De la Universidad a la industria del software. Aplicación de nuevas tecnologías en proyectos finales.

Marcela Daniele, Marcelo Uva, Ariel Arsaute y Franco Brusatti

Departamento de Computación, FCEFQyN, Universidad Nacional de Río Cuarto, Río Cuarto, Argentina. Email:
{marcela,uva,aarsaute,fbrusatti}@dc.exa.unrc.edu.ar

Resumen Durante la última década, la relación Universidad e industria de desarrollo de software se ha vuelto cada vez más estrecha. Pequeñas, medianas y grandes empresas se acercan continuamente a las universidades en busca de recursos humanos calificados. El crecimiento y expansión del mercado informático ha propiciado la generación y adopción de nuevas metodologías de desarrollo, nuevas modalidades de trabajo como desarrollos outsourcing y freelance. Al mismo tiempo, han surgido nuevas tecnologías que brindan el soporte necesario para las actividades de administración, gestión, planificación, implementación, diseño, prueba y control en la producción del software. En este trabajo se presenta una experiencia desarrollada durante los años 2014 y 2015 en el marco de las asignaturas de Ingeniería de Sofware. La propuesta consiste en el desarrollo de un proyecto integrador que aplica un conjunto de herramientas y tecnologías de gran impacto en la industria del desarrollo de software actual, estableciendo un puente directo entre formación académica y necesidad del mercado.

Palabras Claves: Ingeniería de Software, Análisis y Diseño de Sistemas, Industria del Software y Scrum.

1. Introducción

Durante la última década, la relación de la Universidad con la industria del desarrollo de software se ha vuelto cada vez más estrecha. Pequeñas, medianas y grandes empresas se acercan continuamente a las universidades en busca de analistas, programadores, ingenieros de software, etc. El constante crecimiento y expansión del mercado informático, en todos los ámbitos, hace cada vez más visible la necesidad de contar con recursos humanos calificados. Paralelamente, dicho crecimiento ha propiciado en la industria, la generación y adopción de nuevas metodologías de desarrollo y conjuntamente, nuevas modalidades de trabajo como desarrollos outsourcing y freelance. Nuevas tecnologías han surgido con el objeto de dar el soporte necesario para las actividades de administración, gestión, planificación, implementación, diseño, automatización de la prueba, seguimiento y control de las tareas que hacen a la producción del

software. Muchas de estas tecnologías poco a poco han ido incorporándose en las currículas de las carreras de informática. De esta manera, se ha establecido una relación de necesidad entre Universidad e indrustria de desarrollo de sofware en ambos sentidos. En el marco de los proyectos Proyectos de Investigación e Innovación para el Mejoramiento de la Enseñanza de Grado (PIIMEG)[1] pertenecientes a la Universidad Nacional de Río Cuarto (UNRC), se han llevado adelante una serie de propuestas en pos de detectar, analizar y ejecutar acciones concretas con el fin de realizar aportes para solucionar problemáticas observadas en las asignaturas de tercer año de las carreras de Analista, Profesorado y Licenciatura en Ciencias de la Computación dictadas en la UNRC. En este trabajo se presenta una experiencia desarrollada durante los años 2014 y 2015 en el marco de las asignaturas de Ingeniería de Software. La propuesta consiste en el desarrollo de un proyecto donde, además de integrar los contenidos teórico-prácticos abordados en las asignaturas de tercer año, se incorporen un conjunto de tecnologías utilizadas en la industria del desarrollo de software actual. El resto del trabajo está organizado de la siguiente manera: en la sección 2 se presentan los fundamentos de la propuesta. La sección 3 presenta la metodología de desarrollo aplicada. En la sección 4 se presenta la propuesta, planificación, tecnologías y herramientas utlizadas. En la sección 5 se realiza una evalución de la propuesta, y finalmente, las conclusiones.

2. Fundamentos

Desde hace más de una década, en el marco de los proyectos PIIMEG[1], los autores de este trabajo, han realizado acciones tendientes a dar solución a diversas problemáticas identificadas en las carreras de computación dictadas en la UNRC. A continuación se presenta un breve resumen de las temáticas abordadas:

- En el año 2004, se propuso la definición y uso de plantillas genéricas para la descripción de Casos de Uso. El objetivo fue proporcionarle al estudiante una forma clara y concreta de describir las funcionalidades de un sistema a desarrollar mediante soluciones genéricas a problemas similares o recurrentes.
- En el año 2005, continuando en la misma lína de investigación, se realizó la definición de plantillas genéricas para cubrir las etapas siguientes de un proceso de desarrollo orientado a objetos.
- En el año 2006, se trabajó en temáticas vinculadas a la gestión de proyectos de software y la aplicación de herramientas que favorecieran su automatización, poniendo el foco en la mejora de las prácticas vinculadas con la gestión de proyectos de software.
- Durante el año 2007 se elaboró un proyecto de articulación de contenidos con todas las asignaturas de tercer año. Esto permitió a los estudiantes establecer una relación directa entre los contenidos estudiados en cada una se las asignaturas.
- A partir del año 2011 se investigaron y analizaron las causas que hacen que un importante número de estudiantes no logren cumplir con

la planificación temporal establecida para concluir sus proyectos finales de carrera [2].

El cuerpo docente, autor de este artículo, tiene a cargo el dictado de los cursos Análisis y Diseño de Sistemas e Ingeniería de Software, durante el tercer año de las carreras: Licenciatura y Profesorado en Ciencias de la Computación y Analista en Computación de la UNRC. La planificación y ejecución de procesos de enseñanza-aprendizaje para cursos de Ingeniería de Software, plantean un gran desafío a los docentes universitarios involucrados. La necesidad de una actualización dinámica de los contenidos tiene siempre como finalidad la formación integral del estudiante, no solo desde lo académico, sino con una fuerte capacitación en las tecnologías utilizadas en la industria.

3. Metodología de desarrollo aplicada al proyecto integrador

En el marco de las asignaturas de Ingeniería de Sofware, se presenta una visión general sobre las diferentes metodologías utilizadas en el de desarrollo de software[3,4]. Algunos de los ciclos de vida del software estudiados son: el lineal o secuencial, el método del Análisis Estructurado de Yourdon[5], desarrollo basado en componentes, diseño por contratos[6] de Bertrand Meyer, entre otros. Particularmente, se pone énfasis en el Proceso Unificado[7], como método de desarrollo orientado a objetos tradicional y en Scrum[8], como metodología de desarrollo ágil.
En este trabajo se presenta una modificación al proyecto de articulación de contenidos mencionado en la sección 2 meditante la incorporación de un conjunto de herramientas que la industria de desarrollo de software utiliza intensamente en la actualidad. Dichas tecnologías están intimamente ligadas a la aplicación de metodologías ágiles[9], especialmente con Scrum, metodología seleccionada por el equipo docente para el desarrollo de este proyecto integrador. Scrum es una metodología ágil que posibilita trabajar en ambientes muy cambiantes, permitiendo replanteamientos continuos. Por otro lado, reduce el tiempo de producción y de comercialización del producto, aporta un gran beneficio o valor agregado al cliente, minimiza los riesgos de desperdiciar esfuerzo/tiempo en la construcción de artefactos que no serán utilizados o que no son fundamentales para el cliente. Facilita también la comunicación entre todos los integrantes del proyecto. La documentación producida dentro de un proyecto Srum es relativa al usuario, dueño, producto y equipo. Scrum es un marco de trabajo basado sobre la premisa de que el equipo de desarrollo conocerá la mejor manera de resolver el problema que se le presenta. La reunión de planificación de cada conjunto de requerimientos a producir se describe en términos del resultado deseado, en lugar de un conjunto de criterios de ingreso, definiciones y tareas. Scrum se basa en una auto-organización, con un equipo multifuncional y sin líder (dentro del equipo). El equipo es apoyado por dos individuos quienes ocupan los roles de Scrum Master y Product Owner. El Scrum Master es una especie de entrenador para el equipo, su función es ayudar a los miembros del mismo a utilizar el marco que ofrece la metodología para conseguir un alto nivel de productividad. Mientras que

el Product Owner representa los negocios, clientes o usuarios. Éste, guía al equipo hacia la construcción del producto esperado. Los proyectos Scrum avanzan en orden a la definición de los sprints, que son las iteraciones que poseen una duración de entre dos y cuatro semanas. En el inicio de cada sprint, los miembros del equipo se comprometen a producir un cierto número de características que se enumeran en el artefacto conocido como Product Backlog del proyecto. Al final de cada sprint, cada funcionalidad (conocida como user story) debe estar codificada, probada e integrada a una versión demo del sprint anterior. Luego se realiza una revisión, y por último se presenta la nueva funcionalidad frente al Product Owner y las otras partes interesadas que proporcionarán información requerida para el siguiente sprint. Las iteraciones han de continuar hasta obtener el producto deseado. Como se puede observar de lo expuesto, Scrum establece una forma de trabajo en donde todo el equipo se auto-regula y en donde no hay una documentación establecida a priori. Cada equipo utilizará los elementos que le sean necesarios para poder llevar adelante el conjunto de user stories con el cual se ha comprometido. Algunos equipos podrán utilizar diagramas UML[10], otros utilizarán diagramas de flujos de datos, etc. En la figura 1 se esquematiza todo el proceso.

4. Descripción de la propuesta

La propuesta presentada en este trabajo consiste en el desarrollo de un proyecto integrador donde, además de integrar los contenidos teórico-prácticos estudiados en las asignaturas de Ingeniería de Sofware, se incorpore un conjunto de tecnologías y herramientas utilizadas en la industria del desarrollo de software actual, permitiendo establecer un vínculo estrecho entre el futuro egresado y las empresas de desarrollo de software.

4.1. Objetivos

A continuación se enumeran los objetivos de esta propuesta:

- Integrar en un proyecto de desarrollo de software los contenidos trabajados en tercer año, fundamentalmente, aquellos pertenecientes a las asignaturas de Ingeniería de Sofware.
- Aplicar una metodología de desarrollo ágil como Scrum. Aprovechando de esta manera los beneficios que conlleva la misma y a sabiendas de que es una de la más utilizadas por las empresas de desarrolo de software.
- Simular un ambiente de trabajo con características similares a las de un ambiente de trabajo real. Para ello se establecen objetivos grupales y es el mismo grupo quien se auto-gestiona. Si bien los objetivos son grupales, en el marco del proyecto integrador, se realiza un seguimiento y valoración particular de cada uno de los estudiantes, su integración con el resto, su compromiso, su forma de trabajo, etc.
- Desarrollar en los estudiantes habilidades para adoptar y obtener beneficios de un conjunto de herramientas utilizadas en la industria del software. El objetivo no es sólo que aprendan a usar una

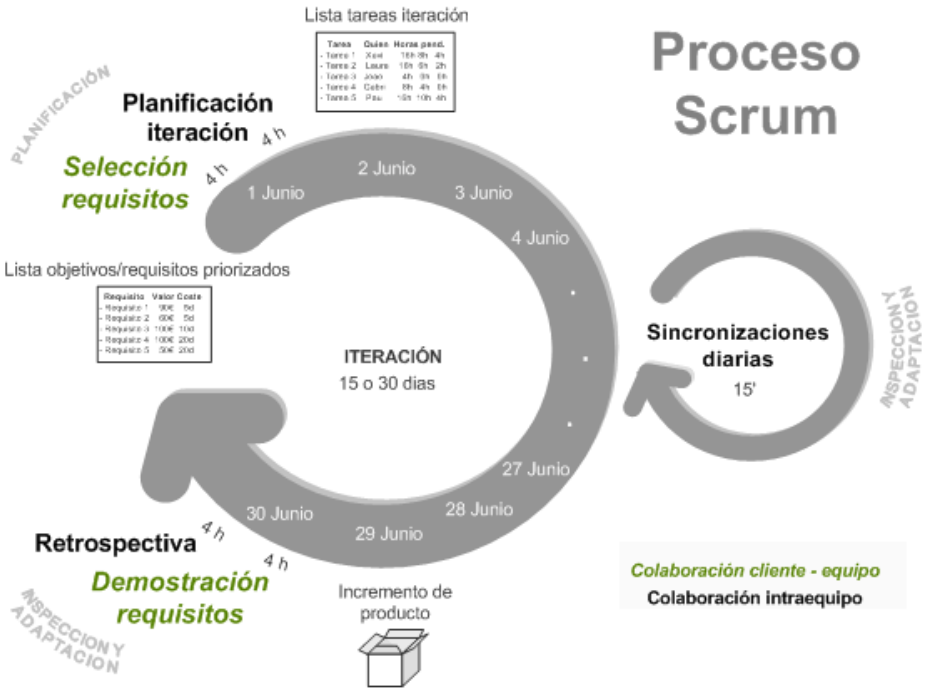

Figura 1. Proceso de desarrollo Scrum

herramienta, sino que sean capaces de seleccionar las herramientas adecuadas en futuros proyectos.

4.2. Organización y planificación

El poyecto integrador es planificado para ejecutarse durante el primer cuatrimestre del año. Cabe señalar que dentro del equipo docente de la asignatura, se ha seleccionado a un docente con dedicación part-time para la coordinación general del poyecto integrador. La selección de este docente fue motivada por la vinculación que el mismo posee con la industria de desarrollo de software. Este docente es fundador de una empresa de desarrollo de software local y está habituado a formar parte de equipos de trabajo con desarrolladores a nivel internacional en proyectos bajo la modalidad outsorcing y freelance. Uno de los objetivos del proyecto integrador es la aplicación de una metodología ágil, es por ello que el proyecto está guiado por Srum. En cada una de las fases del ciclo de vida, se aplica una herramienta específica. No es objetivo de este proyecto agregar complejidad en demasia ni sobrecargar tiempos de desarrollo. Sino que se focaliza en que cada grupo de estudiantes pueda completar el trabajo, usando las tecnologías propuestas.

Se formaron equipos de tres estudiantes, más el Scrum Master[8], rol ocupado por un docente de la asignatura.
El proyecto desarrollado en el año 2014 consistió en la construcción de un sistema web en donde un usuario pueda loguearse y publicar avisos de compra y venta de vehículos. En el año 2015, el proyecto desarrollado fue la construcción de una vesión del juego *Cuatro en línea*. El diseño del problema se planteó de una manera simplificada evitando el crecimiento del proyecto de manera descontrolada. El equipo docente estableció un Product Backlog inicial, y en la medida de que cada grupo avanzaba se incluían nuevas funcionalidades. Todos los grupos asistían a un encuentro semanal de dos horas donde se introducían las herramientas y se establecían los objetivos según la planificación correspondiente. Durante uno de los encuentros, los ayudantes alumno, con los que cuentan las asignaturas involucradas en este trabajo, elaboraron un taller sobre una de las herramientas seleccionadas. Esto permitió realizar aportes a su formación docente, y al mismo tiempo brindó una mirada diferente al aprovechamiento de estas tecnologías. En el resto de los encuentros, los docentes realizaron el seguimiento de cada grupo de estudiantes. A continuación se presenta, a modo de ejemplo, la planificación para el proyecto integrador llevado a cabo durante el año 2014.

- Marzo 18 - Presentación general del taller e introducción de Git.
- Abril 1 - SCRUM y Definnición del SRS (Sofware Requeriment Specification). PivotalTracker - Creación del Product Backlog.
- Abril 8 - Introducción a Java - Eclipse / ActiveJDBC / Postgresql.
- Abril 15 - Taller de Maven y JUnit.
- Abril 22 - Diseño / Implementación.
- Abril 29 - Diseño / Implementación.
- Mayo 6 - Taller de JUnit - definición de la Test Suite.
- Mayo 13 - Diseño / Implementación.
- Mayo 20 - Checkpoint (Test Suite running).
- Mayo 27 - Taller Spark - Sinatra e incorporación de la capa web.
- Junio 3 - Implementación
- Junio 10 - Implementación
- Junio 17 - Presentación de grupos

4.3. Tecnologías y herramientas utilizadas

La industria del software se encuentra en continua expansión. Esto requiere la incorporación de nuevas tecnologías para generar nuevas tecnologías. Sólo incorporando nuevas herramientas de soporte para estos procesos es que se logra producir sistemas para millones de usuarios, como por ejemplo las redes sociales.
Las herramientas utilizadas en este proyecto fueron las siguientes:

- Github[11]: plataforma de desarrollo colaborativo de software para alojar proyectos utilizando el sistema de control de versiones Git. El código se almacena de forma pública, aunque también se puede hacer de forma privada, creando una cuenta de pago.
- Git[12]: sistema de control de versiones distribuido, libre y gratuito.

- Java 8[13]: lenguaje de programación informática de propósito general que es concurrente, basado en clases y orientado a objetos.
- Postgresql[14]: sistema de base de datos objeto-relacional con características de los sistemas de base de datos propietarios tradicionales con mejoras de los sistemas de base de datos de la nueva generación.
- ActiveJDBC[15]: ORM (Object Relational Mapping) para desarrollo ágil. Es una implementación en el lenguaje Java del patrón arquitectural Active Record.
- JUnit[16]: framework para escribir tests (pruebas de software). Es una implementación de la arquitectura xUnix para unit testing.
- Spark[17]: framework inspirado en Sinatra para crear aplicaciones web con Java 8.
- Sinatra[18]: framework para aplicaciones web de software libre y código abierto con una DSL (Domain Specific Language) escrito en Ruby on Rail.
- PivotalTracker[19]: herramienta para la gestión de proyectos ágiles.
- Maven[20]: herramienta de software para la gestión y construcción de proyectos Java creada por Jason van Zyl, de Sonatype, en 2002. Es similar en funcionalidad a Apache Ant (y en menor medida a PEAR de PHP y CPAN de Perl), con un modelo de configuración de construcción más simple, basado en un formato XML.
- GanttProject[21]: herramienta para crear una completa planificación de un proyecto de forma visual. Todo queda bajo control en GanttProject, desde los recursos necesarios en forma de personal, los días festivos, hasta dividir el proyecto en un árbol de tareas y asignar a cada uno los recursos oportunos.

5. Evaluación de la propuesta y resultados

Para evaluar la propuesta, se tuvieron en cuenta los resultados obtenidos en cada uno de los proyectos desarrollados por los estudiantes, los aportes de los docentes involucrados, y también, se generó una encuesta a fin de reflejar el trabajo individual y grupal, la adopción de las nuevas tecnologías y la aplicación de Scrum. La encuesta fue compleatada por 45 estudiantes que cursaron las asignaturas de Ingeniería de Software durante los años 2014 y 2015. En la tabla 1 se presenta la encuesta brindada a los estudiantes. Luego de realizar y analizar las encuestas, y evaluando el desempeño de los grupos es posible destacar lo siguiente: Todos los grupos se mostraron muy entusiasmados e interesados con la propuesta. Algunos grupos lograron ampliamente los objetivos iniciales (85 %) y a otros les resultó más costoso lograr adaptarse a las nuevas herramientas. La modalidad del taller, exige tiempo de investigación fuera del horario de clases. Un par de grupos tuvieron un desempeño sobresaliente, obligando al cuerpo docente a proponer nuevas funcionaliades y nuevas herramientas. La mayoria de los estudiantes desconocían herramientas como Git, JUnit, Pivotal Tracker, ActiveJDBC. Si bien muchas de ellas son citadas en las asignaturas de Ingeniería de Software, fue dentro de este proyecto el primer contacto con las mismas. Un gran porcentaje de los grupos expresa que se siente capaz de escalar esta experiencia a un proyec-

to de dimensiones mayores, aplicando las mismas tecnologias. El trabajo en grupo fue altamente positivo. Se pudo observar el desempeño de cada uno de los integrantes de cada grupo ayudado por una de las herramientas utilizadas (Git). Si bien, desde el cuerpo docente se realizaba un seguimiento particular de cada estudiante, se le otorgó al grupo autonomía para trabajar, de acuerdo a lo establecido por Scrum. En la mayoría de los casos, ellos mismos se dividieron las tareas, por ejemplo cada uno se dedicaba a investigar una herramienta particular para luego mostrarla al resto del grupo. Git fue una de las herramientas más resistidas en un comienzo. Esto, en parte se debe a que Git establece una nueva modalidad de trabajo, en particular para trabajos distribuidos. En un comienzo provocaba problemas en muchos grupos, hasta que pudieron adoptarla, y finalmente sacar a flote los beneficios del versionado establecido por Git. Con respecto a la metodología de desarrollo utilizada Scrum, todos los grupos pudieron seguirla sin inconvenientes, comprendieron los roles dentro del proceso, sus obligaciones como parte de un equipo de desarrollo y lograron producir el software acordado.

Tabla 1 - Encuesta de finalización de cuatrimetre 3er. año

1. ¿Cúales fueron las materias que cursaste en este cuatrimestre ?
2. ¿Cuáles regularizaste?
3. ¿Te pareció interesante la propuesta del taller? Si/No/Parcilmente. Justifique en cada caso.
4. ¿Cúantas horas semanales promedio le dedicaste al taller?
5. ¿Se comprendió el objetivo del taller al momento de presentarlo? Si/No/Parcilmente
6. ¿Habías realizado algún proyecto de materia utilizando alguna metodología de desarrollo? Si(Cuál)/ No
7. ¿ Todos los integrantes del grupo trabajaron de igual manera? Si/No/ En ocasiones
8. Que porcentaje del total aportaste al proyecto.
9. En general, como calificas la experiencia de trabajo grupal. Excelente/Buena/Regular/Mala
10. Del 10 al 1, cuanto conocías de las herramientas antes de comenzar con el taller:
a)Lenguaje de programación Java b)IDEs tales como Eclipse c)Netbeans d)Maven e) Spark web framework f)Git g)ActiveJDBC
11. Describe muy brevemente parqué utilizaste cada una de las herramientas anteriores.
12. Del 10 al 1 califica la utilidad de cada una de las herramientas
utilizadas respecto a tu experiencia en este cuatrimestre.
13 ¿Cuáles fueron las herramientas que te resultaron más
fáciles de adoptar y cuáles fueron las más difíciles.?
14. Con respecto a las otras asignaturas de 3ero. ¿Pudiste relacionarlas con el taller ?
a - Base de datos Si/No/Parcilmente
b - Diseño de Algoritmos Si/No/Parcialmente
15. ¿Tuvieron problemas con el tiempo y las entregas parciales?
16. Te sentis capaz de encarar un proyecto similar a mayor escala.

6. Conclusiones

La propuesta realizada en este trabajo, se fundamenta en el desarrollo de un proyecto integrador de ingeniería de software, sobre un dominio particular, con estudiantes de tercer año de las carreras de computación. Fundamentalmente, la propuesta está centrada en integrar todos los conocimientos adquiridos por el estudiante, y con el fuerte propósito de que se analicen y seleccionen un conjunto de herramientas altamente utilizadas en la actualidad en la industria de software. Con claridad, puede verse que los resultados obtenidos son muy positivos. Esta propuesta permite una capacitación más completa e integral del estudiante, en virtud de que este, pueda integrar fuertemente los conceptos teóricos con los prácticos, visualizar el comportamiento y reacción del mercado ante las nuevas tecnologías, y reflexionar su impacto. Los objetivos planteados en la propuesta se consiguen en su totalidad, y en muchos casos, superan las expectativas que definió el equipo docente. Ello en el sentido, de que algunos estudiantes presentan tanto interés y dedicación, que alcanzan a superar los objetivos y proponen nuevos desafíos que sirven tanto para mejorar sus proyectos, como para una superación de este trabajo. Aplicando esta propuesta, se consigue que el estudiante se convierta en un actor más activo de su propio proceso de enseñanza y aprendizaje, a partir de la investigación a la que se somete para determinar las ventajas y desventajas de las herramientas y técnicas seleccionadas para el desarrollo de su proyecto. Luego, el estudiante es capaz de analizar y seleccionar las herramientas que más se adecúen al proceso de automatización bajo su desarrollo. También, es muy importante destacar que la simulación del desarrollo de un proyecto de software real, con un ambiente de trabajo en equipo y colaborativo, favorece a que los estudiantes se auto-gestionen, distribuyan sus tareas y tiempos, analicen riesgos, y que puedan ser capaces de aplicar acciones correctivas para controlar las desviaciones en la planificación original. Siendo los docentes, los encargados de realizar un seguimiento muy cercano de cada grupo, controlando permanentemente que se adecúen a la planificación prevista y evitando que el proyecto adquiera complejidad en demasia o que crezca en tamaño de forma descontrolada, ya que no es objetivo de este trabajo aumentar su carga horaria en perjuicio de su rendimiento académico general en la carrera. En este año, estamos trabajando en recabar información de los estudiantes, muchos de ellos actualmente egresados, que en 2014 y 2015 desarrollaron el proyecto integrador de las asignaturas de ingeniería de software bajo esta propuesta, y que actualmente trabajan en la industria del software, con el fin de que nos puedan contar sus experiencias y realizar mediciones de cuanto los favoreció la aplicación de esta metodología. También pretendemos contactarnos con empresas de desarrollo empleadoras donde trabajen nuestros recientes graduados, y obtener su opinión, expectativas y sugerencias para mejorar la formación de los actuales estudiantes.

Referencias

1. Proyectos de Innovación e Investigación para el Mejoramiento de la Enseñanza de Grado (PIIMEG), Secretarías de Ciencia y Técnica, y Académica, UNRC.:

- M. Daniele. D. Romero. Definición y uso de Plantillas Genéricas para la descripción de Casos de Uso. RR N° 302/04. 2004.
- M. Daniele, D. Romero. Evolución de Plantillas Genéricas para la descripción de Casos de Uso a Plantillas Genéricas para Análisis y Diseño. RR N° 109/05. 2005.
- M. Daniele. D. Romero. La enseñanza de gestión de proyectos de software y la aplicación de herramientas que favorezcan su automatización. RR N° 499/06. (01/08/2006, 31/07/2008).
- M. Daniele. F. Zorzan. Estimación y Planificación de Proyectos de Software versus duración de proyectos finales en la carrera Analista en Computación. RR N° 171/11. (2011, 2012).
- M. Daniele. F. Zorzan. Causas que producen que los estudiantes de Computación retrasen la culminación de su Trabajo Final. RR N° 923/12. (2013,2014).

2. Fabio Zorzan, Mariana Frutos, Ariel Arsaute, Marcela Daniele, Paola Martellotto, Marcelo Uva, Carlos Luna "Delayed Completion of Final Project of the Career Computer Analyst: Seeking its Causes". XX Congreso Iberoamericano de Educación Superior (CIESC 2012), en el Marco de la XXXVIII Conferencia Latinoamericana en Informática – CLEI 2012 - Octubre 1 al 5 de 2012 - Medellín, Colombia. ISBN 978-1-4673-0792-5.
3. Roger S Pressman. Libro: Software Engineering: A Practitioner's Approach. 8th Edition. McGraw-Hill Education. 2014.
4. An Integrated Approach to Software Engineering. Pankaj Jalote. Springer 2006.
5. E. Yourdon. Libro: Análisis estructurado moderno. Prentice-Hall Hispanoamericana, 1993.
6. Meyer Bertrand. Object Oriented Software Construction. Prentice Hall. 1997.
7. Ivar Jacobson, Grady Booch, James Rumbaugh. Libro: El proceso unificado de desarrollo de software. The Addison-Wesley, 2000
8. Scrum in Action: Agile Software Project Management and Development. Andrew Pham, Phuong Van Pham. Course Technology Ptr. 2011.
9. Highsmith Jim. Agile Software Development Ecosystems. Addison-Wesley 2002. ISBN:0201760136
10. Grady Booch. Libro: The Unified Modeling Language User Guide. The Addison-Wesley, 2005.
11. https://github.com
12. https://git-scm.com
13. http://www.oracle.com
14. http://www.postgresql.org.es
15. http://javalite.io/activejdbc
16. http://junit.org
17. http://sparkjava.com
18. http://www.sinatrarb.com
19. https://www.pivotaltracker.com
20. https://maven.apache.org
21. https://gantt-project-management.com