PROPUESTA PARA LA ENSEÑANZA DE INGENIERÍA DE SOFTWARE EN UN GRUPO DE UNIVERSIDADES PRIVADAS DE LATINOAMÉRICA

Marisa Cecilia Tumino
Juan Manuel Bournissen
Universidad Adventista del Plata, Argentina
Raquel Anaya
Jhon Fredy Niño
Corporación Universitaria Adventista de Colombia, Colombia

RESUMEN

El proyecto busca definir un marco de referencia que apoye la enseñanza de la ingeniería de software en los programas de informática de las universidades a nivel latinoamericano. Se parte de un modelo de enseñanza en contexto de la ingeniería de software, basado en la teoría de la actividad, que tiene tres dimensiones: (a) aprendiz, (b) cuerpo de conocimiento y (c) contexto. Estas dimensiones se trabajan en términos de (a) motivación del estudiante, (b) competencias profesionales y (c) estrategias didácticas, conformando así un escenario pedagógico. El trabajo toma como referente el cuerpo de conocimiento de la ingeniería de software. Se definen escenarios de aprendizaje que buscan articular actividades de aprendizaje centrales del desarrollo de software con estrategias pedagógicas innovadoras que resultan de la revisión de los referentes y de las experiencias de docentes entrevistados. El proyecto finaliza con los resultados obtenidos de la puesta en marcha de un escenario de aprendizaje en el marco de un proyecto piloto.

Palabras clave: ingeniería de software, estrategias de enseñanza, perfil del estudiante informático

Introducción

La investigación acerca de la enseñanza de la ingeniería de software ha estado en plena evolución (Ghezzi y Mandrioli, 2005; Nauman y Uzair, 2007), por tratarse de un reto complejo en el cual se involucran aspectos de tipo técnico, de gestión de proyectos y de interacción social (Bavota, De Lucia, Fasano, Oliveto y Zottoli, 2012). De acuerdo con lo que mencionan Ng y Huang (2013), aplicado a lo que ocurre en China, el rápido crecimiento de la población de desarrollo de software representa un gran desafío para las universidades, lo que explica el interés en esta iniciativa.

A partir del presente estudio se pretende obtener una aproximación de los factores involucrados en la enseñanza de la ingeniería de software que permitan optimizar la formación de profesionales de este campo de conocimiento cada vez más prominente para la sociedad. Para

Marisa Cecilia Tumino y Juan Manuel Bournissen, Universidad Adventista del Plata, Argentina.
Raquel Anaya y Jhon Fredy Niño, Universidad Adventista de Colombia, Colombia.
La correspondencia concerniente a este artículo puede ser enviada a Marisa Cecilia Tumino, correo electrónico: marisa.tumino@uap.edu.ar

ello se procede a enmarcar el escenario pedagógico desde tres elementos clave involucrados en el proceso de enseñanza-aprendizaje: el aprendiz, el objeto de aprendizaje y el contexto, analizados desde la teoría de la actividad (Anaya y Trujillo, 2005) y traducido en términos de (a) motivación del estudiante, (b) competencias profesionales y (c) estrategias didácticas. A lo largo del trabajo se describen estos elementos como andamiaje sobre el que se construye el proceso formativo.

Es importante señalar que en virtud de que en la investigación participaron 15 universidades latinoamericanas que pertenecen a la Red Latinoamericana Adventista de Carreras de Informática (RELACI), los resultados representan una aproximación a propuestas que pretenden mejorar la calidad y eficacia de los programas de enseñanza de ingeniería de software.

Proceso de enseñanza y aprendizaje: escenario pedagógico

Desde la iniciativa de Vygotsky en 1978 (Larripa y Erausquin, 2008), se impulsaron y consolidaron las demandas por conceptualizar nuevas unidades de análisis psicológico que dieran al contexto un papel central en la constitución y explicación del comportamiento humano. La primera generación de la teoría de la actividad se basa en la idea vygotskyana de mediación cultural, concibiéndose a toda acción humana mediada por instrumentos y orientada hacia determinados objetos. El individuo no podría en lo sucesivo ser entendido sin sus medios culturales.

El proceso de enseñanza y aprendizaje puede ser abordado desde la teoría de la actividad (Leontiev, 1981; Talízina, 1988), que interpreta la construcción gradual del conocimiento a partir de la actividad del estudiante que lo relaciona con los objetos del mundo (el contexto) al interactuar con los demás. De acuerdo con Leontiev, se entiende por actividad aquel conjunto de acciones que ejecuta el sujeto, acompañadas por una motivación intrínseca que lo lleva a desarrollarlas activamente.

El proyecto pretende dar continuidad a trabajos de la enseñanza en el contexto de la ingeniería de software, en la cual se establece un modelo conceptual de enseñanza y aprendizaje de la ingeniería de software basado en la teoría de la actividad, acompañada de una propuesta curricular cuyo enfoque está orientado hacia actividades de aprendizaje en lugar de contenidos temáticos (Anaya, 2006; Anaya y Trujillo, 2005). El modelo conceptual que se toma como referencia en este trabajo identifica tres elementos clave involucrados en el proceso de enseñanza-aprendizaje que conforma el escenario pedagógico: el aprendiz, el objeto de aprendizaje y el contexto, analizados desde la teoría de la actividad (Anaya y Trujillo, 2005). Estas dimensiones se trabajan en términos de (a) motivación del estudiante, (b) competencias profesionales y (c) estrategias didácticas, conformando así un escenario pedagógico. Para dar orientación al trabajo, en primer lugar se procedió a indagar los aspectos de la carrera en cuestión que inciden sobre la motivación de los estudiantes.

Motivación de los estudiantes

Álvarez Álvarez, González Mieres y García Rodríguez (2007) comprobaron que una motivación adecuada y una revisión de los métodos de evaluación pueden favorecer el logro del aprendi-

zaje significativo de los estudiantes. Es por ello que el trabajo mantiene, como uno de los focos, la identificación de los niveles de motivación de los estudiantes que participan del estudio.

López Fernández, Alarcón Cavero, Rodríguez Sánchez y Casado Fuente (2014) analizaron diversos instrumentos para evaluar aspectos relativos a la motivación de los estudiantes universitarios, reconociendo la necesidad de construir cuestionarios específicos para el caso de los estudiantes de ingeniería. Luego de presentar una descripción de seis principales teorías de la motivación (teoría de las necesidades de McClelland, teoría de la equidad de Adams, teoría X-Y de McGregor, teoría de las expectativas de Vroom, teoría de la fijación de metas de Locke y teoría del factor dual), los autores presentan un instrumento diagnóstico basado en las seis teorías seleccionadas para evaluar el nivel de motivación de los estudiantes de titulaciones de ingeniería. El cuestionario propuesto quedó compuesto por 51 ítems basados en los indicadores motivacionales encontrados.

Dicho cuestionario fue el instrumento inicial de donde se partió para la construcción del instrumento de medición de la motivación en el presente estudio. Se realizó la validación de contenido con la participación de cinco expertos que evaluaron los ítems, en cuanto a la claridad y a la pertinencia y propusieron la eliminación y la introducción de otros ítems que atendieran a los objetivos del trabajo. Una vez introducidas las modificaciones el instrumento quedó conformado por 40 ítems, utilizando una escala tipo Likert de cinco puntos: de *muy en desacuerdo* (1) a *muy de acuerdo* (5).

Cuerpo de conocimiento: Competencias profesionales de la IS

El enfoque de competencias es una de las estrategias facilitadoras de la globalización en el campo laboral, la movilidad de estudiantes y trabajadores, la calidad de procesos y productos y la competitividad empresarial.

La formación en el desarrollo de software puede verse como un proceso en espiral en donde el aprendiz va adquiriendo las competencias cognitivas, prácticas y actitudinales necesarias no solo para fortalecer el proceso de apropiación progresiva de las diversas áreas de conocimiento relacionadas (ingeniería básica, organizaciones, de comunicaciones, tecnología, entre otras), sino también para lograr la madurez cognitiva y emocional del aprendiz. El modelo conceptual que se toma como referencia en este trabajo identifica tres elementos clave involucrados en el proceso de enseñanza-aprendizaje: el aprendiz, el objeto de aprendizaje y el contexto, analizados desde la teoría de la actividad (Anaya y Trujillo, 2005). Tal como lo define Oliveros (2006), la gestión por competencias es un sistema integrado de evaluación y mejora en el que se distinguen tres momentos: la identificación, el desarrollo y la evaluación de competencias, siendo la identificación de las competencias la base sobre la cual se sustentan los otros dos elementos.

Las competencias profesionales pueden verse desde dos perspectivas: (a) en función de un puesto de trabajo, donde se analizan las competencias de una persona que está desempeñando un trabajo, o (b) en función de un perfil profesional, donde se analizan las competencias de un sujeto en formación. La identificación de las competencias de un plan de estudio es una tarea ardua y compleja,

debido a la formación progresiva a lo largo del programa y a la dificultad de simular escenarios reales de trabajo en un contexto académico (Oliveros, 2006).

A los efectos de definir las competencias, se partió de la lista sugerida por Tumino, Bournissen y Barrios (2016), identificando las competencias genéricas y las competencias específicas en un segundo nivel. El mapa de competencias fue complementado con la propuesta curricular de ingeniería de software de la ACM/IEEE (Ardis et al., 2015; Fuente et al., 2015).

Contexto: estrategias de enseñanza

Tal como se mencionara previamente, la investigación acerca de la enseñanza de la ingeniería de software ha estado en plena evolución (Ghezzi y Mandrioli, 2005; Nauman y Uzair, 2007), por el reto que implica el complejo involucramiento de aspectos de tipo técnico, de gestión de proyectos y de interacción social (Bavota, De Lucia, Fasano, Oliveto y Zottoli, 2012). Entre las estrategias propuestas para enfrentar este reto se pueden destacar las siguientes: (a) el enfoque de aula invertida (Gannod, Burge y Helmick, 2008; Lockwood y Esselstein, 2013), en el que, como su nombre lo indica, se invierten las actividades de un proceso de enseñanza tradicional: fuera de clase los estudiantes se dedican a la lectura del material, utilizando diversos recursos, y en clase se dedican a actividades prácticas de aprendizaje activo; (b) el enfoque basado en proyectos, en los cuales se simulan escenarios de desarrollo de la vida real, con estudiantes de diferentes niveles de formación como pregrado y posgrado (Bavota et al., 2012) o con estudiantes geográficamente distribuidos que permiten simular la realidad del desarrollo global de software (Nordio et al., 2011); (c) uso innovador de la tecnología en el salón de clase (por ejemplo realidad virtual) para facilitar el aprendizaje activo y colaborativo (Razmov y Anderson, 2006) y (d) uso de juegos que buscan que los estudiantes desarrollen, en un ambiente lúdico, habilidades analíticas, técnicas y de colaboración, entre otras (Baker, Oh Navarro y van der Hoek, 2005; Cagiltay, 2007; Hwang, Wu y Chen, 2012).

La reducción del componente presencial, acompañado de la acentuación sobre la carga de trabajo autónomo, revela un cambio en el proceso de enseñanza-aprendizaje que se da en las universidades. El cambio implica un mayor compromiso del alumno con su aprendizaje (Núñez, Solano, González-Pienda y Rosario, 2006). Sin embargo, los resultados de algunas investigaciones, como las mencionadas por Murillo Torrecilla (2003), muestran que la calidad de la enseñanza es un factor fundamental para promover el progreso y el desarrollo de los alumnos, quienes avanzan más cuando los profesores los estimulan y muestran entusiasmo. El autor señala, como especialmente útiles, las afirmaciones y preguntas de orden superior y la insistencia de los docentes en involucrar a los alumnos en estrategias de resolución de problemas. Según Henson y Eller (2000), la investigación indica, entre otras cosas, que el docente eficiente desempeña un papel central en el aula, pero incluye a sus alumnos en la planeación y la organización, lleva la lección a un ritmo ágil en el que requiere la participación abierta y pública de sus alumnos, critica poco, moldea las respuestas de los estudiantes de forma que sean correctas, crea en sus alumnos la responsabilidad con el trabajo y los atiende de manera equitativa. El concepto de estrategias de

enseñanza es definido por Anijovich y Mora (2009) como

> el conjunto de decisiones que toma el docente para orientar la enseñanza con el fin de promover el aprendizaje de sus alumnos. Se trata de orientaciones generales acerca de cómo enseñar un contenido disciplinar considerando qué queremos que nuestros alumnos comprendan, por qué y para qué. (p. 4)

Las autoras afirman que las estrategias de enseñanza que un docente elige y utiliza inciden en (a) los contenidos que transmite a los alumnos, (b) el trabajo intelectual que los alumnos realizan, (c) los hábitos de trabajo, (d) los valores que se ponen en juego en la situación de clase y (e) el modo de comprensión de los contenidos sociales, históricos, científicos, artísticos y culturales, entre otros. Asimismo mencionan "el concepto espiralado de estrategia de enseñanza, que está compuesto por momentos de reflexión y por momentos de acción que interactúan entre sí" (p. 117). Las estrategias que un docente utiliza en una clase son pensadas en forma consciente e intencional con un propósito determinado.

Las estrategias de enseñanza suponen un alto grado de complejidad y necesitan medios para implementarse, además del "control y evaluación de los propósitos que se persiguen al aplicarlas" (Martínez Verde y Bonachea Montero, 2006, p. 3).

En estudios como los de Ng y Huang (2013), se aplicaron estrategias para el tratamiento del ciclo de vida del software. Reconocen que el rápido crecimiento de la población china de desarrollo de software representa un gran desafío para las universidades, lo que explica sus intereses en la iniciativa. El método consistió en dividir a los participantes en grupos pequeños y resolver ejercicios prácticos siguiendo pasos estratégicos de análisis. Sostienen que las universidades no pueden enseñar todo lo que la industria requiere, pero pueden proporcionar a los estudiantes una comprensión sólida de los fundamentos, brindándoles las herramientas para aprender y comprender la diversidad de la ingeniería de software que más adelante aplicarán en su carrera. El método proporciona los fundamentos y la manera de pensar para hacer frente a las cambiantes necesidades de la industria.

Por su parte, Dai, Wei, Wang y Wong (2017) investigaron el impacto de la educación basada en resultados (OBE) en el rendimiento de aprendizaje de los estudiantes de un programa de ingeniería de software. La educación basada en resultados goza de los siguientes beneficios: (a) claridad (los maestros saben lo que van a enseñar en los cursos y los estudiantes tienen objetivos claros que deben alcanzarse después de tomar cursos), (b) flexibilidad (los maestros tienen la libertad de elegir cualquier método para enseñar a los estudiantes) y (c) participación, puesto que se espera que los estudiantes participen activamente en las actividades de aprendizaje en lugar de memorizar pasivamente conocimientos como con los métodos de educación tradicionales. En su estudio, los autores concluyen que la educación basada en resultados puede mejorar sustancialmente la efectividad del aprendizaje de los estudiantes y la calidad de la enseñanza.

Considerando que las estrategias inciden en la comprensión de los contenidos, es esperable que tengan un impacto sobre el rendimiento académico de los estudiantes.

Existen muchos trabajos sobre las mejores prácticas en estrategias de

enseñanza y evaluación (Bain, 2007; Davini, 2008; Finkel, 2008; Martin-Kniep, 2000; Sanjurjo y Rodríguez, 2009; Sanjurjo y Vera, 2006; Strong, Silver y Perini, 2001). Según Joyce, Weil y Calhoun (2000), la forma en que se desarrolla una clase tendrá una enorme influencia en el modo en que los estudiantes concreten su propio aprendizaje. "Los buenos docentes no son simplemente expositores carismáticos y persuasivos" (p. 29). El docente cumple un rol que promueve en el estudiante un trabajo cognitivo y social que se torne en tareas productivas. Es importante reforzar la comprensión acerca del rol facilitador del docente en el proceso de aprendizaje de sus alumnos. El sistema educativo requiere una intervención docente significativa en pro del logro académico de los estudiantes. Las prácticas docentes forjan en muchos casos la percepción del alumno sobre las asignaturas y sus contenidos académicos. El rol del docente como facilitador del aprendizaje asume gran importancia en la formación profesional de los estudiantes. Se requieren estrategias que construyan escenarios propicios para el autoaprendizaje y pensar para ello en aquellas pedagogías que alienten los procesos de pensamiento de orden superior. Anijovich y Mora (2009) sostienen que resultan apropiados algunos principios para tener en cuenta en el momento de planificar las estrategias de enseñanza con el fin de promover aprendizajes significativos: (a) acordar con los alumnos metas de aprendizaje precisas y explícitas, (b) crear situaciones que requieran del uso del conocimiento de los conceptos, de los fenómenos, principios, de las reglas y los procedimientos de las disciplinas en diferentes contextos, (c) plantear la producción de tareas genuinas y de problemas reales propios de las disciplinas, (d) orientar hacia el uso de materiales y fuentes variadas para producir distintos tipos de comunicaciones, (e) desafiar a los alumnos con tareas que vayan más allá de sus habilidades y sus conocimientos, (f) estimular la producción de soluciones alternativas, (g) promover el desequilibrio cognitivo y la sana cautela respecto de la consideración de las verdades establecidas, (h) elaborar dispositivos de diferenciación: según el contenido, según los aprendices, según el contexto, (i) favorecer diferentes usos del tiempo, de espacios y de las formas de agrupamiento, y (j) promover la evaluación continua: la autoevaluación, entre pares, la del docente, escrita y oral, entre otras, que a su vez involucre instancias de metacognición.

De estas declaraciones se infiere claramente que las estrategias de enseñanza deben mantener una vinculación o consistencia con las estrategias de evaluación.

Considerando que el enfoque basado en proyectos será la principal estrategia pedagógica en el área, se realizó la caracterización de los tipos de proyecto que los estudiantes pueden desarrollar en las actividades curriculares. En términos generales, los proyectos se definen según su origen (a partir de un problema, de un diseño, de la necesidad de un cliente o de una idea innovadora) y el número de participantes. Cada docente orienta el proyecto conforme a los objetivos formativos propuestos.

Planteamiento del problema

Se advierten los siguientes problemas: (a) falta de motivación en los estudiantes de ingeniería informática con respecto al área de ingeniería de software, (b) ausencia de una visión unificada de las competencias profesionales que deben ser desarrolladas en un

ingeniero de software, (c) problemas trabajados en el aula de clase aislados de las problemáticas reales que se tienen en la industria, (d) falta de que los contenidos temáticos de las asignaturas del área de ingeniería de software hayan sido analizados con respecto a las tendencias de currículos internacionales sugeridos por la ACM y la IEEE y (e) falta de capacitación por parte de algunos de los docentes del área de software para dirigir algunos de los temas de actualidad en el área de ingeniería de software.

Un problema recurrente que se advierte, desde el comienzo de la carrera, reside en la falta de criterio del estudiante para seleccionar el ciclo de vida más adecuado para cada situación y, comúnmente, cuando los contenidos se imparten mediante exposición teórica, se corre el riesgo de que simplemente los memorice al solo efecto de cumplir con un requisito académico. A partir de los factores que inciden en la motivación de los estudiantes y de la definición de competencias comunes en estudiantes de programas de informática, se pretende identificar las estrategias pedagógicas exitosas, mediante reuniones mantenidas con docentes de cada universidad. La selección de estrategias pedagógicas tuvo el fin de diseñar e implementar un escenario pedagógico adaptado al perfil de la disciplina que responda a los problemas planteados.

Justificación científica

El desarrollo del proyecto responde, entre otras cuestiones, a la demanda creciente de profesionales de software, al desarrollo global de software y al énfasis en innovación y emprendimiento. El auge de la industria de software proyecta a la ingeniería de software como una de las disciplinas de mayor visibilidad e impacto.

Dada la multiplicidad de estrategias de enseñanza de la ingeniería de software, puestas en marcha en diferentes universidades de Latinoamérica, resulta oportuno hacer un relevamiento asociado a los diferentes perfiles de los estudiantes en cuanto al nivel de motivación que podría impactar en el aprendizaje. Las prácticas formativas deben ser coherentes con la naturaleza de la disciplina y, dado que el desarrollo de software es una de las competencias comunes a los programas de ingeniería de sistemas de las universidades latinoamericanas, resulta relevante encontrar modelos efectivos de formación en esta área.

Objetivos

El objetivo general del estudio es definir un marco de referencia que contribuya a mejorar la enseñanza de la ingeniería de software en los programas de informática de las universidades latinoamericanas que participaron en el estudio, teniendo en cuenta los referentes internacionales y el contexto particular de cada programa. Se pretende incidir en el proceso de formación del área de ingeniería de software de los programas de las universidades involucradas.

Los objetivos específicos que se persiguen son los siguientes: (a) identificar el nivel de motivación de los estudiantes de informática por el aprendizaje de la ingeniería de software; (b) identificar un marco de competencias profesionales comunes para los programas de ingeniería de software, analizando su adhesión a los referentes reconocidos a nivel mundial; (c) identificar, analizar y socializar estrategias pedagógicas exitosas en la enseñanza de la ingeniería de

software; y (d) proponer e implementar un escenario de aprendizaje mediante un proyecto piloto y analizar los resultados.

Las preguntas que orientan el proceso de investigación son las siguientes: ¿Cuál es el nivel de motivación de los estudiantes de informática en universidades latinoamericanas, respecto de la ingeniería de software? ¿Es posible definir un conjunto de competencias profesionales comunes del área de ingeniería de software para los programas de informática de las universidades latinoamericanas participantes? ¿Cuáles son las estrategias pedagógicas, en el área de ingeniería de software, que están siendo aplicadas con éxito?

Metodología

A fin de caracterizar el nivel de motivación de los estudiantes de la ingeniería de software, en las diferentes universidades involucradas, se procedió con (a) el análisis de trabajos relacionados, (b) la adaptación del instrumento, (c) la validación de contenido por expertos, (d) la aplicación del instrumento y (e) la identificación del poder predictivo de los factores considerados sobre la motivación general.

A los efectos de identificar un conjunto de competencias profesionales comunes para todos los programas, se procedió a (a) tomar como referente la lista de competencias profesionales del área de ingeniería de software de Tumino et al. (2016), (b) someter el listado al juicio de expertos docentes de los diferentes programas, (c) identificar las competencias de diferente nivel de granularidad y (d) actualizar el listado de competencias a partir de observaciones de los expertos y su evaluación respecto del grado de relevancia.

La lista de competencias profesionales del área de ingeniería de software se constituyó a partir de una búsqueda de competencias profesionales vigentes en carreras afines de instituciones de reconocida trayectoria. Se compilaron las más representativas del desempeño profesional en esta disciplina y se analizaron a la luz de los objetivos de la carrera, con el propósito de someterlas al juicio de empleadores, docentes, estudiantes y egresados. Una vez valoradas por los diferentes colectivos, se propusieron las competencias profesionales que reunieron los requisitos identificados en el estudio,

Como siguiente etapa, se propuso identificar, analizar y socializar las estrategias pedagógicas exitosas en la enseñanza de la ingeniería de software, mediante (a) recopilación de experiencias locales, (b) identificación de nuevas experiencias, (c) divulgación de la documentación entre las universidades participantes, (d) definición de escenarios pedagógicos adecuados para el aprendizaje de la ingeniería de software y (e) selección de un escenario de aprendizaje para ser aplicado en un proyecto piloto. El escenario pedagógico propuesto en el estudio pretendió motivar al estudiante a investigar y crear un producto de calidad que pudiera ser visualizado en la plataforma Youtube, por lo que exigía el desarrollo de la creatividad y responsabilidad de los estudiantes.

Resultados

Los resultados del estudio se describen desde las tres dimensiones enunciadas —aprendiz, cuerpo de conocimiento y contexto—, abordadas en términos de (a) motivación del estudiante, (b) competencias profesionales y (c) escenario pedagógico.

Motivación: análisis descriptivos

La muestra de 409 estudiantes, 333 hombres (81%) y 76 mujeres (19%), provenientes de 15 universidades, cuya edad promedio resultó ser de 21 años, se distribuyó por universidad tal como se detalla en la Tabla 1.

Dado que el estudio se enfoca en el comportamiento de la muestra respecto del área de la ingeniería de software, se obtuvieron las medias y los desvíos típicos de los ítems de motivación vinculados a esta área de interés, tal como se observa en la Tabla 2.

Tabla 1
Distribución de los participantes por instituciones intervinientes

Institución	*n*	%
Universidad Adventista del Plata	30	7,4
Universidad de Navojoa	17	4,2
Universidad Adventista de Chile	12	2,9
Universidad Peruana Unión - Filial Juliaca	51	12,5
Universidad Adventista de Bolivia	19	4,7
Corporación Universitaria Adventista de Colombia	59	14,5
Universidad Peruana Unión - Filial Tarapoto	39	9,6
Universidad Peruana Unión - Filial Lima	99	24,3
Instituto Tecnológico Superior Adventista del Ecuador (ITSAE)	8	2,0
Universidad de Montemorelos	8	2,0
Universidad Linda Vista	13	3,2
Universidad Adventista Dominicana	53	13,0
Universidad Adventista de Centro América	1	0,2
Total	408	100

Se advierte que el primer ítem, "los recursos físicos disponibles en la universidad son adecuados para los aprendizajes de la ingeniería de software: instalaciones, aulas, laboratorios, redes, biblioteca", fue el que obtuvo el menor puntaje con una media de 3,48 y un desvío típico de 1,282; mostrando un bajo nivel de acuerdo con la adecuación de los recursos para los aprendizajes.

Validación de la Escala de Motivación. Las evidencias de validez de la Escala de Motivación se obtuvieron mediante el análisis factorial exploratorio de los 40 ítems de la escala, mostrando un KMO de 0,978, que indica la adecuación muestral para aplicar el análisis. La matriz de factores rotados, utilizando la rotación Varimax, mostró un modelo de cuatro factores que explican el 71,5% de la variabilidad de las variables, lo que indica un buen ajuste del modelo. La saturación de las variables en cada uno de los cuatro factores denotan las siguientes dimensiones:

1. El primer factor está compuesto por los ítems 11, 12, 17, 18, 19, 20, 21, 23, 24, 25, 26, 27, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39 y 40. Estos ítems representan los atributos de desempeño.

2. El segundo factor se compone de los ítems 5, 6, 10, 13, 14, 15, 16, 22 y 28. Estos ítems representan las relaciones con estudiantes y docentes.

3. En el tercer factor se ubican los ítems 7, 8 y 9. Estos ítems representan el desempeño de los docentes.

4. El factor 4 se compone por los ítems 1, 2, 3 y 4, representando los recursos físicos y virtuales.

Se analizó la confiabilidad mediante el cálculo del alfa de Cronbach, donde los ítems correspondientes a la primera dimensión de atributos de desempeño arrojaron un alfa de ,981; los ítems correspondientes a la dimensión *relaciones con estudiantes y docentes* mostraron un valor de ,928, los propios de la dimensión *desempeño de los docentes* obtuvieron uno de ,869 y los ítems que componen la dimensión *recursos físicos y virtuales* alcanzaron un ,821. Cabe señalar que, en todos los casos, los análisis no sugirieron la eliminación de ítems con el propósito de incrementar el valor del estadístico de fiabilidad.

Tabla 2
Estadísticos descriptivos de los ítems de motivación con las mayores medias

Ítem	*M*	*DE*
Los recursos físicos disponibles en la universidad son adecuados para los aprendizajes de la ingeniería de software: instalaciones, aulas, laboratorios, redes, biblioteca	3,48	1,282
El equipamiento disponible en los laboratorios es adecuado para los aprendizajes de la ingeniería de software: sistema operativo, motores de base de datos, entornos de desarrollo, lenguajes de programación, herramientas CASE	3,58	1,264
Los profesores del área de ingeniería de software conocen bien las materias que imparten y saben cómo enseñármelas	3,89	1,218
Los profesores del área de ingeniería de software conocen bien como motivarme; sus estilos y actitudes estimulan mi aprendizaje	3,77	1,208
Los profesores del área de ingeniería de software supervisan adecuadamente mi actividad académica	3,80	1,174
Me gusta participar en la planificación de los trabajos prácticos del área de software	3,81	1,181
Disfruto al aplicar los saberes de ingeniería de software que adquiero en la universidad	4,00	1,184
Las prácticas pedagógicas aplicadas por los docentes en el área de ingeniería de software contribuyen a mi formación profesional	3,91	1,170
Me entusiasman las asignaturas relacionadas con la ingeniería de software	3,91	1,243
En mi vida profesional futura me gustaría desempeñarme en actividades directamente relacionadas con el desarrollo de software	3,89	1,282

Análisis de regresión lineal. Con el propósito de analizar el poder predictor de las variables *relaciones con estudiantes y docentes*, *desempeño de los docentes* y *recursos físicos y virtuales* sobre la variable *atributos de desempeño*, se aplicó un análisis de regresión lineal múltiple, con el método por pasos sucesivos, donde las variables predictoras actuaron como variables de entrada y *atributos de desempeño* actuó como variable dependiente.

El análisis de regresión lineal múltiple mostró la existencia de una relación significativa entre las variables que viene explicada por la ecuación $Y=0{,}102+0{,}782X_1+0{,}111X_2+0{,}071X_3$, donde Y representa a los atributos de desempeño, X_1 las relaciones entre estudiantes y docentes, X_2 el desempeño de los docentes y X_3 los recursos físicos y virtuales.

El coeficiente de determinación R^2 fue de ,832, lo que revela un gran poder predictor de las variables de entrada

sobre los atributos de desempeño de los estudiantes, siendo las relaciones entre estudiantes y docentes la variable con mayor ponderación. El error cuadrático medio fue de ,16. En la Tabla 3 se muestran los coeficientes tipificados y los valores de probabilidad asociados, indicando en todos los casos significación estadística.

Tabla 3
*Coeficientes tipificados y valores de probabilidad (*n = *409)*

Variable	B	SE β	β estandarizado
Relaciones	,828	,032	,782*
Desempeño docente	,109	,029	,111*
Recursos	,066	,030	,071*

* $p < ,05$

Comparación de medias de las dimensiones de motivación. A fin de analizar las varianzas del nivel de motivación percibido según el nivel académico de los estudiantes, para cada una de las cuatro dimensiones resultantes –(a) atributos de desempeño de los estudiantes, (b) relaciones entre estudiantes y docentes, (c) desempeño docente y (d) recursos físicos y virtuales–, se utilizó la variable categorizada de nivel académico como factor inter-sujetos en el análisis multivariante del modelo lineal general. El nivel básico quedó conformado por los estudiantes que cursaban entre el primero y el tercer semestres, el nivel intermedio por aquellos que cursaban entre el cuarto y el séptimo semestres y el nivel avanzado por aquellos que cursaban los últimos semestres de la carrera. Se observó que existe un efecto principal significativo del nivel académico, $F_{(2,377)} = 3,18$, $p < ,05$, $\eta^2 = ,017$, en el nivel de motivación correspondiente a los recursos físicos y virtuales, efecto que no se presenta en las restantes dimensiones. Existieron diferencias estadísticamente significativas en las medias de motivación vinculadas a recursos físicos y virtuales ($p < ,05$) entre los estudiantes del nivel académico avanzado ($M = 3,44$, $DE = 1,12$) y aquellos que cursan el nivel básico ($M = 3,78$, $DE = ,99$). Los datos dan cuenta de un mayor nivel de motivación relacionada con los recursos en los primeros semestres de la carrera.

Mapa de competencias para la IS

Se partió de la lista de competencias propuestas por Tumino et al. (2016), seleccionando las correspondientes a la ingeniería de software y organizándolas a modo de mapa de competencias, en donde las competencias genéricas aparecen en un primer nivel y las competencias específicas en un segundo nivel de granularidad, sin llegar al nivel 3 por su alta especificidad. Es decir, mientras que las competencias genéricas se corresponden con declaraciones generales y abarcantes, las competencias específicas se corresponden con declaraciones particulares contenidas en una competencia más general. La construcción del mapa se obtuvo mediante el juicio de expertos, de las instituciones participantes, quienes valoraron la claridad, la pertinencia y el nivel de granularidad de cada competencia. Como resultado, se actualizó el listado para luego obtener de cada experto la valoración de la relevancia de cada competencia. El mapa de competencias fue complementado con la propuesta curricular de ingeniería de software de la ACM/IEEE (Ardis et al., 2015) y por el trabajo de Fuente et al. (2015), como puede observarse en la Tabla 4.

Tomando como punto de partida las competencias de segundo nivel expresadas en la Tabla 4, se identificaron las competencias de tercer nivel. Por ejemplo, para la competencia 1.1, las competencias de nivel 3 definidas fueron: (a) identificar los requisitos de una aplicación web a través de historias de usuario; (b) realizar el análisis del requisito a través de mockups; (c) identificar los criterios de aceptabilidad y restricciones asociadas a los requisitos funcionales.

Escenarios pedagógicos

A los efectos de proponer e implementar un escenario pedagógico, se definieron algunos detalles que podrían resultar relevantes considerar. La experiencia se llevó a cabo en una universidad argentina, durante el segundo cuatrimestre del año 2016. Dado que se esperaba objetividad en la evaluación de los resultados de los dos módulos, se solicitó la participación de tres docentes de la carrera, dos de nivel doctoral y uno de maestría, con amplia trayectoria en la disciplina. Los ajustes curriculares que permitieron definir escenarios pedagógicos recomendables fueron los siguientes: (a) énfasis en el relacionamiento entre estudiantes y docentes y entre pares; (b) énfasis en la lectura previa del material de clase para orientar el trabajo del curso a la aplicación de conceptos y disminuir el uso de la exposición magistral; (c) implementación de Aprendizaje Basado en Problemas (ABP), donde los estudiantes, previamente divididos en grupos y mediante investigación y consulta con expertos, plantearon problemas, propusieron soluciones, generaron material de estudios sobre el tema y produjeron videos para subirlos a la plataforma Youtube. Respecto de las políticas del curso, se definieron las pautas de código de honor de manera conjunta con los estudiantes, lo que promovió un mayor nivel de responsabilidad del estudiante en sus actividades fuera de clase. En relación con la distribución de unidades, se introdujeron los cambios necesarios para la implementación del escenario pedagógico innovador en un módulo del curso, donde se pretendía marcar la tendencia, mientras que en otro módulo del programa curricular se implementó un escenario pedagógico tradicional.

El escenario pedagógico innovador pretendió reforzar los conocimientos sobre ciclos de vida del software, que luego los estudiantes implementan en varias actividades curriculares de la carrera y en su vida profesional. Este escenario se adecuó a los ciclos de vida, aplicando el aprendizaje basado en problemas, lo que permite a los estudiantes investigar el tema, identificar el problema, reconocer las variables intervinientes y plantear una solución. A esta estrategia se le sumó un requisito grupal mediante el cual cada equipo de estudiantes, organizado según las funciones asignadas por consenso, debía producir un video de 6 a 10 minutos, donde se explicara un ciclo de vida particular, identificando los tipos de problemas en los que se puede aplicar y otorgando los créditos a las fuentes de donde se obtuvo la información. El escenario fue propicio para cultivar el trabajo colaborativo con los pares y con otros colectivos al publicar los resultados del trabajo en YouTube. La evaluación de los dos módulos se adecuó a las modalidades de cursado: presencial o en línea. Mientras que el módulo impartido bajo la modalidad tradicional se evaluó mediante preguntas escritas que incluían opciones múltiples y opciones de desarrollo, la

Tabla 4
Mapa de competencias de Nivel 1 y Nivel 2

Competencia Nivel 1	Competencia Nivel 2
1. Identificar el problema y analizar, diseñar, implementar, mantener y evaluar una solución intensiva en software que satisfaga las necesidades del cliente, según criterios de costos, atributos de calidad e innovación tecnológica.	1.1. Descubrir necesidades del cliente y gestionar y especificar los requisitos y limitaciones del cliente, reconciliando objetivos en conflicto mediante la búsqueda de soluciones aceptables dentro de las restricciones derivadas del costo, del tiempo, de las tecnologías, de la existencia de sistemas ya desarrollados y de las propias organizaciones
	1.2. Diseñar la arquitectura de la solución integrando hardware, software y redes, de acuerdo con el tipo de sistema, teniendo en cuenta los objetivos y restricciones definidos por los interesados (stakeholders) y buscando un balance apropiado costo-beneficio
	1.3. Diseñar y evaluar la interfaz humano-computadora (HCI) que garantice la accesibilidad y usabilidad de los sistemas, servicios y aplicaciones informáticas
	1.4. Realizar el diseño detallado de las estructuras de datos, algoritmos, base de datos y módulos que representan la solución (adaptado de Fuente et al., 2015)
	1.5. Implementar el sistema informático a partir de las especificaciones funcionales y no funcionales
	1.6. Diseñar y realizar pruebas que verifiquen la validez de la solución (adaptado de Fuente et al., 2015)
	1.7. Gestionar la configuración de los componentes del sistema informático
	1.8. Liderar la puesta en marcha de la solución desarrollada, de manera que satisfaga los acuerdos de niveles de servicio establecidos
	1.9. Realizar el mantenimiento de sistemas informáticos de manera que garantice su mejora continua de acuerdo a la dinámica de la realidad que soporta
	1.10. Aprender nuevos modelos, técnicas y tecnologías que emergen y apreciar la necesidad de una actualización continua de la profesión (extraído de Ardis et al., 2015)
	1.11. Diseñar soluciones apropiadas en uno o más dominios de aplicación usando principios de la ingeniería de software que integran consideraciones éticas, legales, sociales y económicas (extraído de Ardis et al., 2015)
2. Planificar y gestionar proyectos en el ámbito de la ingeniería del software haciendo uso de las prácticas, herramientas, métodos y tecnologías adecuadas de acuerdo al tipo de proyecto.	2.1. Aplicar los principios de organización, economía, gestión de recursos humanos, legislación y normalización en proyectos del ámbito de la ingeniería de software
	2.2. Comprender y aplicar los principios de la gestión de riesgos en la elaboración y ejecución de proyectos de ingeniería de software
	2.3. Analizar, seleccionar y aplicar las prácticas, métodos y modelos de ciclos de vida adecuados a las necesidades de la aplicación a construir
	2.4. Trabajar de manera individual o como parte de un equipo, con el propósito de desarrollar y entregar artefactos software de calidad (extraído de Ardis et al., 2015)
	2.5. Definir protocolos para el aseguramiento de la calidad
3. Gestionar las tecnologías de la información y comunicación en los procesos empresariales que contribuyan con soluciones efectivas a las necesidades de información de las organizaciones, otorgándoles ventajas competitivas.	3.1. Mediar entre las comunidades técnicas y de gestión de una organización, aplicando los principios y buenas prácticas de las comunicaciones organizacionales
	3.2. Elaborar pliegos de condiciones técnicas de una instalación informática que cumpla con los estándares y normativas vigentes
	3.3. Evaluar y seleccionar plataformas de hardware y software para la operación de sistemas, servicios y aplicaciones informáticas
	3.4. Realizar tasación, peritaciones e informes de tareas o trabajos de informática, adecuados a las necesidades
	3.5. Aplicar estándares de seguridad, confidencialidad, integridad y privacidad, inherentes a los sistemas de información, dentro del marco de la legislación vigente, la normativa del colegio profesional correspondiente y las políticas de las organizaciones donde se desempeña
	3.6. Gestionar el aseguramiento del sistema y los datos según las necesidades de uso y las condiciones de seguridad establecidas para prevenir fallos y ataques externos
	3.7. Planificar y ejecutar tareas de auditoría de los sistemas de información

evaluación del módulo innovador fue orientada por rúbricas para el trabajo en equipo y la producción de videos. Es importante señalar que el nivel de complejidad en ambos módulos fue similar y consensuado previamente por los docentes.

En la primera evaluación del módulo innovador, los estudiantes lograron un promedio de 93 puntos, considerando una escala de 0 a 100. Finalmente, se aplicó una evaluación parcial que incluyó dos unidades diferenciadas por las estrategias de enseñanza aplicadas. Una de ellas correspondió al tema que investigaron los estudiantes y una segunda se dictó en forma tradicional. En la evaluación de la unidad involucrada en la experiencia, los estudiantes lograron un promedio de 83 puntos, mientras que, en la unidad dictada de forma tradicional, el promedio fue inferior a 60 puntos.

Al comparar las medianas de las calificaciones de los estudiantes entre los dos módulos, mediante la prueba no paramétrica de Wilcoxon para contrastar la hipótesis sobre igualdad de medianas, el estadístico de Wilcoxon estuvo asociado a un nivel crítico menor de ,05, por lo que se rechaza la hipótesis de igualdad de promedios y se concluye que las calificaciones comparadas difieren significativamente; es decir, el promedio de calificaciones logradas por los estudiantes mediante la modalidad innovadora fue significativamente mayor que el obtenido mediante la modalidad tradicional.

Las demás universidades pertenecientes al equipo de trabajo se encuentran en el proceso de implementación de otras estrategias.

Discusión

En el trabajo se han identificado las competencias específicas que permitirán precisar la adhesión de los programas en forma más clara.

El análisis de las dimensiones obtenidas vinculadas a la motivación de los estudiantes permite visualizar el comportamiento de los factores intrínsecos (desarrollo intelectual y superación académica, responsabilidad y predisposición hacia el aprendizaje, perspectiva profesional y reconocimiento, y disposición hacia el trabajo en equipo) y extrínsecos (métodos de evaluación y supervisión de los docentes, recursos formativos disponibles, relaciones con compañeros, definición de tareas y optimización del esfuerzo) en la muestra de estudio.

Es interesante notar que las mayores medias motivacionales se encontraron en los ítems que reflejan la sensación de logro de los estudiantes, tanto por aprender algo nuevo, como por cumplimentar exitosamente una práctica o aprobar las asignaturas, lo que da cuenta de la trascendencia del sentido de superación personal a lo largo de la carrera. Es decir que prevalecen en la motivación factores de carácter intrínseco. Se observó la relevancia que adopta tanto la relación con los estudiantes como cuán expertos son los docentes en el desarrollo de sus clases, lo que podría considerarse a la hora de diseñar las actividades curriculares a los efectos de incrementar la motivación de los estudiantes.

Tal como se observó en la comparación de medias motivacionales, los estudiantes de niveles superiores son los que mejor identifican la importancia de contar con escenarios de prácticas similares a los existentes en la industria.

La experiencia realizada en el estudio, con la implementación de aprendizaje basado en problemas y la generación de videos subidos en la plataforma

Youtube, resultó exitosa, pero se encuentra a la espera de que las demás universidades intervinientes en el proyecto implementen otras estrategias de enseñanza y aprendizaje para disponer de un conjunto de estrategias exitosas y compartirlas con la comunidad académica. Por esta razón, se reconoce al presente trabajo como una aproximación que permite impulsar nuevos estudios tendientes a optimizar la formación de facultativos de la ingeniería de software, profesión cada vez más demandada por el mundo científico y tecnológico.

Referencias

Álvarez Álvarez, A., González Mieres, C. y García Rodríguez, N. (2007). La motivación y los métodos de evaluación como variables fundamentales para estimular el aprendizaje autónomo. *REDU: Revista de Docencia Universitaria, 2*. Recuperado de http://red-u.net/redu/files/journals/1/articles/53/public/53-42-2-PB.pdf

Anaya, R. (2006). Una visión de la enseñanza de la ingeniería de software como apoyo al mejoramiento de las empresas de software. *Universidad Eafit, 42*(141), 60-76.

Anaya, R. y Trujillo, J. (2005, octubre). *Un modelo de enseñanza en el contexto de la ingeniería de software.* Ponencia presentada en el XIII Congreso Iberoamericano de Educación Superior en Computación, Cali, Colombia.

Anijovich, R. y Mora, S. (2009). *Estrategias de enseñanza: otra mirada al quehacer del aula.* Buenos Aires: Aique.

Ardis, M., Budgen, D., Hislop, G. W., Offutt, J., Sebern, M. y Visser, W. (2015). SE 2014: Curriculum guidelines for undergraduate degree programs in software engineering. *Computer, 48*(11), 106-109. doi:10.1109/MC.2015.345

Bain, K. (2007). *Lo que hacen los mejores profesores de universidad*. Valencia: Universidad de Valencia.

Baker, A., Oh Navarro, E. y van der Hoek, A. (2005). An experimental card game for teaching software engineering processes. *Journal of Systems and Software*, *75*(1-2), 3–16. doi:10.1016/j.jss.2004.02.033

Bavota, G., De Lucia, A., Fasano, F., Oliveto, R. y Zottoli, C. (2012). Teaching software engineering and software project management: An integrated and practical approach. En *ICSE '12: Proceedings of the 34th International Conference on Software Engineering* (pp. 1155–1164). Piscataway, NJ: IEEE Press. doi:10.1109/ICSE.2012.6227027

Cagiltay, N. E. (2007). Teaching software engineering by means of computer-game development: Challenges and opportunities. *British Journal of Educational Technology*, 38(3), 405-415. doi:10.1111/j.1467-8535.2007.00705.x

Dai, H. N., Wei, W., Wang, H. y Wong, T. L. (2017). *Impact of outcome-based education on software engineering teaching: A case study.* Ponencia presentada en la IEEE 6th International Conference on Teaching, Assessment, and Learning for Engineering (TALE). doi:10.1109/TALE.2017.8252344.

Davini, M. C. (2008). *Métodos de enseñanza*. Buenos Aires: Santillana.

Finkel, D. (2008). *Dar clase con la boca cerrada.* Valencia: Universidad de Valencia.

Fuente, A. A. J., de Andrés, J., Nieto, C., Suárez, M., Pérez, J. R., Cernuda, A., . . . Fondón, M. D. (2005). El libro azul de la ingeniería en informática: una alternativa al libro blanco. En M. C. Luengo Díez y M. Riesco Albizu (Eds.), *I Jornadas de Innovación Docente de la EUITIO* (pp. 67-75). Oviedo: E. U. de Ingeniería Técnica Informática de Oviedo.

Gannod, G. C., Burge, J. E. y Helmick, M. T. (2008). Using the inverted classroom to teach software engineering. En *ICSE '08: Proceedings of the 30th International Conference on Software Engineering* (pp. 777-786). New York: ACM. doi:10.1145/1368088.1368198

Ghezzi, C. y Mandrioli, D. (2005). The challenges of software engineering education. En *ICSE '05: Proceedings of the 27th International Conference on Software Engineering* (pp. 637-638. New York: ACM. doi:10.1145/1062455.1062578

Henson, K. T. y Eller, B. F. (2000). *Psicología educativa para la enseñanza eficaz*. México: Thomson.

Hwang, G. J., Wu, P. H. y Chen, C. C. (2012). An online game approach for improving students' learning performance in web-based problem-solving activities. *Computers & Education*, *59*(4), 1246–1256. doi:10.1016/j.compedu.2012.05.009

Joyce, B., Weil, M. y Calhoun, E. (2000). *Modelos de enseñanza*. Barcelona: Gedisa.

Larripa, M. y Erausquin, C. (2008). Teoría de la actividad y modelos mentales. Instrumentos para la reflexión sobre la práctica profesional: "aprendizaje expansivo", intercambio cognitivo y transformación de intervenciones de psicólogos y otros agentes en escenarios educativos. *Anuario de Investigaciones, 15*, 109-124.

Leontiev, A. N. (1981). *Problems of the development of the mind.* Moscú: Progress.

Lockwood, K. y Esselstein, R. (2013). The inverted classroom and the CS curriculum. En *SIGCSE '13: Proceeding of the 44th ACM Technical Symposium on Computer Science Education* (pp. 113-118). New York: ACM. doi:10.1145/2445196.2445236

López Fernández, D., Alarcón Cavero, P. P., Rodríguez Sánchez, M. y Casado Fuente, M. L. (2014). Motivación en estudiantes de ingeniería: un caso de estudio con teorías e instrumentos para su medida y desarrollo. *REDU: Revista de Docencia Universitaria, 12*(4), 346-376.

Martínez Verde, R. y Bonachea Montero, O. (2006). ¿Estrategias de enseñanza o estrategias de aprendizaje? *Revista Varela, 6*(13). Recuperado de http://revistavarela.uclv.edu.cu/articulos/rv1305.pdf

Martin-Kniep, G. (2000). *Becoming a better teacher: Eight innovations that work.* Alexandria, VA: ASCD.

Murillo Torrecilla, F. J. (Coord.). (2003). *La investigación sobre eficacia escolar en Iberoamérica: revisión internacional del estado del arte*. Bogotá: Convenio Andrés Bello.

Nauman, M. y Uzair, M. (2007). SE and CS collaboration: Training students for engineering large, complex systems. En *20th Conference on Software Engineering Education & Training (CSEET'07)* (pp. 167-174). doi:10.1109/CSEET.2007.44

Ng, P. W., & Huang, S. (2013, May). Essence: A framework to help bridge the gap between software engineering education and industry needs. In *Software Engineering Education and Training (CSEE&T), IEEE 26th Conference* on (pp. 304-308). IEEE.

Nordio, M., Ghezzi, C., Meyer, B., Di Nitto, E., Tamburrelli, G., Tschannen, J., . . . Kulkami, V. (2011). Teaching software engineering using globally distributed projects: The DOSE course. En *CTGDSD '11: Proceedings of the 2011 Community Building Workshop on Collaborative Teaching of Globally Distributed Software Development* (pp. 36-40). New York: ACM. doi:10.1145/1984665.1984673

Núñez, J. C., Solano, P., González-Pienda, J. A. y Rosario, P., (2006). El aprendizaje autorregulado como medio y meta de la educación. *Papeles del Psicólogo, 27*(3), 141-148.

Oliveros, L (2006). Identificación de competencias: una estrategia para la formación en el Espacio Europeo de Educación Superior. *Revista Complutense de Educación, 17*(1), 101-118.

Razmov, V. y Anderson, R. (2006). Pedagogical techniques supported by the use of student devices in teaching software engineering. En *SIGCSE '06: Proceedings of the 37th SIGCSE Technical Symposium on Computer Science Education* (pp.344-348). New York: ACM. doi:10.1145/1121341.1121449

Sanjurjo, L. y Rodríguez, X. (2009). *Volver a pensar la clase. Las formas básicas de enseñar*. Buenos Aires: Homo Sapiens.

Sanjurjo, L. y Vera, M.T. (2006). *Aprendizaje significativo y enseñanza en los niveles medio y superior.* Buenos Aires: Homo Sapiens.

Strong, R. W, Silver, H. F. y Perini, M. J. (2001). *Teaching what matters most: Standards and strategies for raising student achievement*. Alexandria, VA: ASCD.

Talízina, N. F. (1988). *Psicología de la enseñanza*. Moscú: Progreso.

Tumino, M. C., Bournissen, J. M. y Barrio, K. (Enero, 2016). Sistemas de información: competencias profesionales 2020. *European Scientific Journal, 12*(1), 63-82. doi:10.19044/esi.2016.v12n1.p63

Recibido: 4 de marzo de 2017
Revisado: 10 de abril de 2017
Aceptado: 18 de junio de 2017