HACIA UN ENFOQUE BASADO EN COMPETENCIAS PARA LA ENSEÑANZA DE LA INGENIERÍA DE SOFTWARE UTILIZANDO INVESTIGACIÓN-ACCIÓN

Jhon Fredy Niño Manrique, Raquel Anaya Hernández

Corporación Universitaria Adventista
Medellín, Colombia

Resumen

Este trabajo muestra el esfuerzo conjunto de docentes, profesionales de la industria e investigadores para hacer una reforma curricular en un programa de ingeniería de sistemas, específicamente en el área disciplinar de la ingeniería de software, utilizando el enfoque de investigación acción. Entre los resultados obtenidos a la fecha se pueden destacar: La definición de una mapa de competencias profesionales del área de ingeniería de software (IS), estructurado en competencias de primero y segundo nivel; la identificación de las asignaturas centrales que contribuyen al desarrollo de las competencias y la identificación del nivel de transversalidad de éstas, lo cual fue definido en un trabajo conjunto con los docentes del área de IS; se realizó la caracterización de proyectos de aula, como la estrategia pedagógica central del área de IS. El trabajo más representativo de intervención en este primer ciclo de la investigación se realizó en el curso de Ingeniería de Software en donde se trabajó con el docente de la asignatura para declarar de manera explícita las competencias de tercer nivel, caracterizar el proyecto del curso y enfatizar las actividades de aprendizaje que se desarrollan a lo largo del curso.

Palabras clave: enseñanza de la ingeniería de software; enfoque basado en competencias; investigación-acción

Abstract

This work shows the joint effort of teachers, industry professionals and researchers to make a curricular reform in a system engineering program, specifically in the disciplinary area of software engineering, using the action research approach. Among the results

obtained to date we can remark: the definition of a map of professional skills in the area of software engineering (IS), structured in first and second level skills; The identification of the central subjects that contribute to the development of the skills and the identification of the level of transversality of these, which was defined in a joint work with the teachers of the IS area; The characterization of classroom projects was carried out, such as the central pedagogical strategy of the IS area. The most representative work of intervention in this first cycle of the research was carried out in the Software Engineering course where the work was carried out with the teacher of the subject in order to declare explicity the third level skills, characterize the project of the course and to emphasize the learning activities that are developed throughout the course.

***Keywords**: teaching software engineering; skill-based approach; action- research*

1. Introducción

Son diversos y complejos los perfiles ocupacionales que la industria demanda de un profesional de informática en el siglo XXI. Esta complejidad en la formación se evidencia en la propuesta de la ACM / IEEE que a través del Joint Task Group propuso en el 2005, la definición de programas específicos para las cinco sub-disciplinas de la computación (Shackelford, 2006).

Si bien esta propuesta internacional es un referente importante, la diversidad de programas alrededor de informática en América Latina, tienen denominaciones y perfiles muy diversos que buscan atender, con un solo programa, la demanda de profesionales, tal como lo identificó el estudio de carreras de informática del proyecto Tunning (Latina, 2013). Dicho estudio aplicó una encuesta a empleadores, graduados, académicos y estudiantes de países de Latinoamérica para identificar las competencias disciplinares más importantes, encontrando que las tres competencias identificadas, de forma unánime como más importantes, están directamente relacionadas con el desarrollo de soluciones informáticas. Este énfasis está directamente relacionado con el papel relevante que la industria del software está desempeñando en América Latina. Se observan esfuerzos de la industria de software latinoamericana de incursionar en el mercado mundial y el interés de los gobiernos de promover el software como un renglón de la economía del país (Cepal, 2013) (Martínez, 2015)

De otra parte, se analiza el papel que la academia juega en la formación del recurso humano que debe insertarse en la industria de software y se reconoce, en general, que existe un gap entre las demandas de la industria y el tipo de formación de los egresados de las carreras de informática, tal como lo expresa (Castrillón, 2008): "Las barreras creadas al interior de las universidades, no posibilitan su actuación en la cadena de la industria".

La investigación acerca de la enseñanza de la IS ha estado en plena evolución, por tratarse de un reto complejo en el cual se involucran aspectos de tipo técnico, de gestión de proyectos y de interacción social (Ghezzi, Mandrioli, 2005). Es necesario que el aprendizaje de la práctica del desarrollo de software se convierta en un proceso de

construcción social en donde los participantes (docentes, estudiantes, tutores, universidad, empresas) se relacionen entre sí y propicien situaciones altamente cercanas a la realidad a fin de alcanzar unos objetivos de aprendizaje y así mejorar la práctica del software a través de la educación (Anaya, 2006) (Lethbridge, 2007) (Bavota, et.al 2012).

El enfoque de competencias, es una de las estrategias articuladoras reconocidas con el propósito de eliminar el gap mencionado, a la vez que facilita la globalización del mercado laboral, la movilidad de estudiantes y trabajadores, la calidad de procesos y productos y la competitividad empresarial. Si bien el enfoque de competencias tiene un fuerte soporte conceptual, existen dificultades evidentes a la hora de implementar un currículo basado en competencias (Oliveros, 2006).

El objetivo de este trabajo es fortalecer el enfoque de competencias en la propuesta curricular de un programa de Ingeniería de Sistemas, específicamente en el área de ingeniería de software. El trabajo está estructurado de la siguiente manera: en la sección 2 se define el marco conceptual en el que se soporta el trabajo, en la sección 3 se justifica la metodología de trabajo basado en el enfoque investigación acción, en la sección 4 se sintetizan los resultados de este primer ciclo de la investigación y, finalmente, en la sección 5 se presentan las conclusiones y trabajos futuros.

2. Marco conceptual

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 para abordar dicha transformación cada vez con mayor propiedad, teniendo en cuenta, de una parte, el proceso de apropiación progresiva de las diversas áreas de conocimiento relacionadas (ingeniería básica, organizaciones, de comunicaciones, tecnología, entre otras) y, de otra parte, la madurez cognitiva y emocional del aprendiz. El modelo conceptual que se toma como referencia en este trabajo identifica tres elementos claves involucrados en el proceso de enseñanza aprendizaje: El aprendiz, el objeto de aprendizaje y el contexto, analizadas desde la teoría de la actividad (Anaya & Trujillo, 2006).

El enfoque orientado a competencias puede verse como la estrategia que permite articular de manera coherente estos elementos claves y sus interrelaciones. Tal como lo define (Oliveros, 2006), la gestión por competencias es un sistema integrado de evaluación y mejora de organizaciones y/o personas que la componen, 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 Oliveros (2006): (a) en función de un puesto de trabajo, en donde se analizan las competencias que posee una persona que está desempeñando un puesto de trabajo o en función de un perfil profesional, que son las competencias de un sujeto en formación en un centro

educativo. Mientras que la identificación de competencias en función del puesto de trabajo es relativamente sencilla, la identificación de las competencias de un plan de estudio es una tarea ardua y compleja, debido a la formación progresiva de las mismas a lo largo del programa y a la dificultad de simular en un contexto académico escenarios reales de trabajo (Oliveros, 2006).

Reconociendo la complejidad de la definición de competencias de un perfil profesional, se definieron los siguientes lineamientos para ello: (a) Se debe identificar el nivel de transversalidad de la competencia; cuantas más asignaturas contribuyan al desarrollo de una competencia, el nivel de transversalidad de la misma es mayor. Cada asignatura debe hacer explícita la forma como ésta contribuye al desarrollo de la competencia particular en las dimensiones del conocer y hacer, de manera que se haga evidente el proceso formativo en espiral que ya fue mencionado. (b) De manera inversa, se debe identificar el nivel de relevancia de la asignatura; cuanto mayor sea el número de competencias a las que una asignatura contribuye en su formación se trata de una asignatura central en donde el esfuerzo de formación debe ser mayor (número de créditos/horas presenciales/horas de trabajo fuera de clase). (c) Con el objetivo de acortar la distancia con la realidad empresarial, se ha identificado el enfoque basado en proyectos como la estrategia pedagógica central que permite acercar a los estudiantes a escenarios similares a un puesto de trabajo; este enfoque por proyectos ha sido ampliamente estudiando en la enseñanza de la ingeniería de software (Bavota, 2012) (Rajlich, 2013) (Bruegge, et.al, 2015).

3. Metodología de trabajo

Este trabajo esta soportado en la metodología investigación-acción (IA) que comprende: (a) Se realizan acciones como parte integrante del proceso de investigación, de manera que la acción en si misma representa una fuente de conocimiento; (b) El foco reside en los valores del profesional más que en los aspectos metodológicos; (c) Los profesionales investigan sobre sus propias acciones (Colmenares, 2012); la metodología IA es reconocida como uno de los enfoques apropiado para abordar las reformas curriculares, en los que el docente cumple un papel protagónico porque reflexiona, analiza e indaga acerca de su quehacer formativo y se constituye en investigador de su propia práctica profesional que genera transformaciones reales en el aula de clase (Quintero, et.al, 2007). El enfoque de IA se hizo evidente en el trabajo a través de sesiones de *focus group* con los docentes del área (8 docentes) y a través de un trabajo de acompañamiento al docente de la asignatura de ingeniería de software.

El proyecto siguió las fases utilizadas por (Colmenares, 2012) en sus desarrollos investigativos: Fase 1: Identificación del problema, en donde se caracterizó el programa objeto de estudio y se identificó la brecha existente entre la enseñanza y la práctica de la ingeniería de software. Fase 2: Construcción participativa del plan de acción, en la cual se definieron por consenso con los docentes y administradores del programa las acciones a seguir para la solución de la problemática identificada. Fase 3, Ejecución del plan de acción, en el cual se realizaron las acciones planificadas tendientes a lograr los

cambios curriculares. Fase IV: Reflexión y cierre, es la fase transversal en la cual se realizan los procesos de reflexión permanente y el análisis de resultados en el aula de clase, con el propósito de proponer las aproximaciones teóricas y metodológicas que sirven de orientación para los nuevos ciclos de la investigación.

4. Resultados

4.1 Mapa de competencias de Ingeniería de Software

Se partió de la lista de competencias sugeridas en el trabajo de (Tumino, et.al, 2016) en el cual se encontró que las competencias mencionadas en la lista se encuentran a diferente nivel de granularidad: mientras que unas son declaraciones generales y abarcantes, otras son competencias particulares contenidas en una competencia más general; por lo tanto, se propuso organizarlas a manera de mapa de competencias en donde las competencias genéricas aparecen en primer nivel y las competencias específicas aparecen en un segundo nivel. Este 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 (Fuentes, et.al., 2015). Ver tabla 1.

Competencias Nivel 1	Competencias Nivel 2	Cód.
Identificar el problema y analizar, diseñar, implementar, mantener y evaluar una solución intensiva en software que satisfagan las necesidades del cliente, según criterios de costos, atributos de calidad e innovación tecnológica	Descubrir necesidades del cliente, 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	1.1
	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.	1.2
	Diseñar y evaluar la interfaz humano-computadora (HCI) que garantice la accesibilidad y usabilidad a los sistemas, servicios y aplicaciones informáticas.	1.3
	Realizar el diseño detallado de las estructuras de datos, algoritmos, base de datos y módulos que representan la solución. Adaptada de (Fuentes, et.al 2015)	1.4
	Implementar el sistema informático a partir de las especificaciones funcionales y no funcionales.	1.5
	Diseño y realización de pruebas que verifiquen la validez de la solución. Adaptada de (Fuentes, et.al 2015).	1.6
	Gestionar la configuración de los componentes del sistema informático.	1.7
	Liderar la puesta en marcha de la solución desarrollada, de manera que satisfaga los acuerdos de niveles de servicio establecidos.	1.8
	Realizar el mantenimiento de sistemas informáticos de manera que garantice su mejora continua de acuerdo a la dinámica de la realidad que soporta.	1.9
	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)	1.10

	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).	1.11
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	Aplicar los principios de organización, economía, gestión de recursos humanos, legislación y normalización en proyectos del ámbito de la ingeniería de software	2.1
	Comprender y aplicar los principios de la gestión de riesgos en la elaboración y ejecución de proyectos de ingeniería de software	2.2
	Analizar, seleccionar y aplicar las prácticas, métodos y modelos de ciclos de vida adecuados a las necesidades de la aplicación a construir	2.3
	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)	2.4
	Definir protocolos para el aseguramiento de la calidad	2.5
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.	Mediar entre las comunidades técnicas y de gestión de una organización, aplicando los principios y buenas prácticas de las comunicaciones organizacionales	3.1
	Elaborar de pliegos de condiciones técnicas de una instalación informática que cumpla con los estándares y normativas vigentes	3.2
	Evaluar y seleccionar plataformas hardware y software para la operación de sistemas, servicios y aplicaciones informáticas	3.3
	Realizar tasación, peritaciones e informes de tareas o trabajos de informática, adecuadas a las necesidades	3.4
	Aplicar de 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	3.5
	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	3.6
	Planificar y ejecutar tareas de auditoría de los sistemas de información	3.7

Tabla 1. Competencias de Nivel 1 y 2. Fuente: Elaboración propia

4.2 Cubrimiento de competencias

Para identificar el nivel de transversalidad de las competencias y el nivel de relevancia de las asignaturas, se instrumentó la tabla 2, en la cual se cruzan las competencias del nivel 2 (en las filas) con las asignaturas del área (en las columnas). Cada uno de los ocho docentes participantes asignó un 1 en la casilla correspondiente si consideraba que dicha asignatura debía contribuir al desarrollo de la competencia.

Competencias de nivel 2	Ingeniería de Software	Arquitectura de Software	Taller de Desarrollo de Software
1.1	8	7	6
1.2	8	6	3
1.4	7	5	4

Tabla 2. Cubrimiento de competencias de nivel 2 (Vista parcial). Fuente: Elaboración propia

- **Nivel de transversalidad de las competencias**

Haciendo un análisis horizontal en la tabla 2, se identifica el aporte que cada asignatura hace al desarrollo de la competencia, lo cual indica el nivel de transversalidad de la misma. Se tomaron aquellas competencias que tuvieran al menos 3 asignaturas con una frecuencia mínima de cinco.

Dicha transversalidad implica que las competencias se van desarrollando de forma incremental a través de varias asignaturas, dando diversos énfasis en complejidad, profundidad y alcance práctico en las dimensiones del conocer y hacer. Por ejemplo, la primera competencia relacionada con los requisitos (1.1) se podría trabajar desde Fundamentos de programación (semestre 1) con requisitos funcionales sencillos para proyectos individuales en los cuales los estudiantes se van familiarizando con el concepto de requisitos desde el hacer mismo, pasando por asignaturas como Ingeniería de software (semestre 5) en la cual se formaliza (conocer) y aplica (hacer) el concepto (parcialmente), y Arquitectura de software (semestre 7) en la cual se profundiza en el hacer (identificación de atributos de calidad y restricciones).

- **Nivel de relevancia de las asignaturas**

Haciendo un análisis vertical en la tabla 2, se obtiene el nivel de relevancia de la asignatura, toda vez, que contribuyen al desarrollo de varias competencias de nivel 2. Por cada asignatura se tomaron las competencias en las cuales hubo cinco o más votos de parte de los docentes participantes. Se puede concluir que, a mayor número de competencias a desarrollar en una asignatura, ésta tendría mayor relevancia y llevaría a considerarse la necesidad aumentar el número de horas y el esfuerzo en este espacio de formación, con las implicaciones en el re-diseño del pensum. Se encontró que las asignaturas con mayor nivel de relevancia fueron Ingeniería de Software, Arquitectura de software y Taller de desarrollo de software con al menos cinco competencias relacionadas.

4.3 Caracterización de los proyectos

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 asignaturas. En términos generales los tipos de proyectos se han definido a) según su origen (a partir de la descripción de un problema, a partir de la descripción de un diseño, a partir de la necesidad de un cliente real, y a partir de una idea innovadora) y b) según el número de miembros del proyecto (individual, por equipos y por equipos distribuidos). Cada docente, debe identificar la manera como orienta su trabajo práctico a la luz de dicha caracterización.

4.4 Intervención en la asignatura de Ingeniería de Software

- **Identificación de las competencias de tercer nivel**

Tomando como punto de partida las competencias de segundo nivel expresadas en la tabla 1, se identificó, de manera conjunta con el docente, 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.

- **Otros ajustes micro-curriculares**

Los ajustes micro-curriculares más sobresalientes son los siguientes: a) con respecto a la metodología del curso: se hizo énfasis en la lectura previa del material de clase para orientar el trabajo del curso a aplicación de conceptos, disminuyendo notablemente el uso intensivo de la exposición magistral; la ejecución del proyecto del curso, siguiendo los lineamientos de caracterización de proyectos definido.; b) con respecto a las políticas del curso: se introdujo la estrategia de código de honor definido y pactado de manera conjunta con los estudiantes, lo cual evidenció un mayor nivel de responsabilidad del estudiante en sus actividades fuera de clase; c) con respecto a la distribución de unidades se hicieron cambios para integrar las temáticas del marcos de trabajo ágil que estaban desarticuladas de los temas de proceso y proyecto.

5. Conclusiones

Se han presentado los resultados en su primer ciclo acerca del enfoque basado en competencias en el área de IS. Si bien este trabajo se focaliza en dicha área, puede servir de referente para otras áreas del currículo. Las reflexiones de este ciclo de investigación permitieron identificar los aspectos relevantes a trabajar en el siguiente ciclo, entre los cuales se pueden mencionar: (a) Precisar la manera concreta de contribución de cada asignatura al logro de una competencia transversal. (b) Diseñar las estrategias de evaluación orientada a competencias, que es uno de los retos abiertos para dar evidencia en la práctica de un enfoque realmente orientado a competencias. (c) Definir y aplicar la estrategia para análisis de las competencias transversales.

Uno de los aspectos sobresalientes de este trabajo fue la participación activa de los docentes del área, logrando satisfactoriamente iniciar un proceso de auto-reflexión sobre su ejercicio docente que se espera que vaya madurando hasta convertirlos en agentes dinamizadores del cambio en el aula de clase. Este enfoque participativo se vio fortalecido puesto que seis de los ocho docentes participantes son profesionales activos en la industria del software.

6. Referencias

Artículos de revistas

- Anaya, Raquel (2006). Una visión de la enseñanza de la ingeniería de software como apoyo al mejoramiento de las empresas de software. Revista Universidad EAFIT, Vol 42, No 141, pp. 60-76.
- Ardis, M., Budgen, D., Hislop, G. W., Offutt, J., Sebern, M., And Visser, W. (2015). SE 2014: Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering. Computer, Vol 48, No 11, pp. 106-109.
- Bruegge, B., Krusche, S. And Alperowitz, L. (2015). Software engineering project courses with industrial clients. ACM Transactions on Computing Education (TOCE), Vol 15 No 4 Article 17.
- CEPAL (Comisión Económica para América Latina y el Caribe) (2013), Economía digital para el cambio estructural y la igualdad (LC/L.3602), Santiago de Chile. Publicación de Naciones Unidas, pp. 132
- Colmenares, A. M. (2012). Investigación-acción participativa: una metodología integradora del conocimiento y la acción. Voces y Silencios: Revista Latinoamericana de Educación, Vol 3, No 1, pp. 102 - 115
- Latina, T. A. (2013). Educación Superior en América Latina: reflexiones y perspectivas en Informática. España
- Martínez, S. J., Arango, S., And Robledo, J. (2015). El crecimiento de la industria del software en Colombia: Un análisis Sistémico. Revista EIA, Vol 12, No 23, pp. 95-106.
- Oliveros, L (2006). Identificación de competencias: una estrategia para la formación en el Espacio Europeo de Educación Superior. Revista Complutense de Educación, Vol 17, No 1, pp. 101-118
- Parra, J. (2008). Factores críticos de éxito e hipótesis sobre la Industria del software en Colombia. Consideraciones contextuales y académicas. Avances en sistemas e informática, Vol 5, No 2, pp. 185 - 193.
- Quintero, J., Munévar R. A., And Yepes, J. C. (2007). Investigación-acción y currículo; un recorrido por el mundo. Revista Latinoamericana de Estudios Educativos (Colombia), Vol 3, No 1, pp. 123-142.
- Shackelford, R., McGettrick, A., Sloan, R., Topi, H., Davies, G., Kamali, R., ... And Lunt, B. (2006). Computing curricula 2005: The overview report. In ACM SIGCSE Bulletin. Vol. 38, No. 1, pp. 456-457
- Tumino, M. C., Bournissen, J. M., And Barrio, K. (2016). Sistemas de información: Competencias profesionales 2020. European Scientific Journal, Vol 12, No 1.

Memorias de congresos

- Anaya, R., And Trujillo, J. (2006). Un modelo para la enseñanza en contexto de la ingeniería de software. Memoria 1, 1–13. Memorias del XIII Congreso iberoamericano de educación superior en computación. Pontificia Universidad Javeriana, Vol 1, fasc.1, pp.120 – 132.

- Bavota, G., De Lucia, A., Fasano, F., Olivetto, R. And Zottoli, C. (2012). "Teaching software engineering and software project management: An integrated and practical approach." Proceedings of the 34th International Conference on Software Engineering. IEEE Press, pp. 1115 – 1164
- Fuente, A. A. J., de Andrés, J., Nieto, C., Suárez, M., Pérez, J. R., Cernuda, A., ... And Fondón, M. D. (2005). El libro Azul de la Ingeniería en Informática: una alternativa al Libro Blanco. Conference: I Jornadas de Innovación Docente de la EUITIO, Vol 1, pp. 67.
- Ghezzi, C., Mandrioli, D. (2005). The challenges of software engineering education. Proceedings of the 27th International Conference on Software Engineering - ICSE 2005, Vol 4309, pp. 115 - 127
- Lethbridge, T. C., Diaz-Herrera, J., Le Blanc, R. Jr, And Thompson, J. B. (2007). Improving software practice through education: Challenges and future trends. In Future of Software Engineering, 2007. FOSE'07. pp. 12-28
- Rajlich, V. (2013). Teaching developer skills in the first software engineering course. Proceedings of the 35 th International Conference on Software Engineering – ICSE 2013, IEEE Press.

Libros

- Latorre, A. (2003). La investigación acción. Conocer y cambiar la práctica educativa. España, Editorial Graó, pp. 138

Sobre los autores

- **Jhon Fredy Niño Manrique,** Ingeniero de sistemas, Máster en Ingeniería de la Universidad EAFIT. Decano Facultad de Ingeniería. jfnino@unac.edu.co
- **Raquel Anaya Hernández,** Ingeniera de sistemas, Doctora en Informática de la Universidad Politécnica de Valencia. Investigadora Grupo de Investigación en Ingeniería Aplicada. raquel.anaya.hdez@gmail.com