Fomentando el uso de herramientas en las actividades de Gestión de Proyectos de Software

Marcela Daniele, Paola Martellotto y Daniel Romero

Departamento de Computación,
Universidad Nacional de Río Cuarto
Enlace Ruta 8 y 36 Km. 601,
C.P. X5804BYA, Río Cuarto, Córdoba, Argentina.
{mdaniele, paola, dromero}@dc.exa.unrc.edu.ar

Resumen La enseñanza y el aprendizaje de conceptos y técnicas de ingeniería de software, presenta un importante desafío en las carreras de informática.
Este trabajo propone una metodología para la enseñanza y el aprendizaje de temas vinculados a la Gestión de Proyectos de Software.La implementación de esta metodología en un curso de Ingeniería de Software, brinda al alumno la posibilidad de solidificar los conocimientos teóricos aprendidos en la carrera, a través de su aplicación en proyectos reales y con un uso intensivo de herramientas que faciliten su futuro desempeño profesional.

Palabras claves: Gestión de Proyectos de Software, Herramientas, Ingeniería de Software, Metodología.

1. Introduccion

La ingeniería de software propone desarrollar un sistema de software como un producto ingenieril, con un proceso basado en metodologías, técnicas, teorías y herramientas, que requieren de planeamiento, análisis, diseño, implementación, prueba y mantenimiento. Los métodos de desarrollo de software más usados en la actualidad adoptaron la Ingeniería de Software basada en modelos, es decir la construcción de modelos del sistema antes de la construcción misma del sistema, filosofía propuesta por Tom DeMarco [4]. De esta forma, el modelo de un sistema provee un medio de comunicación entre todos los participantes en el proyecto.

Los modelos de procesos de software, los métodos de ingeniería de software y las herramientas de software han sido adoptados con éxito y aceptación en el amplio espectro de las aplicaciones industriales. Tanto los usuarios como los ingenieros de software reconocen la necesidad de enfocar de manera más disciplinada el desarrollo de software [5, Cap.1]. A pesar de todos los esfuerzos realizados, en las sucesivas etapas del desarrollo y en las actividades de gestión de proyectos de software, ocurren problemas como: carencia de fiabilidad, necesidad de mantenimiento permanente, retrasos en las entregas y costos superiores a los

presupuestados [8,9]. En 1996, la firma internacional Standish Group Internacional, Inc realizó lo que se conoce como "The Chaos Report" [10], reportando que sólo un 16 % de los proyectos de software son exitosos, es decir, terminan dentro de plazos y costos y cumplen los requerimientos acordados, otro 53 % sobrepasa costos y plazos y cumple parcialmente los requerimientos y el resto no llega a término. Los sucesivos reportes (1996, 1998, 2000, 2003 y 2004) muestran que estas cifras siguen tomando importancia en los proyectos de software. En CEIS (Centro Experimental de Ingeniería de Software) [3] definen lo que llaman "peores prácticas y prácticas ausentes del desarrollo de software", entre ellas: no existe medición del proceso ni registro de datos históricos, rechazo a estimaciones imprecisas de plazos y costos, mal uso de herramientas automatizadas para la planificación y estimación, excesiva e irracional presión en los calendarios, crecimiento excesivo de los requerimientos para un producto de software, escaso o deficiente monitoreo en el progreso del proceso de desarrollo, no se hace formalmente la Gestión de Riesgos, no se realiza un proceso formal de pruebas y no se realizan revisiones técnicas formales e inspección de código.

Este trabajo propone una metodología para la enseñanza y el aprendizaje de temas vinculados a la Gestión de proyectos de software en un curso de Ingeniería de Software. La implementación de esta metodología, permite internalizar en el accionar del alumno la utilización de herramientas en las principales actividades de gestión de proyectos de software. De esta manera, se ubica al alumno en el rol de gerente de proyecto formando una actitud crítica hacia el logro de productos de software con las características de calidad deseadas.

Esta propuesta se fundamenta en que un curso de ingeniería de software debe brindar al alumno la posibilidad de solidificar los conocimientos teóricos aprendidos en la carrera, a través de su aplicación en proyectos reales y con un uso intensivo de herramientas que facilite su futuro desempeño profesional.

La metodología propuesta incorpora la realización de *talleres* a la modalidad tradicional de desarrollo del curso, dada por clases teóricas y prácticas. Estos *talleres* consisten en el análisis y aplicación de herramientas para asistir las actividades de gestión sobre un proyecto de software real.

Esta metodología de enseñanza, además de promover el uso de un proceso de desarrollo de software para obtener productos de calidad, fomenta la división de roles operativos y gerenciales entre los integrantes de un equipo de desarrollo, permitiendo que el alumno experimente diferentes responsabilidades en un proyecto de software real y visualice los beneficios aportados por la utilización de herramientas en cualquiera de las actividades desarrolladas por la gerencia de un proyecto de software.

El trabajo está organizado de la siguiente forma: la Sección 2 presenta una descripción de la metodología propuesta, que incluye la metodología de trabajo y los objetivos planteados. La Sección 3 describe la implementación y los resultados obtenidos con los talleres propuestos en el año 2006. Por último, la Sección 4, expone las conclusiones.

2. Descripción de la Metodología

La metodología de enseñanza descripta en este trabajo se aplica a un curso de Ingeniería de Software de tercer año de las carreras de computación de la Universidad Nacional de Río Cuarto. El principal objetivo de este curso es que el alumno tome conocimiento de conceptos básicos de ingeniería de software que van desde la planificación y Gestión de proyectos hasta técnicas de prueba, incluyendo otros métodos de desarrollo de software complementarios a los vistos por el alumno, como así también conceptos transversales a las etapas de desarrollo como el gerenciamiento de la configuración de software.

Esta metodología incorpora la realización de *talleres*, abordando específicamente el desafío de la utilización de herramientas que asisten en las actividades de gestión de proyectos de software [2]. El principal propósito es conseguir que los alumnos vivencien situaciones muy cercanas a la realidad y, de esta manera, disminuir la brecha entre la teoría universitaria y la realidad cotidiana.

Los objetivos de la propuesta son:

- Fomentar el uso de una metodología de desarrollo de software en proyectos reales.
- Promover la distribución de roles operativos y gerenciales entre los integrantes de un equipo de desarrollo de software.
- Capacitar a los alumnos para la selección y evaluación de diferentes herramientas aplicables a las actividades de gestión de software.

se divide en el dictado de clases teóricas, clases prácticas y talleres.

Cada clase teórica del curso son tomada por todo el grupo de alumnos, donde se imparte el contenido general de cada unidad temática y se desarrollan ejemplos concretos de casos particulares, que ayudan a los alumnos a reflexionar sobre la aplicación del conocimiento teórico en un caso práctico específico.

Para las clases prácticas, se divide el total de alumnos en grupos más pequeños. Se cuenta con una abundante ejercitación para cada unidad temática. Se utilizan problemas de creciente complejidad que exigen, en algunos casos, el uso de herramientas para obtener y visualizar los resultados. En cada clase práctica, el docente y los alumnos realizan la tarea de confrontar las diferentes soluciones planteadas para cada problema y se debaten las ventajas y desventajas de cada solución.

Los talleres son incorporados desde el año 2006, en la siguiente sección se detalla la aplicación de los mismos.

3. Talleres

La incorporación de talleres es el principal aporte de esta propuesta. Se plantean como una actividad de seguimiento y trabajo en equipo que tienen por finalidad potenciar en el alumno el proceso de aprendizaje y grupal. En los talleres, el alumno se enfrenta con la necesidad de:

(i) Intercambiarse e interrelacionarse con sus pares.

(ii) Realizar un uso intenso de herramientas para la automatización de las actividades de gestión de proyectos de software.
(iii) Valorar las ventajas que obtiene la gerencia del proyecto con la utilización de herramientas.

Por otro lado, los talleres favorecen la investigación, la interacción y la coordinación para el trabajo en equipo de los futuros profesionales.

Para el desarrollo de cada taller, cada grupo de alumnos debe:

- Evaluar las características de las herramientas propuestas, destacando que pueden hacer y lograr con su uso.
- Elegir y aplicar una de las herramientas evaluadas al problema presentado en el taller.
- Confeccionar un informe con los resultados obtenidos, mostrando el problema resuelto con la herramienta elegida y justificando la elección.

Algunas de las herramientas utilizadas para resolver los talleres son provistas por los docentes del curso, mientras que otras son aportes de los alumnos. En el Cuadro 2 se presenta una lista de las herramientas que surgen del la implementación de la propuesta en el año 2006.

3.1. Implementación de Talleres

Para la implementación de esta metodología en el año 2006, se asignó un proyecto real a cada equipo de alumnos y se desarrollaron cinco *talleres*. Los temas abordados se resumen en el Cuadro 1.

Una vez finalizadas las actividades solicitadas en el taller, cada equipo elaboró un informe detallando los resultados obtenidos, las herramientas utilizadas y las conclusiones arribadas.

3.2. Características de los proyectos

La totalidad de los proyectos utilizados para el desarrollo de los talleres en el año 2006, corresponden a sistemas que informatizan la gestión administrativa de una empresa y, por sus características similares, permiten garantizar que el nivel de complejidad de los mismos sea uniforme para los diferentes equipos de alumnos.

Estos proyectos fueron desarrollados siguiendo la metodología del Proceso Unificado de desarrollo de software [6]. Utilizan UML [1] como lenguaje de modelado estándar para los artefactos del proceso. La notación usada para modelar los procesos de negocio es BPMN (Business Process Modeling Notation) [7]. Además utilizan, el lenguaje de programación JAVA, en el ambiente de programación Eclipse, MySql como administrador de base de datos, la capa persistencia TriActive JDO (TJDO) y para la creación de reportes utilizan JasperReport.

De cada proyecto se dispone de la documentación completa, incluyendo el informe técnico con todos los artefactos construidos o utilizados en el proyecto, el manual del usuario y el código fuente.

- **I. Gestión de Proyectos de Software:**
  Identificar actores y sus roles, usuarios, clientes y otros. Identificar el Producto. Identificar las actividades realizadas del proceso de desarrollo del software. Enunciar los objetivos principales definidos para este proyecto, e identificar las funcionalidades (casos de uso) más interesantes que plantea el sistema para la concreción de dichos objetivos.
- **II. Estimación – Métricas:**
  Calcular el valor de punto de función del proyecto. Utilizar los tres valores de PF y calcular el esfuerzo según el modelo COCOMO.
- **III. Métricas de software OO:**
  Calcular métricas orientadas a clases: Árbol de Profundidad de Herencia (APH) y Número de Descendientes (NDD). Medir el tamaño de las clases, comparar los valores obtenidos y sacar conclusiones. Calcular el Número medio de parámetros por operación y analizar si el valor obtenido representa la mayoría de las operaciones incluidas o no.
- **IV. Planificación Temporal:**
  Realizar la planificación temporal del proyecto usando un Diagrama de Gantt. Construir un Diagrama de Pert para las actividades realizadas en la primera iteración del proyecto, para un caso de uso. Definir el camino crítico. Indicar si alguna actividad se puede retrasar y cuánto tiempo.
- **V. Prueba de Software:**
  Definir casos de prueba para los casos de uso de la primera iteración del proyecto. Realizar las pruebas utilizando una o más herramientas de caja negra. Seleccionar el código fuente del caso de uso que considere más crítico de la primera iteración y evaluar con una herramienta de caja blanca.

Cuadro 1. Talleres implementados en el año 2006

3.3. Evaluación de la propuesta

La evaluación de los resultados obtenidos con la implementación de *talleres*, es realizada por el grupo de docentes que dicta el curso, con aporte de alumnos, docentes de otras asignaturas, asesores pedagógicos y directivos.

Los instrumentos de recolección de información empleados son:

- Reuniones de discusión entre los docentes de la asignatura.
- Análisis del puntaje obtenido por los alumnos en los exámenes parciales.
- Análisis de eficiencia en los resultados obtenidos por los alumnos en el examen final de la asignatura.
- Evaluación de los informes de los talleres presentado por los alumnos.
- Entrevistas individuales a alumnos, auxiliares y docentes participantes.
- Entrevistas a grupos de alumnos de cursos superiores que realizan asignaturas correlativas.
- Reuniones de discusión con docentes de asignaturas correlativas.

Los alumnos participaron de manera activa y entusiasta, lo que permitió obtener buenos resultados. Para la evaluación de cada equipo, se utilizó una plantilla

- **Métricas (Punto de Función)**
  - Software Engineering Tiny Tools
    http://www.engin.umd.umich.edu/CIS/course.des/cis525/js/f00/harvey/FP_Calc.html
  - Computing Function Points
    http://www.engin.umd.umich.edu/CIS/course.des/cis525/js/f00/sekhar/fp.html
  - Function Oriented Metrics
    http://www.engin.umd.umich.edu/CIS/course.des/cis525/js/f00/artan/functionpoints.htm
  - Function Pointer Counter
    http://www.engin.umd.umich.edu/CIS/course.des/cis525/js/f00/jager/index.html
- **Métricas (CoCoMo)**
  - Costar http://www.softstarsystems.com/
  - USC COCOMO II
    http://sunset.usc.edu/research/COCOMOII/index.html
  - Basic Cocomo Model
    http://www.engin.umd.umich.edu/CIS/course.des/cis525/js/f00/gamel/introduction.html
- **Métricas OO**
  - SDMetrics
    http://www.sdmetrics.com/
  - Practice
    http://practise.cs.tut.fi
  - Analizar e Interpretar Métricas Orientadas a Objeto
    http://satc.gsfc.nasa.gov/support/STC_APR98/apply_oo/apply_oo.html
- **Planificación de Proyectos**
  - Continoplant
    http://www.continoplan.de/e/index.html
  - Critical Tools
    http://www.criticaltools.com/
  - Smartdraw
    http://www.smartdraw.com/exp/gan/home/
- **Pruebas de Software**
  - JUnit
    http://www.junit.org/index.htm
  - Donde buscar herramientas
    http://opensourcetesting.org/
- **SQA**
  - Tabla con link de herramientas
    http://www.sqa-test.com/download.html
    http://www.qualsearch.com/resources/tools.htm
- **Gestión de Configuración de Software**
  - WinCVS
    http://www.wincvs.org/
  - El cvs de Eclipse
    http://www.eclipse.org/
  - CS-CVS
    http://www.componentsoftware.com/products/cvs/

Cuadro 2. Herramientas propuestas para el año 2006

que indica los ítems a medir en cada taller, que fue completada por los docentes con el fin de llevar un control de la evaluación, confrontar los resultados obtenidos entre diferentes equipos e informar los resultados de la evaluación a los alumnos involucrados. A modo de ejemplo, en el Cuadro 3, se muestra la plantilla utilizada en el taller "Gestión de proyectos de software".

Grupo Nro:	
Integrantes:	
Taller 1: Gestión de proyestos de software	
Presentación:	(MB | B | R)
Personas:	(MB | B | R)
Producto:	(MB | B | R)
Proceso:	(MB | B | R)
Objetivos:	(MB | B | R)
Uso de herramienta:	(MB | B | R)
Comentarios:	.....
	.....
Devuelto:	(SI, fecha: ..... | NO). Reevaluado, fecha: .....

Cuadro 3. Plantilla de evaluación del taller "Gestión de proyectos de software"

3.4. Evaluación de resultados

Con los resultados obtenidos en la evaluación de los talleres durante el año 2006, se hicieron tres tipos de análisis que consideran: (i) la condición final de los alumnos en el curso, (ii) el rendimiento de los alumnos en los parciales y (iii) el examen final del curso.

i) Condición final de los alumnos

Un alumno se considera *regular*, si a superado la evaluación de los talleres y de los parciales. En el Cuadro 4, se presentan los resultados correspondientes a estas evaluaciones junto con las condiciones de regularidad de los alumnos, del cual se observa que del total de alumnos inscriptos en el curso, el 83 % regularizó mientras que solo el 17 % quedo en condición de libre, 2 de los cuales no agotaron las instancias de evaluación.

Otro punto que se observa, es que el total de los alumnos que asistieron aprobaron los talleres. Este factor fue motivante para los alumnos en el momento de preparase para los parciales (ver "eficiencia de los alumnos en los parciales" mas adelante).

Inscriptos	42 Alumnos		
Asistieron	40 Asistieron		2 No asistieron
Talleres	40 Aprobaron		
Parciales	35 Aprobaron	5 No aprobaron	
	83 % Regulares	17 % Libres	

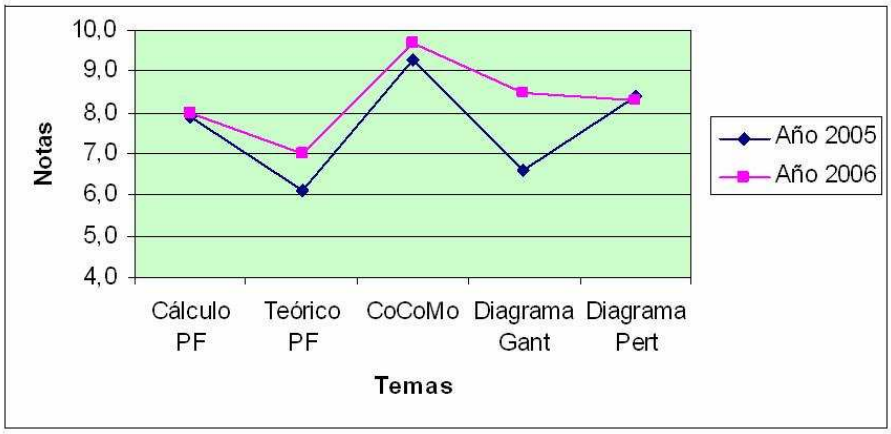
Cuadro 4. Alumnos regulares y libres en el año 2006

ii) Rendimiento de los alumnos en los parciales

El rendimiento de los alumnos en los parciales se obtiene midiendo los valores obtenidos en los parciales en las preguntas relacionados con los temas de los talleres.

El gráfico del Cuadro 5, muestra un análisis comparativo del rendimiento obtenido por los alumnos en los exámenes parciales en el año 2005 y 2006. En particular, se analizan los temas abordados en los talleres y permite deducir que en la mayoría de dichos temas los alumnos obtuvieron una mejor calificación en el año 2006.

Esta mejora en los exámenes parciales, se traduce en que los alumnos han afianzado y comprendido los temas abordados.

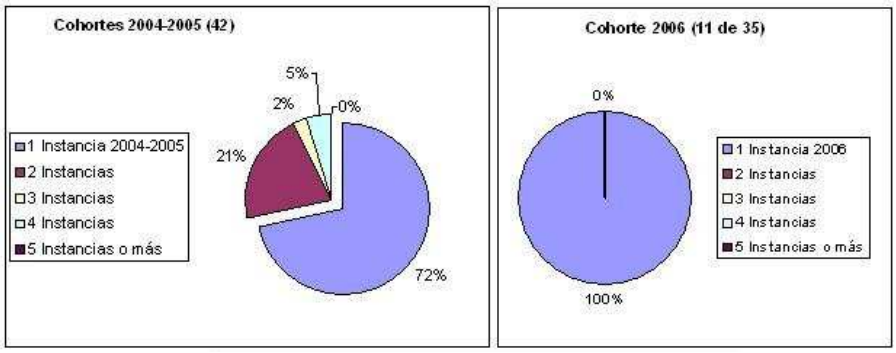


Cuadro 5. Puntajes obtenidos en los parciales de los años 2005 y 2006

iii) Examen final del curso

Respecto al análisis en los resultados obtenidos por los alumnos en el examen final de la asignatura, se recabaron los datos correspondientes a los años 2004 y 2005, y se los confrontó con los datos obtenidos del año 2006. Los gráficos del

Cuadro 6, muestran que de un total de 42 alumnos que cursaron la asignatura antes del 2006, un 72 % aprobó el examen final en la primera instancia, un 21 % necesito dos instancias para aprobar, un 2 % lo hizo en la tercera instancia y un 5 % necesitó rendir cuatro veces para lograr la aprobación final de la asignatura. De los 42 alumnos que cursaron la asignatura Ingeniería de Software en el año 2006, 35 regularizaron. De estos 35, los 11 alumnos que rindieron el examen final lograron su aprobación en la primera instancia.



Cuadro 6. % de Alumnos que aprueban el examen final en 1era. instancia o en más

Si bien estos datos aún no son sumamente relevantes para obtener conclusiones contundentes respecto del funcionamiento de la nueva metodología de enseñanza, ya se puede vislumbrar una notoria convergencia hacia un mejoramiento en el rendimiento final de los alumnos.

Respecto al futuro accionar profesional de los alumnos que cursaron la asignatura antes y durante el año 2006, resulta muy difícil de medir y solo se podrá obtener información subjetiva de la opinión que puedan dar los graduados.

4. Conclusiones

Los esfuerzos realizados para mejorar la enseñanza y el aprendizaje de los temas involucrados en el área de ingeniería de software, son bien recibidos si se realizan con una planificación clara y una inserción adecuada en la currícula de estos cursos. No basta con agregar horas de tediosas prácticas, en su lugar es conveniente pensar estrategias que incentiven a los alumnos a utilizar y aplicar los conceptos teóricos adquiridos en situaciones problemáticas reales.

La propuesta metodológica presentada en este trabajo, refuerzan en el accionar de los alumnos la aplicación de herramientas que permiten asistir en la gestión de proyectos de software, mejorando considerablemente la toma de decisiones gerenciales. Además, permite la adquisición de conductas profesionales

efectivas y ordenadas, hacia el uso y la búsqueda de elementos que colaboren con la obtención de software de alta calidad.

En este trabajo se presentan los resultados obtenidos en las evaluaciones realizadas sobre los alumnos. Se puede observar un alto porcentaje de alumnos que regularizaron el curso (83 %), una mejora en los resultados obtenidos en los temas propios de los talleres con respecto a años anteriores, y una proyección favorable en en la evaluación final de los alumnos.

Por otra parte, el interés que demostraron los alumnos y el equipo de docentes participantes en esta actividad, permitió mejorar la asignatura en su totalidad junto con la metodología de enseñanza y de aprendizaje en general.

Como conclusión final, se superaron las expectativas con respecto a los resultados esperados La realización de los talleres descriptos produjo un impacto positivo en el alumno, permitiendole adquirir la habilidad y la actitud para introducir estas actividades en su futuro accionar profesional.

Referencias

1. Grady Booch, Jim Rumbaugh, and Ivar Jacobson. *The Unified Modeling Language*. Addison-Wesley, 1999.
2. M. Daniele, D. Romero, P. Martellotto, and M. Frutos. La enseñanza de gestión de proyectos de software y la aplicación de herramientas que favorezcan su automatización. Technical report, 2006. Proyecto de Innovación e Investigación para el Mejoramiento de la Enseñanza (PIIMEG 2006). Finaciado por la Secretaría de Ciencia y Técnica y la Secretaría Académica de la Universidad Nacional de Río Cuarto. RR N° 499/06.
3. Centro Experimental de Ingeniería de Software. Available at: `http://www.ceis.cl/Gestacion/Gestacion.htm`.
4. T. DeMarco. Structured analysis and system specification. pages 409–424, 1979.
5. Carlo Ghezzi, Mehdi Jazayeri, and Dino Mandrioli. *Fundamentals of software engineering*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1991.
6. Ivar Jacobson, Grady Booch, and James Rumbaugh. *The unified software development process*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1999.
7. Object Management Group. Business Process Modeling Notation. Available at: `http://www.bpmn.org/`.
8. Roger S. Pressman. *Software Engineering: A Practitioner's Approach*. McGraw-Hill Higher Education, 2001.
9. Ian Sommerville. *Software engineering (5th ed.)*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1996.
10. Inc Standish Group Internacional. The chaos report, 1996. Available at: `http://www.standishgroup.com/`.