.0	11	23.0	1	2.0	0	0.0

4.3.3. Fortalecimiento de la participación de la industria

La Tabla 17 muestra que los estudiantes también sintieron que el uso de HESESPI durante el curso ayudó a fortalecer la participación de la industria y ayudó a organizar las discusiones entre el equipo (preguntas 1, 2, y 3). Durante las entrevistas con los estudiantes destacaron dos respuestas. La primera fue la siguiente: "*En la fase de pilotaje, el maestro dirigió on-line nuestras ideas para comprender las debilidades identificadas en el proceso de software de las empresas, antes de que se las presentáramos a los jefes de proyectos. Pero lo interesante es que también a través de la herramienta obtuvimos la realimentación a través de la comunicación con estos jefes de proyectos y así podíamos discutir sus comentarios en la próxima clase*". Como esta afirmación lo indica, HESESPI permitió al profesor, mediar la colaboración y monitorear cómo trabajan los alumnos en colaboración con su empresa durante el curso, lo que ayudó a mantener las actividades dentro del plan establecido. La segunda respuesta que surgió de la discusión arrojó la razón de este éxito: "*Las sesiones de evaluación requirieron que aprendiéramos a recoger y analizar las respuestas concentradas en los cuestionarios; estos datos junto con la documentación objetiva (evidencia de los proyectos) sirvieron para presentar en clase la evolución del proceso de software de las empresas con la autorización previa de los jefes de proyectos*". Es evidente que a través de estas sesiones, los estudiantes aprendieron a construir pruebas objetivas de las empresas de una manera organizada, y luego se comunicaron constantemente con los responsables de cada proyecto al reunir pruebas y evidencias.

Tabla 17. Resultados obtenidos para la categoría relacionada al fortalecimiento de la participación de la industria

Elementos de evaluación	Totalmente de acuerdo		De acuerdo		En desacuerdo		Totalmente en desacuerdo	
	#	%	#	%	#	%	#	%
1. El curso me ayudó a beneficiarme del conocimiento y participación de la red de colaboración de la industria.	33	69.0	15	31.0	0	0.0	0	0.0
2. HESESPI me ayudó a fortalecer la colaboración y mejorar la discusión con los jefes de proyectos de la industria local de software.	25	52.0	10	21.0	7	15.0	6	12.0
3. El curso soporta las lecciones de "cómo hacerlo" a través de experimentos realistas en la red de colaboración de empresas.	30	63.0	18	37.0	0	0	0	0.0

4.3.4. Fortalecimiento del entorno de colaboración

De acuerdo con [Iversen et al., 2004], la mejora de los procesos de software es altamente demandante y dinámica. Durante su desarrollo, una variedad de problemas dificultan el desarrollo e implementación de los nuevos procesos. Por ejemplo, el jefe de proyectos puede solicitar un conocimiento específico para la reasignación de las personas que poseen el conocimiento básico sobre cómo aplicar procesos maduros y los métodos de mejora. En este sentido, HESESPI proporcionó un espacio educativo de trabajo para que los estudiantes manejaran este tipo de actividades dinámicas. En la discusión sobre una colaboración continua, que corresponde a las preguntas 1, 2 y 3 de la Tabla 18, los estudiantes respondieron que utilizaron el módulo de modelado que la herramienta computacional proporciona para controlar los cambios en los procesos de software de las empresas, lo que les permitió obtener una mejor comprensión de cómo trabaja la industria. Además, los estudiantes argumentaron que les gustó utilizar HESESPI puesto que introduce el uso de notificaciones automáticas a todos los participantes en la iniciativa de mejora, y también solicita continuamente que se verifique si un proceso o modificación debe ser revisado (una actualización menor) o reelaborado. Se determinó que esta información es útil para ayudar a los estudiantes a mantener la colaboración. HESESPI se utiliza no sólo para la realización de las tareas colaborativas del modelado, sino también para ayudar a mantener el esfuerzo requerido del grupo de trabajo en el entorno dinámico de las mejoras en los procesos.

En cuanto a los resultados obtenidos de la pregunta 3, el 85% de los encuestados consideró que les gustaría usar de nuevo la herramienta. En una discusión posterior sobre cómo HESESPI había ayudado en el fortalecimiento de la colaboración, se encontró que los estudiantes consideraron que es una herramienta útil para ayudarles a establecer y conducir iniciativas de mejora en colaboración con la industria. Por ejemplo, a la mayoría de los estudiantes (el 90%) les agradó contar con las funciones de colaboración en la red, tales como el envío de notificaciones, la creación de modelos, el intercambio de información, la recepción y envío del estado de los procesos de software de las empresas, etc. Por otra parte, los estudiantes también reconocieron el hecho de que HESESPI proporcionó más orientación sobre el cómo, y no solo el qué, para mejorar sus habilidades en el desarrollo de software.

Tabla 18. Resultados obtenidos para la categoría relacionada al fortalecimiento del entorno de colaboración

Elementos de evaluación	Totalmente de acuerdo		De acuerdo		En desacuerdo		Totalmente en desacuerdo	
	#	%	#	%	#	%	#	%
1. Siento que la gestión de la red de colaboración a través de HESESPI fue efectiva y continua a través de la iniciativa de mejora.	31	65.0	12	25.0	2	4.0	3	6.0
2. Siento que el módulo de modelado, particularmente el diseño colaborativo del proceso, hace más efectiva la comprensión sobre la forma de trabajo actual de la industria.	38	79.0	9	19.0	1	2.0	0	0.0
3. Me gustaría usar HESESPI en mi siguiente proyecto grupal relacionado con la mejora de procesos de software en una empresa real.	29	60.0	12	25.0	7	15.0	0	0.0

4.3.5. Percepción de las empresas colaboradoras en la experimentación

Como resultado adicional, se recopiló información para examinar la sostenibilidad de esta experiencia de colaboración mediante los beneficios de la acción de “proporcionar y tomar” entre las universidades y las empresas locales en la red. El enfoque adoptado se centró en “los resultados

relacionados con el comportamiento" (por ejemplo, el grado de satisfacción) [Lee, 2000], es decir, los beneficios percibidos. En este sentido, el beneficio más significativo alcanzado por las empresas que participaron es un mayor acceso a las nuevas investigaciones e innovaciones propuestas por los profesores de la UTM en relación al campo de la mejora de los procesos. Por otra parte, todas las empresas dentro de la red de colaboración han adquirido conocimientos sobre la aplicación práctica de las actividades de mejora en su labor cotidiana. En relación al uso de HESESPI y su incorporación en el curso, las empresas creen que el uso de la herramienta les ayudará a reclutar graduados más calificados en el campo de la Mejora del Proceso de Software. La industria local está de acuerdo en que el enfoque del curso facilita a los estudiantes la adquisición de conocimientos acerca de los problemas prácticos y es útil para la enseñanza a nivel universitario. De manera similar, la percepción general de las empresas después de la colaboración indica que el uso continuo de HESESPI representará un apoyo importante para ayudarles a aprender cómo resolver los problemas de su personal y dominar técnicas específicas relacionadas con la conducción de iniciativas de mejora. Por último, como consecuencia de este trabajo las empresas mantienen su compromiso con la red de colaboración, ya que consideran que la capacitación proporcionada (seminarios y talleres) los ha motivado a mejorar la calidad de sus productos.

4.4. Limitaciones finales de la experimentación

Se han reportado algunas experiencias tempranas en la incorporación de un curso colaborativo y su soporte computacional, la herramienta HESESPI, como la base para la enseñanza de las principales actividades de un ciclo de mejora de procesos en un curso de posgrado. Esta investigación demuestra que simplemente HESESPI puede apoyar el enfoque educativo tradicional utilizado en los cursos de Ingeniería de Software y que incluyen temas relacionados con la calidad del software. Estos aspectos han sido analizados con base en datos subjetivos obtenidos de los equipos de estudiantes al final del curso impartido. El enfoque colaborativo propuesto tiene una buena cobertura[13] sobre las fases del ciclo de vida de la mejora a nivel de un curso universitario. Sin embargo, es importante mencionar que los profesores y estudiantes deben asegurarse que los planes de acción entregados a las empresas sean seguidos fielmente por todos los equipos y jefes de proyectos para asegurar el éxito pronosticado.

Como se mencionó anteriormente, el curso colaborativo propuesto presenta una buena cobertura de las fases de Evaluación y Pilotaje a nivel de equipo. Sin embargo, la mayoría de las implementación de los modelos de referencia (por ejemplo, CMMI-DEV v1.2 y v1.3) se realizan a nivel de la organización. Incluso si todos los equipos están siguiendo un modelo, todavía quedan muchos aspectos organizacionales que tienen que ser cubiertos. Por lo tanto, se considera que estas cuestiones deben ser reforzadas por una parte teórica más amplia en el curso (aspectos organizacionales de la Mejora del Proceso de Software) y con el apoyo de la industria (*coaching*). Cabe mencionar que se considera una buena decisión el fortalecer la enseñanza a nivel posgrado sobre cómo conducir una iniciativa de mejora de procesos a través de HESESPI, la incorporación de un enfoque colaborativo entre todas las partes interesadas, la orientación hacia el establecimiento de una red de colaboración con la industria local de software, y el enfoque en el modelo MoProSoft. En este sentido, los estudiantes pueden aprender cómo funciona un modelo de referencia ampliamente utilizado en México desde la perspectiva de un contexto organizacional, y comprender cómo el costo de una iniciativa de mejora puede reducirse si se realiza de una manera sistemática. De manera

[13] Tomando como referencia al modelo IDEAL [McFeeley, 1996], el curso colaborativo ofrece un contexto educativo que abarca cuatro de las cinco fases del modelo: Inicio, Diagnóstico, Establecimiento, y Ejecución. Actualmente se está trabajando en la incorporación de la última fase al curso, la Adopción.

similar, los estudiantes pueden asimilar cómo la productividad se puede mejorar dentro de una empresa cuando ésta es expuesta a mejoras realistas que son probadas y monitoreadas a través de proyectos piloto.

Por lo tanto, esta experimentación ha definido un ciclo de mejora y un método de evaluación. Estos elementos proporcionan un conjunto coherente y complementario de herramientas que facilitan el aprendizaje en la implementación de una iniciativa de mejora de procesos desde su lanzamiento hasta su cierre. Sin embargo, a pesar de que este curso colaborativo puede representar un gran atajo en el aprendizaje de las habilidades para conducir una iniciativa de mejora, todavía es demasiado difícil de aprender sin un guía especializado.

Por último, una consideración final que es importante mencionar en el contexto de la experimentación es que todas las empresas que integran la red de colaboración fueron formadas por estudiantes egresados de los niveles de pregrado y posgrado de la UTM. Así, estas empresas tienen, desde hace más de 5 años, un estrecho vínculo de trabajo con el director de la tesis a través de proyectos de titulación anteriores o consultoría especializada. En este sentido, existe la posibilidad de que el nivel de confianza establecido entre la universidad y las empresas haya sido un factor determinante en el éxito de la experimentación.

4.5. Validación de la hipótesis de la tesis

Por último, este apartado pretende reunir la evidencia que permita aceptar o rechazar la hipótesis planteada en el Capítulo 1 de esta tesis.

Hipótesis: "La introducción de un enfoque colaborativo permitirá mejorar las habilidades de los estudiantes de posgrado para desarrollar iniciativas de mejora en un entorno real de trabajo.".

Esta hipótesis es aceptada dado que con la implementación del enfoque colaborativo (a través de un curso y una herramienta computacional de soporte) mostrado en los Capítulos 3 y 4 de esta tesis, se estableció una iniciativa enfocada a desarrollar cuatro habilidades en los estudiantes: motivación, discusión, colaboración, e interacción. En este sentido, se definieron cuatro categorías a explorar (trabajo en equipo, entorno de trabajo, participación de la industria, entorno de colaboración) que fueron monitoreadas para incorporar un proceso eficiente de gestión de las iniciativas de mejora de procesos, dado que éste fue soportado por una red de colaboración que ayudó para que los estudiantes planificaran adecuadamente, efectuaran el seguimiento apropiado, cerraran eficientemente sus ciclos de trabajo y realimentaran los datos obtenidos para nuevos proyectos de mejora. Concretamente las Tablas 15, 16, 17 y 18 mostraron que las apreciaciones de los estudiantes que participaron en el curso experimental son positivas.

Es importante mencionar que esta hipótesis, que ha sido establecida en el contexto de la tesis, es aceptada para el caso de los estudiantes que participaron en la experimentación. De ninguna manera es posible generalizar que la solución propuesta en esta tesis proporcionará los mismos resultados en cualquier entorno estudiantil debido a que esto requiere de forma obligatoria continuar con la experimentación incluyendo una muestra más significativa. Así pues, en el contexto de la experimentación en Ingeniería de Software se afirma que los resultados positivos sobre una efectividad parcial, mostrados en un trabajo de investigación o reporte, evidencian que es posible establecer un caso de estudio con un mayor número de participantes y proyectos [Juristo & Moreno, 2010].

5. Discusión y Conclusiones

La presente tesis intenta fortalecer el proceso de enseñanza/aprendizaje en el área de mejora de procesos de software a través de la participación activa de los estudiantes en un curso de posgrado con un enfoque colaborativo que es soportado por una red de colaboración formada por empresas representativas de la industria local de software. Bajo este contexto de colaboración, en las iniciativas de mejora que se realizaron a lo largo de un semestre, los estudiantes evaluaron el estado actual de los procesos de software y determinaron acciones de mejora en base al resultado obtenido, dichas acciones de mejora estuvieron basadas en el modelo MoProSoft.

Durante las actividades de evaluación, los estudiantes aprendieron a valorar los entregables generados en un proceso de software, a definir y asignar los roles de una iniciativa de mejora, a identificar las prácticas fuertes y débiles de un proceso, y a recopilar las lecciones aprendidas como medio de comprobación de la mejora. Todos estos aspectos contribuyeron, junto con la supervisión del profesor y la red de colaboración, para que el estudiante pusiera en práctica sus habilidades suaves (observación, revisión, presentación oral de la información, escritura, planificación, cooperación, reflexión y juicio), aspectos que solamente pueden mejorarse bajo un entorno práctico. De esta manera, los estudiantes también pudieron darse cuenta que el factor humano tienen un gran impacto en el éxito de una iniciativa de mejora, además de otros factores externos que no están bajo su control, tales como la disposición del tiempo de la empresa, los recursos financieros, la cultura organizacional, la visión organizacional y el nivel de compromiso con el programa de mejora. Así pues, la introducción de este enfoque colaborativo cambió el comportamiento de los estudiantes frente a las iniciativas de mejora mediante el contacto directo con problemas reales de la industria local de software. Es decir, aspectos como la motivación, la cooperación y la comunicación fueron claramente observados durante las actividades que desarrollaron en la experimentación del enfoque colaborativo.

Como resultado del *benchmarking* de la literatura revisada, esta tesis propuso como elemento esencial la modificación del temario de la materia de "Modelos y Métodos de Evaluación y Mejora" dentro de la Maestría en Tecnologías de Cómputo Aplicado de la UTM, presentada en el Capítulo 3 de esta tesis, que es soportado por un enfoque colaborativo para planificar, conducir y "pilotar" iniciativas de mejora de procesos en empresas reales, a fin de preparar a los estudiantes de posgrado a hacer frente a los desafíos que se encontrarán en su vida profesional.

En el contexto de nuestra universidad, el curso propuesto fue implementado en el programa de Maestría en Tecnologías de Computo Aplicado con una generación de cuatro alumnos. Sin embargo, se consideró que se requería de una población estudiantil mayor para recoger y presentar resultados en pro de sustentar el desarrollo de la tesis, por lo cual se incluyeron grupos de estudiantes de otras universidades. En todos los casos, el curso fue apoyado por HESESPI y la misma red de colaboración con las empresas locales, con el objetivo de aumentar el efecto de aprendizaje,

específicamente, a nivel de aplicación, y reforzar la comprensión de los conceptos teóricos. No obstante, se considera que aún y con todo este apoyo la enseñanza en la aplicación real de los conceptos de la mejora de procesos es todo un reto. Por lo tanto este trabajo de tesis ha desarrollado un instrumento para enseñar y poner en práctica las actividades de mejora para contribuir así en la renovación del estado actual de la educación sobre la mejora de procesos de software. Como resultado directo de esta investigación, se ha generado información con respecto a la comunidad educativa y los problemas asociados con el desarrollo de software a través de la conformación de una "red de colaboración con la industria local". El concepto de la colaboración universidad-industria ha demostrado ser un importante experimento social en los sistemas educativos de muchos países. En este sentido, se ha observado que, si bien la colaboración con las empresas locales es en sí una oportunidad de cambio, los profesores y estudiantes pueden beneficiarse de la información valiosa que las empresas les proporcionen puesto que podrían establecer vínculos de trabajo sobre un área de investigación muy particular. Además de que el acceso a experimentos reales dentro de las empresas, reduciría la inversión que éstas deberían hacer en en una iniciativa de mejora real.

De este modo, las experiencias obtenidas con la incorporación de las empresas en ejercicios prácticos han proporcionado una premisa preliminar de un impacto positivo en el aprendizaje. Sin embargo, esta información necesita ser extraída, organizada y analizada con más cursos con el fin de sustentar con mayor evidencia los resultados obtenidos, fuera del objetivo indicado al principio de esta tesis. En este sentido, es importante mencionar que como consecuencia de este trabajo las empresas mantendrán su compromiso de colaborar en la red en futuras tesis y proyectos de investigación. Así pues, se tiene la intención de repetir la implementación del enfoque colaborativo en más cursos de posgrado y recoger la información a una escala más grande con la misma red de colaboración.

Por otro lado, considerando los objetivos iniciales de esta tesis se puede concluir que:

- Se realizó un estudio de la literatura existente relacionada con enfoques educativos como soporte a la educación de la mejora del proceso en el contexto de la Ingeniería de Software.
- Se realizó una base comparativa de las propuestas analizadas y a modo de *benchmarking* se identificaron aspectos comunes que permitieron dar pauta a este trabajo.
- Como parte fundamental del enfoque colaborativo, se propuso un temario con indicadores de trabajo y desempeño.
- Se eligió el método incremental como base para la construcción de una herramienta computacional de soporte al enfoque colaborativo, y se consideró a MoProSoft en sus niveles 1 y 2 como el modelo de referencia a seguir en el curso.
- Se probó la solución propuesta en esta tesis con alumnos de nivel posgrado y profesores de diferentes niveles educativos de cuatro universidades diferentes.
- Como resultados se su aplicación, se obtuvieron los datos presentados en el Capítulo 4 donde se muestra la precepción de los estudiantes y representantes de las empresas.

Por último se puede concluir que la incorporación de HESESPI y la manera en la que ésta fue diseñada, permitió la automatización de las actividades de evaluación de los procesos eliminando así las interrupciones en el trabajo diario de los jefes de proyectos que participaron activamente en las actividades dentro de la red de colaboración. Además, la evaluación del proceso a través de su representación gráfica ayudó a los estudiantes a interpretar los datos obtenidos y establecer una

justificación para cada decisión que tomaban. Es decir, esto les permitió establecer, a su propio juicio, las acciones correctivas que deberían tomarse para la mejora de los procesos evaluados.

En este contexto, a pesar de que los esfuerzos sobre la mejora de procesos se han disparado en el mercado como una estrategia a seguir cuando las empresas de software intentan mejorar la calidad del software producido, aún existen muchas universidades que no están enseñando cómo hacerlo en un contexto real. Así, MoProSoft es considerado como uno de los modelos más conocidos a nivel nacional y que se enfocan en la mejora de los procesos de software como medio para incrementar la calidad del software en las empresas. Sin embargo, este modelo es relativamente nuevo por lo que ha habido poca investigación escrita en la que se recopilen datos y herramientas de generación de planes de acción que se puedan emplear cuando se utiliza este enfoque en la educación.

Finalmente, como trabajo futuro se enuncian las siguientes líneas de investigación:

- Es necesario incorporar las actividades del modelo CMMI-DEV v1.3 [CMMI, 2010] y adaptarlas al menos en sus niveles 1 y 2 en HESESPI, tal y como se hizo en esta tesis con MoProSoft.
- Es muy importante que se continúe con la experimentación incluyendo una mayor cantidad de alumnos y empresas de software no necesariamente MiPyMEs.
- Se considera que es viable el modificar el enfoque colaborativo propuesto a modo que se busque promover, en los estudiantes de posgrado, el desarrollo de competencias para la gestión de proyectos software y otros temas similares.

6. Anexo A.- Acrónimos

ABET	Accreditation Board of Engineering Technology
ACM	Association for Computing Machinery
ANIEI	Asociación Nacional de Instituciones de Educación en Informática
APE	Administración de Proyectos Específicos
APEC	Foro de Cooperación Económica Asia-Pacífico
ARC	Appraisal Requirements for CMMI
CBOK	Core Body Of the Knowledge
CMM	Capability Maturity Model
CMMI-DEV	Capability Maturity Model Integration for Development
COMPETISOFT	Proyecto de Mejora de Procesos para Fomentar la Competitividad de la Pequeña y Mediana Industria del Software de Iberoamérica
CONACYT	Consejo Nacional de Ciencia y Tecnología
CONOCER	Consejo Nacional de Normalización y Certificación de Competencias Laborales
DMS	Desarrollo y Mantenimiento de Software
EvalProSoft	Modelo de Evaluación del Proceso Software
GSwE2009	Plan de Estudios para Posgrados en Ingeniería de Software 2009
HESESPI	Herramienta de Soporte para la Enseñanza de SPI
HTML	HyperText Markup Language
I+D	Investigación + Desarrollo
IDEAL	Initiating, Diagnostic, Establishing, Acting, Leveraging
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronics Engineers
IEEE CS	Institute of Electrical and Electronics Engineers Computer Society
IMPULSA-TI	Sociedad Academia-Industria-Gobierno en Tecnologías de la Información
IMS	Instructional Management System
INEGI	Instituto Nacional de Estadística y Geografía
IS	Ingeniería de Software

ISO	International Organization for Standardization
ISSEC	Integrated Software & Systems Engineering Curriculum
LMS	Learning Management System
LO	Learning Object
LOC	Line Of Code
MoProSoft	Modelo de Procesos para la Industria del Software
MiPyMEs	Micro, Pequeñas y Medianas Empresas de Software
MR-MPS	Modelo de Referencia del Programa para la Mejora del Proceso Software Brasileño
MSG	Management Steering Group
NATO	North Atlantic Treaty Organization
NCp	Nivel de Cobertura por pregunta
NYCE	Normalización y Certificación Electrónica S.C.
OMG	Object Management Group
PBL	Project Based Learning
PI	Pregunta de Investigación
PIB	Producto Interno Bruto
PMBoK	Project Management Body of Knowledge
PROMEDIA	Programa para el Desarrollo de la Industria de Medios Interactivos
PROSOFT	Programa para el Desarrollo de la Industria del Software
PSP	Personal Software Process
ROI	Return Of Investment
RUP	Rational Unified Process
RWL	Real Word Lab
SCAMPI	Standard CMMI Appraisal Method for Process Improvement
SCORM	Sharable Content Object Reference Model
SEEK	Software Engineering Education Knowledge
SEI	Software Engineering Institute
SEPG	Software Engineering Process Group
SIICAP	Sistema Inteligente de Información en Capacidades de TI
SOAP	Simple Object Access Protocol
SPI	Software Process Improvement
SPICE	Software Process Improvement Capability Determination
SQPI	Software Quality and Process Improvement Course
SUNEO	Sistema de Universidades Estatales de Oaxaca
SWEBOK	Software Engineering Body of Knowledge
SWECC	Software Engineering Coordinating Committee
TI	Tecnologías de Información
TSP	Team Software Process
TSPi	Introduction to the Team Software Process

TWG	Technical Working Group
UNAM	Universidad Nacional Autónoma de México
UNICEN	Universidad Nacional del Centro de la Provincia de Buenos Aires
UNIVALI	Universidad del Valle de Itajaí
UTM	Universidad Tecnológica de la Mixteca
WS	Web Service
XML	eXtensible Markup Language

7. Anexo B.- Diseño de la herramienta computacional de soporte

7.1 Requisitos funcionales

Esta sección contiene la lista de los requisitos funcionales que considera la herramienta que soporta al presente trabajo de tesis. Los requisitos son numerados con la intención de seguir una agrupación por áreas funcionales, misma que se utiliza para el desarrollo de la solución propuesta. Por cada requisito funcional se proporciona una breve descripción, así como los valores de entrada y salida (si los hubiera), estando los datos obligatorios marcados como "ob" y los opcionales como "op". Dado que más adelante los casos de uso son descritos para descomponer funcionalmente el software a desarrollar, se hace referencia también a cuál caso de uso corresponde cada requisito para un mejor seguimiento.

7.1.1 Gestionar acceso

7.1.1.1 Requisito funcional: acceso de usuarios registrados

Descripción	Módulo inicial para la validación de los datos de autenticación del usuario y su acceso al sistema.
Entrada	Se introducirá nombre de usuario y contraseña previamente asignados.
Salida	Se inicia la sesión del usuario. Se gestionara el nivel de acceso de cada usuario según se trate de alumno, jefe de proyecto o profesor, siendo este último un tipo de administrador. Mostrar errores en caso de autenticación fallida.
Proceso	El sistema muestra en pantalla los datos principales correspondientes al usuario autenticado correctamente, dándole acceso a las ventanas correspondientes a su nivel de usuario. Prohibir el acceso al sistema en caso de introducir datos erróneos.
Referencias	Alta de usuarios.
Casos de uso	Gestionar acceso.

7.1.1.2 Requisito funcional: alta de usuarios

Descripción	Módulo donde se introducirán datos de identificación del usuario para ser registrados y permitir su acceso al sistema.

Entrada	Se introducirán los siguientes datos: Datos de usuario: Usuario (ob), Contraseña (ob), Repetir contraseña (ob). Datos personales: Nombre (ob), Apellido Paterno (ob), Apellido Materno (ob), E-Mail (ob), Teléfono (op) Datos de tipo de usuario: Tipo de usuario (ob) (“Profesor”, “Estudiante”, “Jefe de proyecto”). Dependiendo del tipo de usuario se pedirán los siguientes datos: Profesor: Materia (ob), ciclo escolar (ob). Estudiante: Materia (ob). Jefe de proyecto: Empresa (ob).
Salida	Se envía un mensaje de éxito en la creación del usuario.
Proceso	Se tienen datos en comunes para el alta de usuarios, pero en cuanto definan qué tipo de usuario serán, se le pedirán los diferentes datos definidos en las entradas. Una vez confirmados los datos y que éstos hayan sido correctos, se mostrará un mensaje de confirmación.
Referencias	Acceso de usuarios registrados.
Casos de uso	Gestionar acceso.

7.1.1.3 Requisito funcional: baja de usuarios

Descripción	Módulo que elimina permanentemente los datos de un usuario.
Entrada	Se introducirán los siguientes datos: Datos de usuario: Usuario (ob).
Salida	Se pedirá confirmación para eliminación de usuario.
Proceso	Primero se debe mostrar un mensaje de confirmación de eliminación del usuario en caso que coincida exactamente con los datos de entrada. Mostrar mensajes de error en caso de no ser encontrado.
Referencias	Alta de usuarios.
Casos de uso	Gestionar acceso.

7.1.1.4 Requisito funcional: listado de usuarios

Descripción	Módulo donde se obtienen datos referentes a la cuenta de un usuario, se contará con un buscador que tendrá varios filtros de búsqueda opcionales que se pueden combinar para hacer más específica la búsqueda.
Entrada	Se debe introducir al menos uno de los siguientes datos: Palabras clave del nombre de usuario (op), username (op), teléfono (op), Correo (op).
Salida	Se mostrarán los resultados correspondientes separándolos por los tipos de usuario y mostrando sus nombres.
Proceso	Al introducir los datos opcionales, para comenzar a realizar la búsqueda de los usuarios cuyos datos coinciden con los criterios de búsqueda, se debe presionar el botón buscar. Solamente un profesor puede hacer estas búsquedas.
Referencias	Acceso de usuarios registrados.
Casos de uso	Gestionar acceso.

7.1.1.5 Requisito funcional: restauración de contraseña

Descripción	En caso que el usuario tenga problemas para ingresar al sistema, implementar un modo de establecer una nueva contraseña.
Entrada	Se introducirán los siguientes datos: Del panel de listado de usuarios, seleccionar al usuario correspondiente (ob) y se accederá al panel de modificación de datos de usuarios.
Salida	Se mostrará el panel de modificación de datos de usuarios.
Proceso	Validar los datos que proporciona el usuario.
Referencias	Acceso de usuarios registrados, modificación de datos de usuarios.
Casos de uso	Gestionar acceso.

7.1.1.6 Requisito funcional: modificación de datos de usuarios

Descripción	Se pueden hacer modificaciones con respecto a los datos de los usuarios del sistema.
Entrada	Los campos que se mostrarán son los mismos mostrados en el caso de uso de alta de usuarios, y se mostrará un botón para confirmar los cambios realizados.
Salida	Se mostrará un mensaje confirmando que los cambios se han efectuado en el sistema.
Proceso	El profesor ingresará al módulo de lista de usuarios donde accederá a la opción de editar datos, y podrá modificar todos los datos correspondientes al nombre de usuario, contraseña y datos personales. Antes de guardar los cambios se mostrará un mensaje de confirmación de modificación de datos y al aceptar, los datos quedarán actualizados en el sistema.
Referencias	Acceso de usuarios registrados, Alta de usuarios, Listado de usuarios.
Casos de uso	Gestionar acceso.

7.1.1.7 Requisito funcional: alta de empresas

Descripción	Módulo que permite el registro de datos de empresas en el sistema.
Entrada	Se introducirán los siguientes datos: Datos de la empresa: Nombre de la empresa (ob), Dirección (ob), Teléfono (op).
Salida	Se guarda en la base de datos y se muestra un mensaje de confirmación.
Proceso	En el caso de un alta de usuario y su empresa no exista previamente en la base de datos, se pueden dar de alta datos de una nueva empresa en el sistema a través del panel Empresas. Se debe evitar dar de alta nombres de empresas duplicadas para este caso primero se debe mostrar una lista de empresas dadas de alta en la base de datos para buscarla y verificar su existencia.
Referencias	Ninguna.
Casos de uso	Gestionar acceso.

7.1.1.8 Requisito funcional: listado de empresas

Descripción	Se listan todas las empresas existentes que coinciden con el nombre introducido en la entrada de búsqueda o se listan todas las empresas existentes por default de manera paginada.
Entrada	Se introducirá el nombre de una empresa.
Salida	Se llenará una lista de empresas, mostrando su nombre, dirección y ubicación.

Proceso	La lista de empresas será generada a partir de las palabras clave introducidas en el nombre de la empresa, una vez desplegados los datos, estos pueden ser modificados en caso de requerirse.
Referencias	Alta de usuarios.
Casos de uso	Gestionar acceso.

7.1.1.9 Requisito funcional: modificación de empresas

Descripción	Modifica los datos de las empresas registradas.
Entrada	Se introducirá al menos uno de estos campos: Datos de la empresa: Nombre de la empresa, Dirección, Teléfono.
Salida	Se actualizan los cambios en el sistema.
Proceso	Accediendo desde el panel de listado de empresas se accede a la opción de modificar datos, donde se podrán modificar uno o los tres datos al mismo tiempo especificados en la entrada. Antes de actualizar los datos en el sistema, se debe mostrar un mensaje de confirmación.
Referencias	Alta de usuarios, Alta de empresas, Listado de empresas.
Casos de uso	Gestionar acceso.

7.1.1.10 Requisito funcional: baja de empresas

Descripción	Elimina permanentemente los datos referentes a una empresa.
Entrada	Se selecciona la opción eliminar empresa
Salida	Se actualizan los cambios en la base de datos.
Proceso	Desde el panel de listado de empresas se accede a este módulo desde la opción eliminar la empresa, a continuación se debe mostrar un mensaje de confirmación de eliminación de la empresa.
Referencias	Alta de usuarios, Alta de empresas, Listado de empresas.
Casos de uso	Gestionar acceso.

7.1.2 Gestionar proyectos

7.1.2.1 Requisito funcional: alta de proyectos

Descripción	Creación de proyectos.
Entrada	Con un usuario tipo profesor se introducirán los siguientes datos: Nombre del proyecto (ob). Completitud (ob), Estado (ob), Fecha de inicio (ob). Jefes (ob), Participantes (ob).
Salida	El sistema creará el proyecto y vinculará a los usuarios participantes en los paneles de Jefes y Participantes con el proyecto creado.
Proceso	Para la creación de un proyecto se necesita una lista de usuarios participantes, para ello se utilizará el listado de usuarios. La creación de la lista de usuarios y el número de participantes queda a la decisión del profesor.
Referencias	Acceso de usuarios registrados, altas de empresas- equipos de estudiantes.
Casos de uso	Gestionar proyectos.

7.1.2.2 Requisito funcional: alta de empresas – equipos de estudiantes

Descripción	Este módulo permite realizar la vinculación entre equipos de estudiantes y empresas con los diversos proyectos que se irán creando.
Entrada	En un panel, el profesor seleccionará del listado de estudiantes, aquellos que formarán un equipo.
Salida	El sistema asignará a cada usuario el proyecto correspondiente.
Proceso	En base al proyecto seleccionado, el sistema vinculará los jefes de proyecto con los usuarios seleccionados en las entradas y al profesor en sesión. Una vez hecho esto, un nuevo proyecto de mejora será creado.
Referencias	Listado de estudiantes, Listado de empresas, Listado de usuarios, Alta de proyectos.
Casos de uso	Gestionar proyectos.

7.1.2.3 Requisito funcional: modificación de empresas – equipos de estudiantes

Descripción	Este módulo permite modificar la vinculación entre equipos de estudiantes y proyectos
Entrada	El profesor seleccionará de una lista de proyectos, uno y se mostrarán opciones para modificar: La lista de estudiantes que participan en el proyecto pulsando el botón llamado Participantes. La lista de jefes de proyectos que participan en el proyecto pulsando el botón llamado Jefes.
Salida	El sistema actualizará las asignaciones correspondientes.
Proceso	De un panel como el usado en el caso de uso alta de empresas-equipos de estudiantes se podrá seleccionar y modificar las listas usuarios vinculados al proyecto de mejora. Antes de actualizar los datos en el sistema se debe mostrar un mensaje confirmación al usuario.
Referencias	Listado de proyectos, Listado de empresas, Alta de proyectos.
Casos de uso	Gestionar proyectos.

7.1.2.4 Requisito funcional: baja de empresas – equipos de estudiantes

Descripción	Este módulo permite realizar la eliminación de vínculos entre equipos de estudiantes, jefes y proyectos.
Entrada	Se seleccionará en un panel de listado de proyectos al menos uno y se seleccionara la opción de quitar de la lista.
Salida	El sistema borrará todos los vínculos que existen entre los diversos usuarios con el proyecto.
Proceso	De un panel donde se identifiquen listas de proyectos el profesor seleccionará uno y en los paneles de Jefes y/o Participantes, se editarán las listas de usuarios participantes.
Referencias	Listado de proyectos, Alta de empresas – equipos de estudiantes.
Casos de uso	Gestionar proyectos.

7.1.2.5 Requisito funcional: iniciar proyecto de mejora

Descripción	Una vez creado el proyecto y se cuente con la participación de al menos un jefe de proyecto, y se haya echo la vinculación con su empresa, se inicializa el proyecto de mejora.
Entrada	El profesor puede inicializar un proyecto.
Salida	El sistema podrá en un estado de activo este proyecto.

Proceso	Este es un módulo para asegurarse que en un proyecto ya existen los tres tipos de usuarios especificados anteriormente participando en un proyecto. Una vez pasada esta comprobación el proyecto ya queda activo para poder trabajar en él.
Referencias	Intercambio de mensajes directos, Acceso de usuarios registrados.
Casos de uso	Gestionar proyectos.

7.1.2.6 Requisito funcional: listado de proyectos

Descripción	Lista de proyectos.
Entrada	Por default el sistema mostrará una lista de los proyectos activos y más recientes vinculados a la cuenta de los usuarios. En caso de tener muchos proyectos dar criterios de búsqueda de los cuales al menos uno se puede introducir y se pueden crear combinaciones de las tres opciones que son: Nombre del proyecto (op), Fecha de inicio del proyecto (op), Estado (op), Completitud (op).
Salida	Se mostrará en una lista todos los proyectos vinculados con la cuenta del usuario.
Proceso	Una vez cualquiera de los tipos de usuarios identificados en este sistema haya ingresado al sistema, el primer panel que verá es donde se le mostrará una lista de proyectos en los que ha participado.
Referencias	Acceso de usuarios registrados.
Casos de uso	Gestionar proyectos.

7.1.2.7 Requisito funcional: listado de detalles de proyectos

Descripción	Se muestran todos los detalles de un proyecto seleccionado por el requisito de listado de proyectos.
Entrada	De la lista de proyectos, se selecciona la opción ver detalles de proyecto.
Salida	Se mostrarán los datos a detalle del proyecto seleccionado, como son, nombre del proyecto, estado del proyecto con respecto a un porcentaje de completes con respecto a las fases del modelo IDEAL, usuarios participantes en el proyecto, nombre de la empresa que será evaluada. Además del panel de comunicación mencionado en el caso de uso gestión de comunicación.
Proceso	De la lista de proyectos mencionada en el requisito de listado de proyectos, por cada proyecto en la lista, se mostrará una opción para ver sus detalles
Referencias	Listado de proyectos.
Casos de uso	Gestionar proyectos.

7.1.2.8 Requisito funcional: modificación de detalles de proyectos

Descripción	En el panel donde se muestran los detalles de cada proyecto, se pueden actualizar en cualquier momento los datos.
Entrada	Se introducen o actualizan los campos referentes a los datos del proyecto, deben ser los mismos campos utilizados en el alta de proyectos.
Salida	Se mostrará un mensaje confirmando los datos guardados.
Proceso	Durante las fases del modelo IDEAL, los usuarios modificarán el estado del proyecto de acuerdo a las fases del modelo.
Referencias	Listar detalles de proyectos, Acceso de usuarios registrados.

Casos de uso	Gestionar proyectos.

7.1.2.9 Requisito funcional: intercambio de mensajes directos

Descripción	Módulo que permite la comunicación entre todos los usuarios.
Entrada	En un panel donde divida por proyectos, mostrará el panel de chat que permite la comunicación y visualización de mensajes de texto entre los usuarios involucrados en el proyecto mismo proyecto.
Salida	Se mostrarán mensajes de todos los usuarios del mismo proyecto.
Proceso	En el panel de chat cualquier usuario verá los mensajes de texto escritos por otro, siempre y cuando pertenezcan al mismo proyecto. Se tiene que hacer una distinción entre ellos mostrando su nombre de usuario.
Referencias	Listar detalles de proyectos.
Casos de uso	Gestionar proyectos.

7.1.3 Gestionar documentos

7.1.3.1 Requisito funcional: alta de documentos

Descripción	Módulo donde se deposita la documentación que servirá para futuros cursos.
Entrada	Los alumnos agregarán al sistema los documentos que considere necesarios y estos serán sujetos a una revisión por parte del profesor.
Salida	El sistema guardará todos estos documentos y aparecerán en el módulo de aprobación de documentos.
Proceso	Los alumnos serán capaces de subir documentos de diferentes formatos (Word, PDF, imágenes, etc.). El profesor aprobará la permanencia de estos. El sistema debe encargarse de gestionar el almacenamiento de los documentos, a fin de que estén disponibles a futuro para su consulta y/o aprobación.
Referencias	Aprobación de documentos.
Casos de uso	Revisar lecciones.

7.1.3.2 Requisito funcional: aprobación de documentos

Descripción	Módulo que revisa el profesor para verificar la utilidad de los documentos.
Entrada	El profesor ingresa al módulo de aprobación de documentos y en forma de lista se le muestra la opción de descargarlos y una vez revisados se cuenta con la opción para aprobar o rechazar.
Salida	En caso de ser aprobados, los documentos serán almacenados en el sistema y estarán disponibles para su consulta. En caso de ser rechazado los alumnos podrán volver a subir una nueva versión de ese documento.
Proceso	Se mostrarán todos los documentos recientemente agregados al sistema para poder ser consultados y evaluados, mediante el listado de documentos, aparecerá una subsección para modificar el estado de aprobación de los documentos, en caso de ser rechazados serán marcados con esa etiquetay en el caso de ser aceptados los documentos estarán disponibles en el panel de listado de documentos aprobados.
Referencias	Alta de documentos, Listar documentos, Listado de documentos aprobados.
Casos de uso	Gestionar documentos.

7.1.3.3 Requisito funcional: modificación de documentos

Descripción	Módulo para modificar el nombre de los documentos.
Entrada	Se selecciona un documento y se elige la opción de cambiar nombre, a continuación se deberá escribir directamente.
Salida	El sistema guardará los cambios en la base de datos.
Proceso	A fin de tener los datos de una manera correcta, se puede modificar el nombre en caso de establecer nomenclaturas, prefijos o sufijos para los nombres de los archivos. En caso de subir un documento con una nueva versión de un archivo que ya fue subido antes, se sobrescribirá este nuevo por el antiguo ya existente.
Referencias	Listado de documentos aprobados.
Casos de uso	Gestionar documentos.

7.1.3.4 Requisito funcional: listado de documentos aprobados

Descripción	Módulo para mostrar los documentos y posteriormente poder modificar su estado de aprobación.
Entrada	De un conjunto de filtros, se puede escribir al menos alguna de estas opciones como son: Nombre (op), fecha de modificación (op), tipo de documento (op).
Salida	Se mostrarán por defecto las carpetas (si las hubiese) con los archivos guardados en ella. El buscador mostrará los resultados correspondientes a los criterios de búsqueda.
Proceso	Se consultará en la base de datos las coincidencias con respecto a los criterios de búsqueda y se mostrarán en un panel para visualizarlos. Se mostrará una sección para documentos a ser aprobados.
Referencias	Aprobación de documentos.
Casos de uso	Gestionar documentos.

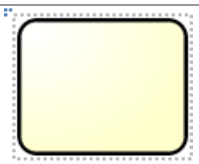
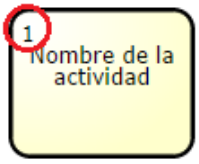
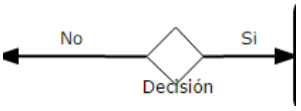
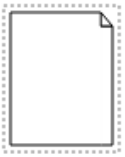
7.1.3.5 Requisito funcional: baja de documentos aprobados

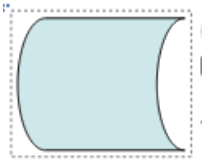
Descripción	Módulo que permite eliminar los documentos guardados en el sistema.
Entrada	De un conjunto de filtros, se puede escribir al menos alguna de estas opciones como son: Nombre (op), fecha de modificación (op), tipo de documento (op).
Salida	Se eliminarán permanentemente del sistema aquellos documentos seleccionados.
Proceso	Del panel de listado de documentos aprobados se podrán seleccionar aquellos documentos que quieran ser eliminados, sin antes mencionar un mensaje de confirmación para realizar dicha acción.
Referencias	Listado de documentos aprobados.
Casos de uso	Gestionar documentos.

7.1.4 Evaluar proceso de la organización

7.1.4.1 Requisito funcional: modelar proceso de la organización

Descripción	Módulo que de manera gráfica representa algún proceso de la organización

<table>
<tr><td>Entrada</td><td>Mediante un módulo gráfico, usando diagramas de actividades definidos en el proceso, se dibujarán las actividades que el jefe de proyecto con algún proceso específico dentro de su empresa y dentro de algún área que el modelo MoProSoft identifique.</td></tr>
<tr><td>Salida</td><td>El sistema guardará el diagrama para ser evaluado posteriormente y se asociará a un nombre de proceso en la organización.</td></tr>
<tr><td>Proceso</td><td>
<table>
<tr><th>Componente</th><th>Símbolo</th><th>Descripción</th></tr>
<tr><td>Secuencia</td><td>  </td><td>Las flechas indican la secuencia de una actividad.</td></tr>
<tr><td>Caja</td><td>  </td><td>Una caja representa una actividad.</td></tr>
<tr><td>Números en cajas.</td><td> 
</td><td>El número indica el orden de las actividades.</td></tr>
<tr><td>Diamantes</td><td> 
</td><td>Un diamante representa una decisión binaria</td></tr>
<tr><td>Hoja blanca</td><td>  </td><td>La hoja blanca representa cualquier documento necesario para la ejecución de una actividad.</td></tr>
</table>
</td></tr>
</table>

	<table><tr><td>Media elipse</td><td>  </td><td>Representa la obtención/relación de algún producto o servicio con la actividad</td></tr><tr><td>Rol/Persona</td><td>  </td><td>Las personas representan a las partes relacionadas en la consecución de una actividad</td></tr><tr><td>Producto de entrada/salida</td><td>  </td><td>Requeridos para la evaluación en el modelo MoProSoft, representa los entregables. Debe tener como atributo, rol del responsable, rol implicado, producto de entrada, producto de salida, criterio de medición.</td></tr></table>
Referencias	Listado de documentos aprobados.
Casos de uso	Gestionar documentos.

7.1.4.2 Requisito funcional: modificación de diagramas del proceso de la organización

Descripción	Mediante el mismo panel que crea diagramas, se pueden abrir y modificar los ya existentes.
Entrada	Mediante el menú de diagrama de procesos los alumnos podrán abrir modelos y editarlos a su consideración.
Salida	Se guardarán los cambios en la base de datos.
Proceso	Se accederá al panel de modelar el proceso de la organización, de una lista de archivos se seleccionará el que se desee y se abrirá en el mismo panel. De acuerdo como especifica el caso de uso modelas proceso de la organización se agregarán, quitarán o eliminarán los elementos de los diagramas y se guardarán reemplazando el diagrama anteriormente guardado.
Referencias	Modelar proceso de la organización.
Casos de uso	Evaluar proceso de la organización.

7.1.4.3 Requisito funcional: baja de diagramas del proceso de la organización

Descripción	Mediante el listado de diagramas, se puede elegir entre borrar uno de ellos.
Entrada	Se elige del listado de diagramas y se selecciona la opción borrar.
Salida	Se mostrará un mensaje de confirmación.
Proceso	A fin de evitar duplicidad en diagramas, incorrectos o limpiar la lista de diagramas, se debe contar con la eliminación de estos. Después de seleccionarlos desde la opción borrar diagramas se pedirá confirmación para realizar la acción.
Referencias	Modelar proceso de la organización, listado de diagramas del proceso de la organización.
Casos de uso	Evaluar proceso de la organización.

7.1.4.4 Requisito funcional: vista del diagrama del proceso de la organización

Descripción	Se muestra el diagrama relacionados al proyecto abierto.
Entrada	Desde el listado de detalles de proyectos, se podrá acceder al diagrama asociado a cada proceso.
Salida	Se visualizará el diagrama correspondiente.
Proceso	Desde los detalles de un proyecto se debe mostrar la opción para visualizar los procesos identificados y su diagrama correspondiente, al acceder a esa opción se cargará una visualización del diagrama y se deben dar opciones para exportación a formatos estándares como imágenes jpg.
Referencias	Listado de detalles de proyecto.
Casos de uso	Evaluar proceso de la organización.

7.1.4.5 Requisito funcional: evaluación del modelado del proceso de la organización

Descripción	Módulo donde se hará la comparativa entre el estado actual de los procesos de la organización contra el modelo de referencia MoProSoft.
Entrada	Al ejecutarse este módulo el sistema verificará el diagrama creado en el caso de uso modelar proceso de la organización.
Salida	Se guardará la calificación sobre el proceso analizado.
Proceso	El sistema hará hacer una comparación del diagrama creado por el jefe de proyecto con uno creado previamente cuyo contenido este basado en las prácticas del modelo MoProSoft a fin de mostrarle al alumno las actividades que puedan hacerle falta al proceso que está siendo evaluado y agilizar el proceso de evaluación.
Referencias	Modelar proceso de la organización.
Casos de uso	Evaluar proceso de la organización.

7.1.4.6 Requisito funcional: evaluación de cuestionarios del proceso de la organización

Descripción	Módulo donde se mostrarán los cuestionarios a los jefes de proyectos.
Entrada	Todas las respuestas a los cuestionarios.
Salida	El sistema guardará las respuestas de los jefes de proyectos a fin de ser evaluadas.

Proceso	Los datos tomados del cuestionario se guardarán en la base de datos, y se realizaran los cálculos como se especifica a continuación: Para cada pregunta existen 5 posibles respuestas: Siempre, Usualmente, Algunas veces, Rara vez y Nunca. Elegir siempre significa que la práctica se realiza entre 75%-100% de las veces, Usualmente significa entre 25%-74%, Rara vez 0%-24%, Nunca 0%. Por convención de nomenclatura se usarán las siguientes: no.Jp = Número de jefes de proyecto que contestaron el cuestionario, #S=Número de veces que fue contestada la opción Siempre, #U= Número de veces que fue contestada la opción Usualmente, #A= Número de veces que fue contestada la opción Algunas Veces, #N= Número de veces que fue contestada la opción Nunca. Para calcular la cobertura por pregunta la operación queda como: Cp = (#S*1 + #U*0.75+ #A*0.5 + #R*0.25)/no.JP. Media (sobre 4). Media = (#S*4 + #U*3+ #A*2 + #R*1)/no.JP. Desviación típica. Desvp $= \frac{\sqrt{(np.JP*(4^2*\#S+3^2*\#U+2^2*\#A+1^2*\#R))-np.JP^2*Media^2)}}{np.JP^2}$)) Cobertura total que es el promedio de todas las coberturas por pregunta del cuestionario, donde n es el número de preguntas. $Cp(XY) = \frac{\sum_{i=1}^{n} Cp_i}{n}$
Referencias	Alta de empresas, Alta de usuarios, Alta de proyectos, Inicializar proyecto de mejora, evaluación de modelado de procesos de la organización.
Casos de uso	Evaluar proceso de la organización.

7.1.4.7 Requisito funcional: mostrar resultados de la evaluación

Descripción	Módulo donde se mostrarán los datos de la evaluación.
Entrada	Seleccionar los resultados de los cuestionarios aplicados en la evaluación.
Salida	El sistema representará mediante gráficas de barras horizontales los datos calculados después de rellenar los cuestionarios, al fin de que los usuarios puedan hacer una mejor interpretación de ellos y generen prioridades posteriormente para su plan de acción.
Proceso	En el caso de las gráficas correspondientes a la cobertura por pregunta, mostrar en color rojo la barra que muestre el 50% o menos, y en color azul en caso de ser mayor al 50%. Mostrar en gráficas de barras la representación de las desviaciones típicas, si la desviación es menor a 0.8 mostrar la gráfica en color gris que representará una práctica fuerte y en caso de ser mayor o igual que 0.8 se debe poner que es un punto débil y expresar la barra en color rojo.
Referencias	Evaluación de cuestionarios del proceso de la organización, Aprobación de documentos.
Casos de uso	Evaluar proceso de la organización.

7.1.4.8 Requisito funcional: baja de evaluación

Descripción	Módulo donde se eliminan datos de la evaluación.
Entrada	Se selecciona la opción de baja de evaluación completa.
Salida	Se eliminarán del sistema todos los datos de la evaluación.
Proceso	Debe ser posible borrar toda la evaluación completa, incluyendo los datos y diagramas generados en cada área de proceso identificada.
Referencias	Evaluación de modelado de proceso de la organización.

Casos de uso	Evaluar proceso de la organización.

7.1.5 Generar plan de mejora

7.1.5.1 Requisito funcional: generar plan preliminar

Descripción	Módulo dedicado a la agrupación de prácticas débiles y su asignación de prioridades.
Entrada	En base a los resultados de evaluaciones el plan preliminar mostrará aquellas prácticas que resultaron ser las más débiles.
Salida	Las listas ahora mostrarán las actividades que se deben realizar para corregir esas prácticas débiles y para ser agendadas en el plan de mejora.
Proceso	El equipo de evaluación analizara los documentos previos y las actividades de mejora.
Referencias	Ninguna.
Casos de uso	Generar plan de mejora.

7.1.5.2 Requisito funcional: priorizar mejoras

Descripción	Este módulo corresponde al acuerdo que se realizará para establecer el conjunto de prioridades hacia las prácticas débiles encontradas.
Entrada	Un estudiante establecerá una prioridad por cada práctica débil identificada, por cada una de ellas se establece al menos uno de los siguientes estados: Alta, Media y Baja. Al establecer la prioridad aparecerá un cuadro de texto donde deberá explicar la justificación de esta toma de decisión con respecto a la práctica seleccionada y a la prioridad.
Salida	Se actualizarán las prioridades en cada área seleccionada.
Proceso	Como parte de establecimiento del plan de mejora, en base a las prácticas más débiles encontradas, los alumnos serán quienes establezcan la priorización.
Referencias	Generar plan preliminar, Generar plan de acción, Crear calendarios.
Casos de uso	Generar plan de mejora.

7.1.5.3 Requisito funcional: generar plan de acción

Descripción	De manera ordenada se mostrará el conjunto de áreas priorizadas que necesitan corregirse, y a partir de éstas se hará el plan de acción.
Entrada	Una vez establecida la prioridad a la actividad, se mostrará un botón para agendar dicha actividad, al pulsar sobre él se mostrará el calendario.
Salida	Se mostrará el panel calendario.
Proceso	Se guardarán las fechas programadas y se anexará a la agenda de cada usuario participante.
Referencias	Priorizar mejoras, crear calendario, modificar calendario, borrar calendario.
Casos de uso	Generar plan de mejora.

7.1.5.4 Requisito funcional: crear calendario

Descripción	Módulo que de manera gráfica representa un calendario y se dan opciones para calendarizar actividades.
Entrada	Una vez pulsado el botón Agendar actividad, se mostrará el panel donde establecerá el título (ob) de la actividad, Fecha de inicio (ob), Fecha de fin (ob), y en una lista agregará a los usuarios que serán notificados por email de dicha actividad.
Salida	Se guardará en el calendario las fechas y horas propuestas, además de notificar a todos los usuarios involucrados mediante email, en el cuerpo del mail ira la descripción de la actividad a realizar junto con la justificación que el alumno redactó.
Proceso	Mediante una herramienta gráfica se reservarán rangos de fechas y/o horas para programar una actividad, escribiendo el nombre del evento, lugar, fecha de inicio, fecha de fin, hora de inicio, hora de fin. El rango puede ser desde horas, días, semanas y meses. Se puede seleccionar a todos los usuarios que participan en el proyecto para notificarles automáticamente, además que se puede elegir manualmente por la dirección de correo electrónico. Estos recibirán una notificación en su correo y se les dará una liga para aceptar, rechazar o proponer otra fecha para realizar las actividades.
Referencias	Generar plan de acción.
Casos de uso	Generar plan de mejora.

7.1.5.5 Requisito funcional: modificación de calendario

Descripción	Módulo que de manera gráfica se dan opciones para modificar las actividades calendarizadas especificadas en el requisito crear calendario.
Entrada	En caso que un usuario proponga otra fecha para realizar actividades, lo hará de la misma manera que el caso de uso crear calendario.
Salida	Se guardará en el calendario las fechas y horas propuestas, además de notificar a todos los usuarios involucrados.
Proceso	Se puede seleccionar a todos los usuarios que participan en el proyecto parta notificarles automáticamente, además que se puede elegir manualmente por la dirección de correo electrónico. Estos recibirán una notificación en su correo y se les dará una liga para aceptar, rechazar o proponer otra fecha para realizar las actividades.
Referencias	Generar plan de acción, Crear calendario.
Casos de uso	Generar plan de mejora.

7.1.5.6 Requisito funcional: baja de calendario

Descripción	Módulo que elimina permanentemente un rango de fecha programado
Entrada	Seleccionar el calendario y la opción eliminar calendario.
Salida	Se eliminarán las fechas de actividades asociadas a ese calendario y la agenda de los participantes también se actualizará.
Proceso	Se puede eliminar un calendario completo y todas sus fechas programadas, al hacer esto la agenda de todos los usuarios que estaban involucrados y las fechas que habían sido programadas serán también eliminadas. Esto sucede en caso de que los usuarios estén de acuerdo en programar otra calendarización para sus actividades.
Referencias	Generar plan de acción, Crear calendarios.
Casos de uso	Generar plan de mejora.

7.2 Casos de uso del sistema

En esta sección se incluyen los diferentes casos de uso en los que se ha dividido el soporte tecnológico de esta tesis. En primer lugar se incluye el diagrama de contexto general que proporciona una idea aproximada sobre las áreas funcionales en las que se divide el sistema y a continuación se da una explicación más detallada de cada una de éstas.

7.2.2 Casos de uso de contexto general

La Figura B.1 muestra el diagrama de contexto de casos de uso general definido para el sistema.

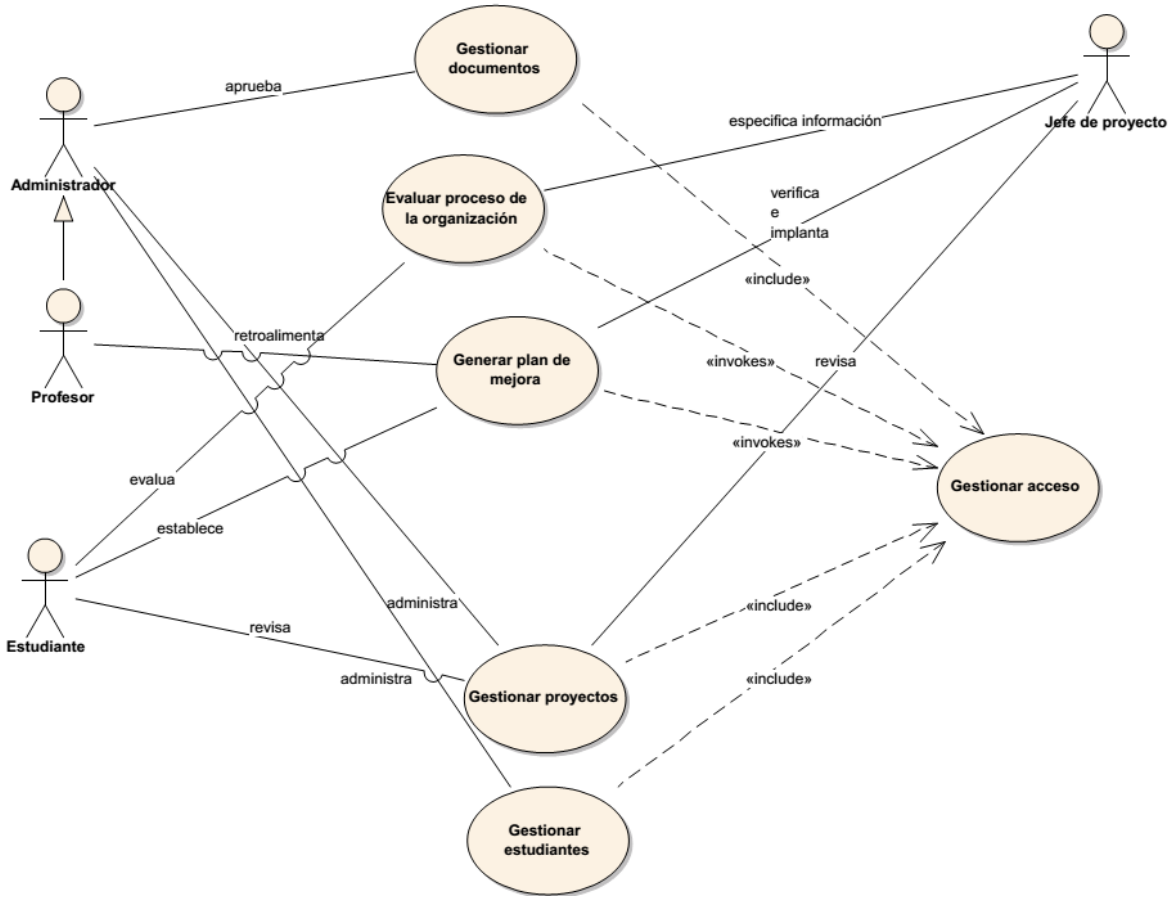


Figura B.1. Modelo del caso de uso de contexto general

En este diagrama de contexto se pueden observar a los cuatro actores que intervienen en el sistema:

- Administrador: Persona que estará dedicada a la administración de la herramienta y tendrá los privilegios especiales para eliminar, agregar, modificar y crear nuevos datos.
- Profesor: Persona que, además de ser un administrador, tendrá participación en los diferentes módulos para revisar y aprobar diversos documentos. En este documento se referirá de manera natural al profesor, pero se entiende que es un tipo de usuario que tiene privilegios de administrador.

- Jefe de proyecto: Persona que se dedicará a modelar los procesos de su empresa, usando el módulo de evaluar proceso de la organización, responder los cuestionarios, recibir informes sobre planes de mejora y apoyar y evaluar a los estudiantes durante su participación en los proyectos.
- Estudiante: Persona que evaluará los procesos de la organización, establecerá planes de mejora y realizará labores propias de la iniciativa de mejora dentro de un equipo de trabajo con un rol específico dentro de su equipo de estudiantes.

Los casos de uso que componen el diagrama de contexto general se definen como:

- Gestionar acceso. Módulo encargado de realizar el *login* o acceso a la herramienta y creación de catálogos de datos básicos que serán referenciados durante todos los proyectos.
- Gestionar estudiantes. Módulo que se encargará de mostrar los datos a detalle de cada estudiante.
- Gestionar proyectos. Este módulo se encargará de listar los proyectos principalmente y el modulo para editarlos donde se podrán relacionar a los equipos de estudiantes, jefes de proyectos y profesores que participarán en un proyecto de mejora.
- Evaluar proceso de la organización. Este módulo comprenderá la realización del diagrama actual de proceso de la organización y su comparación contra el propuesto por el modelo de referencia (MoProSoft), además de la evaluación basada en cuestionarios tipo likert.
- Establecer plan de mejora. Módulo donde se proporcionarán herramientas para planificar actividades para la iniciativa de mejora.
- Gestionar documentos. Este módulo se encargará de gestionar los documentos que formarán el historial de la iniciativa de mejora, además de facilitar la revisión y aprobación de estos.

7.2.2.1 Gestionar acceso

La Figura B.2 muestra al modelo del primer caso de uso del sistema que permite el acceso a la herramienta. Los casos de uso que lo componen se pueden definir de la siguiente manera:

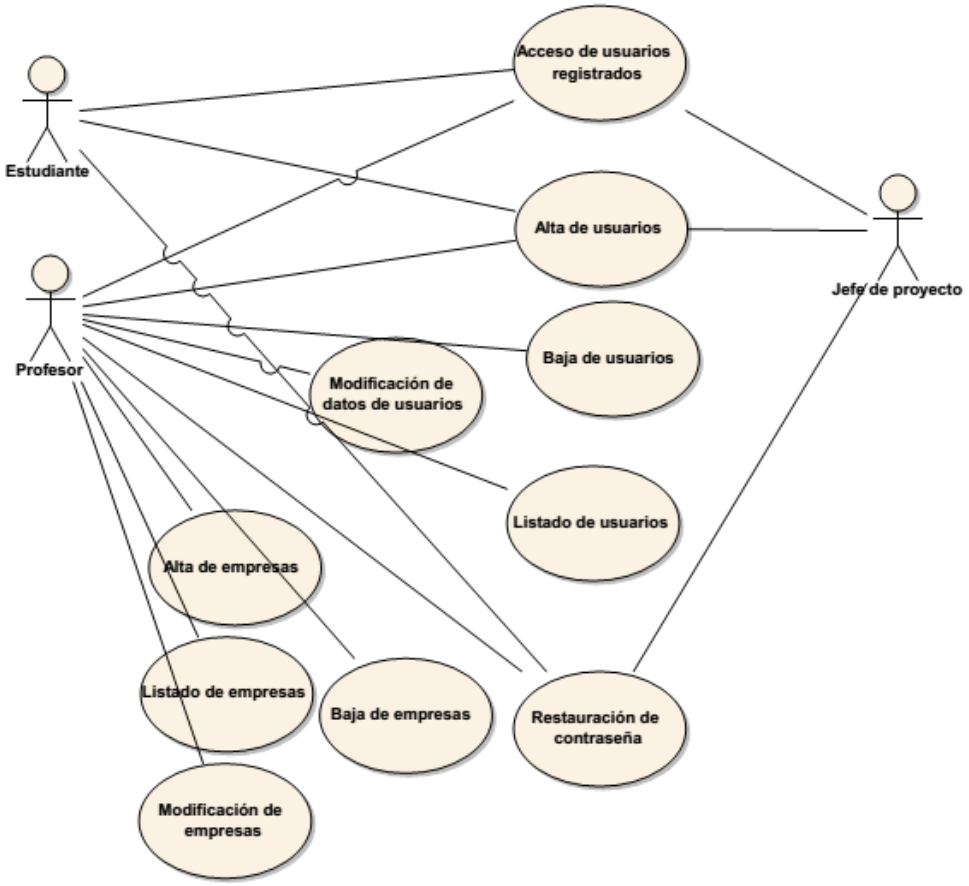


Figura B.2. Modelo del caso de uso para gestionar acceso

- **Caso de uso:** Alta de usuarios.
  - **Actores:** Administrador.
  - **Propósito:** Realizar el alta de usuarios en el sistema a fin de acceder a sus componentes.
  - **Visión general:** Al hacerlo se deberá identificar si se trata de un alumno, profesor o jefe de proyecto. Una vez que la alta haya sido realizada, un administrador tendrá que verificar los datos y activará la cuenta en caso de considerarlo conveniente.
  - **Referencias:** Requisito N° 7.1.1.2 Alta de usuarios
  - **Curso típico de eventos:**

Usuario (alumno, profesor, jefe de proyecto)	Sistema
1. El usuario accede a la pantalla de creación de nueva cuenta desde el menú principal del sistema en usuarios al panel de Alta de usuarios.	2. El sistema muestra una pantalla de registro.

3. El usuario rellena los datos obligatorios para el alta de usuarios.	4. El sistema valida los campos obligatorios y su formato. Una vez comprobada la validez de los datos se almacenan en la base de datos, se muestra un mensaje de confirmación del alta del usuario.

- **Cursos alternativos:**

  Paso 2a: Para cada tipo de usuario diferente, el sistema requerirá los campos apropiados, en el caso de alta de jefes de proyectos, se pedirá el nombre de su empresa y en caso de no existir, se le proporcionara un enlace al panel de alta de empresas.

  Paso 5: En caso que exista un problema de conexión con la base de datos, mostrar un mensaje de error al usuario.

- **Caso de uso:** Listado de usuarios.
  - **Actores:** Profesor.
  - **Propósito:** Mostrar una lista de usuarios para acceder a las diferentes acciones que se pueden realizar sobre una cuenta (baja o modificación).
  - **Visión general:** Esta lista podrá ser definida con mayor precisión utilizando criterios de búsqueda que sirvan como filtros para identificar con mayor facilidad a usuarios en particular.
  - **Referencias:** Requisito N° 7.1.1.4 Listado de usuarios, 7.1.3.3 Baja de usuarios.
  - **Curso típico de eventos:**

Profesor	Sistema
1. El profesor acede desde el menú principal del sistema en la opción usuarios al panel de listado de usuarios.	2. Se cargan los usuarios en un *gridpanel*, utilizando paginado en los datos, separándolos por el tipo de usuario y mostrando las opciones para ejecutar las acciones que son modificar y eliminar. Además se proporcionan filtros de búsqueda en los encabezados del *gridpanel* para establecer búsquedas en base al nombre de usuario o correo electrónico para ser rellenados opcionalmente.
3. El profesor realiza la búsqueda de los usuarios utilizando los filtros de búsqueda opcionales y presiona el botón buscar.	4. Se comienza a realizar la búsqueda y presentación de los datos (si existiesen) en el *gridpanel*.

  - **Cursos alternativos:**

    Ninguno.

- **Caso de uso:** Modificación de datos de usuarios.

- **Actores:** Profesor.
- **Propósito:** Actualizar los distintos datos de los usuarios así como el tipo de usuario en caso de ser necesario.
- **Visión general:** El profesor puede modificar todos los datos generales de una cuenta en particular y actualizar el tipo de usuario.
- **Referencias:** Requisito N° 7.1.1.4 Listado de usuarios, 7.1.1.6 Modificación de datos de usuarios.
- **Curso típico de eventos:**

Profesor	Sistema
1. Se accede al panel de listado de usuarios.	2. Se carga el *gridpanel* que contiene los usuarios activos en el sistema y se proporcionan filtros de búsqueda para usuarios.
3. Escribe en los filtros de búsqueda, se localiza al usuario en el *gridpanel* y se accede a la opción modificar.	4. Se presenta un panel donde se le piden todos los campos correspondientes asociados a la cuenta del usuario y los botones guardar y cancelar.
5. Modifica los campos presentados y accede a la opción guardar.	6. Se validan los datos obligatorios así como el formato de cada uno validando su formato y excluyendo caracteres especiales, se presenta un mensaje de confirmación de la acción al usuario.
7. El usuario confirma la realización de los cambios.	8. Los cambios se guardan en la base de datos y se presenta un mensaje de éxito de la acción realizada.

- **Cursos alternativos:**

  Paso 4a: En caso que el profesor desee cambiar el tipo de usuario (jefe de equipo, alumno, profesor) el sistema deberá pedir los campos necesarios de acuerdo al tipo de usuario nuevo al que se desee cambiar.

  Paso 6a: En caso de encontrar datos con formato erróneo o caracteres especiales, mostrar un mensaje de error al usuario.

  Paso 8a: En caso de presentarse un problema para actualizar los datos en la base de datos, se presenta un mensaje de error.

- **Caso de uso:** Baja de usuarios.
  - **Actores:** Profesor.
  - **Propósito:** Eliminar las cuentas relacionadas con los usuarios que presenten inactividad o duplicidad.
  - **Visión general:** El profesor decide en que momento eliminar usuarios de acuerdo a diferentes situaciones que se pudieran presentar.

- **Referencias:** Requisito N° 7.1.1.4 Listado de usuarios, 7.1.1.3 Baja de usuarios.
- **Curso típico de eventos:**

Profesor	Sistema
1. Se accede al panel de listado de usuarios.	2. Se carga el *gridpanel* que contiene los usuarios activos en el sistema y se proporcionan filtros de búsqueda para usuarios. Además que por cada campo de un usuario se proporciona la opción eliminar.
3. El profesor busca al usuario y accede a la opción eliminar cuenta.	4. Se emite un mensaje de confirmación para eliminar el usuario.
5. El profesor confirma la eliminación de la cuenta.	6. Se elimina de la base de datos toda la información relacionada únicamente con datos de la persona, se muestra un mensaje de confirmación y se actualiza la lista de usuarios mostrados en el *gridpanel* descrito en listado de usuarios.

- **Cursos alternativos:**

  Paso 4a: El sistema detecta algún error en el borrado y da un mensaje de error al no poder completar la acción.

- **Caso de uso:** Acceso de usuarios registrados.
  - **Actores:** Todos los usuarios.
  - **Propósito:** Validar sus datos mediante un *login* y contraseña a fin de garantizar su acceso al sistema.
  - **Visión general:** Todo usuario previamente registrado debe ingresar sus datos de acceso.
  - **Referencias:** Requisito N° 7.1.1.1 Acceso de usuarios registrados.
  - **Curso típico de eventos:**

Usuario	Sistema
1. Accede a la url del sitio web.	2. Se muestran en pantalla los campos usuario, contraseña y aceptar.
3. Introduce su nombre de usuario, contraseña y hace clic en el botón aceptar.	4. Se procede a validar los campos ingresados, descartando aquellos que contienen caracteres no válidos, después de pasar por este filtro, se procede a realizar la conexión con la base de datos comprobando la existencia del usuario y contraseña. Una vez realizada esa comprobación y de ser existentes los datos, se procederá a cargar el menú principal del sistema que permite

	acceder a los componentes asociados para cada tipo de usuario.

- **Cursos alternativos:**

  Paso 4a: en caso de detectar datos no validos o un usuario y/o contraseña incorrecta, presentar un mensaje de error.

  Paso 4b: En caso de presentarse un problema de conexión a la base de datos, mostrar un mensaje de error.

- **Caso de uso:** Restauración de contraseña.
  - **Actores:** Profesor.
  - **Propósito:** Mecanismo para recuperar la contraseña en caso de ser olvidada por los usuarios.
  - **Visión general:** La contraseña es enviada a la dirección de correo que se registró previamente.
  - **Referencias:** Requisito N° 7.1.1.5 Restauración de contraseña.
  - **Curso típico de eventos:**

Usuario	Sistema
1. Accede al módulo de listado de usuarios.	2. Se presentan las listas de usuarios y las opciones para modificar o borrar.
3. El profesor busca el usuario y accede a la opción de editar.	4. Se carga un panel donde se muestran los campos que se pueden editar.
5. Introduce una nueva contraseña y hace clic en guardar.	6. Se validan los nuevos datos guardados y se actualizan en la base de datos, enviando un mensaje de datos actualizados correctamente.

  - **Cursos alternativos:**

    Paso 6a: En caso de tener un formato no válido del correo electrónico, mostrar un mensaje de error.

    Paso 6b: En caso de no existir el correo en la base de datos mostrar un mensaje de error.

    Paso 6c: En caso de presentarse un problema de conexión con la base de datos, presentar un mensaje de error.

- **Caso de uso:** Alta de empresas.
  - **Actores:** Profesor, Jefe de proyectos.
  - **Propósito:** Se deben dar de alta todos los datos de aquellas empresas con las que se estará trabajando durante los proyectos.
  - **Visión general:** Cuando se crea la cuenta de un jefe de proyecto, se le pide que también proporcione su empresa, en caso de no existir, la puede dar de alta desde este mismo panel.
  - **Referencias:** Requisito N° 7.1.1.7 Alta de empresas, 7.1.1.2 Alta de usuarios.
  - **Curso típico de eventos:**

Usuario (profesor, jefe de proyectos)	Sistema
1. Accede al panel de alta de empresas desde su panel de menús.	2. Se carga el panel de alta de empresas y sus campos correspondientes para ser rellenados además de botones de aceptar y cancelar.
3. Se deben proporcionar los datos obligatorios y se da clic en aceptar.	4. Se validan los datos en cuanto a formato y caracteres permitidos. Se guardan en el sistema mostrando un mensaje de confirmación.

  - **Cursos alternativos:**

    Paso 1b: En el caso que se da de alta una cuenta de jefe de proyectos. Accede al panel de alta de empresas desde el panel de alta de usuario en caso de no existir su empresa en el listado de empresas.

    Paso 4b: Se debe verificar que no hayan sido dadas de alta las empresas anteriormente a fin de evitar tener datos duplicados, si existen mostrar mensaje de error.

    Paso 4c: En caso de presentarse un problema con la base de datos, presentar un mensaje de error.

- **Caso de uso:** Listado de empresas.
  - **Actores:** Profesor.
  - **Propósito:** A modo de establecer acciones con datos de una empresa en particular, se debe proporcionar este mecanismo para su búsqueda.
  - **Visión general:** Los dos tipos de usuario deberán tener acceso a estos datos a fin de buscar una empresa en particular, para el jefe de proyecto le sirve para completar su alta en el sistema y al profesor para gestionar los datos de las empresas.
  - **Referencias:** Requisito N° 7.1.1.8 Listado de empresas, 7.1.1.2 Alta de usuarios, 7.1.1.6 Modificación de datos de usuarios.
  - **Curso típico de eventos:**

Usuario	Sistema
1. Accede al menú Empresas, panel listado de empresas desde su menú principal.	2. Se carga el panel de listado de empresas, mostrando en un *gridpanel* todas las empresas dadas de alta y proporcionando filtros de búsqueda de acuerdo al nombre de la empresa. Por cada empresa se muestra la opción modificar y eliminar empresa.

- **Cursos alternativos:**

  Ninguno.

- **Caso de uso:** Modificación de empresas.
  - **Actores:** Profesor.
  - **Propósito:** Modificar datos de empresas.
  - **Visión general:** En caso de presentarse problemas con datos de las empresas, estos pueden ser modificados solamente por el profesor.
  - **Referencias:** Requisito N° 7.1.1.9 Modificación de empresas.
  - **Curso típico de eventos:**

Profesor	Sistema
1. Desde el menú principal accede a Empresas y luego al panel de listado de empresas.	2. Se carga el panel de listado de empresas, mostrando en un *gridpanel* todas las empresas dadas de alta y proporcionando filtros de búsqueda de acuerdo al nombre de la empresa. Por cada empresa se muestra la opción modificar y eliminar datos de empresa.
3. Identifica un empresa en particular y selecciona la opción modificar datos de empresa.	4. Se muestra el panel de modificación de empresas, sus campos correspondientes a los datos de la empresa para ser modificados y los botones aceptar y cancelar.
5. Se introducirán los datos referentes a la empresa y se da clic en aceptar.	6. Se pide un mensaje de confirmación y se validan los datos obligatorios, su formato correspondiente, se actualizan en la base de datos y se muestra un mensaje que confirma la actualización de los datos.

  - **Cursos alternativos:**

    Paso 6a: En caso de presentarse problemas internos con la base de datos y de no ser posible realizar los cambios, mostrar un mensaje de error.

- **Caso de uso:** Baja de empresas.
  - **Actores:** Profesor.

- **Propósito:** Eliminar de la base de datos una empresa en particular.
- **Visión general:** En caso de presentarse datos duplicados o problemas relacionados con datos de una empresa, se da esta opción para eliminar todos los datos referentes a una empresa de un solo paso.
- **Referencias:** Requisito N° 7.1.3.10 Baja de empresas.
- **Curso típico de eventos:**

Profesor	Sistema
1. Desde el menú principal accede a Empresas y luego al panel de listado de empresas.	2. Se carga el panel de listado de empresas, mostrando en un *gridpanel* todas las empresas dadas de alta y proporcionando filtros de búsqueda de acuerdo al nombre de la empresa. Por cada empresa se muestra la opción modificar y eliminar datos de empresa.
3. Selecciona la opción eliminar en la empresa que desee.	4. Se muestra un mensaje para que el usuario confirme la acción. Se eliminan los datos de la empresa en la base de datos y en caso de estar asociados a un usuario en particular se deja en *null* la referencia. Se muestra un mensaje confirmando la eliminación de la empresa.

- **Cursos alternativos:**

  Paso 3a. En caso de que no se pueda eliminar de la base de datos, se muestra un mensaje de error.

7.2.1.2 Gestionar estudiantes

La Figura B.3 muestra el modelo del caso de uso que permite realizar la búsqueda por estudiante a fin de facilitar el acceso a sus datos. Los casos de uso que lo describen son los dos siguientes:

Figura B.3. Modelo del caso de uso para gestionar estudiantes

- **Caso de uso:** Listado de estudiantes.
  - **Actores:** Profesor.
  - **Propósito:** Realizar la búsqueda de estudiantes.
  - **Visión general:** El profesor realiza búsquedas al fin de poder acceder después a sus datos referentes a las estadísticas por estudiante.
  - **Referencias:** Requisito N° 7.1.2.1 Listado de estudiantes.
  - **Curso típico de eventos:**

Profesor	Sistema
1. Desde el menú principal en estudiantes, accede al panel de listado de estudiantes.	2. Se muestra un *gridpanel* con la lista de todos los estudiantes, mostrando sus principales datos además de la opción ver detalles.
3. Desde los filtros de búsqueda del *gridpanel* se introducen datos para refinar la búsqueda y se presiona el botón buscar.	4. Se mostrarán los resultados en el mismo *gridpanel*. Si no hubo datos que se introdujeron en los filtros de búsqueda, mostrar por default todos los datos existentes, pero paginándolos en diez resultados.

  - **Cursos alternativos:**

    Ninguno.

- **Caso de uso:** Estadísticas por estudiante.
  - **Actores:** Profesor.
  - **Propósito:** Después de revisar el listado de estudiantes el profesor podrá visualizar con mayor detalle los datos de un estudiante en particular.
  - **Visión general:** Esta funcionalidad le permitirá revisar y actualizar todos los datos correspondientes a un alumno.
  - **Referencias:** Requisito N° 7.1.2.2 Estadísticas por estudiante.
  - **Curso típico de eventos:**

Profesor	Sistema
1. Desde el listado de alumnos se accede a la opción ver detalles.	2. Se muestra un panel que mostrará datos referentes a los proyectos en los que ha participado el estudiante y el estado del proyecto.

  - **Cursos alternativos:**

    Ninguno.

7.2.1.3 Gestionar proyectos

La Figura B.4 muestra el modelo del caso de uso relacionado con la manipulación de los datos de los proyectos, su inicialización, visualización y actualización. Los casos de uso que lo componen se definen de la siguiente manera:

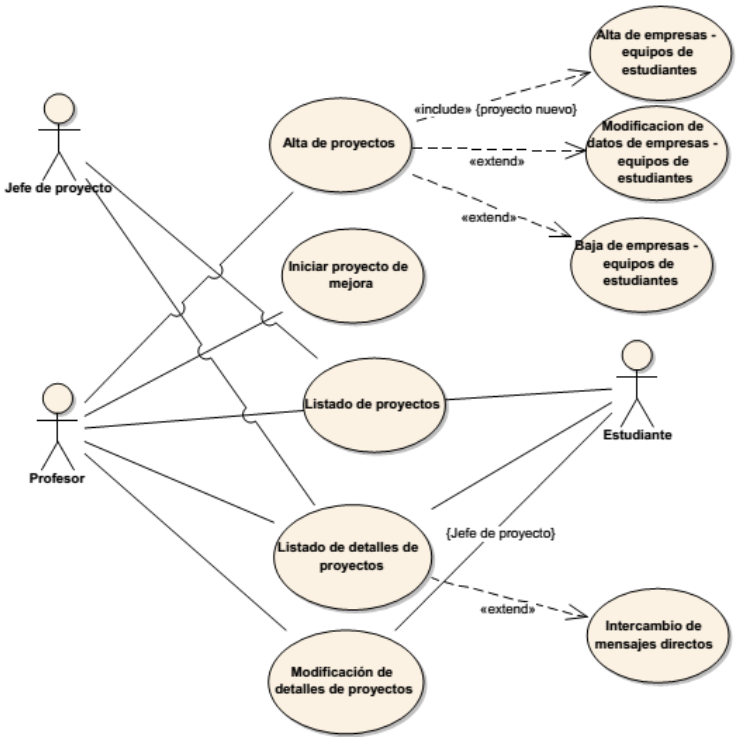


Figura B.4. Modelo del caso de uso para gestionar proyectos

- **Caso de uso:** Listado de proyectos.
  - **Actores:** Profesor, jefe de proyecto y estudiante.
  - **Propósito:** Acceder a los proyectos que cada usuario tiene asignado para trabajar con las iniciativas de mejora.
  - **Visión general:** Los usuarios accederán a este módulo y podrán ver las diferentes acciones que se pueden realizar con cada proyecto, dependiendo del tipo de usuario.
  - **Referencias:** Requisito N° 7.1.3.6 Listado de proyectos.
  - **Curso típico de eventos:**

Usuario (profesor, jefe de proyecto, estudiante)	Sistema
1. Del menú principal accederá a proyectos y accede al panel listado de proyectos.	2. Se presentará mediante un *gridpanel* una lista de todos los proyectos ordenados descendentemente utilizando fecha de creación, mostrando solamente los proyectos en los que el usuario está involucrado. Solamente para el profesor se mostrarán las opciones modificar, eliminar e iniciar proyecto. Para todos los usuarios se mostrará la opción ver detalles.

- **Cursos alternativos:**

  Ninguno.

- **Caso de uso:** Alta de empresas – equipos de estudiantes.
  - **Actores:** Profesor.
  - **Propósito:** Vinculación de los diferentes usuarios con un proyecto específico.
  - **Visión general:** Dentro del alta de proyectos también se mostrarán los paneles para dar de alta los diferentes usuarios a los equipos del proyecto.
  - **Referencias:** Requisito N° 7.1.3.2 Alta de empresas-equipos de estudiantes.
  - **Curso típico de eventos:**

Profesor	Sistema
1. Del menú principal del listado de proyectos, teniendo seleccionado uno, accederá al menú Participantes.	2. Se mostrarán en el panel: Equipos de estudiantes. En una lista se podrán agregar los nombres de estudiantes que conformarán un equipo.
3. El profesor selecciona los grupos, personas participantes y selecciona la opción guardar datos.	4. Se guardan en la base de datos las relaciones con el proyecto y se manda un mensaje de confirmación de éxito.
5. Del menú principal del listado de proyectos, teniendo uno seleccionado, accederá al menú Jefes.	6. Se mostrarán en el panel: Una lista de jefes de proyectos.
7. Se agregan a una lista los jefes de proyectos que asociará al proyecto y pulsa el botón guardar.	8. Los datos se guardan en la base de datos y se muestra un mensaje de confirmación de éxito.

  - **Cursos alternativos:**

    Paso 3a y 8a: En caso que no se pueda guardar en la base de datos mostrar un mensaje de error.

- **Caso de uso:** Modificación de empresas – equipos de estudiantes.
  - **Actores:** Profesor.
  - **Propósito:** Modificar la lista de usuarios involucrados en un proyecto.
  - **Visión general:** El profesor puede editar las listas de los distintos usuarios involucrados en un proyecto en particular.
  - **Referencias:** Requisito N° 7.1.3.3 Modificación de empresas – equipos de estudiantes.
  - **Curso típico de eventos:**

Profesor	Sistema
1. Del menú principal seleccionará proyectos y accede al panel de listado de proyectos.	2. Se presentará mediante un *gridpanel* una lista de todos los proyectos ordenados descendentemente utilizando fecha de creación, mostrando solamente los proyectos en los que el usuario está involucrado. Solamente para el profesor se mostrarán las opciones modificar, eliminar e iniciar proyecto.
3. Selecciona la opción modificar para un proyecto en particular	4. Se carga un nuevo panel donde se muestran los diferentes paneles que permiten modificar los equipos de alumnos, profesores, empresa y jefes de proyectos creados anteriormente para el proyecto específico.
5. Modifica las listas agregando o quitando participantes y da clic al botón aceptar.	6. Se muestra un mensaje de confirmación al usuario. Se actualizan las relaciones en la base de datos y se muestra un mensaje confirmando el éxito de la acción.

- **Cursos alternativos:**

  Paso 6b: En caso que se produzca un problema para actualizar la base de datos, mostrar un mensaje de error al usuario.

- **Caso de uso:** Baja de empresas – equipos de estudiantes.
  - **Actores:** Profesor.
  - **Propósito:** Eliminar todos los vínculos existentes entre los diferentes usuarios a un proyecto, de un solo paso.
  - **Visión general:** El profesor seleccionará un proyecto en particular y lo eliminará en caso de encontrar problemas con este.
  - **Referencias:** Requisito N° 7.1.3.4 Baja de empresas-equipos de estudiantes.
  - **Curso típico de eventos:**

Profesor	Sistema
1. Del menú principal selecciona proyectos y accede listado de proyectos para acceder al módulo.	2. Se presentará mediante un *gridpanel* una lista de todos los proyectos ordenados descendentemente utilizando fecha de creación, mostrando solamente los proyectos en los que el usuario está involucrado. Solamente para el profesor se mostrarán las opciones modificar, eliminar e iniciar proyecto.

3. Selecciona la opción eliminar para un proyecto en particular	4. Se abre un mensaje de confirmación preguntando si realmente desea eliminar el proyecto. Si acepta se eliminaran todos los vínculos en la base de datos que hay con respecto a ese proyecto y se mostrará un mensaje de eliminación exitosa.
5. Visualiza la información de pantalla.	6. Se actualiza el *gridpanel*.

- **Cursos alternativos:**

  Paso 4a. En caso de presentarse un problema con la base de datos, mostrar un mensaje de error al usuario.

- **Caso de uso:** Iniciar proyecto de mejora.
  - **Actores:** Profesor.
  - **Propósito:** Establecer el estado de un proyecto como iniciado a fin que todos los usuarios involucrados comiencen a trabajar sobre él.
  - **Visión general:** El profesor cambiara el estado del proyecto desde el listado de proyectos.
  - **Referencias:** Requisito N° 7.1.3.5 Iniciar proyecto de mejora.
  - **Curso típico de eventos:**

Profesor	Sistema
1. Del menú principal selecciona proyectos y accede listado de proyectos para acceder al módulo.	2. Se presentará mediante un *gridpanel* una lista de todos los proyectos ordenados descendentemente utilizando fecha de creación, mostrando solamente los proyectos en los que el usuario está involucrado. Solamente para el profesor se mostrarán las opciones modificar, eliminar e iniciar proyecto.
3. Selecciona un proyecto y da clic a la opción iniciar proyecto.	4. Se muestra un mensaje pidiendo la confirmación de la acción. Si es aceptado, se modifica en la base de datos el estado del proyecto como iniciado.

  - **Cursos alternativos:**

    Paso 4a: En caso que no se pueda modificar el estado del proyecto, enviar u mensaje de error.

- **Caso de uso:** Listado de detalles de proyectos.
  - **Actores:** Profesor, jefe de proyectos, estudiante.
  - **Propósito:** Trabajar a través de este módulo con las actividades que llevarán las iniciativas de mejora.

- **Visión general:** Los usuarios accederán a este módulo para visualizar todos los elementos que están asociados a un proyecto que serán especificados en los siguientes casos de uso y que en el diagrama general de casos de uso se puede observar la relación que tiene el caso de uso gestionar proyectos con otros tres.
- **Referencias:** Requisito N° 7.1.3.7 Listado de detalles de proyectos.
- **Curso típico de eventos:**

Profesor	Sistema
1. Del menú principal accederá a proyectos y accede al panel listado de proyectos.	2. Se presentará mediante un *gridpanel* una lista de todos los proyectos ordenados descendentemente utilizando fecha de creación, mostrando solamente los proyectos en los que el usuario está involucrado. Solamente para el profesor se mostrarán las opciones modificar, eliminar e iniciar proyecto. Para todos los usuarios se mostrará la opción ver detalles.
3. Una vez localizado su proyecto en el *gridpanel* de listado de proyectos, selecciona la opción ver detalles.	4. Se carga un panel nuevo donde se muestran los datos del proyecto, además del módulo de chat y dependiendo del estado actual del proyecto con respecto al modelo IDEAL se habilitarán los diferentes módulos que son, generar plan de mejora y evaluar proceso de la organización.

- **Cursos alternativos:**

  Ninguno.

- **Caso de uso:** Modificación de detalles de proyectos.
  - **Actores:** Profesor, jefe de proyecto, estudiante (líder de equipo).
  - **Propósito:** Modificar el estado del proyecto con respecto a las fases del modelo IDEAL.
  - **Visión general:** Los usuarios pueden modificar el estado del proyecto, pero es el alumno con rol de líder de equipo quien regularmente ejecutará esta acción.
  - **Referencias:** Requisito N° 7.1.3.8 Modificación de detalles de proyectos.
  - **Curso típico de eventos:**

Usuario	Sistema
1. Del menú principal accederá a proyectos y accede al panel listado de proyectos.	2. Se presentará mediante un *gridpanel* una lista de todos los proyectos ordenados descendentemente utilizando fecha de creación, mostrando solamente los proyectos en los que el usuario está involucrado. Solamente para el profesor se mostrarán las opciones modificar, eliminar e iniciar proyecto. Para todos los usuarios se mostrará la opción ver detalles.

3. Accede a la opción ver detalles.	4. Se carga el panel de listado de detalles de proyectos donde se muestran todos los datos correspondientes al avance del proyecto, su estado con respecto al modelo IDEAL, el panel de chat y el acceso a los diferentes paneles que están asociados a este.
5. Desde el mismo panel de listado de detalles de proyectos se modifica el estado del proyecto, y datos generales sobre éste.	6. Se valida el perfil de usuario para poder realizar cambios. Los profesores, jefes de equipo y el alumno jefe de equipo solamente pueden modificar el estado del proyecto, en caso de alumnos que no son líderes de equipo, se mostrarán los datos como solo lectura. Una vez guardados los cambios, mostrar un mensaje de éxito.
7. Hace clic al botón guardar cambios.	8. Se muestra un mensaje confirmación de la acción. Una vez aceptado se actualiza la base de datos y se muestra un mensaje confirmando el éxito de la acción.

- **Cursos alternativos:**

  Paso 8a: En caso de presentarse un problema con los cambios en la base de datos, presentar un mensaje de error al usuario.

- **Caso de uso:** Intercambio de mensajes directos.
  - **Actores:** Todos los usuarios.
  - **Propósito:** Intercambio de mensajes de texto entre los usuarios involucrados en un proyecto en particular.
  - **Visión general:** Este módulo de chat es un canal de comunicación entre todos los usuarios.
  - **Referencias:** Requisito N° 7.1.3.9 Intercambio de mensajes directos.
  - **Curso típico de eventos:**

Usuarios	Sistema
1. Del menú principal accederá a chat.	2. Se presentará mediante un *gridpanel* una lista de todos los proyectos ordenados descendentemente utilizando fecha de creación, mostrando solamente los proyectos en los que el usuario está involucrado.
3. Realiza doble clic sobre un proyecto.	4. Se carga el panel de chat correspondiente al proyecto donde podrá intercambiar mensajes de texto directamente con los involucrados en el mismo proyecto.
5. Escribe mensajes de texto a través del panel de chat y oprime la tecla [enter] o da	6. Se encarga que el mensaje lo reciban todos los miembros del mismo proyecto y que

clic en enviar.	tengan su sesión activa.

- **Cursos alternativos:**

  Paso 6a: En caso de que no se puedan enviar los mensajes por un problema de conexión, mostrar un mensaje de error.

7.2.1.4 Gestionar proyectos

La Figura B.5 muestra el modelo de caso de uso que se encargará de almacenar todos los documentos y entregables que se generan a lo largo de la iniciativa de mejora. Este módulo estará disponible durante todos los estados del proyecto. Los casos de uso que lo componen se describen de la siguiente manera:

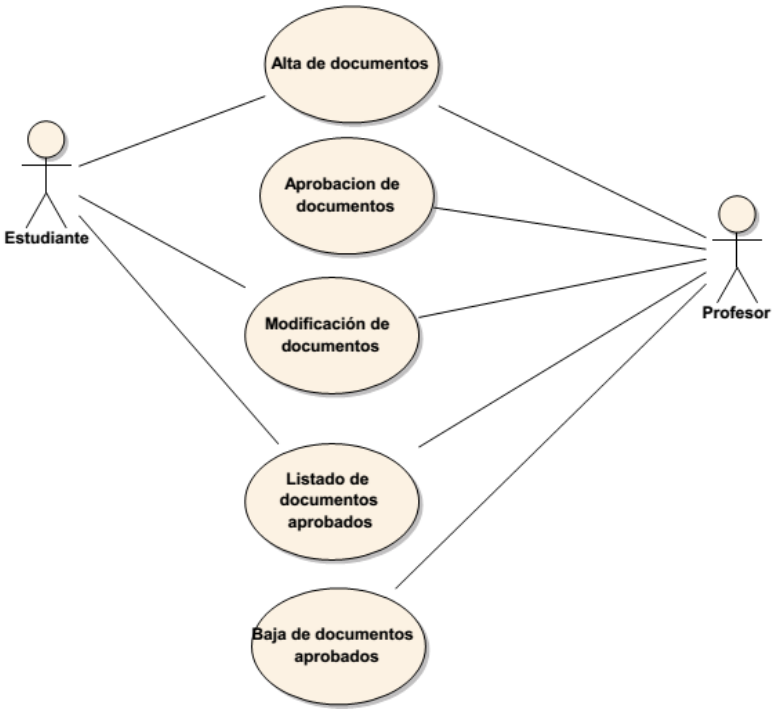


Figura B.5. Modelo del caso de uso para gestionar documentos

- **Caso de uso:** Alta de documentos.
  - **Actores:** Todos los usuarios.
  - **Propósito:** Alta de archivos en el repositorio del sistema para que sean compartidos entre los usuarios.
  - **Visión general:** Este módulo estará regulado por un profesor quien se encargará de revisar los archivos subidos por un usuario, antes que los demás puedan visualizarlos.
  - **Referencias:** Requisito N° 7.1.4.1 Alta de documentos.
  - **Curso típico de eventos:**

Usuario (alumno, profesor)	Sistema
1. Del menú principal accederá a proyectos y accede al panel Alta de documentos.	2. Se presentará mediante un *gridpanel* una lista de todos los documentos descendentemente utilizando fecha de creación. Para todos los usuarios se mostrará la opción ver detalles.
3. Una vez localizado su proyecto en el *gridpanel* de listado de proyectos, selecciona la opción ver detalles.	4. Se carga un panel nuevo donde se muestran los archivos subidos previamente, agrupándolos por proyecto.
5. Accede a la opción Agregar.	6. Se muestra un cuadro de selección de examinar, para abrir la ventana de explorador de archivos, se escribe el nombre del documento y se selecciona el proyecto.
7. Selecciona el/los archivos, introduce el nombre y selecciona el proyecto y da clic en aceptar.	8. Se comienza a subir al servidor el archivo y se almacena en una carpeta temporal para ser revisada posteriormente en el módulo de aprobación de documentos. Se muestra un mensaje de éxito del usuario.

- **Cursos alternativos:**

  Paso 7a: El usuario puede volver a hacer clic en subir archivos y repetir el paso 7.

  Paso 8a: En caso que ocurra un problema en la carga de archivos al servidor, mostrar un mensaje de error al usuario.

- **Caso de uso:** Aprobación de documentos.
  - **Actores:** Profesor.
  - **Propósito:** Descargar y revisar los documentos subidos en el repositorio.
  - **Visión general:** Una vez que el profesor le haya dado el visto bueno a los archivos, los aprobará para que permanezcan en el sistema y sean visibles para los demás usuarios alumnos.
  - **Referencias:** Requisito N° 7.1.4.2 Aprobación de documentos.
  - **Curso típico de eventos:**

Usuario	Sistema
1. Del menú principal accederá al panel Aprobar Documentos.	2. Se presentará mediante un *gridpanel* una lista de todos los archivos subidos previamente en el panel alta de documentos, agrupados por proyectos, cada uno con un *checkbox* y en la parte inferior los botones aprobar y rechazar.
3. De la lista hace doble clic a un archivo.	4. El archivo se descarga desde el servidor a la computadora local del profesor, para proceder a su revisión.

5. Después de darle el visto bueno, puede elegir si lo aprueba o rechaza con los botones.	6. En caso de ser aprobado, el archivo ya podrá ser visible para todos los demás usuarios en otra bandeja de archivos aprobados, en caso de ser rechazado, será marcado su estado como rechazado pero no será eliminado.

- **Cursos alternativos:**

  Paso 6a: En caso de presentarse un error con el manejo de un archivo, presentar un mensaje de error.

- **Caso de uso:** Listado de documentos aprobados.
  - **Actores:** Profesor, Alumno.
  - **Propósito:** Visualizar en una lista todos los archivos que pertenecen al repositorio de archivos a fin de tenerlos como apoyo a las iniciativas de mejora durante todas las fases del modelo IDEAL.
  - **Visión general:** Los usuarios accederán a este módulo para subir archivos que representen datos históricos sobre evaluaciones anteriores.
  - **Referencias:** Requisito N° 7.1.4.4 Listado de documentos aprobados.
  - **Curso típico de eventos:**

Usuario (alumno, profesor)	Sistema
1. Del menú principal accederá a proyectos y accede mediante el botón listado de proyectos.	2. Se presentará mediante un *gridpanel* una lista de todos los proyectos ordenados descendentemente utilizando fecha de creación, mostrando solamente los proyectos en los que el usuario está involucrado. Solamente para el profesor se mostrarán las opciones modificar, eliminar e iniciar proyecto. Para todos los usuarios se mostrará la opción ver detalles.
3. Una vez localizado su proyecto en el *gridpanel* de listado de proyectos, este es seleccionado y realiza clic sobre el botón Documentos.	4. Se muestra en un *gridpanel* todos los archivos ordenados descendentemente por fecha de subida y se muestran las opciones ver detalles y eliminar por cada uno.
5. Se hace doble clic o clic sobre el icono descargar.	6. El archivo se descarga en la computadora local.

  - **Cursos alternativos:**

    Paso 6a: Mostrar mensaje de error en caso de no haber archivo.

- **Caso de uso:** Baja de documentos aprobados.
  - **Actores:** Profesor.

- **Propósito:** Con el fin de establecer un mecanismo que permita eliminar documentos en caso de estar mal clasificados, o duplicados, se establece esta opción para eliminación permanente de documentos del repositorio.
- **Visión general:** De la lista de documentos aprobados y por aprobar se tiene la opción para eliminar el documento.
- **Referencias:** Requisito N° 7.1.4.5 Baja de documentos aprobados.
- **Curso típico de eventos:**

Profesor	Sistema
1. Del menú principal accederá a proyectos y accede al panel listado de proyectos.	2. Se presentará mediante un *gridpanel* una lista de todos los proyectos ordenados descendentemente utilizando fecha de creación, mostrando solamente los proyectos en los que el usuario está involucrado. Solamente para el profesor se mostrarán las opciones modificar, eliminar e iniciar proyecto. Para todos los usuarios se mostrará la opción ver detalles.
3. Una vez localizado su proyecto en el *gridpanel* de listado de proyectos, selecciona uno de la lista y accede mediante el botón Documentos.	4. Se muestra en un *gridpanel* todos los archivos ordenados descendentemente por fecha de subida y se muestran las opciones ver detalles y eliminar por cada uno.
7. Se localiza un archivo en particular y se selecciona la opción eliminar.	8. Se pide confirmación por parte del usuario para realizar la acción y una vez confirmado, se elimina del repositorio de archivos el documento y se muestra un mensaje confirmando la eliminación.

- **Cursos alternativos:**

  Paso 8a: Mostrar un mensaje de error en caso de haber un problema con el archivo.

7.2.1.5 Evaluar proceso de la organización

La Figura B.6 muestra el modelo del caso de uso que contiene todas las opciones gráficas para modelar el proceso actual de las empresas y evaluarlo ya sea a través del mismo diagrama generado o de la aplicación de cuestionarios de evaluación basados en el modelo de referencia (MoProSoft), este módulo estará disponible cuando el estado del proyecto corresponda a la fase de evaluación en el modelo IDEAL. Los casos de uso que lo componen se definen como:

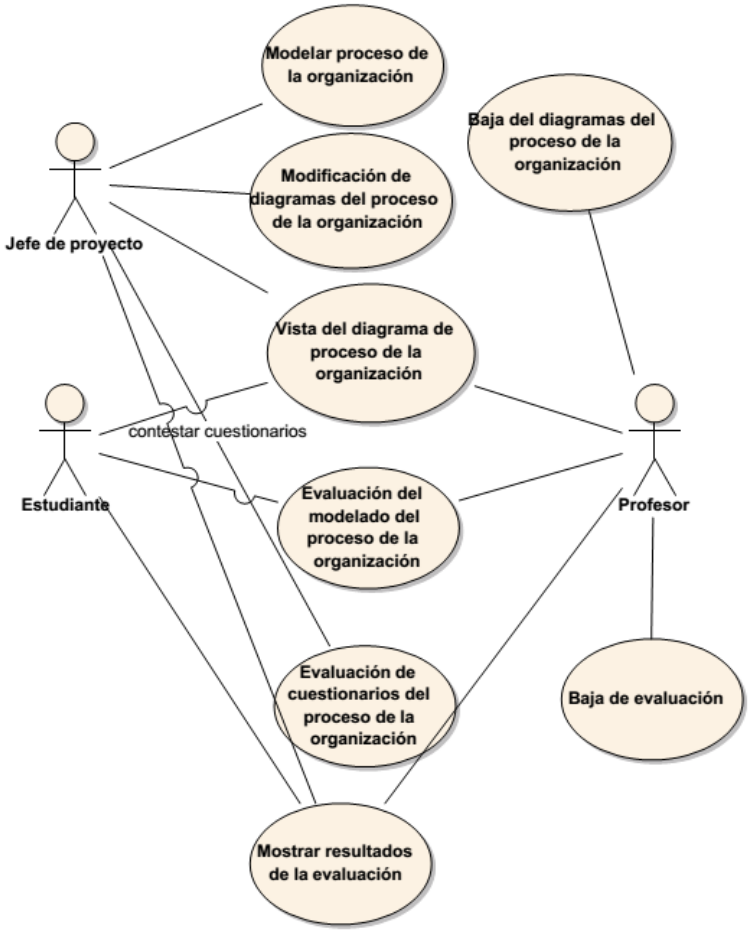


Figura B.6. Modelo del caso de uso para evaluar el proceso de la organización

- **Caso de uso:** Modelar proceso de la organización.
  - **Actores:** Jefe de proyecto.
  - **Propósito:** Este módulo permitirá que el jefe de proyecto modele el proceso de su organización a través de una herramienta gráfica de dibujo.
  - **Visión general:** De esta forma, se generará más adelante un proceso de evaluación que comparará el diagrama del jefe de proyecto con el diagrama establecido por el modelo de referencia (MoProSoft).
  - **Referencias:** Requisito N° 7.1.5.1 Modelar proceso de la organización.
  - **Curso típico de eventos:**

Jefe de Proyecto	Sistema
1. Del menú principal accederá a proyectos y accede al panel listado de proyectos.	2. Se presentará mediante un *gridpanel* una lista de todos los proyectos ordenados descendentemente utilizando fecha de creación, mostrando solamente los proyectos en los que el usuario está involucrado. Solamente para el profesor se mostrarán las opciones modificar, eliminar e iniciar

	proyecto. Para todos los usuarios se mostrará la opción ver detalles.
3. Una vez localizado su proyecto en el *gridpanel* de listado de proyectos, selecciona la opción propiedades del proyecto.	4. Se carga un panel nuevo donde se habilitarán los diferentes módulos que son, generar plan de mejora (Diagramas) y evaluar proceso de la organización (Cuestionarios).
5. Accede al panel de Diagramas.	6. Se carga un panel nuevo donde se muestran la herramienta gráfica que permitirá al usuario modelar las actividades de un proceso en particular de acuerdo a lo especificado en el requisito funcional 7.1.5.1. Se muestran las opciones de guardar y cerrar. Es posible crear varios diagramas para un solo proyecto.
7. Dibuja el diagrama mediante la herramienta que modele su proceso y lo guarda.	8. Se actualiza la base de datos con el nuevo diagrama y este ya aparece para su uso en el caso de uso *Evaluación del modelado del proceso de la organización*.

- **Cursos alternativos:**

  Paso 8a: En caso de presentarse un problema con la actualización en la base de datos, presentar un mensaje de error.

- **Caso de uso:** Modificación de diagramas del proceso de la organización.
  - **Actores:** Jefe de proyecto.
  - **Propósito:** Permitir que la misma herramienta que sirvió para el modelado, sirva para la actualización del mismo, sobrescribiendo el último cambio guardado.
  - **Visión general:** El jefe de proyecto realizará modificaciones al diagrama las veces que considere necesarias.
  - **Referencias:** Requisito N° 7.1.5.2 Modificación de diagramas del proceso de la organización.
  - **Curso típico de eventos:**

Jefe de Proyecto	Sistema
1. Del menú principal accederá a proyectos y accede al panel listado de proyectos.	2. Se presentará mediante un *gridpanel* una lista de todos los proyectos ordenados descendentemente utilizando fecha de creación, mostrando solamente los proyectos en los que el usuario está involucrado. Solamente para el profesor se mostrarán las opciones modificar, eliminar e iniciar proyecto. Para todos los usuarios se mostrará la opción ver propiedades del proyecto.
3. Una vez localizado su proyecto en el *gridpanel* de listado de proyectos, selecciona la opción propiedades de	4. Se carga un panel nuevo donde se habilitarán los diferentes módulos que son, generar plan de mejora (Diagramas) y

proyectos.	evaluar proceso de la organización (Cuestionarios).
5. Accede al panel de Diagramas.	6. Se abre la herramienta gráfica para editar el diagrama existente.
7. Modifica el diagrama existente y guarda los cambios.	8. Se actualizan los cambios en la base de datos.

- **Cursos alternativos:**

  Paso 8a: En caso de presentarse un problema con la actualización en la base de datos, presentar un mensaje de error.

- **Caso de uso:** Baja de diagramas del proceso de la organización.
  - **Actores:** Jefe de proyecto.
  - **Propósito:** Mecanismo para eliminar permanentemente el diagrama que está asociado a un proyecto.
  - **Visión general:** El profesor elimina un diagrama en caso de presentarse problemas durante la evaluación.
  - **Referencias:** Requisito N° 7.1.5.3 Baja de diagramas del proceso de la organización.
  - **Curso típico de eventos:**

Jefe de Proyecto	Sistema
1. Del menú principal accederá a proyectos y accede al panel listado de proyectos.	2. Se presentará mediante un *gridpanel* una lista de todos los proyectos ordenados descendentemente utilizando fecha de creación, mostrando solamente los proyectos en los que el usuario está involucrado. Solamente para el profesor se mostrarán las opciones modificar, eliminar e iniciar proyecto. Para todos los usuarios se mostrará la opción ver detalles.
3. Una vez localizado su proyecto en el *gridpanel* de listado de proyectos, selecciona la opción ver detalles.	4. Se carga un panel nuevo donde se habilitarán los diferentes módulos que son, generar plan de mejora (Diagramas) y evaluar proceso de la organización (Cuestionarios).
5. Accede al panel Diagramas.	6. Se carga en pantalla las listas de diagramas y se muestra una opción para eliminar.
7. Selecciona la opción eliminar.	8. Se pide confirmación del usuario. Una vez confirmado se realiza la eliminación en la base de datos y se muestra un mensaje de éxito confirmando la eliminación.

  - **Cursos alternativos:**

    Paso 8a: En caso de presentarse un problema con la actualización en la base de datos, presentar un mensaje de error.

- **Caso de uso:** Vista del diagrama del proceso de la organización.
  - **Actores:** Profesor, jefe de proyecto, alumno.
  - **Propósito:** Visualizar el diagrama creado en un formato para impresión.
  - **Visión general:** Todos los usuarios pueden acceder al diagrama en modo solo lectura para fin de analizarlo y compararlo contra el modelo de referencia MoProSoft.
  - **Referencias:** Requisito N° 7.1.5.4 Vista del diagrama del proceso de la organización.
  - **Curso típico de eventos:**

Usuario (profesor, jefe de proyecto, estudiante)	Sistema
1. Del menú principal accederá a proyectos y accede al panel listado de proyectos.	2. Se presentará mediante un *gridpanel* una lista de todos los proyectos ordenados descendentemente utilizando fecha de creación, mostrando solamente los proyectos en los que el usuario está involucrado. Solamente para el profesor se mostrarán las opciones modificar, eliminar e iniciar proyecto. Para todos los usuarios se mostrará la opción ver detalles.
3. Una vez localizado su proyecto en el *gridpanel* de listado de proyectos, selecciona la opción ver detalles.	4. Se carga un panel nuevo donde se habilitarán los diferentes módulos que son, generar plan de mejora (Diagramas) y evaluar proceso de la organización (Cuestionarios).
5. Accede al panel de Diagramas.	6. Se muestra el listado de diagramas.
7. Mediante doble clic selecciona uno del listado de diagrama.	8. Se carga en pantalla el diagrama y se muestran opciones para impresión del documento.

  - **Cursos alternativos:**

    Paso 7a: En caso de no existir el diagrama, mostrar un mensaje de error.

- **Caso de uso:** Evaluación del modelado del proceso de la organización.
  - **Actores:** Profesor, estudiante.
  - **Propósito:** Realizar una comparación del modelo creado por el jefe de proyecto y el modelo propuesto por MoProSoft de manera automática que arroje las diferencias.
  - **Visión general:** El sistema muestra las actividades que le hacen falta incorporar a la organización.
  - **Referencias:** Requisito N° 7.1.5.5 Evaluación del modelado del proceso de la organización.
  - **Curso típico de eventos:**

Usuario (profesor, estudiante)	Sistema
1. Del menú principal accederá a proyectos y accede al panel listado de proyectos.	2. Se presentará mediante un *gridpanel* una lista de todos los proyectos ordenados descendentemente utilizando fecha de creación, mostrando solamente los proyectos en los que el usuario está involucrado. Solamente para el profesor se mostrarán las opciones modificar, eliminar e iniciar proyecto. Para todos los usuarios se mostrará la opción propiedades de proyecto.
3. Una vez localizado su proyecto en el *gridpanel* de listado de proyectos, selecciona la opción propiedades de proyecto.	4. Se carga un panel nuevo donde se habilitarán los diferentes módulos que son, generar plan de mejora (Diagramas) y evaluar proceso de la organización (Cuestionarios).
5. Selecciona la opción Diagramas.	6. Se muestran las listas de diagramas y el botón comparar.
7. Se selecciona un diagrama de la lista y se da clic sobre el botón comparar.	8. En base al diagrama creado sobre un proyecto, se verifica a que área de proceso pertenece para hacer la comparativa con el repositorio de diagramas que ya se tienen precargados y que son referentes al modelo MoProSoft, y se tiene que presentar a modo de informe, las actividades que les hacen falta implantar al modelo que la organización tiene y se hacen sugerencias en base a las actividades que le faltan.

- **Cursos alternativos:**

  8a. En caso que el diagrama creado presente faltas, como es el caso de tener diagramas sin *start* o *end event*, elementos sin asociar mediante flechas, etc., mostrar un mensaje de elementos faltantes en el diagrama.

- **Caso de uso:** Evaluación de cuestionarios del proceso de la organización.
  - **Actores:** Jefe de proyecto.
  - **Propósito:** Como parte complementaria a la evaluación con el modelado de procesos, este módulo generará cuestionarios que los jefes de proyectos deberán responder y que los estudiantes analizarán posteriormente.
  - **Visión general:** Los jefes de proyecto responderán los cuestionarios que el sistema enviará en base al área de proceso que está siendo evaluada.
  - **Referencias:** Requisito N° 7.1.5.6 Evaluación de cuestionarios del proceso de la organización.
  - **Curso típico de eventos:**

Jefe de proyecto	Sistema
1. Del menú principal accederá a proyectos y accede al panel listado de proyectos.	2. Se presentará mediante un *gridpanel* una lista de todos los proyectos ordenados descendentemente utilizando fecha de creación, mostrando solamente los proyectos en los que el usuario está involucrado. Solamente para el profesor se mostrarán las opciones modificar, eliminar e iniciar proyecto. Para todos los usuarios se mostrará la opción ver detalles.
3. Una vez localizado su proyecto en el *gridpanel* de listado de proyectos, selecciona la opción ver detalles.	4. Se carga un panel nuevo donde se habilitarán los diferentes módulos que son, generar plan de mejora (Diagramas) y evaluar proceso de la organización (Cuestionarios).
6. Accede a la opción cuestionarios.	7. Se muestran los cuestionarios almacenados en la base de datos que corresponden al área de proceso que está siendo evaluada en el proyecto en cuestión, para la creación de los formularios y el modo de evaluación ver la estructura en el requisito funcional 7.1.5.6. Se debe habilitar la opción de guardar sesión y continuar con las preguntas del cuestionario en caso de una desconexión de internet o cerrar el navegador.

- **Cursos alternativos:**

  Ninguno.

- **Caso de uso:** Baja de evaluación.
  - **Actores:** Profesor.
  - **Propósito:** Limpiar la base de datos y volver a realizar una evaluación en caso de haber introducido datos erróneos.
  - **Visión general:** El profesor mediante este módulo permitirá la eliminación de todos los datos relacionados con la evaluación.
  - **Referencias:** Requisito N° 7.1.5.8 Baja de evaluación.
  - **Curso típico de eventos:**

Jefe de proyecto	Sistema
1. Del menú principal accederá a proyectos y accede al panel listado de proyectos.	2. Se presentará mediante un *gridpanel* una lista de todos los proyectos ordenados descendentemente utilizando fecha de creación, mostrando solamente los proyectos en los que el usuario está involucrado. Solamente para el profesor se mostrarán las opciones modificar, eliminar e iniciar proyecto. Para todos los usuarios se mostrará la opción ver detalles.

3. Una vez localizado su proyecto en el *gridpanel* de listado de proyectos, selecciona la opción ver detalles.	4. Se carga un panel nuevo donde se habilitarán los diferentes módulos que son, generar plan de mejora (Diagramas) y evaluar proceso de la organización (Cuestionarios).
5. Selecciona la opción baja de evaluación.	6. Se pide que confirme la acción a realizar, una vez confirmado se eliminan de la base de datos, todos los resultados referentes a los cuestionarios realizados y de la comparación en los diagramas y se muestra un mensaje de éxito en la eliminación.

- **Cursos alternativos:**

  Paso 6a. En caso que exista un problema con la base de datos para eliminar, mostrar un mensaje de error.

- **Caso de uso:** Mostrar resultados de la evaluación.
  - **Actores:** Profesor, jefe de proyecto, estudiante.
  - **Propósito:** Visualizar los resultados de las evaluaciones.
  - **Visión general:** todos los usuarios podrán acceder a este módulo para visualizar los resultados de la evaluación especificados en el requisito funcional 7.1.5.7.
  - **Referencias:** Requisito N° 7.1.5.7 Baja de evaluación.
  - **Curso típico de eventos:**

Usuario (profesor, jefe de equipo, estudiante)	Sistema
1. Del menú principal accederá a proyectos y accede al panel listado de proyectos.	2. Se presentará mediante un *gridpanel* una lista de todos los proyectos ordenados descendentemente utilizando fecha de creación, mostrando solamente los proyectos en los que el usuario está involucrado. Solamente para el profesor se mostrarán las opciones modificar, eliminar e iniciar proyecto. Para todos los usuarios se mostrará la opción ver detalles.
3. Una vez localizado su proyecto en el *gridpanel* de listado de proyectos, selecciona la opción Resultados de evaluación.	4. Mediante gráficas de barras horizontales se muestran los datos calculados mostrando las prácticas fuertes y débiles mediante las fórmulas que se definieron en el requisito funcional 4.1.5.7. Las gráficas se agruparan de acuerdo al área que fue evaluada, en cada agrupación de mostrarán las gráficas correspondientes al CPi, Desviación y Media calculadas en cada pregunta.

- **Cursos alternativos:**

  4a. En caso de no existir evaluación alguna o que los cuestionaros no se haya contestado aún, mostrar los campos vacíos.

7.2.1.6 Generar plan de mejora

La Figura B.7 muestra el modelo del caso de uso relacionado con la gestión de los documentos que se generarán a partir de la fase de evaluación, y la calendarización de las actividades que se planificarán para la fase de establecer la mejora. Los casos de uso que lo componen se definen como:

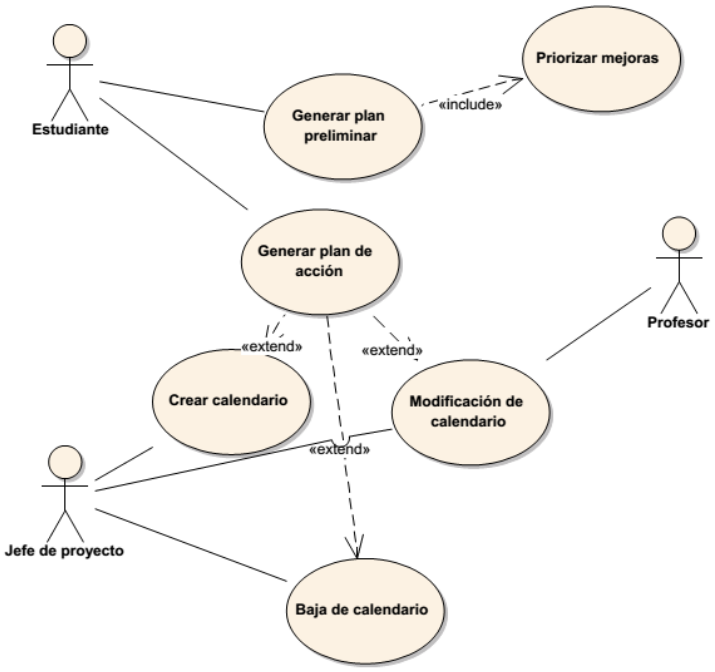


Figura B.7. Modelo del caso de uso generar plan de mejora

- **Caso de uso:** Generar plan preliminar.
  - **Actores:** Estudiante.
  - **Propósito:** Este módulo permitirá mostrar los documentos generados y las prácticas débiles encontradas en la evaluación de una manera concentrada.
  - **Visión general:** El alumno podrá tomar decisiones con respecto a la generación del plan de mejora y podrá subirlo al repositorio para su revisión.
  - **Referencias:** Requisito N° 7.1.6.1 Generar plan preliminar.
  - **Curso típico de eventos:**

Usuario (estudiante)	Sistema
1. Del menú principal accederá a proyectos y accede al panel listado de proyectos.	2. Se presentará mediante un *gridpanel* una lista de todos los proyectos ordenados descendentemente utilizando fecha de creación, mostrando solamente los proyectos

	en los que el usuario está involucrado. Solamente para el profesor se mostrarán las opciones modificar, eliminar e iniciar proyecto. Para todos los usuarios se mostrará la opción ver detalles.
3. Una vez localizado su proyecto en el *gridpanel* de listado de proyectos, selecciona la opción Resultados de evaluación.	4. Se abre el panel de resultados y aparece la opción Generar plan de mejora, donde en una lista y en base a las prácticas débiles que salieron en la evaluación, se mostrarán las actividades de mejora que se deben realizar para corregir esa debilidad.

- **Cursos alternativos:**

  Ninguno.

- **Caso de uso:** Priorizar mejoras.
  - **Actores:** Estudiante (líder de equipo).
  - **Propósito:** Considerando las prácticas débiles encontradas, el alumno dará prioridad a aquellas que estén más alineadas con los objetivos de la organización.
  - **Visión general:** El alumno seleccionará a su criterio las prácticas que tienen mayor prioridad.
  - **Referencias:** Requisito N° 7.1.6.1 Generar plan preliminar y 7.1.6.2 priorizar mejoras.
  - **Curso típico de eventos:**

Usuario (estudiante)	Sistema
1. Del menú principal accederá a proyectos y accede al panel listado de proyectos.	2. Se presentará mediante un *gridpanel* una lista de todos los proyectos ordenados descendentemente utilizando fecha de creación, mostrando solamente los proyectos en los que el usuario está involucrado. Solamente para el profesor se mostrarán las opciones modificar, eliminar e iniciar proyecto. Para todos los usuarios se mostrará la opción ver detalles.
3. Una vez localizado su proyecto en el *gridpanel* de listado de proyectos, selecciona la opción resultados de evaluación.	4. Se carga un panel donde a manera de lista mediante un *gridpanel* se muestran todas las prácticas débiles encontradas. Para cada una de ellas se mostrará un *select box*, donde las opciones serán alta, baja y medio. Se proporciona un botón guardar.
5. Selecciona la prioridad que se le asignará a cada práctica débil y pulsa el botón guardar.	6. Se guardan las prioridades en la base de datos para cada práctica débil encontrada.

- **Cursos alternativos:**

  Paso 6a. En caso de presentarse un problema para guardar en la base de datos, mostrar un mensaje de error.

- **Caso de uso:** Generar plan de acción.
  - **Actores:** Estudiante.
  - **Propósito:** Identificar a las personas que estarán involucradas en el plan de mejora a nivel de responsabilidades y actividades que tendrá que desempeñar cada uno.
  - **Visión general:** Cada práctica se convierte en una actividad para planificar la mejora en base a los calendarios.
  - **Referencias:** Requisito N° 7.1.6.3 Generar plan de acción.
  - **Curso típico de eventos:**

Usuario (estudiante)	Sistema
1. Del menú principal accederá a proyectos y accede al panel listado de proyectos.	2. Se presentará mediante un *gridpanel* una lista de todos los proyectos ordenados descendentemente utilizando fecha de creación, mostrando solamente los proyectos en los que el usuario está involucrado. Solamente para el profesor se mostrarán las opciones modificar, eliminar e iniciar proyecto. Para todos los usuarios se mostrará la opción ver detalles.
3. Una vez localizado su proyecto en el *gridpanel* de listado de proyectos, selecciona la opción resultados de evaluación.	4. Se carga un panel donde a manera de lista mediante un *gridpanel* se muestran todas las prácticas débiles encontradas. Para cada una de ellas se mostrará un *select box*, donde las opciones serán alta, baja y medio. Se proporciona un botón Agendar actividad.
5. Pulsa sobre el botón de generar actividad.	6. Se carga el panel calendario.

  - **Cursos alternativos:**

    Ninguno.

- **Caso de uso:** Crear calendario.
  - **Actores:** Jefe de proyecto.
  - **Propósito:** Calendarizar las actividades a desarrollar e invitar a los participantes.
  - **Visión general:** El jefe de proyecto genera un calendario y en base a fechas y horas, planifica actividades y envía la invitación a todos los implicados en el proyecto.
  - **Referencias:** Requisito N° 7.1.6.4 Crear calendario.
  - **Curso típico de eventos:**

Usuario (jefe de proyecto)	Sistema
1. Del menú principal accederá a proyectos y accede al panel listado de proyectos.	2. Se presentará mediante un *gridpanel* una lista de todos los proyectos ordenados descendentemente utilizando fecha de creación, mostrando solamente los proyectos en los que el usuario está involucrado. Solamente para el profesor se mostrarán las opciones modificar, eliminar e iniciar proyecto. Para todos los usuarios se mostrará la opción ver detalles.
3. Una vez localizado su proyecto en el *gridpanel* de listado de proyectos, selecciona la opción resultados de evaluación.	4. Se carga un panel donde a manera de lista mediante un *gridpanel* se muestran todas las prácticas débiles encontradas. Para cada una de ellas se mostrará un *select box*, donde las opciones serán alta, baja y medio. Se proporciona un botón Agendar actividad.
5. Pulsa sobre el botón de generar actividad.	6. Se carga el panel calendario. Con las opciones típicas para agendar una actividad además de un cuadro para agregar a los usuarios que serán notificados por correo electrónico.
7. Rellena los campos requeridos y guarda los cambios.	8. Se guarda en la base de datos los cambios y se envía el correo electrónico, donde en el cuerpo del correo se mostrará la actividad que se desarrollará, además de la justificación del estudiante en cuanto a la prioridad establecida.

- **Cursos alternativos:**

  Paso 7a. En caso de error mostrar un mensaje.

- **Caso de uso:** Modificación de calendario.
  - **Actores:** Jefe de proyecto, profesor, alumno.
  - **Propósito:** En caso de proponer un cambio, este módulo permitirá la modificación de las fechas propuestas en el calendario inicial.
  - **Visión general:** El profesor o el mismo jefe de equipo pueden proponer cambios a los calendarios creados en un inicio.
  - **Referencias:** Requisito N° 7.1.6.5 Modificación de calendario.
  - **Curso típico de eventos:**

Usuario (jefe de proyecto)	Sistema
1. Del menú principal accederá a proyectos y accede al panel listado de proyectos.	2. Se presentará mediante un *gridpanel* una lista de todos los proyectos ordenados descendentemente utilizando fecha de creación, mostrando solamente los proyectos en los que el usuario está involucrado. Solamente para el profesor se mostrarán las

	opciones modificar, eliminar e iniciar proyecto. Para todos los usuarios se mostrará la opción ver detalles.
3. Una vez localizado su proyecto en el *gridpanel* de listado de proyectos, selecciona la opción resultados de evaluación.	4. Se carga un panel donde a manera de lista mediante un *gridpanel* se muestran todas las prácticas débiles encontradas. Para cada una de ellas se mostrará un *select box*, donde las opciones serán alta, baja y medio. Se proporciona un botón Agendar actividad.
5. Pulsa sobre el botón de generar actividad.	6. Se carga el panel calendario. Con las opciones típicas para agendar una actividad además de un cuadro para agregar a los usuarios que serán notificados por correo electrónico.
7. Rellena los campos requeridos y guarda los cambios.	8. Se guarda en la base de datos los cambios y se envía el correo electrónico, donde en el cuerpo del correo se mostrará la actividad que se desarrollará, además de la justificación del estudiante en cuanto a la prioridad establecida.

- **Cursos alternativos:**

  Ninguno.

- **Caso de uso:** Baja del calendario.
  - **Actores:** Jefe de proyecto, profesor.
  - **Propósito:** Eliminar de un solo paso todas las actividades programadas sobre una práctica específica en un calendario.
  - **Visión general:** En caso de presentarse problemas con las actividades, el jefe de proyecto tiene la posibilidad de eliminar toda la calendarización programada para una práctica.
  - **Referencias:** Requisito N° 7.1.6.6 Baja de calendario.
  - **Curso típico de eventos:**

Usuario (jefe de proyecto)	Sistema
1. Del menú principal accederá a proyectos y accede al panel listado de proyectos.	2. Se presentará mediante un *gridpanel* una lista de todos los proyectos ordenados descendentemente utilizando fecha de creación, mostrando solamente los proyectos en los que el usuario está involucrado. Solamente para el profesor se mostrarán las opciones modificar, eliminar e iniciar proyecto. Para todos los usuarios se mostrará la opción ver detalles.
3. Una vez localizado su proyecto en el *gridpanel* de listado de proyectos, selecciona la opción resultados de	4. Se carga un panel donde a manera de lista mediante un *gridpanel* se muestran todas las prácticas débiles encontradas. Para cada una

evaluación.	de ellas se mostrará un *select box*, donde las opciones serán alta, baja y medio. Se proporciona un botón Agendar actividad.
5. Pulsa sobre el botón de generar actividad.	6. Se carga el panel calendario y se muestran las opciones para modificar las actividades agendadas.
7. Elimina los campos seleccionados y guarda los cambios.	8. Se guarda en la base de datos los cambios.

- **Cursos alternativos:**

  Paso 8a. En caso de presentarse un problema con la base de datos, mostrar un mensaje de error.

7.3 Diseño de bajo nivel

A continuación se presenta el modelo entidad-relación que representa el diseño de la base de datos utilizada para esta propuesta de tesis. Esta base de datos fue diseñada en Visual Paradigm Enterprise Edition y fue exportada directamente a MySQL.

7.3.1 Diseño del modelo entidad-relación

La Figura B.8 presenta el modelo entidad-relación de la base de datos que se utiliza para implementar el sistema de soporte al enfoque colaborativo.

Dadas las limitaciones del tamaño de la página, el modelo entidad-relación se encuentra disponible en formato digital y editable en el CD que acompaña a la tesis.

Nombre	**Valor**
Nombre	Modelo Entidad-Relación
Autor	Carlos Perez
Fecha de Creación	03/02/2014 10:48:10 PM
Última Modificación	28/02/2014 11:39:22 AM

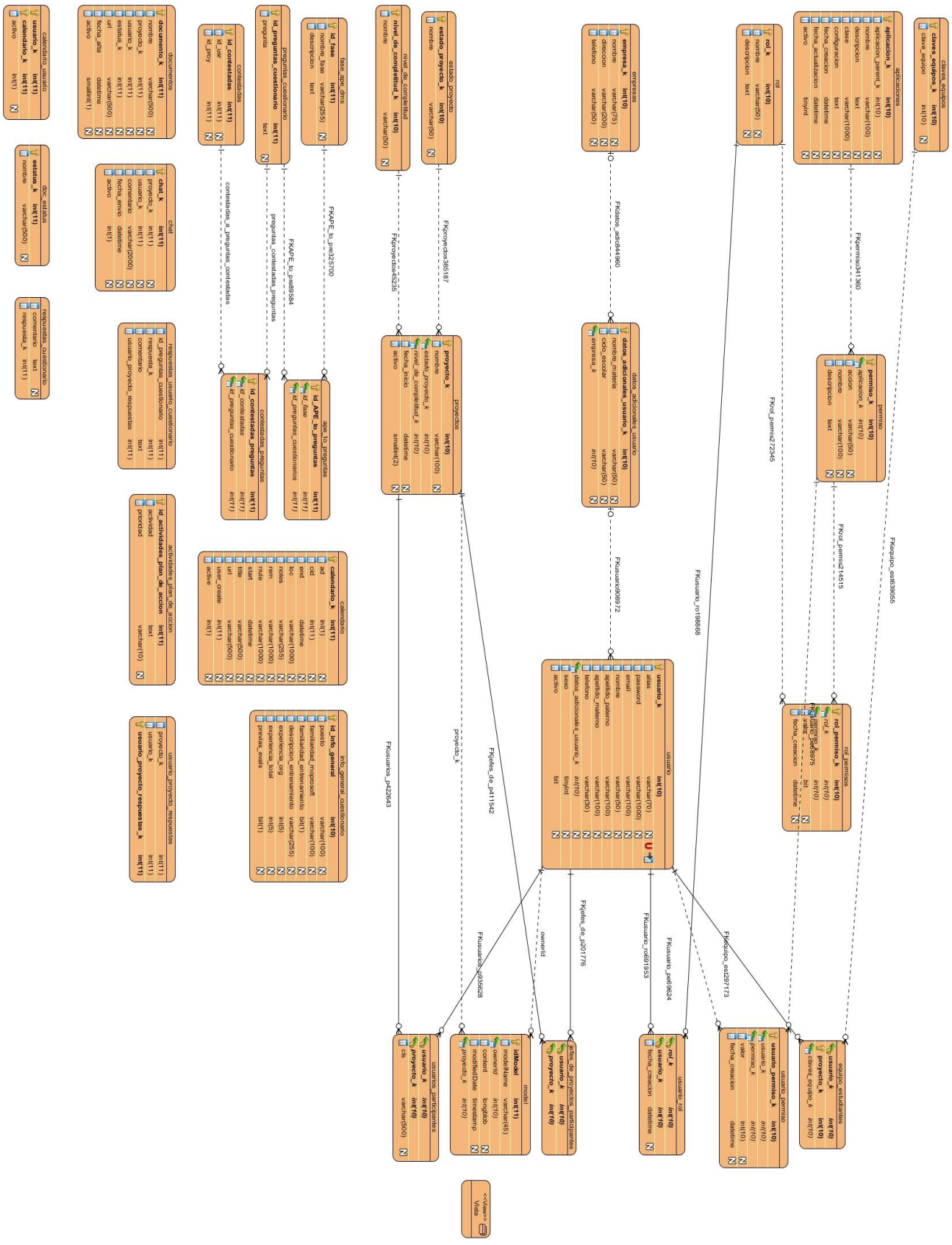


Figura B.8. Modelo entidad-relación diseñado

7.3.1.2 Diseño de tablas y atributos

A continuación se detallan una a una, las diversas tablas que componen el sistema. Para cada una de ellas, se hará una descripción de cada uno de sus campos, así como las claves propias y las ajenas que las componen. Para cada tabla se mostrará un resumen de los atributos con el siguiente formato:

Nombre	DataType	Limitaciones	Anulable	Documentación

Este formato presenta el nombre del atributo, el tipo de dato, si es clave primaria (PK) o si es clave ajena (FK), y si el dato puede ser nulo. La tabla que se muestra a continuación es un resumen de todas las tablas que se detallan en las siguientes secciones:

Nombre
claves_equipos
aplicaciones
permiso
rol_permisos
equipo_estudiantes
usuario_permiso
rol
usuario_rol
usuario
empresas
datos_adicionales_usuario
jefes_de_proyectos_participantes
model
proyectos
estado_proyecto
usuarios_participantes
nivel_de_completitud
fase_ape_dms
calendario
info_general_cuestionario
ape_to_preguntas

preguntas_cuestionario
contestadas_preguntas
contestadas
documentos
chat
respuestas_usuario_cuestionario
actividades_plan_de_accion
usuario_proyecto_respuestas
calendario_usuario
doc_estatus
respuestas_cuestionario

7.3.1.3 Tabla “claves_equipos”

Esta tabla guarda un catálogo de equipos de estudiantes, de esta manera se sabe que esquipos de estudiantes son asignados a un proyecto. La Figura B.9 muestra la tabla “claves_equipos”.

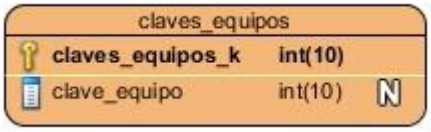


Figura B.9. Tabla “claves_equipos”

Columnas

Nombre	DataType	Limitaciones	Anulable
claves_equipos_k	int(10)	PK	No
clave_equipo	int(10)		Yes

Relaciones

Unnamed Relación	
To	equipo_estudiantes
Autor	Carlos Perez
Fecha de Creación	21-oct-2014 11:46:15
Última Modificación	21-oct-2014 12:05:22
Quality Score	Good
Identifying	false

Subtype	false
To Multiplicity	0..*
From Multiplicity	1
Sync To Association	Yes
Datos Modelo	Physical

7.3.1.4 Tabla “model”

Esta tabla guarda los modelos de procesos utilizados en el caso de uso evaluar proceso de la organización. La Figura B.10 muestra la tabla “model”.

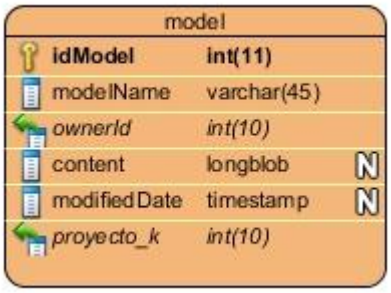


Figura B.10. Tabla “model”

Columnas

Nombre	DataType	Limitaciones	Anulable
idModel	int(11)	PK	No
modelName	varchar(45)		No
ownerId	int(10)	FK (usuario.usuario_k)	No
content	longblob		Yes
modifiedDate	timestamp		Yes
proyecto_k	int(10)	FK (proyectos.proyecto_k)	No

Relaciones

Unnamed Relación	
From	proyectos
Autor	Carlos Perez
Fecha de Creación	21/08/2014 11:46:03 AM
Última Modificación	21/08/2014 12:05:22 PM
Quality Score	Good
Identifying	false

Subtype	false
To Multiplicity	0..*
From Multiplicity	1
Sync To Association	Yes
Datos Modelo	Physical

7.3.1.5 Tabla “fase_ape_dms”

Esta tabla guarda un catálogo de las fases de MoProSoft en donde las preguntas se distribuyen, entre estos fases se encuentra: “Planeación”, “Realización”, “Evaluación y Control”, “Cierre”, “Inicio”, “Requisitos”, “Análisis y Diseño”, “Construcción”, “Integración y pruebas” y “Cierre” y son guardadas en el campo nombre_fase y en el otro campo se guarda la descripción de cada fase para ser presentada en los cuestionarios. La Figura B.11 muestra la tabla “fase_ape_dms”.

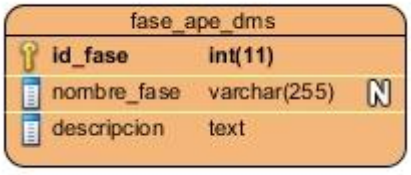


Figura B.11. Tabla “fase_ape_dms”

Columnas

Nombre	DataType	Limitaciones	Anulable
id_fase	int(11)	PK	No
nombre_fase	varchar(255)		Yes
descripcion	text		No

Relaciones

Unnamed Relación	
To	ape_to_preguntas
Autor	Carlos Perez
Fecha de Creación	5/04/2014 12:33:03 AM
Última Modificación	5/04/2014 12:38:44 AM
Quality Score	Good
Identifying	false
Subtype	false
To Multiplicity	0..*
From Multiplicity	1
Sync To Association	Yes
Datos Modelo	Physical

7.3.1.6 Tabla “calendario”

Esta tabla guarda todos los datos que se utilizarán en la ventana de calendarios. La Figura B.12 muestra la tabla “estado_proyecto”.

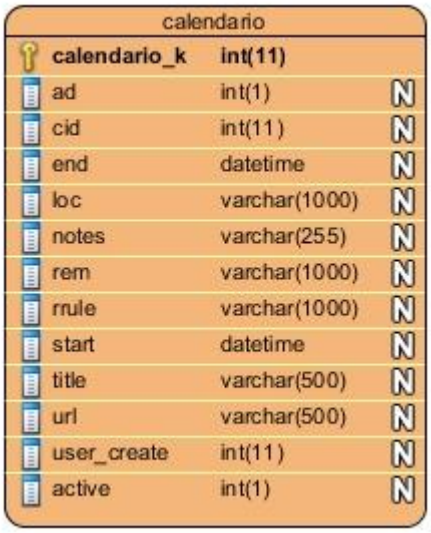


Figura B.12. Tabla “calendario”

Columnas

Nombre	DataType	Limitaciones	Anulable
calendario_k	int(11)	PK	No
ad	int(1)		Yes
cid	int(11)		Yes
end	datetime		Yes
loc	varchar(1000)		Yes
notes	varchar(255)		Yes
rem	varchar(1000)		Yes
rrule	varchar(1000)		Yes
start	datetime		Yes
title	varchar(500)		Yes
url	varchar(500)		Yes
user_create	int(11)		Yes
active	int(1)		Yes

7.3.1.7 Tabla “preguntas_cuestionario”

Esta tabla guarda guarda todas las preguntas que se lanzan en los cuestionarios. La Figura B.13 muestra la tabla “preguntas_cuestionario”.

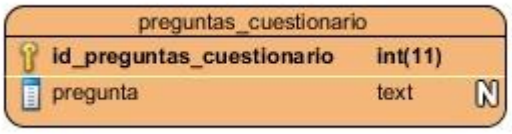


Figura B.13. Tabla "preguntas_cuestionario"

Columnas

Nombre	DataType	Limitaciones	Anulable
id_preguntas_cuestionario	int(11)	PK	No
pregunta	text		Yes

Relaciones

Unnamed Relación	
To	contestadas_preguntas
Autor	Carlos Perez
Fecha de Creación	5/04/2014 12:33:03 AM
Última Modificación	5/04/2014 12:38:44 AM
Quality Score	Good
Identifying	false
Subtype	false
To Multiplicity	0..*
From Multiplicity	1
Sync To Association	Yes
Datos Modelo	Physical

Unnamed Relación	
To	ape_to_preguntas
Autor	Carlos Perez
Fecha de Creación	5/04/2014 12:33:03 AM
Última Modificación	5/04/2014 12:38:44 AM
Quality Score	Good
Identifying	false
Subtype	false
To Multiplicity	0..*
From Multiplicity	1
Sync To Association	Yes

Datos Modelo	Physical

7.3.1.8 Tabla “contestadas_preguntas”

Esta tabla guarda la sesión de las preguntas que ha contestado el usuario para poder posteriormente retomar desde la pregunta donde se quedó en caso de haber una desconcexión o simplemente cerrar sesión en el sistema. La Figura B.14 muestra la tabla “contestadas_preguntas”.

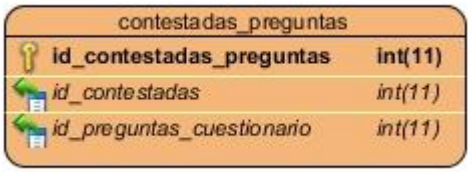


Figura B.14. Tabla “contestadas_preguntas”

Columnas

Nombre	DataType	Limitaciones	Anulable
id_contestadas_preguntas	int(11)	PK	No
id_contestadas	int(11)	FK (contestadas.id_contestadas)	No
id_preguntas_cuestionario	int(11)	FK (preguntas_cuestionario.id_preguntas_cuestionario)	No

Relaciones

Unnamed Relación	
From	preguntas_cuestionario
Autor	Carlos Perez
Fecha de Creación	5/04/2014 12:33:03 AM
Última Modificación	5/04/2014 12:38:44 AM
Quality Score	Good
Identifying	false
Subtype	false
To Multiplicity	0..*
From Multiplicity	1
Sync To Association	Yes
Datos Modelo	Physical
Unnamed Relación	
From	contestadas

Autor	Carlos Perez
Fecha de Creación	5/04/2014 12:33:03 AM
Última Modificación	5/04/2014 12:38:44 AM
Quality Score	Good
Identifying	false
Subtype	false
To Multiplicity	0..*
From Multiplicity	1
Sync To Association	Yes
Datos Modelo	Physical

7.3.1.9 Tabla “contestadas”

Esta tabla guarda el catálogo de preguntas que el usuario ha contestado. La Figura B.15 muestra la tabla “contestadas_preguntas”.

Figura B.15. Tabla “contestadas”

Columnas

Nombre	DataType	Limitaciones	Anulable
id_contestadas	int(11)	PK	No
id_usr	int(11)		Yes
id_proy	int(11)		Yes

Relaciones

Unnamed Relación	
To	contestadas_preguntas
Autor	Carlos Perez
Fecha de Creación	21/10/2014 12:33:03 AM
Última Modificación	21/10/2014 12:38:44 AM
Quality Score	Good
Identifying	false
Subtype	false
To Multiplicity	0..*

From Multiplicity	1
Sync To Association	Yes
Datos Modelo	Physical

7.3.1.10 Tabla “documentos”

Funciona como un índice y guarda la referencia de todos los archivos que se han subido por los usuarios para ser usados por los diferentes proyectos. La Figura B.16 muestra la tabla “documentos”.

Figura B.16. Tabla “documentos”

Columnas

Nombre	DataType	Limitaciones	Anulable
documento_k	int(11)	PK	No
nombre	varchar(500)		Yes
proyecto_k	int(11)		Yes
usuario_k	int(11)		Yes
estatus_k	int(11)		Yes
url	varchar(500)		Yes
fecha_alta	datetime		Yes
activo	smallint(1)		Yes

7.3.1.11 Tabla “chat”

Esta tabla guarda todos los mensajes intercambiados entre los usuarios. La Figura B.17 muestra la tabla “chat”.

Figura B.17. Tabla "chat"

Columnas

Nombre	DataType	Limitaciones	Anulable
chat_k	int(11)	PK	No
proyecto_k	int(11)		Yes
usuario_k	int(11)		Yes
comentario	varchar(2000)		Yes
fecha_envio	datetime		Yes
activo	int(1)		Yes

7.3.1.12 Tabla "estado_proyecto"

Esta tabla guarda un catálogo de estados en los que un proyecto puede entrar, entre estos estados se encuentra: "en pausa", "no iniciado", "iniciado", "cancelado". La Figura B.18 muestra la tabla "estado_proyecto".

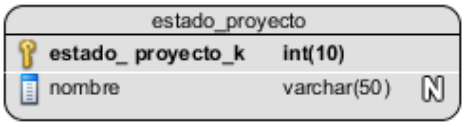


Figura B.18. Tabla "estado_proyecto"

Columnas

Nombre	DataType	Limitaciones	Anulable
estado_ proyecto_k	int(10)	PK	No
nombre	varchar(50)		Yes

Relaciones

Unnamed Relación	
To	proyectos
Autor	Carlos Pérez

Fecha de Creación	5/04/2014 12:33:03 AM
Última Modificación	5/04/2014 12:38:44 AM
Quality Score	Good
Identifying	false
Subtype	false
To Multiplicity	0..*
From Multiplicity	1
Sync To Association	Yes
Datos Modelo	Physical

7.3.1.13 Tabla “proyectos”

Esta tabla contiene los detalles de un proyecto en específico, además a través de esta tabla, se lleva un control de todos los proyectos de mejora que existirán en el sistema. La Figura B.19 muestra la tabla “proyectos”.

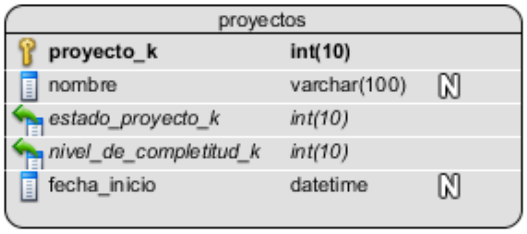


Figura B.19. Tabla “proyectos”

Columnas

Nombre	DataType	Limitaciones	Anulable
proyecto_k	int(10)	PK	No
nombre	varchar(100)		Yes
estado_proyecto_k	int(10)	FK (estado_proyecto.estado_proyecto_k)	No
nivel_de_completitud_k	int(10)	FK (nivel_de_completitud.nivel_de_completitud_k)	No
fecha_inicio	datetime		Yes

Relaciones

Unnamed Relación	
To	usuarios_participantes

Autor	Carlos Pérez
Fecha de Creación	5/04/2014 12:35:00 AM
Última Modificación	5/04/2014 12:38:44 AM
Quality Score	Good
Identifying	false
Subtype	false
To Multiplicity	0..*
From Multiplicity	1
Sync To Association	Yes
Datos Modelo	Physical
Unnamed Relación	
To	jefes_de_proyectos_participantes
Autor	Carlos Pérez
Fecha de Creación	5/04/2014 12:37:55 AM
Última Modificación	5/04/2014 10:08:16 AM
Quality Score	Good
Identifying	false
Subtype	false
To Multiplicity	0..*
From Multiplicity	1
Sync To Association	Yes
Datos Modelo	Physical
Unnamed Relación	
From	nivel_de_completitud
Autor	Carlos Pérez
Fecha de Creación	5/04/2014 12:32:44 AM
Última Modificación	5/04/2014 12:38:44 AM
Quality Score	Good
Identifying	false
Subtype	false
To Multiplicity	0..*
From Multiplicity	1
Sync To Association	Yes
Datos Modelo	Physical
Unnamed Relación	

From	estado_proyecto
Autor	Carlos Pérez
Fecha de Creación	5/04/2014 12:33:03 AM
Última Modificación	5/04/2014 12:38:44 AM
Quality Score	Good
Identifying	false
Subtype	false
To Multiplicity	0..*
From Multiplicity	1
Sync To Association	Yes
Datos Modelo	Physical

7.3.1.14 Tabla “nivel_de_completitud”

Esta tabla guarda un catálogo de los distintos niveles de completitud que puede tener un proyecto, para este caso el nivel de completitud se utilizará según las mismas fases del modelo ideal. La Figura B.20 muestra la tabla “nivel_de_completitud”.

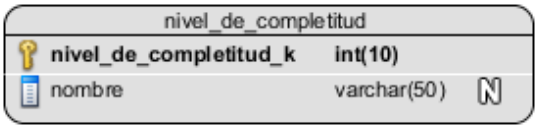


Figura B.20. Tabla “nivel_de_completitud”

Columnas

Nombre	DataType	Limitaciones	Anulable
nivel_de_completitud_k	int(10)	PK	No
nombre	varchar(50)		Yes

Relaciones

Unnamed Relación	
To	proyectos
Autor	Carlos Pérez
Fecha de Creación	5/04/2014 12:32:44 AM
Última Modificación	5/04/2014 12:38:44 AM
Quality Score	Good
Identifying	false
Subtype	false
To Multiplicity	0..*

From Multiplicity	1
Sync To Association	Yes
Datos Modelo	Physical

7.3.1.15 Tabla “usuarios_participantes”

Esta tabla ayuda a tener la relación de los usuarios Alumnos que estarán involucrados en uno o varios proyectos. La Figura B.21 muestra la tabla “usuarios_participantes”.

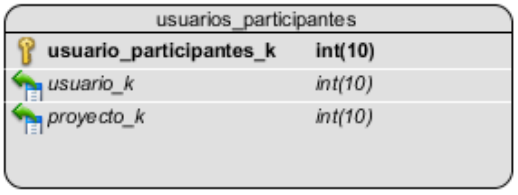


Figura B.21. Tabla “usuarios_participantes”

Columnas

Nombre	DataType	Limitaciones	Anulable
usuario_participantes_k	int(10)	PK	No
usuario_k	int(10)	FK (usuario.usuario_k)	No
proyecto_k	int(10)	FK (proyectos.proyecto_k)	No

Relaciones

Unnamed Relación	
From	usuario
Autor	Carlos Pérez
Fecha de Creación	5/04/2014 12:34:14 AM
Última Modificación	5/04/2014 12:38:44 AM
Quality Score	Good
Identifying	false
Subtype	false
To Multiplicity	0..*
From Multiplicity	1
Sync To Association	Yes
Datos Modelo	Physical

Unnamed Relación	
From	proyectos
Autor	Carlos Pérez
Fecha de Creación	5/04/2014 12:35:00 AM
Última Modificación	5/04/2014 12:38:44 AM
Quality Score	Good
Identifying	false
Subtype	false
To Multiplicity	0..*
From Multiplicity	1
Sync To Association	Yes
Datos Modelo	Physical

7.3.1.16 Tabla “equipo_estudiantes”

Tabla que guarda la relación de los equipos de estudiantes que se crearán en los proyectos para realizar las actividades de mejora. La Figura B.22 muestra la tabla “equipo_estudiantes”.

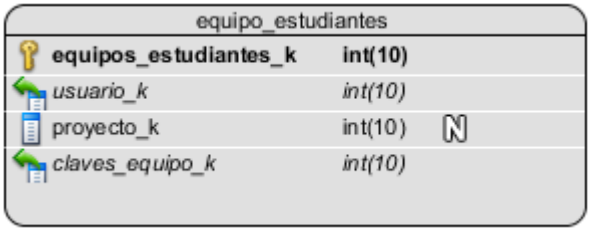


Figura B.22. Tabla “equipos_estudiantes”

Columnas

Nombre	**DataType**	**Limitaciones**	**Anulable**
equipos_estudiantes_k	int(10)	PK	No
usuario_k	int(10)	FK (usuario.usuario_k)	No
proyecto_k	int(10)		Yes
claves_equipo_k	int(10)	FK (claves_equipos.claves_equipos_k)	No

Relaciones

Unnamed Relación	
From	usuario
Autor	Carlos Pérez
Fecha de Creación	5/04/2014 12:19:05 AM
Última Modificación	5/04/2014 12:28:13 AM
Quality Score	Good
Identifying	false
Subtype	false
To Multiplicity	0..*
From Multiplicity	1
Sync To Association	Yes
Datos Modelo	Physical
Unnamed Relación	
From	claves_equipos
Autor	Carlos Pérez
Fecha de Creación	5/04/2014 03:12:00 PM
Última Modificación	5/04/2014 03:23:11 PM
Quality Score	Good
Identifying	false
Subtype	false
To Multiplicity	0..*
From Multiplicity	1
Sync To Association	Yes
Datos Modelo	Physical

7.3.1.17 Tabla “claves_equipos”

Esta tabla es utilizada como un control interno del sistema, para tener un identificador único para cada equipo de alumnos creado. La Figura B.23 muestra la tabla “claves_equipos”.

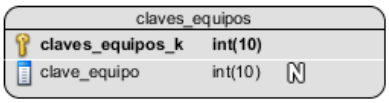


Figura B.23. Tabla “claves_equipos”

Columnas

Nombre	**DataType**	**Limitaciones**	**Anulable**

claves_equipos_k	int(10)	PK	No
clave_equipo	int(10)		Yes

Relaciones

Unnamed Relación	
To	equipo_estudiantes
Autor	Carlos Pérez
Fecha de Creación	5/04/2014 03:12:00 PM
Última Modificación	5/04/2014 03:23:11 PM
Quality Score	Good
Identifying	false
Subtype	false
To Multiplicity	0..*
From Multiplicity	1
Sync To Association	Yes
Datos Modelo	Physical

7.3.1.18 Tabla "jefes_de_proyectos_participantes"

Descripción: Esta tabla contiene una relación entre los usuarios Jefe de Proyecto y los proyectos en los que está participando. La Figura B.24 muestra la tabla "jefes_de_proyectos_participantes".

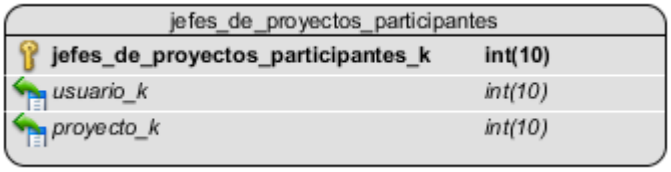


Figura B.24. Tabla "jefes_de_proyectos_participantes"

Columnas

Nombre	**DataType**	**Limitaciones**	**Anulable**
jefes_de_proyectos_participantes_k	int(10)	PK	No
usuario_k	int(10)	FK (usuario.usuario_k)	No
proyecto_k	int(10)	FK (proyectos.proyecto_k)	No

Relaciones

Unnamed Relación	
From	proyectos
Autor	Carlos Pérez
Fecha de Creación	5/04/2014 12:37:55 AM
Última Modificación	5/04/2014 10:08:16 AM
Quality Score	Good
Identifying	false
Subtype	false
To Multiplicity	0..*
From Multiplicity	1
Sync To Association	Yes
Datos Modelo	Physical
Unnamed Relación	
From	usuario
Autor	Carlos Pérez
Fecha de Creación	5/04/2014 12:39:33 AM
Última Modificación	5/04/2014 10:08:16 AM
Quality Score	Good
Identifying	false
Subtype	false
To Multiplicity	0..*
From Multiplicity	1
Sync To Association	Yes
Datos Modelo	Physical

7.3.1.19 Tabla “empresas”

Tabla que guarda el nombre de la empresa a la que pertenecen los jefes de proyecto. La Figura B.25 muestra la tabla “empresas”.

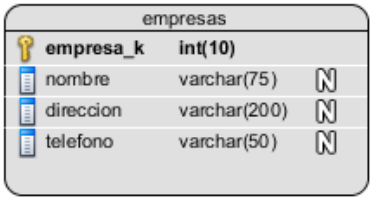


Figura B.25. Tabla “empresas”

Columnas

Nombre	DataType	Limitaciones	Anulable
empresa_k	int(10)	PK	No
nombre	varchar(75)		Yes
direccion	varchar(200)		Yes
telefono	varchar(50)		Yes

Relaciones

Unnamed Relación	
To	datos_adicionales_usuario
Autor	Carlos Pérez
Fecha de Creación	5/04/2014 12:02:37 AM
Última Modificación	5/04/2014 12:07:13 AM
Quality Score	Good
Identifying	false
Subtype	false
To Multiplicity	0..*
From Multiplicity	1
Sync To Association	Yes
Datos Modelo	Physical

7.3.1.20 Tabla “datos_adicionales_usuario”

Descripción: Esta tabla se creó a parte para guardar exclusivamente los datos que corresponden a un alumno. La Figura B.26 muestra la tabla “datos_adicionales_usuario”.

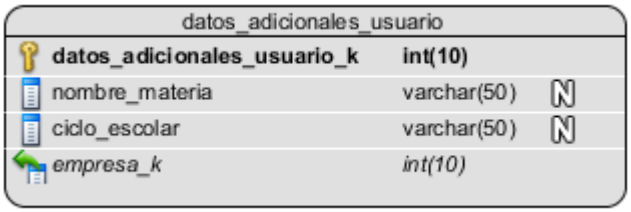


Figura B.26. Tabla “datos_adicionales_usuario”

Columnas

Nombre	DataType	Limitaciones	Anulable
datos_adicionales_usuario_k	int(10)	PK	No
nombre_materia	varchar(50)		Yes

ciclo_escolar	varchar(50)		Yes
empresa_k	int(10)	FK (empresas.empresa_k)	No

Relaciones

Unnamed Relación	
To	usuario
Autor	Carlos Pérez
Fecha de Creación	5/04/2014 12:10:22 AM
Última Modificación	5/04/2014 12:17:43 AM
Quality Score	Good
Identifying	false
Subtype	false
To Multiplicity	1
From Multiplicity	1
Sync To Association	Yes
Datos Modelo	Physical
Unnamed Relación	
From	empresas
Autor	Carlos Pérez
Fecha de Creación	5/04/2014 12:02:37 AM
Última Modificación	5/04/2014 12:07:13 AM
Quality Score	Good
Identifying	false
Subtype	false
To Multiplicity	0..*
From Multiplicity	1
Sync To Association	Yes
Datos Modelo	Physical

7.3.1.21 Tabla “usuario”

Tabla donde se guardan todos los datos principales relacionados a un usuario del sistema. La tabla B.27 muestra la tabla “usuario”.

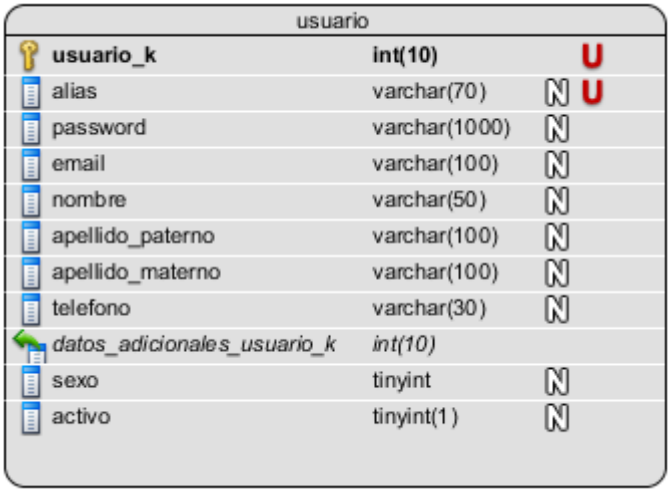


Figura B.27. Tabla "usuario"

Columnas

Nombre	DataType	Limitaciones	Anulable
usuario_k	int(10)	PKUnique	No
alias	varchar(70)	Unique	Yes
password	varchar(1000)		Yes
email	varchar(100)		Yes
nombre	varchar(50)		Yes
apellido_paterno	varchar(100)		Yes
apellido_materno	varchar(100)		Yes
telefono	varchar(30)		Yes
datos_adicionales_usuario_k	int(10)	FK (datos_adicionales_usuario.datos_adicionales_usuario_k)	No
sexo	tinyint		Yes
activo	tinyint(1)		Yes

Relaciones

Unnamed Relación	
To	usuario_rol
Autor	Carlos Pérez
Fecha de Creación	3/04/2014 11:10:52 PM
Última Modificación	3/04/2014 11:19:18 PM
Quality Score	Good

Identifying	false
Subtype	false
To Multiplicity	1
From Multiplicity	1
Sync To Association	Yes
Datos Modelo	Physical
Unnamed Relación	
To	usuario_permiso
Autor	Carlos Pérez
Fecha de Creación	4/04/2014 09:38:28 AM
Última Modificación	4/04/2014 11:46:13 PM
Quality Score	Good
Identifying	false
Subtype	false
To Multiplicity	0..*
From Multiplicity	1
Sync To Association	Yes
Datos Modelo	Physical
Unnamed Relación	
To	equipo_estudiantes
Autor	Carlos Pérez
Fecha de Creación	5/04/2014 12:19:05 AM
Última Modificación	5/04/2014 12:28:13 AM
Quality Score	Good
Identifying	false
Subtype	false
To Multiplicity	0..*
From Multiplicity	1
Sync To Association	Yes
Datos Modelo	Physical
Unnamed Relación	
To	usuarios_participantes
Autor	Carlos Pérez
Fecha de Creación	5/04/2014 12:34:14 AM

Última Modificación	5/04/2014 12:38:44 AM
Quality Score	Good
Identifying	false
Subtype	false
To Multiplicity	0..*
From Multiplicity	1
Sync To Association	Yes
Datos Modelo	Physical
Unnamed Relación	
To	jefes_de_proyectos_participantes
Autor	Carlos Pérez
Fecha de Creación	5/04/2014 12:39:33 AM
Última Modificación	5/04/2014 10:08:16 AM
Quality Score	Good
Identifying	false
Subtype	false
To Multiplicity	0..*
From Multiplicity	1
Sync To Association	Yes
Datos Modelo	Physical
Unnamed Relación	
From	datos_adicionales_usuario
Autor	Carlos Pérez
Fecha de Creación	5/04/2014 12:10:22 AM
Última Modificación	5/04/2014 12:17:43 AM
Quality Score	Good
Identifying	false
Subtype	false
To Multiplicity	1
From Multiplicity	1
Sync To Association	Yes
Datos Modelo	Physical

7.3.1.22 Tabla “usuario_rol”

La tabla es usada como un catálogo de los diferentes roles y permisos que tiene un usuario para ver los diferentes módulos del sistema. La Figura B.28 muestra la tabla “usuario_rol”.

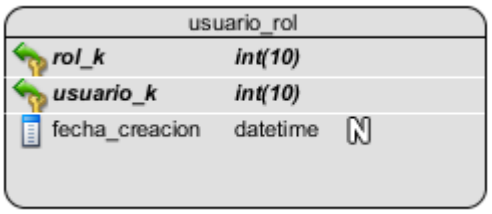


Figura B.28. Tabla “usuario_rol”

Columnas

Nombre	DataType	Limitaciones	Anulable
rol_k	int(10)	PK/FK (rol.rol_k)	No
usuario_k	int(10)	PK/FK (usuario.usuario_k)	No
fecha_creacion	datetime		Yes

Relaciones

Unnamed Relación	
From	usuario
Autor	Carlos Pérez
Fecha de Creación	3/04/2014 11:10:52 PM
Última Modificación	3/04/2014 11:19:18 PM
Quality Score	Good
Identifying	false
Subtype	false
To Multiplicity	1
From Multiplicity	1
Sync To Association	Yes
Datos Modelo	Physical
Unnamed Relación	
From	rol
Autor	Carlos Pérez
Fecha de Creación	3/04/2014 11:14:14 PM
Última Modificación	3/04/2014 11:19:18 PM

Quality Score	Good
Identifying	false
Subtype	false
To Multiplicity	0..*
From Multiplicity	1
Sync To Association	Yes
Datos Modelo	Physical

7.3.1.23 Tabla “rol”

La tabla contiene un catálogo con los posibles roles que pueden existir en el sistema, se pensó de esta manera por si existiese una expansión futura del sistema y se quisieran definir nuevos roles. La Figura B.29 representa la tabla “rol”.

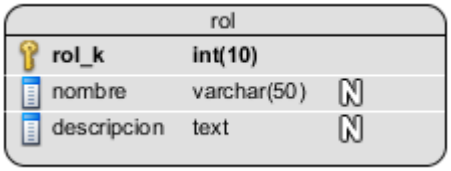


Figura B.29. Tabla “rol”

Columnas

Nombre	DataType	Limitaciones	Anulable
rol_k	int(10)	PK	No
nombre	varchar(50)		Yes
descripcion	text		Yes

Relaciones

Unnamed Relación	
To	usuario_rol
Autor	Carlos Pérez
Fecha de Creación	3/04/2014 11:14:14 PM
Última Modificación	3/04/2014 11:19:18 PM
Quality Score	Good
Identifying	false
Subtype	false
To Multiplicity	0..*
From Multiplicity	1
Sync To Association	Yes
Datos Modelo	Physical

Unnamed Relación	
To	rol_permisos
Autor	Carlos Pérez
Fecha de Creación	3/04/2014 11:23:34 PM
Última Modificación	3/04/2014 11:29:48 PM
Quality Score	Good
Identifying	false
Subtype	false
To Multiplicity	0..*
From Multiplicity	1
Sync To Association	Yes
Datos Modelo	Physical

7.3.1.24 Tabla “rol_permisos”

Tabla destinada para un control interno de la visualización de las ventanas que se mostrarán y sus diferentes atributos que son la posibilidad de agregar datos nuevos, editarlos y eliminarlos. La Figura B.30 muestra la tabla “rol_permisos”.

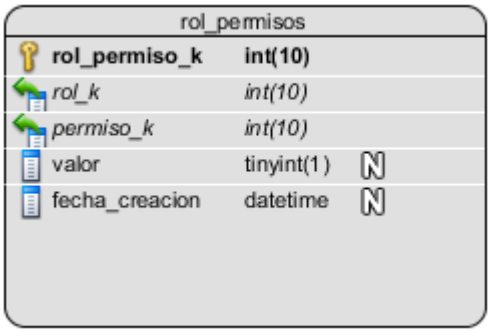


Figura B.30. Tabla “rol_permisos”

Columnas

Nombre	DataType	Limitaciones	Anulable
rol_permiso_k	int(10)	PK	No
rol_k	int(10)	FK (rol.rol_k)	No
permiso_k	int(10)	FK (permiso.permiso_k)	No
valor	tinyint(1)		Yes
fecha_creacion	datetime		Yes

Relaciones

Unnamed Relación	
From	permiso
Autor	Carlos Pérez
Fecha de Creación	3/04/2014 11:22:52 PM
Última Modificación	3/04/2014 11:29:48 PM
Quality Score	Good
Identifying	false
Subtype	false
To Multiplicity	0..*
From Multiplicity	1
Sync To Association	Yes
Datos Modelo	Physical
Unnamed Relación	
From	rol
Autor	Carlos Pérez
Fecha de Creación	3/04/2014 11:23:34 PM
Última Modificación	3/04/2014 11:29:48 PM
Quality Score	Good
Identifying	false
Subtype	false
To Multiplicity	0..*
From Multiplicity	1
Sync To Association	Yes
Datos Modelo	Physical

7.3.1.25 Tabla “usuario_permiso”

La tabla es usada para control interno sobre los usuarios y las distintas acciones que éstos pueden ejecutar en los diferentes módulos del sistema. La Figura B.31 muestra la tabla “usuario_permiso”.

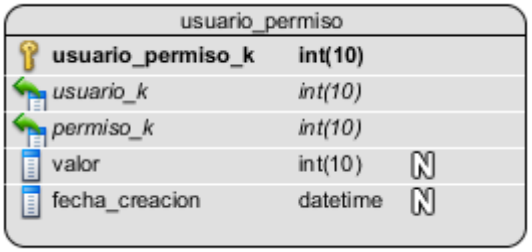


Figura B.31. Tabla "usuario_permiso"

Columnas

Nombre	DataType	Limitaciones	Anulable
usuario_permiso_k	int(10)	PK	No
usuario_k	int(10)	FK (usuario.usuario_k)	No
permiso_k	int(10)	FK (permiso.permiso_k)	No
valor	int(10)		Yes
fecha_creacion	datetime		Yes

Relaciones

Unnamed Relación	
From	usuario
Autor	Carlos Pérez
Fecha de Creación	4/04/2014 09:38:28 AM
Última Modificación	4/04/2014 11:46:13 PM
Quality Score	Good
Identifying	False
Subtype	False
To Multiplicity	0..*
From Multiplicity	1
Sync To Association	Yes
Datos Modelo	Physical
Unnamed Relación	
From	permiso
Autor	Carlos Pérez
Fecha de Creación	4/04/2014 09:40:40 AM

Última Modificación	4/04/2014 11:46:13 PM
Quality Score	Good
Identifying	false
Subtype	false
To Multiplicity	0..*
From Multiplicity	1
Sync To Association	Yes
Datos Modelo	Physical

7.3.1.26 Tabla "aplicaciones"

Tabla para control interno que contiene las mismas ventanas implementadas en el sistema, además contiene los datos del módulo y a través de los datos de identificación de éstas se cargan los módulos existentes en el código fuente. La Figura B.32 muestra la tabla "aplicaciones".

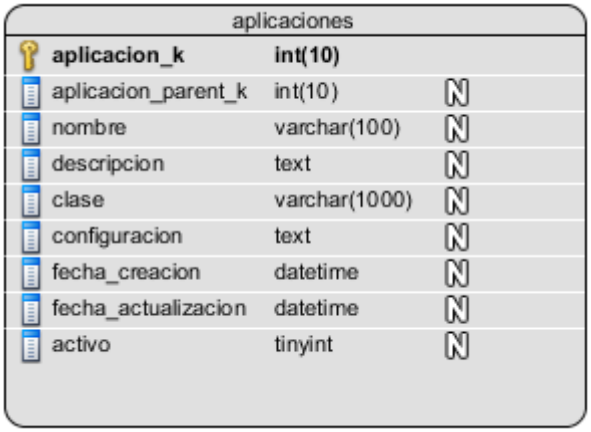


Figura B.32. Tabla "aplicaciones"

Columnas

Nombre	DataType	Limitaciones	Anulable
aplicacion_k	int(10)	PK	No
aplicacion_parent_k	int(10)		Yes
nombre	varchar(100)		Yes
descripcion	text		Yes
clase	varchar(1000)		Yes
configuracion	text		Yes
fecha_creacion	datetime		Yes
fecha_actualizacion	datetime		Yes
activo	tinyint		Yes

Relaciones

Unnamed Relación	
To	permiso
Autor	Carlos Pérez
Fecha de Creación	4/04/2014 09:28:34 AM
Última Modificación	4/04/2014 09:40:06 AM
Quality Score	Good
Identifying	false
Subtype	false
To Multiplicity	0..*
From Multiplicity	1
Sync To Association	Yes
Datos Modelo	Physical

7.3.1.27 Tabla “permiso”

Catálogo de las diferentes acciones que se pueden realizar sobre un módulo en particular. La Figura B.33 muestra la tabla “permiso”.

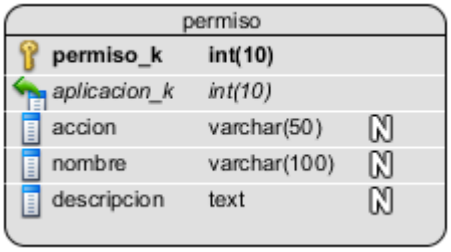


Figura B.33. Tabla “permiso”

Columnas

Nombre	DataType	Limitaciones	Anulable
permiso_k	int(10)	PK	No
aplicacion_k	int(10)	FK (aplicaciones.aplicacion_k)	No
accion	varchar(50)		Yes
nombre	varchar(100)		Yes
descripcion	text		Yes

Relaciones

Unnamed Relación	
To	rol_permisos
Autor	Carlos Pérez
Fecha de Creación	3/04/2014 11:22:52 PM
Última Modificación	3/04/2014 11:29:48 PM
Quality Score	Good
Identifying	false
Subtype	false
To Multiplicity	0..*
From Multiplicity	1
Sync To Association	Yes
Datos Modelo	Physical
Unnamed Relación	
To	usuario_permiso
Autor	Carlos Pérez
Fecha de Creación	4/04/2014 09:40:40 AM
Última Modificación	4/04/2014 11:46:13 PM
Quality Score	Good
Identifying	false
Subtype	false
To Multiplicity	0..*
From Multiplicity	1
Sync To Association	Yes
Datos Modelo	Physical
Unnamed Relación	
From	aplicaciones
Autor	Carlos Pérez
Fecha de Creación	4/04/2014 09:28:34 AM
Última Modificación	4/04/2014 09:40:06 AM
Quality Score	Good
Identifying	false
Subtype	false
To Multiplicity	0..*

From Multiplicity	1
Sync To Association	Yes
Datos Modelo	Physical

8. Anexo C.- Manual de usuario

La siguiente documentación pretende auxiliar a los lectores a comprender el funcionamiento básico de la herramienta computacional que soporta el curso colaborativo propuesto en la tesis.

8.1. Inicio de sesión al sistema

Cuando el sistema sea instalado la primera vez, por defecto será creada una cuenta de usuario tipo administrador con los siguientes datos:

Usuario: admin

Password: 0dp28oe8

Se accederá al sistema a través de la dirección http://54.187.32.93/hesespi, donde aparecerá una ventana como la mostrada en la Figura C.1. En este punto será necesario introducir el nombre usuario y contraseña proporcionados. Al hacer clic en "Entrar" o pulsar la tecla [enter] se realizará la validación del usuario correspondiente y se concederá acceso al sistema. En el caso de proporcionar datos incorrectos se mostrará un mensaje de error.

Figura C.1. Inicio de sesión de usuario

8.2. Gestión de usuarios

Solamente un usuario tipo administrador podrá acceder a esta función. Para acceder a la ventana mostrada en la Figura C.2, será necesario hace clic sobre el botón del menú Inicio-> Administración -> Usuarios.

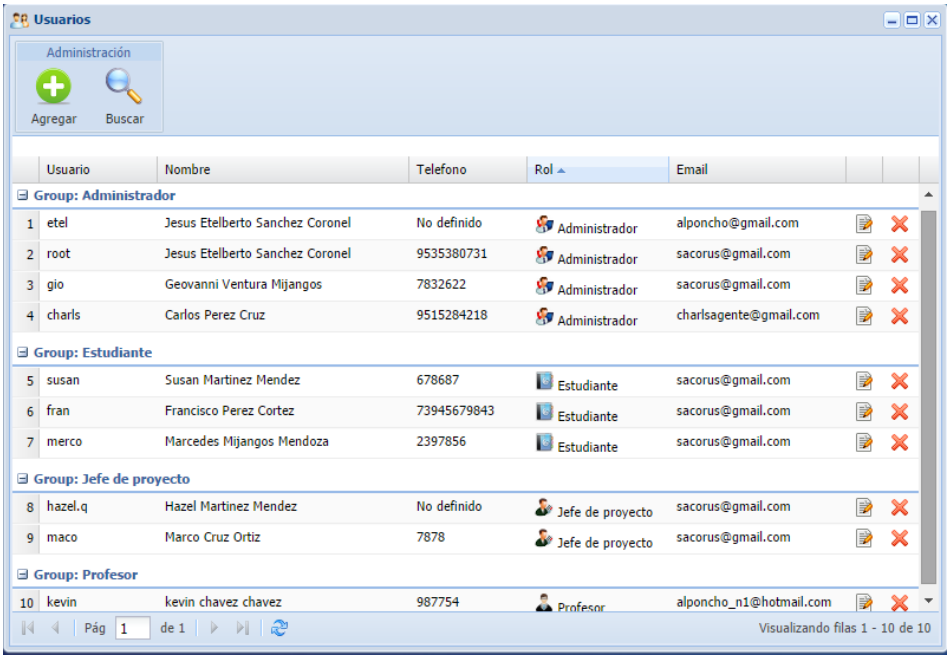


Figura C.2. Panel de administración "usuarios".

La gestión a través de esta ventana permite las siguientes acciones:

- Eliminar un usuario: por cada usuario listado se muestra una "X", al hacer clic sobre ésta saldrá un cuadro requiriendo la confirmación para eliminar el usuario.
- Editar un usuario: por cada usuario listado se mostrará un botón "Editar", al presionarlo se mostrará un cuadro de dialogo donde se mostrarán los campos correspondientes a la información del usuario y en espera de edición.
- Agregar un usuario nuevo: se deberá hacer clic sobre el icono "Agregar" que aparece en la parte superior izquierda de la ventana y se mostrará una nueva ventana donde se solicitarán los datos requeridos por el sistema.

8.3. Gestión de proyectos

Para comenzar a crear un proyecto desde cero, se debe acceder al panel de proyectos en el menú Inicio-> Proyectos y se abrirá una ventana como la mostrada en la Figura C.3.

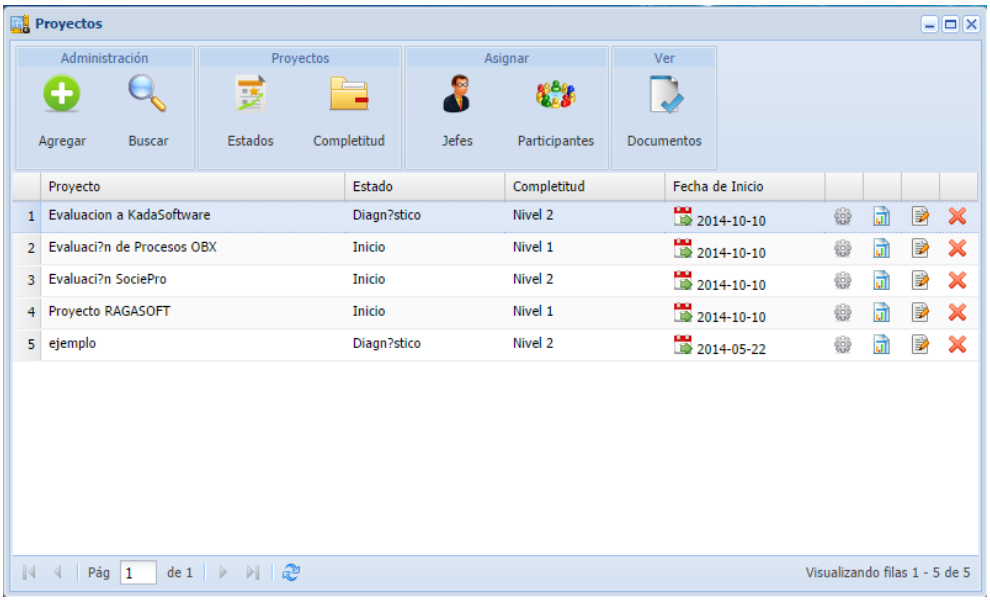


Figura C.3. Ventana "Proyectos".

Para crear un proyecto nuevo se debe hacer clic en el botón "Agregar" y se deberá introducir un nombre para el proyecto y la fase (estado) en la que se piensa arrancar. En el caso de intoducir un estado por default se mostrará una lista con los estados correspondientes al modelo IDEAL. La completitud en este caso será una etiqueta que representa el avance del proyecto y la fecha en la que inicia el mismo. Una vez que esta información haya sido guardada, el nuevo proyecto aparecerá en la lista de proyectos y se deberá asignar a los jefes de proyectos que participarán en él a través del botón "Jefes". Así, se mostrará una lista con los jefes de proyectos (incorporados a la red de colaboración) y los botones correspondientes para asignarlos a la lista de jefes.

De igual manera, pulsando el botón "Participantes" el profesor creará los grupos de estudiantes que participarán en ese proyecto.

Por último, en el botón "Documentos" aparecerán las listas de los archivos que fueron aprobados y están relacionados al proyecto.

8.4. Alta y aprobación de documentos

Para subir un documento, cualquier usuario puede acceder a través del menú Inicio->Alta de Documentos, donde aparecerá una ventana similar a la mostrada en la Figura C.4.

Al hacer clic en "Agregar", el sistema mostrará una ventana emergente que pedirá se seleccione el archivo a subir, un nombre para identificar al documento, y que se seleccione el proyecto al que será agregado, al "Guardar" el documento se subirá al servidor y estará listo para ser revisado y/o aprobado por un profesor.

Para aprobar un documento el usuario profesor deberá acceder al panel de aprobación de documentos haciendo clic en Inicio->Aprobar Documentos y seguir las instrucciones de la ventana mostrada en la Figura C.5.

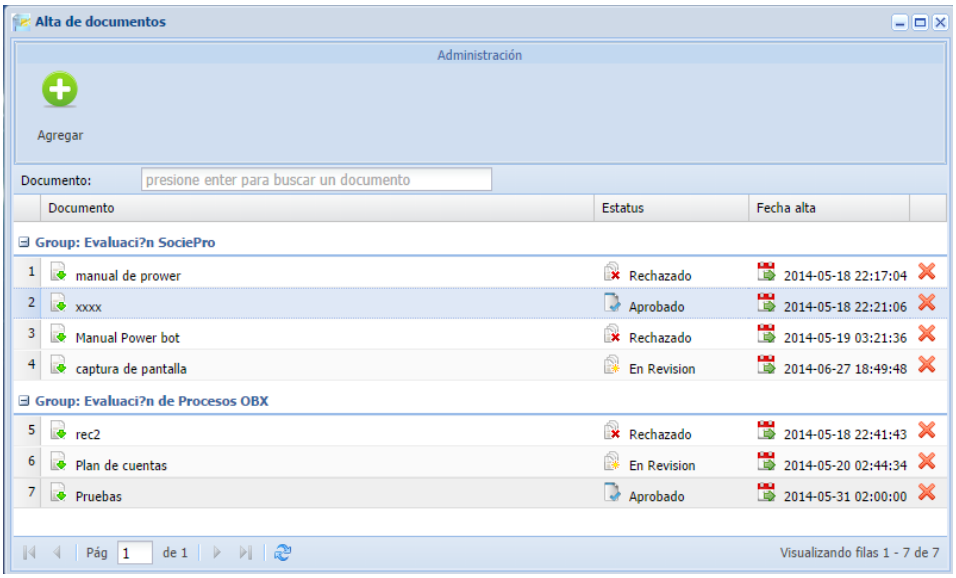


Figura C.4. Ventana "Alta de documentos".

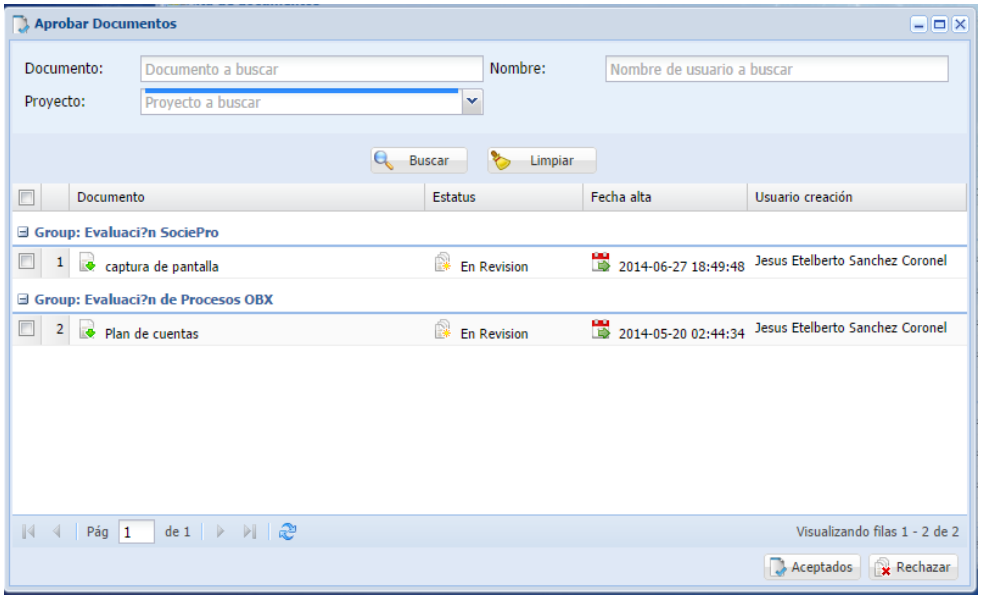


Figura C.5. Ventana "Aprobar documentos".

Adicionalmente, se cuenta con herramientas de búsqueda para ubicar cualquier documento a través del nombre que se le dio, del proyecto al que se asignó o del nombre del usuario que lo subió. Un usuario podrá descargar el documento que desee al hacer clic sobre éste directamente en la lista visualizada.

Para aprobar un documento, de la lista de documentos, se debe seleccionar su *checkbox* y a continuación hacer clic en el botón llamado "Aceptados" situado en la parte inferior derecha de la ventana. En caso contrario el usuario deberá pulsar el botón "Rechazar" para indicar que no es un documento aprobado para el proyecto.

8.5. Uso del chat

Una vez que los usuarios hayan sido dados de alta en los proyectos, éstos podrán acceder al módulo de chat ubicado en el menú Inicio->Chat donde verán una ventana principal similar a la mostrada en la Figura C.6. Es importante mencionar que al seleccionar un proyecto, los integrantes de equipo podrán intercambiar mensajes de texto con todos los participantes relacionados a ese proyecto.

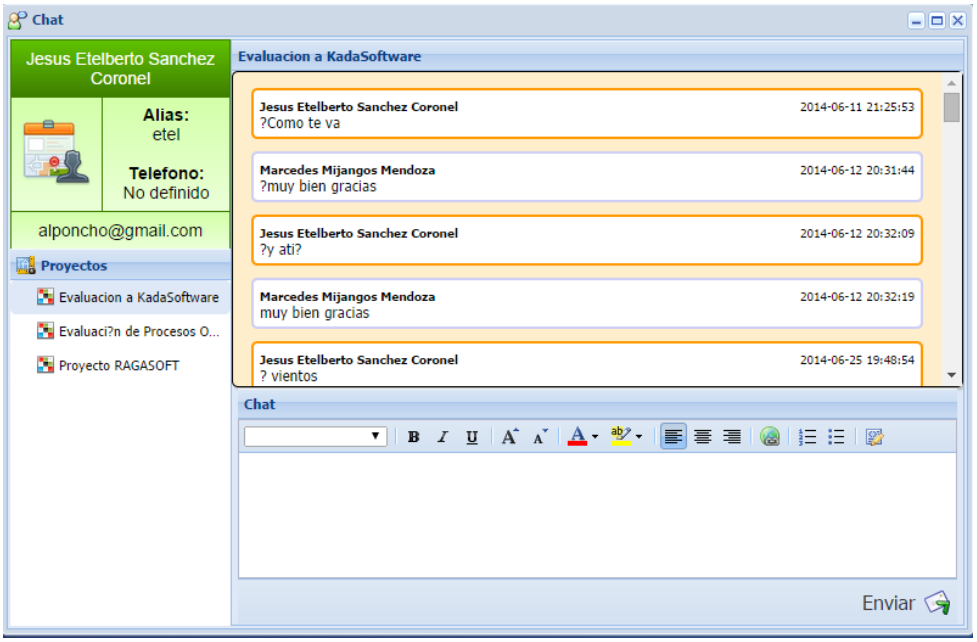


Figura C.6. Ventana "Chat".

8.6. Evaluación de procesos.

8.6.1. Uso de diagramas.

Dentro de la ventana de proyectos, en la lista desplegada, por cada proyecto se mostrará un botón con forma de engrane, al hacer clic sobre él se visualizará por default en la pestaña Diagramas una ventana similar a la mostrada en la Figura C.7.

La lista desplegada permite ver los nombres de los diagramas que han sido creados. Si el usuario pretende crear un diagrama desde cero, deberá pulsar sobre el botón "Nuevo" ubicado en la parte superior izquierda de la Figura. El sistema le pedirá que escriba el nombre para este diagrama antes de comenzar a editarlo, y hasta hacer clic sobre "Guardar" es que su nombre aparecerá en la lista. Para editar un diagrama, el usuario debe primero seleccionarlo de la lista y pulsar sobre el botón "Editar diagrama", lo que hará que se muestre una nueva ventana como la mostrada en la Figura C.8.

Para comenzar con la creación de un diagrama, el usuario debe acceder al panel Propiedades del Diagrama (en caso de no estar visible se deberá pulsar el icono ">>" mostrado en rojo en la parte derecha de la Figura C.8) y ajustar ciertos atributos. El usuario deberá pulsar sobre la opción Categoría para visualizar una lista con las áreas de proceso que MoProsoft contiene, en este caso se debe seleccionar el área de proceso sobre la cual se realizará la evaluación. En la opción Documentos, los usuarios deberán seleccionar y asociar al diagrama los archivos que fueron subidos

previamente y que servirán como evidencia objetiva que soporte la creación del diagrama y sus elementos.

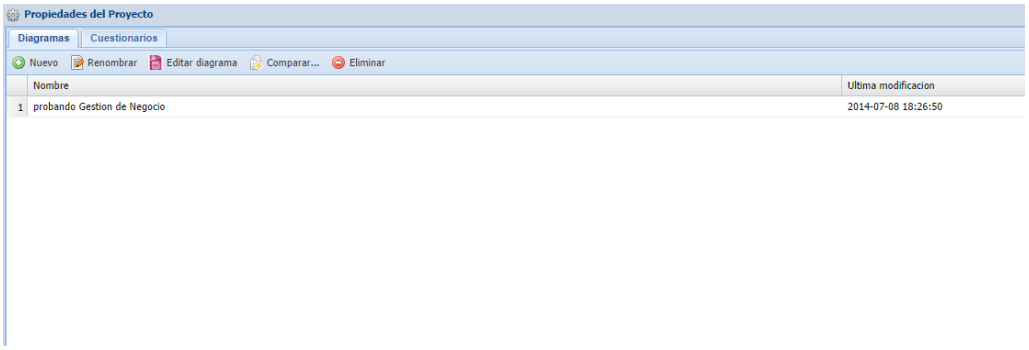


Figura C.7 Ventana "Propiedades de Proyecto".

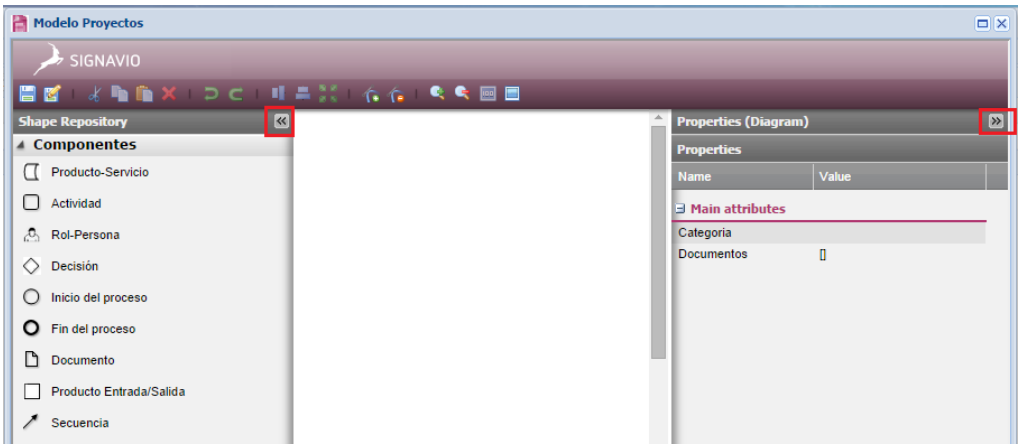


Figura C.8 Ventana "Modelo Proyectos".

Una vez que los requisitos anteriores hayan sido cumplidos, el usuario podrá comenzar a dibujar el diagrama del proceso a través de la Barra de Herramientas de Componentes que se visualiza en la parte izquierda de la Figura. La única restricción al iniciar esta tarea, es que sólo puede existir un *start event* y un *end event* en cada diagrama, así que será necesario utilizar las flechas de secuencia para reutilizarlos. Por cada elemento actividad, será posible seleccionar propiedades adicionales como su nombre y el número de actividad y otras que deberán requisitarse, tal y como lo muestra la Figura C.9.

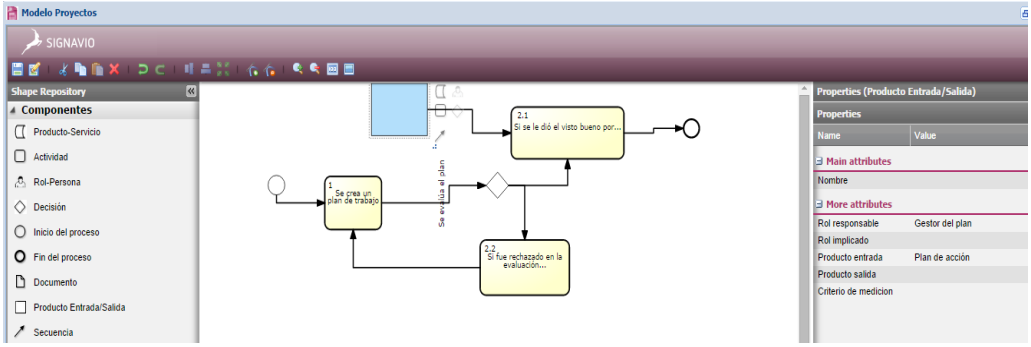


Figura C.9 Ventana "Modelo Proyectos".

Una vez que el usuario considere que ha finalizado la edición de su diagrama, deberá pulsar sobre el botón "Guardar" y cerrar la ventana de edición. Esto lo regresará a la ventana anterior mostrada en la Figura C.7. Puesto que se pretende comparar un diagrama que ha sido creado por los estudiantes de forma colaborativa, y con el visto bueno del jefe de proyectos, con un diagrama "ideal" introducido por MoProSoft, el usuario deberá seleccionar de la lista el diagrama que acaba de crear y pulsar sobre el botón "Comparar". El sistema le mostrará todos los errores encontrados en el diagrama creado. Algunos errores típicos pueden estar relacionados con la acción de haber dejado elementos del diagrama sin nombre o sin asociación mediante las flechas de secuencia, no tener un start/end event o tener más de un start/end event.

Fuera de los errores propios de la edición, el sistema comparará un diagrama interno que está basado en MoProSoft contra el diagrama que fue creado y seleccionado por el usuario y mostrará el porcentaje de completitud/cobertura que tiene en relación al modelo de referencia MoProSoft. Una lista resumirá las actividades que hace falta realizar para cubrir el área de proceso que se pretende modelar, tal y como se muestra en la Figura C.10.

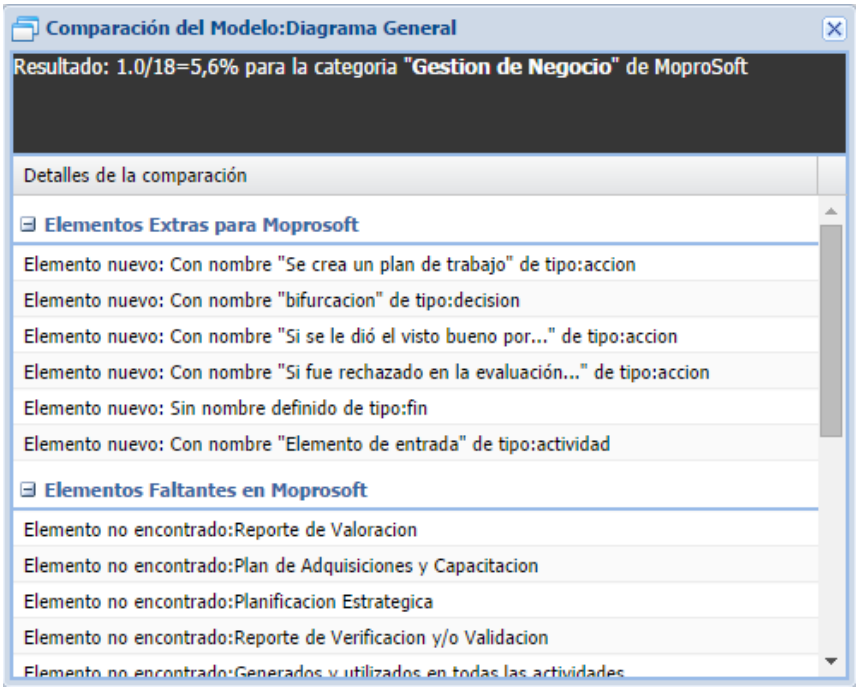


Figura C.10 Ventana "Resultados de la evaluación".

8.6.2. Uso de cuestionarios.

Para iniciar con la segunda fase de evaluación, el usuario deberá acceder a la pestaña Cuestionarios a través de la ventana principal de "Propiedades de proyecto". Así, los jefes de proyecto deberán contestar *on-line* todos los cuestionarios que el sistema les proporcione (véase Figura C.11). El sistema guarda continuamente la sesión en caso de que ocurra algún problema de red o simplemente el usuario se desconecte y decida continuar más tarde con el proceso de evaluación.

El usuario deberá responder a cada pregunta seleccionando una de las posibles respuestas que se le proporcionan y añadir comentarios de manera obligatoria en caso de escoger "No sabe" o "No aplica". Para avanzar en cada pregunta el usuario deberá pulsar el botón "Guardar". Es importante mencionar que es responsabilidad de los usuarios contestar todas las preguntas para que la evaluación de los cuestionarios sea exitosa.

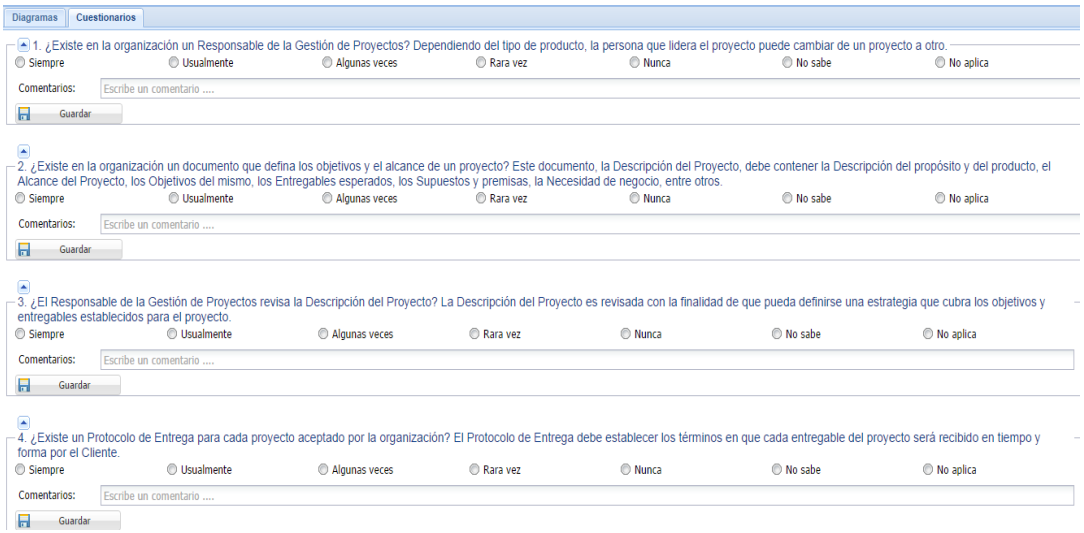


Figura C.11 Ventana "Cuestionarios".

8.6.3. Resultados de la evaluación

En la ventana principal de proyectos se visualizará un icono en forma de gráfica de barras por cada proyecto listado. El usuario deberá pulsar sobre dicho icono para visualizar las gráficas asociadas a los resultados obtenidos en la evaluación y correspondientes a todas las áreas evaluadas distribuidas en pestañas como se muestra en la Figura C.12.

Figura C.12 Ventana "Resultados de Evaluación".

En el ejemplo mostrado en la Figura C.12, del lado izquierdo se muestra una gráfica que contiene barras en color rojo que denotan la desviación estándar y en color azul las barras correspondientes a la media de las preguntas 130 a 145 del área Integración y Pruebas. Del lado derecho se muestra otra gráfica de barras que muestra la Cobertura por Pregunta para el área indicada. Para el caso de la gráfica de la izquierda, que la desviación se muestre en color rojo significa que es una práctica débil y si se muestra en color gris le indica al usuario que es una práctica fuerte. Para el caso de la gráfica derecha, si la cobertura se muestra roja es una práctica débil y si es azul se trata de una práctica fuerte. Cabe mencionar que el usuario también podrá

visualizar una gráfica de keviat que le mostrará el nivel de cobertura total de todas las áreas de proceso que hayan sido evaluadas.

8.7. Generar plan de mejora.

Después de haber revisado las gráficas y haber interpretado los datos, los estudiantes deben generar el plan de mejora que establece el modelo IDEAL. A través de la ventana de resultados mostrada en la Figura C.12, el usuario deberá seleccionar la pestaña llamada "Generar Plan de Mejora". Con esta acción el usuario visualizará todas las actividades que la evaluación arrojó como prácticas débiles en la evaluación y en este caso se mostrará la actividad que debe realizarse como mejora (véase Figura C.13). Con ayuda del profesor, los estudiantes deberán determinar y asignar una prioridad a cada una de las actividades y justificar su decisión mediante un cuadro de texto presentado en la misma herramienta. Esta última acción pretende realimentar al resto de la clase en cuanto a las decisiones tomadas sobre el proceso a nivel organizacional.

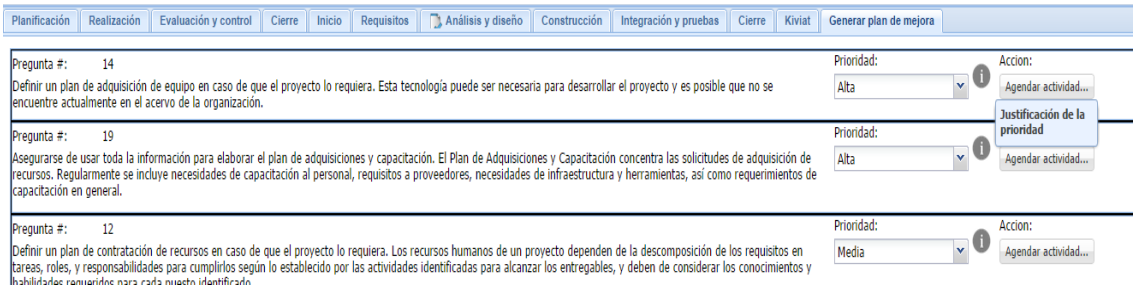


Figura C.13 Ventana "Generar plan de mejora".

Una vez asignada la prioridad y rellenado el campo de texto que pide la justificación sobre la decisión tomada, el usuario deberá pulsar sobre el botón "Agendar actividad" y deberá requisitar información adicional, tal y como lo muestra la Figura C.14.

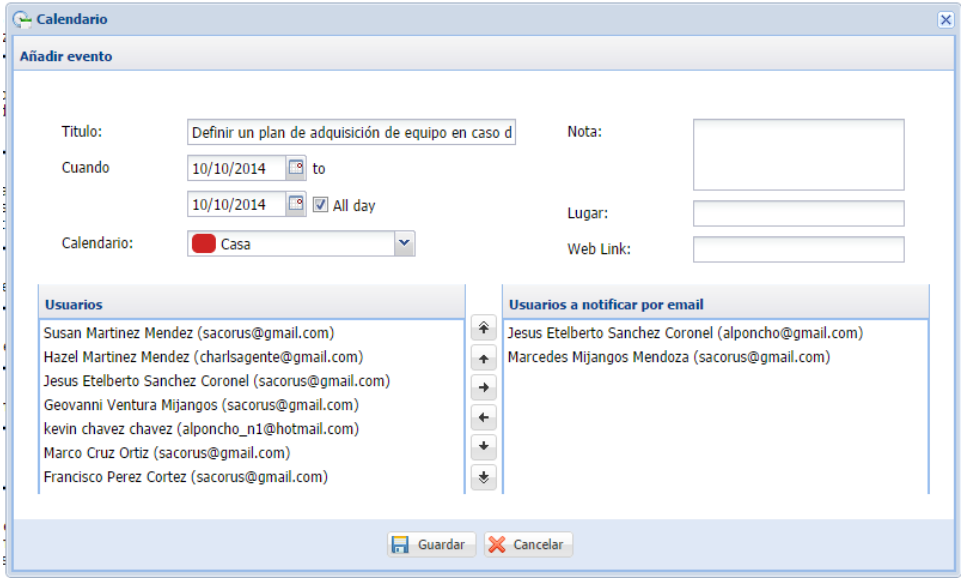


Figura C.14 Ventana "Añadir evento".

Esta ventana permitirá que el usuario agende la fecha en la cual se llevará a cabo dicha actividad de mejora. Como parte del trabajo en equipo, será necesario añadir los datos de los usuarios que serán notificados por email para que estén enterados de esta acción.

Posteriormente los participantes involucrados podrán revisar las fechas agendadas accediendo al calendario del proyecto a través del menú Inicio->Calendario (véase Figura C.14).

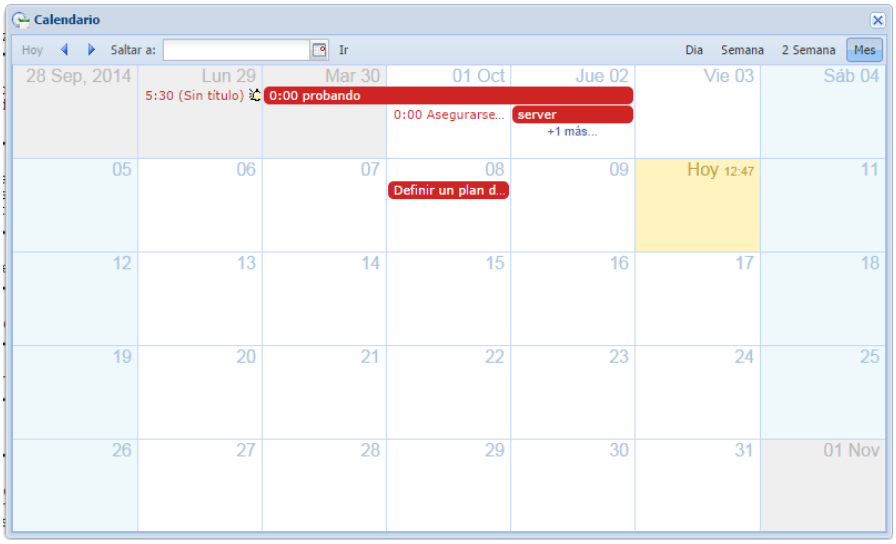


Figura C.14 Ventana “Calendario”.

Si una la fecha actual del sistema coincidiera con la última fecha en la que fue asignada una actividad de mejora, el sistema emitirá un mensaje de aviso para dicha actividad.

9. Anexo D.- Acta de publicación

Software Engineering Education for a Graduate Course: A Web-Based Tool for Conducting Process Improvement Initiatives With Local Industry Collaboration

I. A. GARCIA,[1] J. A. CALVO-MANZANO,[1] C. L. PACHECO,[2] C. A. PEREZ[2]

[1]*Languages and Information Systems and Software Engineering Department, Technical University of Madrid, 28040, Madrid, Spain*

[2]*Postgraduate Division, Technological University of the Mixtec Region, Oaxaca, Mexico*

Received 20 February 2013; accepted 25 August 2013

ABSTRACT: In recent years, there has been an on-going demand for better services and functionality in software products; as a consequence many models, techniques, and tools have been developed such as CMMI-DEV v1.2, TSP, or Scrum. However, software products still suffer from excessive costs, delays in delivery, and low quality. Furthermore, there is a lack of educational material providing high levels of interaction between students and the software industry to learn about how enterprises adopt these models, techniques, and tools into their daily work. This article describes a web-based Tool (EduSysProVAL) to support a graduate course in collaboration with the local software industry. The main goal of this research is to demonstrate that a Software Engineering course may use the EduSysProVAL tool to improve students' practical and professional skills, thus increasing their participation and effort in improvement initiatives, in comparison to traditional educational approaches which are only based on theory classes. This research uses a four-category questionnaire and follow-up interviews to evaluate a satisfactory level of tool effectiveness with undergraduate students and summarizes the industry's positive perceptions about its contribution to the course. © 2013 Wiley Periodicals, Inc. Comput Appl Eng Educ; View this article online at wileyonlinelibrary.com/journal/cae; DOI 10.1002/cae.21584

Keywords: Software Engineering education; web-based tool; graduate level; software industry collaboration

INTRODUCTION

Since the beginning of the eighties, the software industry has tried to increase its quality and productivity by applying new methods and techniques. It has been recognized that unfortunately the fundamental problem for many companies is the incapacity to manage the software process [1]. According to researchers such as Okiraba [2] and Johnson [3], the software development process is far from being a "mature process." Both large enterprises and small enterprises have made an important effort to improve their software process. One indicator of this is the increasing number of international initiatives related to process improvement, such as the Capability Maturity Model Integration for Development v1.3 (CMMI-DEV v1.3) [4], ISO/IEC 15504:2004 [5], SPICE [6], and ISO/IEC 12207:2008 [7]. In addition, many methods for assessing processes in organizations, such as the Standard CMMI Appraisal Method for Process Improvement (SCAMPI) [8] and ISO/IEC 15504:2004 [5] and improvement lifecycle models such as IDEAL [9] and ISO/IEC 15504:2004 [5] have been developed. Thus a process improvement initiative has become one of the main aims of software enterprises (SEs), because of the fact that the

Correspondence to I. A. Garcia (ivan@mixteco.utm.mx)

10. Bibliografía

[**Abran et al., 2001**] Abran, A., Moore, J. W., Bourque, P., Dupuis, R. and Tripp, L. L. "Guide to the Software Engineering Body of Knowledge (SWEBOK), Version 1.00" Los Alamitos, CA, USA: IEEE Computer Society Press. 2001.

[**Aguinis & Kraiger, 2009**] Aguinis, H. and Kraiger, K. "Benefits of training and development for individuals and teams, organizations, and society" *Annual Review of Psychology*, 60(1): 451-474, 2009.

[**Akgün et al., 2011**] Akgün, A. E., Keskin, H., Byrne, J. C. and Gunsel, A. "Antecedents and results of emotional capability in software development project teams" *Journal of Product Innovation Management*, 28(6): 957-973, 2011.

[**Alanis-Funes et al., 2011**] Alanis-Funes, G. J., Neri, L. and Noguez, J. "Virtual collaborative space to support active learning" *Proc. of the 41th IEEE Frontiers in Education Conference*, IEEE Computer Society, pp. 1-6, 2011.

[**Almeida et al., 2012**] Almeida, E., Dali, L., Faulk, S., Lima, C., Rui, Z., Weiss, D., Ying, J., Young, M. and Yu, L. "Teaching globally distributed software development: An experience report" *Proc. of the 25th Conference on Software Engineering Education and Training (CSEET&T)*, IEEE Computer Society, pp. 105-109, 2012.

[**Ardis & Ford, 1989**] Ardis, M. A. and Ford, G. "SEI report on graduate software engineering education" CMU/SEI 89-TR-21, Software Engineering Institute, Carnegie Mellon University. 1989.

[**Ardis et al., 2008**] Ardis, M. A., Chenoweth, S. V. and Young, F. H. "The 'soft' topics in software engineering education" *Proc. of the 38th ASSE/IEEE Frontiers in Education Conference*, IEEE Computer Society, pp. 1-6, 2008.

[**Ardis et al., 2011**] Ardis, M. A., Bourque, P., Hilburn, T., Lasfer, K., Lucero, S., McDonald. J., Pyster, A. and Shaw, M. "Advancing software engineering professional education" *IEEE Software*, 28(4): 58-63, 2011.

[**Barnes, 2007**] Barnes, J. "Implementing the IBM® Rational Unified Process® and solutions: A guide to improving your software development capability and maturity" Upper Saddle, NJ: IBM Press. 2007.

[**Beck, 2008**] Beck, R. J., "What are learning objects?" Center for International Education, University of Wisconsin-Milwaukee. 2008.

[**Bézivin & Gerbé, 2001**] Bézivin, J. and Gerbé, O. "Towards a precise definition of the OMG/MDA framework" *Proc. of the 16th International Conference on Automated Software Engineering*, IEEE Computer Society, pp. 273-282, 2001.

[**Biberoglu & Haddad, 2002**] Biberoglu, E. and Haddad, H. "Survey of industrial experiences with CMM and the teaching of CMM practices" *Journal of Computing Sciences in Colleges*, 18(2): 143-152, 2002.

[**Bonwell & Sutherland, 1996**] Bonwell, C. C. and Sutherland, T. E. "The active learning continuum: Choosing activities to engage students in the classroom" *New Directions for Teaching and Learning*, 1(67): 3-16, 1996.

[**Bourque et al., 1999**] Bourque, P., Dupuis, R., Abran, A., Moore, J. W. and Tripp, L. "The guide to the software engineering body of knowledge" *IEEE Software*, 16(6): 35-44, 1999.

[**Budimac et al., 2011**] Budimac, Z., Putnik, Z., Ivanović, M., Bothe, K. and Schuetzler, K. "On the assessment and self-assessment in a students teamwork based course on software engineering" *Computer Applications in Engineering Education*, 19(1): 1-9, 2011.

[**Bustos & Nussbaum, 2009**] Bustos, H. and Nussbaum, M. "An experimental study of the inclusion of technology in higher education" *Computer Applications in Engineering Education*, 17(1): 100-107, 2009.

[**Calvo-Manzano et al., 2008**] Calvo-Manzano, J. A., Bayona, S., Cuevas, S. and Feliu, T. S. "Teaching team software process in graduate courses to increase productivity and improve software quality" *Proc. of the 32nd Annual IEEE International Computer Software and Applications Conference*, IEEE Computer Society, pp. 440-446, 2008.

[**Cárdenas, 2009**] Cárdenas, C. "Social design in multidisciplinary engineering design courses" *Proc. of the 39th IEEE Frontiers in Education Conference*, IEEE Computer Society, pp. 1-6, 2009.

[**Carver eta al., 2003**] Carver, J., Jaccheri, M. L., Morasca, S. and Shull, F. "Issues in using students in empirical studies in software engineering education" *Proc. of the 9th International Software Metrics Symposium*, IEEE Computer Society, pp. 239-249, 2003.

[**Casas et al., 2010**] Casas, I., Collazos, C., Guerrero, L. A., Leiva, C., Ochoa, S. and Puente, J. "Addressing computer-supported collaborative learning in the classroom: Experiences in engineering education" *Procedia - Social Behavioral Sciences*, 2 (2): 2685-2688, 2010.

[**Cataldo et al., 2008**] Cataldo, M., Herbsleb, J. and Carley, K. "Socio-technical congruence: A framework for assessing the impact of technical and work dependencies on software development productivity" *Proc. of the 2nd ACM-IEEE International Symposium on Empirical Software Engineering and Measurement*, ACM Publisher, pp. 2-11, 2008.

[**Cater-Steel et al., 2006**] Cater-Steel, A., Toleman, M. and Rout, T. "Process improvement for small firms: an evaluation of the rapid assessment-based method" *Information and Software Technology*, 48(5): 323-334, 2006.

[**Cavalcanti et al., 2008**] Cavalcanti, A. P. C, Santos, S., Moraes, M. d. C., Albuquerque, J. and Meira, S. "An evaluation approach based on the problem-based learning in a software engineering master course" *Journal of Technology Management & Innovation*, 3(2): 18-28, 2008.

[**Chao, 2007**] Chao, J. "Student project collaboration using Wikis" *Proc. of the 20th Conference on Software Engineering Education & Training*, IEEE Computer Society, pp. 255-261, 2007.

[**Chen & Chong, 2011**] Chen, C. Y. and Chong, P. P. "Software engineering education: A study on conducting collaborative senior project development" *Journal of Systems and Software*, 84(3): 479-491, 2011.

[**Chen & Li, 2011**] Chen, M. H. and Li, T. L. "Construction of a high-performance computing cluster: a curriculum for engineering and science students" *Computer Applications in Engineering Education*, 19(4): 678-684. 2011.

[Chen & Teng, 2011] Chen, C. Y. and Teng, K. C. "The design and development of a computerized tool support for conducting senior projects in software engineering education" *Computers & Education*, 56(3): 802-817, 2011.

[**Chickering & Ehrmann, 1987**] Chickering, A. W. and Ehrmann, S. C. "Implementing the seven principles: Technology as lever" *American Association of Higher Education Bulletin*, 39(1): 3-7, 1987.

[**Chrissis et al., 2009**] Chrissis, M. B., Konrad, M. and Shrum, S. "CMMI - Guía para la integración de procesos y la mejora de productos" Madrid, Spain: Pearson Addison Wesley. 2009.

[**Chrissis et al., 2012**] Chrissis, M. B., Konrad, M. and Shrum, S. "CMMI para el desarrollo versión 1.3: Guía para la integración de procesos y la mejora de productos" Madrid, Spain: Editorial Universitaria Ramón Areces. 2012.

[**Clinton & Hokanson, 2012**] Clinton, G. and Hokanson, B. "Creativity in the training and practice of instructional designers: the design/creativity loops model" *Educational Technology Research and Development*, 60(1): 111-130, 2012.

[**CMMI, 2002**] CMMI Product Team. "CMMI for systems engineering, software engineering, integrated product and process development, and supplier sourcing (CMMI-SE/SW/IPPD/SS, v1.1). Continuous representation" CMU/SEI-2002-TR-011, Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University. 2002.

[**CMMI, 2006**] CMMI Product Team. "CMMI for Development (CMMI-DEV, v1.2)" CMU/SEI-2006 TR-008, Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University. 2006.

[**CMMI, 2010**] CMMI Product Team. "CMMI for Development (CMMI-DEV, v1.3)" CMU/SEI-2010-TR-033, Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University. 2010.

[**Coallier, 1995**] Coallier, F. "TRILLIUM: A model for the assessment of Telecom product development & support capability" *Software Process Newsletter*, 2(1): 3-8, 1995.

[**Cohn, 2005**] Cohn, M. "Agile estimating and planning" Upper Saddle River, NJ: Prentice Hall, 2005.

[**Collins et al., 2008**] Collins, D., Deck, A. and McCrickard, M. "Computer aided instruction: A study of student evaluations and academic performance" *Journal of College Teaching & Learning*, 5(11): 49-58, 2008.

[**Cramer et al., 2010**] Cramer, S. F., Tetewsky, S. J. and Marczynski, K. S. "What facilitates and impedes collaborative work during higher education software implementation projects?" *Journal of Educational Change*, 11(4): 457-486, 2010.

[**Daniels et al., 2010**] Daniels, M., Cajander, A. and Pears, A. "Engineering education research in practice: evolving use of open ended group projects as a pedagogical strategy for developing skills in global collaboration" *International Journal on Engineering Education*, 26(4): 1-12, 2010.

[**Dawson, 2000**] Dawson, R. "Twenty dirty tricks to train software engineers" *Proc. of the 22nd International Conference on Software Engineering*, ACM, pp. 209-218, 2000.

[Debnath & Pandey, 2011] Debnath, M. and Pandey, M. "Enhancing engineering education learning outcomes using project-based learning: A case study" *International Journal of Quality Assurance in Engineering and Technology Education*, 1(2): 23-34, 2011.

[**Dingsøyr et al., 2000**] Dingsøyr, T., Jaccheri, M. L. and Wang, M. L. "Teaching software process improvement through a case study" *Computer Applications in Engineering Education*, 8(3-4): 229-234, 2000.

[**Dubisnky et al., 2010**] Dubisnky, Y., Yaeli, A. and Kofman, A. "Effective management of roles and responsibilities: Driving accountability in software development teams" *IBM Journal of Research and Development*, 54(2): 1-11, 2010.

[**Dyba, 2005**] Dyba, T. "An empirical investigation of the key factors for success in software process improvement" *IEEE Transactions on Software Engineering*, 31(5): 410-424, 2005.

[**Favela & Peña-Mora, 2001**] Favela, J. and Peña-Mora, F. "An experience in collaborative software engineering education" *IEEE Software*, 18(2): 47-53, 2001.

[**Figueiredo et al., 2010**] Da C. Figueiredo, R. M., De Sales, Ribeiro, L. C. M. R. Jr., Laranjeira, L. A. F. and Rocha, A. "Teaching software quality in an interdisciplinary course of engineering" *Proc. of the 7th International Conference on the Quality of Information and Communications Technology*, IEEE Computer Society, pp. 144-149, 2010.

[**Fornaro et al., 2007**] Fornaro, R. J., Heil, M. R. and Tharp, A. L. "Reflections on 10 years of sponsored senior design projects: students win–clients win" *Journal of Systems and Software*, 80(8): 1209-1216, 2007.

[**Freeman et al., 1976**] Freeman, P., Wasserman, A. I. and Fairley, R. E. "Essential elements of software engineering education" *Proc. of the 2nd International Conference on Software Engineering (ICSE 76)*, IEEE Computer Society, pp. 116-122, 1976.

[**Freeman & Wasserman, 1978**] Freeman, P. and Wasserman, A. I. "A proposed curriculum for software engineering education" *Proc. of the 3rd International Conference on Software Engineering (ICSE 78)*, IEEE Computer Society, pp. 56-62, 1978.

[**Fuller & Moreno, 2004**] Fuller, D. A. and Moreno, A. F. "Experimenting with a computer-mediated collaborative interaction model to support engineering courses" *Computer Applications in Engineering Education*, 12(3): 175-188, 2004.

[**Garcia et al., 2007**] Garcia, I., Calvo-Manzano, J. A., Cuevas, G. and San Feliu, T. "Determining practice achievement in project management using a two-phase questionnaire on small and medium enterprises" *Software process improvement*, P. Abrahamsson, N. Baddoo, T. Margaria, and R. Messnarz (Eds.), Springer-Verlag, Berlin Heidelberg, 2007, pp. 46-58.

[**Garcia et al., 2010**] Garcia, I., Pacheco, C. and Coronel, N. "Learn from practice: defining an alternative model for software engineering education in Mexican universities for reducing the breach between industry and academia" *Proc. of the 2010 International Conference on Applied Computer Science*, WSEAS Press, pp. 120-124, 2010.

[**Garcia et al., 2011**] Garcia, I., Pacheco, C., Cruz, D. and Calvo-Manzano, J. A. "Implementing the modeling-based approach for supporting the software process assessment in SPI initiatives inside a small software company" *Studies in Computational Intelligence*, R. Lee (Ed.), Springer-Verlag, Berlin Heidelberg, 2011, pp. 1-13.

[**Garcia et al., 2011a**] Garcia, I., Pacheco, C., Calvo-Manzano, J., Cuevas, G., San Feliu, T. and Mendoza, E. "Managing the software process with a software process improvement tool in a small enterprise" *Journal of Software Maintenance and Evolution: Research and Practice*, 24(5): 481-491, 2011.

[**Garcia & Pacheco, 2012**] Garcia, I. and Pacheco, C. "Using TSPi and PBL to support software engineering education in an upper-level undergraduate course" *Computer Applications in Engineering Education*, Published online in Wiley Online Library, DOI: 10.1002/cae.21566.

[**Garcia et al., 2014**] Garcia, I., Pacheco, C. and Calvo-Manzano, J. A. "Changing the software engineering education: a report from current situation in Mexico" in Lee, R. (Eds.). *Studies in Computational Intelligence*, Springer-Verlag, pp. 43-58, 2014.

[**Gillet et al., 2003**] Gillet, D., Geoffroy, F., Zeramdini, K., Nguyen, A. V., Rekik, Y. and Piguet, Y. "The cockpit: An effective metaphor for web‐based experimentation in engineering education" *International Journal of Engineering Education*, 19(3): 389-397, 2003.

[**Gillham, 2000**] Gillham, B. "Developing a questionnaire" London, New York: Continuum. 2000.

[**Göl & Nafalski, 2007**] Göl, Ö. and Nafalski, A. "Collaborative learning in engineering education" *Global Journal of Engineering Education*, 11(2): 173-180, 2007.

[**Habra et al., 2008**] Habra, N., Alexandre, S., Desharnais, J-M., Laporte, C. Y. and Renault, A. "Initiating software process improvement in very small enterprises: Experience with a light assessment tool" *Information and Software Technology*, 50(7-8), 763-771, 2008.

[**Hassan, 2008**] Hassan, A. "A methodology for combining development and research in teaching undergraduate software engineering" *International Journal of Engineering Education*, 24(3): 567-580, 2008.

[**Hamann, 2006**] Hamann, D. "Towards an integrated approach for software process improvement: combining software process assessment and software process modeling" *PhD Thesis in Experimental Software Engineering*, D. Rombach, F. Bomarius and P. Liggesmeyer (Editors), vol. 19, Fraunhofer IESE, 2006.

[**Hawker, 2009**] Hawker, J. S. "A software process engineering course" *Proc. of the 2009 American Society for Engineering Education Annual Conference*, American Society for Engineering Education, pp. 25-31, 2009.

[**Hainey et al., 2011**] Hainey, T., Connolly, T. M., Stansfield, M. and Boyle, E. A. "Evaluation of a game to teach requirements collection and analysis in software engineering at tertiary education level" *Computers & Education*, 56(1): 21-35, 2011.

[**Hislop, 1999**] Hislop, G. W. "Teaching process improvement in a graduate software engineering course" *Proc. of the 29th Annual Frontiers in Education Conference*, IEEE Computer Society, pp. 19-21, 1999.

[**Humphrey, 1996**] Humphrey, W. S. "Introduction to the personal software process" 1st. Edition, Reading MA: Addison-Wesley Professional, 1996.

[**Humphrey, 2000**] Humphrey, W. S. "Introduction to the team software process" 1st. Edition, Reading MA: Addison-Wesley Professional, 2000.

[**Humphrey, 2008**] Humphrey, W. S. *Software: The competitive edge*. International Conference in Software Engineering and Applications, Guadalajara, Jalisco, Mexico. 2008.

[**IEEE, 2006**] Institute of Electrical and Electronics Engineers, IEEE Standard for Developing a Software Project Life Cycle Process, New York: IEEE, 2006.

[**ISO/IEC, 1998**] ISO/IEC, "ISO/IEC TR 15504:1998(e): Information technology – software process assessments. Parts 1-9" International Organization for Standardization, Geneva, 1998.

[**ISO/IEC, 2004**] ISO/IEC, "ISO/IEC 15504:2003/cor.1:2004(e): Information technology – process assessment. Parts 1-5" International Organization for Standardization, Geneva, 2004.

[**ISO/IEC, 2008**] ISO/IEC, "ISO/IEC 12207: 2008, systems and software engineering – software life cycle processes" International Organization for Standardization/ International Electrotechnical Commission, Geneva, 2008.

[**Iversen et al., 2004**] Iversen, J. H., Mathiassen, L. and Nielsen, P. A. "Managing risk in software process improvement: an action research approach" *Journal MIS Quarterly*, 28(3): 395-433, 2004.

[**Jaccheri, 2002**] Jaccheri, M. L. "Software quality and software process improvement course based on interaction with the local software industry" *Computer Applications in Engineering Education*, 9(4): 265-272, 2002.

[**Janz et al., 1997**] Janz, B. D., Colquitt, J. A. and Noe, R. A. "Knowledge worker team effectiveness: the role of autonomy, interdependence, team development, and contextual support variables" *Personnel Psychology*, 50(4): 877-904, 1997.

[**Johnson, 2006**] Johnson, J. "My life is failure: 100 things you should know to be a better project leader" Boston MA: Standish Group International Publisher, 2006.

[**Johnson & Johnson, 1994**] Johnson, R. T. and Johnson, D. W. "An overview of cooperative learning", *Creativity and collaborative learning: A practical guide to empowering students and teachers*, in Thousand, J., Villa, A. and Nevin, A. (Eds.), Brookes Press, Baltimore, MD, pp. 31-44, 1994.

[**Joyce et al., 2013**] Joyce, T., Evans, L., Pallan, W. and Hopkins, C. "A hands-on project-based mechanical engineering design module focusing on sustainability" *Engineering Education*, 8(1): 65-80, 2013.

[**Juristo & Moreno, 2010**] Juristo, N. and Moreno, A. "Basics of software engineering experimentation" Dordrecht, the Netherlands: Kluwer Academic Press. 2010.

[**Kamthan, 2013**] Kamthan, P. "An exploration of the social web environment for collaborative software engineering education", *Web-based and Blended Educational Tools and Innovations*, in Karacapilidis, N., Raisinghani, M. S. and Ng, E. M. W. (Eds.), pp. 1-23, 2013.

[**Kirkpatrick & Kirkpatrick, 2006**] Kirkpatrick D. L. and Kirkpatrick, J. D. "Evaluating training programs: The four levels" 3rd Edition, San Francisco, CA: Berrett-Koehler Publishers. 2006.

[**Kirschner, 2004**] Kirschner, P. A. "Design, development, and implementation of electronic learning environments for collaborative learning" *Educational Technology Research and Development*, 52(2): 39-46, 2004.

[**Kitchenham, 2004**] Kitchenham, B. A. "Procedures for performing systematic reviews" Joint Technical Report Software Engineering Group. Department of Computer Science Keele University (UK) and Empirical Software Engineering, National ICT Australia, 2004.

[**Komi-Sirviö, 2004**] Komi-Sirviö, S. "Development and Evaluation of Software Process Improvement Methods" Espoo 2004, VTT Publications 535, pp. 175, 2004.

[**Kramarski & Mizrachi, 2006**] Kramarski, B. and Mizrachi, S. "Online discussion and self-regulated learning: effects of instructional methods on mathematical literacy" *Journal of Educational Research*, 99(4): 218-230, 2006.

[**Krathwohl et al., 1964**] Krathwohl, D. R., Bloom, B. S. and Masia, B. B. "Taxonomy of educational objectives. The classification of educational goals. Handbook II: Affective domain" New York, NY: David McKay. 1964.

[**Kreijns et al., 2007**] Kreijns, K., Kirschner, P. A., Jochems, W. and Buuren, H. V. "Measuring perceived sociability of computer-supported collaborative learning environments" *Computers & Education*, 49(2): 176-192, 2007.

[**LeBlanc & Sobel, 2004**] LeBlanc, R. and Sobel, A. "Software Engineering 2004 Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering" ACM, 2004.

[**Lee, 2000**] Lee, Y. S. "The sustainability of university-industry research collaboration: an empirical assessment" *The Journal of Technology Transfer*, 25(2): 111-133, 2000.

[**Lethbridge, 1998**] Lethbridge, T. C. "The relevance of software education: A survey and some recommendations" *Annals of Software Engineering*, 6(1-4): 91-110, 1998.

[**Likert, 1932**] Likert, R. "A technique for the measurement of attitudes" *Archives of psychology*, 22(140): 1-55, 1932.

[**Liu et al., 2009**] Liu, S., Takahashi, K., Hayashi, T. and Nakayama, T. "Teaching formal methods in the context of software engineering" *ACM SIGCSE Bulletin*, 41(2): 17-23, 2009.

[**Llorens et al., 2013**] Llorens, A., Llinàs-Audet, X., Ras, A. and Chiaramonte, L. "The ICT skills gap in Spain: industry expectations versus university preparation" *Computer Applications in Engineering Education*, 21(2): 256-264, 2013.

[**Lu & DeClue, 2011**] Lu, B. and DeClue, T. "Teaching agile methodology in a software engineering capstone course" *Journal of Computing Sciences in Colleges*, 26(5): 293-299, 2011.

[**Mahmood, 2011**] Mahmood, Z. "Teaching software engineering to undergraduate computing students" *International Journal of Teaching and Case Studies*, 3(2): 112-122, 2011.

[**Marquardt, 1992**] Marquardt, D. "ISO 9000: A universal standard of quality" *Management Review*, 81(1): 50-52, 1992.

[**Maxinez et al., 2011**] Maxinez, D. G., Garcia-Galvan, M. A., Sanchez-Rangel, F. J., Chavez-Cuayahuitl, E., Gonzalez, E. A., Rodriguez-Bautista, R., Ferreyra-Ramirez, A., Aviles-Cruz, C. and Siller-Alcala, I. I. "Interactive scenario development", *Recent Researches in Mathematical Methods in Electrical Engineering and Computer Science*, in Thomas, G., Fleaurant, C., Panagopoulos, T. & Chevassus-Lozza, E. (Eds.), pp. 35-39. 2011.

[**McFeeley, 1996**] McFeeley, B. "IDEAL: A user's guide for software process improvement" CMU/SEI-96- HB-001, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA., 1996.

[**Mead, 2009**] Mead, N. R. "Software engineering education: how far we've come and how far we have to go" *Journal of Systems and Software*, 82(4): 571-575, 2009.

[**Mendoza et al., 2009**] Mendoza, R., Morales, M., Morgado, M., Oktaba, H., Ibargüengoitia, G., Pino, F. and Piattini, M. "Supporting the software process improvement in very small entities through e- learning: the HEPALE! project" *Proc. of the Mexican International Conference on Computer Science*, IEEE Computer Society, pp. 221-231, 2009.

[**Mills et al., 1980**] Mills, H. D., O'Neill, D., Linger, R. C., Dyer, M. and Quinnan, R. E. "The management of software engineering" *IBM Systems Journal*, 24(2): 414-477, 1980.

[**Moe & Šmite, 2008**] Moe, N. B. and Šmite, D. "Understanding a lack of trust in global software teams: A multiple-case study" *Software Process Improvement and Practice*, 13(3): 217-231, 2008.

[**Moore & Brennan, 1995**] Moore, M. M. and Brennan, T. "Process improvement in the classroom" *Proc. of the SEI Conference on Software Engineering Education*, Springer-Verlag, pp. 123-130, 1995.

[**Nichols, 2008**] Nichols, M. "E-learning in context" New Zealand: Lidlaw College. 2008.

[**Nichols & Salazar, 2009**] Nichols, W. R. and Salazar, R. "Deploying TSP on a national scale: an experience report from pilot projects in Mexico", CMU/SEI-2009-TR-011, Software Engineering Institute, Carnegie Mellon University, 2009.

[**NYCE, 2005**] Normalización y Certificación. NMX-I-059/01-NYCE-2005. Tecnología de la Información -Software- Modelos de procesos y evaluación para desarrollo y mantenimiento de software. Parte 01: Definición de conceptos y productos. México, DF: NYCE. 2005.

[**NYCE, 2005a**] Normalización y Certificación. NMX-I-059/02-NYCE-2005. Tecnología de la Información -Software- Modelos de procesos y evaluación para desarrollo y mantenimiento de software. Parte 02: Requisitos de procesos (MoProSoft). México, DF: NYCE. 2005.

[**NYCE, 2005b**] Normalización y Certificación. NMX-I-059/03-NYCE-2005. Tecnología de la Información -Software- Modelos de procesos y evaluación para desarrollo y mantenimiento de software. Parte 03: Guía de implantación de procesos. México, DF: NYCE. 2005.

[**NYCE, 2005c**] Normalización y Certificación. NMX-I-059/04-NYCE-2005. Tecnología de la Información -Software- Modelos de procesos y evaluación para desarrollo y mantenimiento de software. Parte 04: Directrices para la evaluación de procesos (EvalProSoft). México, DF: NYCE. 2005.

[**Noguez & Espinosa, 2004**] Noguez, J. and Espinosa, E. "Improving learning and soft skills using project oriented learning in software engineering courses" *Proc. of the 2nd International Workshop on Designing Computational Models of Collaborative Learning Interaction*, pp. 83-88, 2004.

[**O'Regan, 2010**] O'Regan, G. "Introduction to Software Process Improvement (Undergraduate Topics in Computer Science)" London, UK: Springer-Verlag. 2010.

[**Oktaba et al., 2004**] Oktaba, H., Alquicira, C., Su Ramos, A., Palacios, J., Pérez, C. and López, F. *Método de Evaluación de procesos para la industria del software EvalProSoft, Versión 1.1*, México. 2004.

[**Oktaba, 2006**] Oktaba, H. "MoProSoft: a software process model for small enterprises" *Proc. of the First International Research Workshop for Process Improvement in Small Settings*, Software Engineering Institute, pp. 93-100, 2006.

[**Oktaba et al., 2007**] Oktaba, H., Garcia, F., Piattini, M., Ruiz, F., Pino, F. J. and Alquicira, C. "Software process improvement: The COMPETISOFT Project" *Computer*, 40(10): 21-28, 2007.

[**Oktaba & Piatinni, 2008**] Oktaba, H. and Piatinni, M. "Software process improvement for small and medium enterprises: techniques and case studies" New York: Information Science Reference Publisher, 2008.

[**Paulk et al., 1993**] Paulk, M., Weber, V., Garcia, S., Chrissis, M. and Bush, M. "Key Practices of the Capability Maturity Model (Version 1.1)" Technical Report CMU/SEI-93-TR-25, Software Engineering Institute, Pittsburgh, PA, 1993.

[**Peredo-Valderrama, 2011**] Peredo-Valderrama, R., Canales-Cruz, A. and Peredo-Valderrama, I. "An approach toward a software factory for the development of educational materials under the paradigm of WBE" *Interdisciplinary Journal of E-Learning and Learning Objects*, 7: 55-67, 2011.

[**Pino et al., 2010**] Pino, F. J., Pardo, C., García, F. and Piattini, M. "Assessment methodology for software process improvement in small organizations" *Information and Software Technology*, 52(10): 1044-1061, 2010.

[**PMI, 2009**] Project Management Institute. "A guide to the project management body of knowledge (PMBOK guide)" 4th Edition, Newton Square: PA, Project Management Institute. 2009.

[**Polanco et al., 2004**] Polanco, R., Calderon, P. and Delgado, F. "Effects of a problem-based learning program on engineering students' academic achievements in a Mexican university" *Innovations in Education and Teaching International*, 41(2): 145-155, 2004.

[**Pyster, 2009**] Pyster, A. *Graduate Software Engineering 2009 (GSwE2009): Curriculum Guidelines for Graduate Degree Programs in Software Engineering, Integrated Software & Systems Engineering Curriculum Project*, Stevens Institute of Technology, September, 2009.

[**Pyster et al., 2009**] Pyster, A., Lasfer, K., Turner, R., Bernstein, L. and Henry, D. "Master's degrees in software engineering: An analysis of 28 university programs" *IEEE Software*, 26(5): 94-101, 2009.

[**Ramirez-Hernandez & Leon-Rovira, 2005**] Ramirez-Hernandez, D. and Leon-Rovira, N. "Active learning in engineering: examples at Tecnológico de Monterrey in México", *Research and practice*

of active learning in Engineering Education, de Graff, E., Saunders-Smits, G. N. and Nieweg, M. R. (Eds.), pp. 56-62, 2005.

[**Razmov & Anderson, 2006**] Razmov, V. and Anderson, R. "Pedagogical techniques supported by the use of student devices in teaching software engineering" *Proc. of the 37th SIGCSE Technical Symposium on Computer Science Education*, ACM Publisher, pp. 344-348, 2006.

[**Reichlmayr, 2006**] Reichlmayr, T. J. "Collaborating with industry – strategies for an undergraduate software engineering program" *Proc. of the 2006 International Workshop on Summit on Software Engineering Education*, ACM, pp. 13-16, 2006.

[**Regueras et al., 2011**] Regueras, L. M. Verdú, E. Verdú, M. J. and de Castro, J. P. "Design of a competitive and collaborative learning strategy in a communication networks course" *IEEE Transactions in Education*, 54(2): 302-307, 2011.

[**Resta & Laferrière, 2007**] Resta, P. and Laferrière, T. "Technology in support of collaborative learning" *Educational Psychology Review*, 19(1): 65-83, 2007.

[**Richardson et al., 2010**] Richardson, I., Casey, V., Burton, J. and McCaffery, F. "Global software engineering: A software process approach" *Collaborative Software Engineering*, I. Mistrík, J. Grundy, A. Hoek, J. Whitehead (Eds.), vol. 1, Springer-Verlag Berlin, Heidelberg, pp. 35-56, 2010.

[Richardson et al., 2011] Richardson, I., Reid, L., Seidman, S. B., Pattinson, B. and Delaney, Y. "Educating software engineers of the future: Software quality research through problem-based learning" *Proc. of the 24th IEEE CS Conference on Software Engineering Education and Training*, IEEE Computer Society, 2011, pp. 91–100.

[**Robillard et al., 1994**] Robillard, P. N., Mayrand J. and Drouin, J. N. "Process self-assessment in an educational context" *Software Engineering Education*, J. L. Diaz-Herrera (Ed.), vol. 750, Springer-Verlag Berlin, Heidelberg, pp. 211-225, 1994.

[Rodriguez et al., 2012] Rodriguez, G., Soria, A. and Campo, M. "Supporting virtual meetings in distributed scrum teams" *IEEE Latin America Transactions*, 10(6): 2316-2323, 2012.

[**Rocha et al., 2005**] Rocha, A., Montoni, M., Santos, G., Mafra, S., Figueiredo, S., Albuquerque, A. and Mian, P. "Reference Model for Software Process Improvement: A Brazilian Experience" *Software Process Improvement*, I. Richardson, P. Abrahamsson, R. Messnarz (Eds.), vol. 3792, Springer-Verlag Berlin, Heidelberg, pp. 130-141. 2005.

[**Rooji, 2009**] Rooji, S. W. "Scaffold project-based learning with the project management body of knowledge" *Computers & Education*, 52(1): 210–219, 2009.

[**Roschelle & Teasley, 1995**] Roschelle, J. and Teasley, S. "The construction of shared knowledge in collaborative problem solving" *Computer supported collaborative learning*, C. E. O'Malley (Ed.), vol. 128, Springer-Verlag, Berlin, Heidelberg, pp 69-197, 1995.

[**Sampedro, 2011**] Sampedro, J. L. "Conocimiento y empresa: la industria del software en México" México, D.F: Editorial Plaza y Valdés, S. A. de C.V. / UAM Cuajimalpa, 2011.

[**Sancho-Tomas et al., 2009**] Sancho-Thomas, P., Fuentes-Fernandez, R. and Fernandez-Manjon, B. "Learning teamwork skills in university programming courses" *Computers & Education*, 53(2): 517–531, 2009.

[**SCAMPI, 2006**] Members of the Assessment Method Integrated Team, "Standard CMMI® Appraisal Method for Process Improvement (SCAMPI), version 1.2" CMU/SEI-2006-HB-002, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA., 2006.

[**Schwaber, 2007**] Schwaber, K. "The enterprise and Scrum" Microsoft Press, Redmond, Washington, 2007.

[**Sobel, 2003**] Sobel, A. E. K. "Computing curricula - software engineering volume" *Proc. of the Final draft of the software engineering education knowledge (SEEK)*, IEEE with ACM Press, 2003.

[**Syazwan et al., 2002**] Syazwan, M., Benest, I., Evans, A. and Kimble, C. "Knowledge modeling for developing knowledge management systems" *Proc. of the 3rd European Conference on Knowledge Management*, IEEE Computer Society, 2002, pp. 15-25.

[**Tomayko & Hazzan, 2004**] Tomayko, J. E. and Hazzan, O. "Human aspects of software engineering" Hingham, MA: Charles River Media Inc., 2004.

[**Unterkalmsteiner, 2012**] Unterkalmsteiner, M., Gorschek, T., Islam, A. K. M. M., Cheng, C. K., Permadi, R. B. and Feldt, R. "Evaluation and measurement of software process improvement - a systematic literature review" *IEEE Transactions on Software Engineering*, 38(2): 398-424, 2012.

[**van der Duim, 2007**] van der Duim, L., Andersson, J. and Sinnema, M. "Good practices for educational software engineering projects" *Proc. of the 29th International Conference on Software Engineering*, ACM, pp. 698-707, 2007.

[**van Vliet, 2006**] van Vliet, H. "Reflections on software engineering education" *IEEE Software*, 23(3): 55-61, 2006.

[**Viana et al., 2012**] Viana, D., Conte, T., Vilela, D., de Souza, R. B., Santos, G., and Prikladnicki, R. "The influence of human aspects on software process improvement: qualitative research findings and comparison to previous studies" *Proc. of the 16th International Conference on Evaluation & Assessment in Software Engineering*, IEEE Computer Society, pp. 121-125, 2012.

[**von Wangenheim & Hauck, 2010**] von Wangenheim, C. G. and Hauck, J. C. R. "Teaching software process improvement and assessment" *Proc. of the 17th European Systems and Software Process Improvement and Innovation Conference*, Grenoble Institute of Technology, 2010, pp. 25-34.

[**Young et al., 2006**] Young, H., Fang, T. and Hu, C. "A successful practice of applying software tools to CMMI process improvement" *Journal of Software Engineering Studies*, 1(2): 78-95, 2006.

[**Weisberg, 2009**] Weisberg, H. F. "The total survey error approach" Chicago: University of Chicago Press, 2009.

[**Werth, 1994**] Werth, L. "An Adventure in Software Process Improvement" *Software Engineering Education*, J. L. Diaz-Herrera (Ed.), vol. 750, Springer–Verlag Berlin Heidelberg, 1994, pp. 191-210.

[**Wichstraum, 1995**] Wichstraum, L. "Harter's self-perception profile for adolescents: reliability, validity, and evaluation of the question format" *Journal of personality assessment*, 65(1): 100-116, 1995.

[**Wohlin & orn Regnell, 1999**] Wohlin, C. and orn Regnell, B. "Strategies for industrial relevance in software engineering education" *Journal of Systems and Software*, 49(2-3): 125–134, 1999.

[**Zahran, 1998**] Zahran, S. "Software process improvement: practical guidelines for business success" Addison-Wesley, Essex, England, 1998.

[**Zhu, 2004**] Zhu, Q., Wang, T. and Tan, S. "Adapting Game Technology to Support Software Engineering Process Teaching: From SimSE to MO-SEProcess" *Proc. of the Third International Conference on Natural Computation*, IEEE Computer Society, 2007, pp. 777-780.

10.1. Sitios de Internet

[**URL-1**] Secretaría de Economía. Plan Nacional de Desarrollo 2007-20012. Disponible en: http://pnd.calderon.presidencia.gob.mx/pdf/PND_2007-2012.pdf. Última consulta: Abril 2013.

[**URL-2**] Criteria for Accrediting Engineering Programs (Effective for Evaluations during the 2010-2011 Accreditation Cycle), Accreditation Board for Engineering and Technology. Disponible en: http://www.abet.org/. Última consulta: Abril 2013.

[**URL-3**] Software Engineering 2004: Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering, ACM/IEEE CS Joint Task Force on Computing Curricula. Disponible en: http://www.acm.org/education/curricula-recommendations. Última consulta: Abril 2013.

[**URL-4**] Sparx Systems. Enterprise Architect. Descripción técnica disponible en: http://www.sparxsystems.com.au/. Última consulta: Abril 2013.

[**URL-5**] IMS Global Learning Consortium. IMS Learning Standards. Disponible en: http://www.imsglobal.org/specifications.html. Última consulta: Abril 2013.

[**URL-6**] Advanced Distributed Learning. SCORM. Disponible en: http://www.adlnet.org/scorm. Última consulta: Abril 2013.

[**URL-7**] Google. Google Analytics. Disponible en: http://www.google.com/analytics/. Última consulta: Abril 2013.