METODOLOGÍA DE ENSEÑANZA PARA PROCESOS DE ENTRADA/SALIDA EN LA CONSTRUCCIÓN DE SOFTWARE MEDIANTE APRENDIZAJE COLABORATIVO Y MODELADO GRÁFICO

Javier Alejandro Jiménez Toledo

Institución Universitaria CESMAG
Pasto, Colombia

César Collazos Ordóñez, Julio Ariel Hurtado Alegría, Wilson Libardo Pantoja Yépez

Universidad del Cauca
Popayán, Colombia

Resumen

Este artículo presenta los resultados de investigación obtenidos al diseñar una metodología de enseñanza basada en aprendizaje colaborativo y modelado gráfico, que permite de una manera sencilla presentar al estudiante de fundamentos de programación, el desarrollo de la fase de análisis como de diseño de los procesos de entrada/salida en la construcción de una solución de software. La investigación se desarrolló bajo el paradigma positivista, con enfoque cuantitativo, utilizando el método empírico analítico, bajo un tipo de investigación correlacional y con un diseño experimental basado en G_1 X O_1 ,G_2 - O_2, G_3 X O_3, G_4 – O_4, G_5 X O_5 y G_6 – O_6; donde G_1, G_3 y G_5 son los tres grupos experimentales conformados por estudiantes de ingenierías del primer curso de fundamentos de programación de tres universidades de la ciudad de San Juan de Pasto y G_2, G_4 y G_6 son sus correspondientes grupos de control, además X corresponde al tratamiento experimental basado en la estrategia didáctica propuesta y desde O_1 hasta O_6 corresponde a las pre pruebas realizadas a cada grupo. Los datos obtenidos fueron analizados con la distribución de probabilidad t de Student con la que se comprobó que la diferencia de notas entre los grupos experimentales y los grupos de control para cada universidad participante es estadísticamente significativa, lo que concluyó el éxito del tratamiento experimental.

Palabras clave: aprendizaje colaborativo; modelado gráfico; entradas/salidas; construcción de software

Abstract

This paper presents the research results obtained in the design of a teaching methodology based on collaborative learning and graphic modeling, which allows to present the programming fundamentals student in a simple manner, develop the analysis phase and design processes input / output in building software. The research was conducted in the positivist paradigm, with quantitative approach, using the analytical empirical method, under a kind of correlational research and an experimental design based on G_1 X O_1, G_2 - O_2, G_3 X O_3, G_4 – O_4, G_5 X O_5 and G_6 – O_6; where G1, G3 and G5 are the three experimental groups shaped by Systems Engineering students of the first course of Programming Fundamentals in three universities from the city of San Juan de Pasto and G_2, G_4 and G_6 are their control groups corresponding, besides, X is to experimental treatment based on the didactical strategy proposal and from O_1 to O_6 It corresponds to the pre tests made to each group. The data obtained were analyzed with the probability distribution T-student checking that the difference in notes between experimental groups and control groups for each university were statistically significant, concluding the good success of the experimental treatment.

Keywords: *collaborative learning; graphic modeling; input / output; software construction*

1. Introducción

La enseñanza de los fundamentos de programación en los últimos años ha sido objeto de estudio del mismo campo ingenieril como de diversas áreas de la Psicología, la Neurociencia, el Diseño gráfico, la pedagogía, entre otras. Esto debido a la importancia que actualmente tiene el Software en todos los procesos del accionar humano donde los programadores, constructores y arquitectos de soluciones computacionales son los actores principales de estos procesos. Pese al estado tecnológico actual existen aún problemas relacionados con la fundamentación de los futuros constructores de software cuyo origen se inicia desde el primer curso de programación recibido, el cual es clave en el proceso de formación que tendrá un desarrollador de soluciones software en su etapa de aprendizaje y más tarde en su vida profesional (Hernández, Jiménez, and Martínez, n.d.). En el estudio realizado por Affleck y Smith (1999) se ha encontrado que el principal problema de los programadores principiantes es el acceso a los conocimientos previos y la adopción de un enfoque para estudiar, que va más allá de la memorización explícita de conocimientos necesarios para aplicar y transferir el dominio de conceptos a situaciones nuevas.

Es evidente la preocupación existente en los docentes de los primeros cursos de programación con relación a los resultados obtenidos en el proceso de aprendizaje, por ello se han llevado a cabo en los últimos años varios proyectos encaminados a mejorar dichos procesos donde la mayor atención se enfoca en los primeros niveles de formación (Jiménez, Collazos, Hurtado, & Pantoja, 2015). El término Pensamiento Computacional ha sido popularizado por Jeannette M. Wing en el 2006 (Selby and Woollard, 2010); desde entonces, ha venido cobrando gran importancia por ser considerado como una habilidad del Siglo 21, la cual deben desarrollar todas las personas para lograr ser competitivos en una economía global (Gómez, 2014). Existen distintas iniciativas y herramientas educativas para enseñar el pensamiento computacional (Espino & González, 2015) como

ChildProgramming (Hurtado, Collazos, Cruz and Rojas, 2012), Scratch (MIT, 2008), Alice (Mellon, 2003), entre otros, lo que ha conllevado a que los sistemas educativos estén incorporando en sus currículos oficiales nuevos conocimientos relacionados con el pensamiento computacional (Valverde, Fernández and Garrido, 2015).

Por lo anterior, el pensamiento computacional es un campo que toma especial atención en la formación de constructores de software el cual consiste en la resolución de problemas, el diseño de los sistemas, y la comprensión de la conducta humana haciendo uso de los conceptos fundamentales de la informática (Wing, 2006). Además, son muchas los beneficios que conlleva la apropiación del pensamiento computacional, entre ellos, permite la transformación de una sociedad formada por meros consumidores de tecnología, en una de potenciales desarrolladores de ésta (Barcelo, 2015), permite desarrollar sistemáticamente las habilidades de pensamiento crítico y resolución de problemas con base en los conceptos de la computación, donde los estudiantes y profesionales tendrán la necesidad de aprender y practicar las habilidades para poder utilizar las nuevas tecnologías y confrontar los desafíos del Siglo XXI (Zapotecatl, 2014).

Por otro lado, el aprendizaje colaborativo (cooperativo) es el uso instruccional de pequeños grupos de tal forma que los estudiantes trabajen juntos para maximizar su propio aprendizaje y el de los demás (Jonhson, Jonhson and Holubec, 1993) donde cada miembro del grupo de trabajo es responsable no solo de su aprendizaje, sino de ayudar a sus compañeros a aprender, creando con ello una atmósfera de logro (Monterrey, 2008). En el aprendizaje colaborativo los estudiantes trabajan colaborando, este tipo de aprendizaje no se opone al trabajo individual ya que puede observarse como una estrategia de aprendizaje complementaria que fortalece el desarrollo global del alumno, además, se establece que los métodos de aprendizaje colaborativos traen consigo una renovación en los roles asociados a profesores y alumnos, en el caso de los profesores se establece tres tipos: Profesores como Mediador cognitivo, Instructor y Diseñador Instruccional (Collazos, Guerrero, and Vergara, 2012).

En la enseñanza de los fundamentos de programación los procesos de entrada/salida tienen especial tratamiento debido a que son los insumos (entradas) que permitirán desencadenar el accionar interno del software y son los resultados (salidas) que el cliente espera obtener. Por lo anterior, en este artículo se presenta una alternativa diferente para acercar al estudiante de programación computacional, de una manera práctica, a tener una visión general del comportamiento de un proceso de entrada/salida, mediante la una metodología de enseñanza basada en aprendizaje colaborativo y modelado gráfico.

2. Propuesta metodológica de enseñanza para procesos de entrada/salida en la construcción de software mediante aprendizaje colaborativo y modelado gráfico

Los resultados presentados en este artículo toman como base la enseñanza para la unidad de competencia denominada ¨análisis y diseño de procesos de captura y salida de datos¨ en los cursos de fundamentos de programación bajo un entorno colaborativo que combinado con una propuesta de modelado gráfico, permitió mejorar el rendimiento académico de los estudiantes participantes, en la figura 1 se presenta el diagrama de actividad bajo

el cual se llevó a cabo el proceso metodológico del presente estudio.

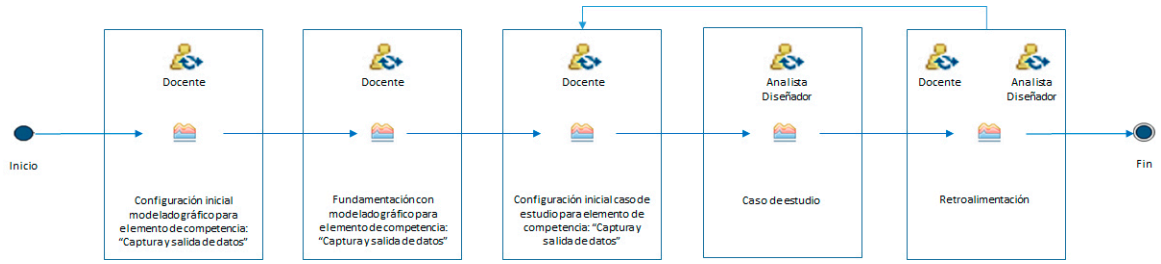

Fig. 1. Diagrama de actividad

En la figura 1 se muestra tanto los roles (docente y estudiante el cual asume los sub roles de analista y diseñador) como las actividades requeridas para la estrategia didáctica propuesta. El rol docente asumirá características de mediador cognitivo, instructor y diseñador instruccional para la realización de una configuración inicial por cada temática orientada la cual tendrá dos momentos: la primera corresponde a nivel de la orientación al grupo de estudiantes respecto de la fundamentación conceptual de las temáticas bajo un modelado gráfico y en las segunda presentará casos de estudio utilizando el método JigSaw. Por otra parte, el estudiante actuará en dos sub roles: Analista y diseñador orientado por principios colaborativos que le permitan interactuar con los demás compañeros en los casos de estudio planteados. Apoyados en el modelo presentado por Collazos y Mendoza (Aronso, Blaney, Stephan, Sikes and Snapp, 1978) (Collazos and Mendoza, 2006), en la figura 2 se presenta tanto las herramientas como las actividades del docente para llevar a cabo una colaboración efectiva que basa su fundamento en la acción del docente como diseñador instruccional, como mediador cognitivo y como profesor instructor.

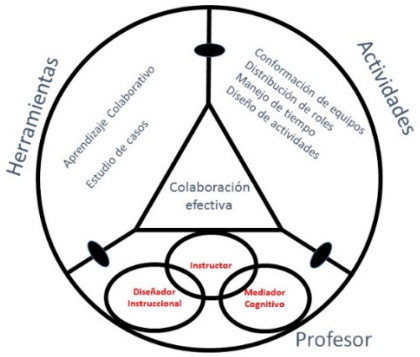

Fig. 2. Rol docente

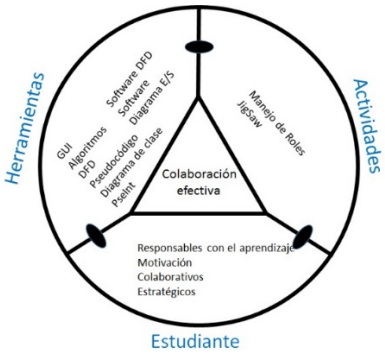

Fig. 3. Rol estudiante

El docente como diseñador instruccional es quien realiza la planeación tanto de las unidades temáticas como de las actividades de aprendizaje y evaluación que se llevarán a cabo durante el desarrollo del curso (Collazos and Mendoza, 2006). A su vez, el docente como mediador cognitivo, es el encargado de validar el conocimiento adquirido por el estudiante mediante la utilización de diversas estrategias de seguimiento que deben ir desde la observación directa con la cual se puede evidenciar el interés y los alcances conceptuales logrados, la utilización de preguntas de verificación, la realización de actividades de aprendizaje y evaluación que permitan directamente comprobar el aprendizaje de las temáticas. Finalmente el docente como profesor instructor se apoya en la clase magistral como estrategia didáctica en la que se encarga tanto de la enseñanza de las unidades temáticas en este caso de la

asignatura de fundamentos de programación, como de desarrollar en los estudiantes habilidades sociales, de trabajo en grupo y fundamentalmente actividades colaborativas, modelando habilidades interpersonales positivas y conllevándolos a su práctica (Collazos and Mendoza, 2006). De igual manera en la figura 3 se presenta el rol del estudiante bajo el esquema efectivo de colaboración en el que intervienen herramientas, actividades y las características necesarias para el trabajo del estudiante bajo un entorno colaborativo. A continuación, se describe el modelo propuesto en el diagrama de actividad de la figura 1.

Configuración inicial de los fundamentos conceptuales con modelado gráfico para "Captura y salida de datos". En la cual el docente prepara previo a la sesión de clase los materiales y saberes necesarios para abordar esta unidad de competencia. En esta etapa se llevaron a efecto las siguientes actividades: Identificación del curso: área, componente, créditos, semestre, docente; descripción del curso; competencia central; definición de unidades de aprendizaje con sus correspondientes subtemas; diseño de ejemplos; estimación del tiempo para desarrollo de temática explicativa; planeación de recursos y materiales necesarios para las actividades propuestas; planeación de la distribución física de los estudiantes en el aula de clase y diseño de tareas y evaluaciones.

Fundamentación inicial al grupo de estudiantes sobre "capturas y salidas de datos" utilizando modelado gráfico. La estrategia propuesta para modelado gráfico parte del objeto de análisis más elemental a la hora de concebir un proyecto de software que es el "diagrama de entrada salida", con el cual es posible identificar elementos claves como estructuras condicionales, estructuras repetitivas, clases, variables generales, variables de clase, métodos y otros elementos necesarios en los fundamentos de programación. El objetivo de esta estrategia es inculcar en el estudiante desde su primer curso de programación que existen unas fases importantes en la construcción de software (independiente de las metodologías para construcción de software existentes) las cuales debe identificar desde etapas tempranas. Es así como entre las fases iniciales propuestas para un primer curso de fundamentación para programación se recomienda:

Fase de recolección de requerimientos: El docente debe informar al estudiante la importancia de recolectar información como una etapa importante en la construcción de proyectos de software. Es necesario que el docente concientice al estudiante que si se lleva a cabo un buen proceso de recolección de información, la probabilidad de éxito del proyecto será alta

Fase de análisis. Es necesario que el docente le informe al estudiante que después de apropiar y realizar la fase de recolección de requerimientos (que para efectos de la presente propuesta metodológica se tomará a través del enunciado del problema) es necesario un proceso de análisis de la información.

Fase de diseño. En esta fase el docente le informará al estudiante quien ya tiene claro la fase de recolección de requerimientos y la fase de análisis, que es importante construir un modelo que cumpla con los requerimientos exigidos y con los elementos realizados en la fase de análisis

Fase de codificación. El docente informará al estudiante que, tras un proceso de diseño de software adecuado, la fase de codificación tendrá un alto porcentaje de éxito. Además, se debe destacar la importancia que tiene la

realización de diseño en todo proceso ingenieril.

Configuración inicial para caso de estudio. El docente debe antes de la sesión de clase preparar los casos de estudio que les presentará a los estudiantes para que a través de un entorno colaborativo mediado por JigSaw haya una apropiación significativa de la unidad de competencia tratada. En esta etapa se deben llevar a efecto las siguientes actividades: descripción del curso, definición de objetivos individuales y grupales, diseño de ejemplos y tareas, estimación del tiempo para realización de ejercicios y tareas, planeación de recursos y materiales necesarios para las actividades propuestas, conformación de los equipos de trabajo y planeación de la distribución física de los estudiantes en el aula de clase.

Presentación y realización de casos de estudio con el grupo utilizando JigSaw. Para ello el docente ejecutará la planeación realizada en la configuración inicial para el caso de estudio, organizando el salón de clase de tal manera que los estudiantes queden reunidos en grupos de dos y distribuidos en todo el recinto, demás el docente será el encargado únicamente de presentar los enunciados para cada ejercicio propuesto. Cada ejemplo debe ser analizado por los dos estudiantes cada uno en el sub rol de analista y diseñador. Para ello una vez presentado el enunciado del ejercicio, cada estudiante propondrá una solución desde su rol y la compartirá con su compañero de grupo base, luego cada uno se reunirá con otro "par" de otro grupo para analizar cada propuesta y concluir una sola y posteriormente esos dos se reunirán con otros "pares" hasta que la mitad del grupo quede reunido bajo un rol y la otra mitad con el otro rol los cuales obtendrán una solución al ejercicio, finalmente cada estudiante volverá a su grupo base para compartir la solución con su compañero y entre los dos proponer la solución final.

Retroalimentación por parte del docente. Al terminar cada ejercicio el docente realizará una retroalimentación que consistirá en concluir junto con los estudiantes la solución final y las consideraciones importantes y pertinentes tanto al ejercicio como al tema tratado. Luego de realizar la retroalimentación, el docente presentará el siguiente ejercicio y si es necesario intercambiará los roles entre los dos integrantes de cada grupo o los renovará totalmente.

3. Resultados de la investigación

La metodología de investigación planteada se desarrolló bajo el paradigma positivista, con enfoque cuantitativo, utilizando el método empírico analítico, bajo un tipo de investigación correlacional y con un diseño experimental basado en G1 X O1, G2 - O2, G3 X O3, G4 – O4, G5 X O5 y G6 – O6. La investigación fue realizada en seis grupos diferentes pertenecientes a tres instituciones de educación superior del municipio de Pasto (Nariño) para el primer curso de fundamentación de programación. Los grupos G1, G3 y G5 corresponden a los grupos experimentales de cada institución y G2, G4 y G6 fueron los grupos de control, además X fue el tratamiento experimental que consistió en la estrategia didáctica bajo un entorno colaborativo a través de modelado grafico para la enseñanza para la unidad de competencia "Captura y salida de datos". A su vez, O1, O2, O3, O4, O5, O6, fueron las post pruebas realizadas al final del tratamiento experimental tanto para los grupos experimentales como los de control.

El primer grupo experimental G1 fue conformado por 33 estudiantes de la asignatura de Introducción a la programación de primer semestre de Ingeniería de Sistemas, el segundo grupo experimental G3 estuvo constituido por 11 estudiantes de la asignatura de Lógica Computacional de primer semestre de Ingeniería Electromecánica y el tercer grupo experimental G5 estuvo constituido por 29 estudiantes de la asignatura de fundamentos de programación de primer semestre de Ingeniería Electrónica, todos correspondientes al periodo académico II-2015 a quienes se les aplicó el tratamiento experimental X y finalmente se le aplicó una prueba posterior. Los grupos de control estuvieron conformados por: G2 por 27 estudiantes, G4 14 por estudiantes y G6 por 33, todos del periodo académico I -2015 del mismo semestre y asignatura del grupo experimental a quienes no se les aplicó tratamiento experimental y las notas obtenidas del tema de estudio fueron consideradas como O2. Una vez aplicado el tratamiento investigativo en cada grupo experimental (G1, G3 y G5) se procedió a la realización de cuatro actividades evaluativas que consistieron en la aplicación de dos talleres grupales con la participación de dos estudiantes (40%) y dos seguimientos individuales (60%) con el propósito de establecer cuantitativamente el proceso de apropiación de dicha temática y cuyos resultados se aprecian en la tabla 1

Tabla 1. Promedio de notas grupos experimentales y de control

Introducción a la programación			Fundamentos de programación			Lógica computacional		
Grupo	Periodo	Promedio	Grupo	Periodo	Promedio	Grupo	Periodo	Promedio
G_1	II-2015	4.2	G_3	II-2015	4.4	G_5	II-2015	4.9
G_2	I-2015	3.3	G_4	I-2015	3.7	G_6	I-2015	4.4

Con los resultados obtenidos tanto para los grupos experimentales como de control se realizó un análisis estadístico mediante la distribución de probabilidad T de Student con un nivel de confianza del 95%. En la tabla 2 se muestran los resultados obtenidos para el curso de introducción a la programación para el curso de introducción a la programación.

Tabla 2. T de Student grupos G_1, G_2, G_3, G_4, G_5 y G_6

Concepto	Introducción a la Programación		Lógica Computacional		Fundamentos de Programación	
	Grupo Experimental	Grupo de Control	Grupo Experimental	Grupo Control	Grupo Experimental	Grupo Control
Media	4,2161	3,2926	4,4445	3,6807	4,8552	4,3939
Varianza	1,4293	0,4969	0,7548	0,6092	0,0147	1,3712
Observaciones	33	27	11	14	29	33
Varianza agrupada	1,0113		0,6725		0,738	
Diferencia hipotética de las medias	0		0		0	
Grados de libertad	58		23		60	
Estadístico t	3,5387		2,3118		2,1091	
P(T<=t) una cola	0,0004		0,0150		0,0195	
Valor crítico de t (una cola)	1,6716		1,7139		1,6706	
P(T<=t) dos colas	0,0008		0,0301		0,0391	
Valor crítico de t (dos colas)	2,0017		2,0686		2,0002	

La tabla 2 evidencia el resultado de la aplicación del tratamiento experimental al curso de Introducción a la programación, los resultados estadísticos obtenidos para el grupo experimentales G_1 adquiere un valor estadístico t (3,53867884) mayor tanto al valor crítico de t de una cola (1,67155276) como al valor crítico para dos colas (0,00079949) y el valor de P (para una y dos colas) es menor al 5% (0,00039975 y 0,00079949 respectivamente), lo cual concluye que la diferencia de notas entre grupo experimental y grupo de control es estadísticamente significativa para t=5%. Además, En la tabla 2 se evidencia los resultados estadísticos obtenidos para el curso de Lógica Computacional y Fundamentos de Programación donde G_3 y G_5 adquieren un valor estadístico t (2,3118 y 2,1091) mayor tanto al valor crítico de t de una cola (1,7139 y 1,6706) como al valor crítico para dos colas (2,0686 y 2,0002) y el valor de P (para una y dos colas) es menor a al 5 en los cuatro casos, lo cual también concluye que la diferencia de notas entre grupo experimental y grupo de control para cada curso es estadísticamente significativa para t=5%.

4. Conclusiones y trabajos futuros

Por lo tanto, el análisis estadístico anterior demuestra la incidencia que tiene el tratamiento presentado en esta investigación en los grupos experimentales frente a los grupos de control, estableciendo que al incorporar estrategias didácticas adecuadas se obtienen resultados académicos que benefician de una manera directa a los estudiantes. Además, El proceso colaborativo llevado a cabo en la investigación permitió que los estudiantes se inter relacionara de forma activa, logrando así una dinámica interactiva que mejoró las relaciones interpersonales del grupo. Por otro lado, en el estudio se pudo evidenciar que al mezclar la rigurosidad de una teoría (en este caso el estudio de la unidad de competencia denominada "Captura y salida de datos") con procesos interactivos soportados en una propuesta gráfica y mediado con una didáctica de participación activa (estrategia colaborativa utilizando el método JigSaw), es más sencillo obtener por tiempos más prolongados la atención del estudiante, que involucrado en un proceso totalmente interactivo comienza a construir las bases de su lógica computacional cuya importancia es vital para los procesos que se requieren en posteriores semestres.

Como trabajo futuro se contempla la inclusión del estudio de estructuras condicionales y cíclicas que finalmente puedan concluir en la construcción de una metodología que utilizando aprendizaje colaborativo y modelado gráfico permitan que el estudiante de los primeros cursos de fundamentación en programación incorpore los aprendizajes de una manera significativa a sus condiciones personales mediante la interacción en un ambiente propicio colaborativo.

5. Referencias

- Affleck, G. and Smith, T. (1999). Dentifying a need for web - based course support.
- Aronso, E., Blaney, N., Stephan, C., Sikes, J. and Snapp, M. (1978). The Jigsaw Classroom. Beverly Hills: Sage.

- Barcelo, P. (2015). Editorial. Bits de Ciencia, Vol. 1, No. 1.
- Collazos, C., Guerrero, L., & Vergara, A. (2012). Aprendizaje Colaborativo: un cambio en el rol del profesor.
- Collazos, C., & Mendoza, J. (2006). Cómo aprovechar el "aprendizaje colaborativo" en el aula. Educación Y Educadores, Vol. 9, No. 2, pp. 61–76.
- Espino, E., & González, C. (2015). Estudio sobre diferencias de género en las competencias y las estrategias educativas para el desarrollo del pensamiento computacional. Revista de Educación a Distancia., Vol. 46, pp. 1–20.
- Gómez Giraldo, L. Y. (2014). Competencias Mínimas En Pensamiento Computacional Que Debe Tener Un Estudiante Aspirante a La Media Técnica Para Mejorar Su Desempeño En La Media T{é}cnica De Las Instituciones Educativas De La Alianza Futuro Digital Medellín. UNIVERSIDAD EAFIT.
- Hernández, G., Jiménez, R., & Martínez, Á. (n.d.). Creencias docentes sobre la importancia de la didáctica en la orientación de la enseñanza del primer curso de programación de computadoras. Revista Universitaria: Docencia, Investigación E Innovación, Vol. 2, pp. 87–103.
- Hurtado, J. A., Collazos, C. A., Cruz, S. T., & Rojas, O. E. (2012). Child Programming: una estrategia de aprendizaje y construcción de Software basada en la lúdica, la colaboración y la agilidad. Revista Universitaria RUDIC, Vol. 1, No. 1.
- Jiménez, J., Collazos, C., Hurtado, J., & Pantoja, W. (2015). Estrategia colaborativa en entornos tridimensionales como estrategia didáctica de aprendizaje de estructuras iterativas en programación computacional. Investigium IRE Ciencias Sociales Y Humanas, Vol. 6, No. 2, 80–92. http://doi.org/10.1007/s13398-014-0173-7.2
- Jonhson, D. W., Jonhson, R., & Holubec, E. (1993). Circles of learning (4th ed.).
- Mellon, U. C. (2003). Alice. Http://alice.org/index.php.
- MIT. (2008). Scratch.
- Monterrey, I. T. (2008). Aprendizaje Colaborativo, técnicas didácticas. Programa de Desarrollo de Habilidades Docentes. Retrieved from de http://www.itesca.edu.mx/documentos7desarrollo_academico/metodo_apr
- Selby, C., and Woollard, J. (2010). Computational Thinking: The Developing Definition. SIGCSE 2014.
- Valverde Berrocoso, J., Fernández Sánchez, M. R., & Garrido Arroyo, M. del Ć. (2015). El pensamiento computacional y las nuevas ecologías del aprendizaje. RED - Revista de Educación a Distancia, (46), 1–18. http://doi.org/10.6018/red/46/3
- Wing, J. M. (2006). Computational Thinking. It represents a universally applicable attitude and skill set everyone, not just computer scientists, would be eager to learn and use. COMMUNICATIONS OF THE ACM, Vol. 49, No. 3. Retrieved from https://www.cs.cmu.edu/~15110-s13/Wing06-ct.pdf
- Zapotecatl, J. L. (2014). Pensamiento Computacional. In Insituto Nacional de Astrofísica, Óptica y Electrónica (1st ed.). Purbla, México: Luis Enrique Erro.

Sobre los autores

- **Javier Alejandro Jiménez Toledo**: Ingeniero de Sistemas, especialista en Docencia Universitaria, Candidato a Magister en Computación, Estudiante de doctorado en Ciencias de la Electrónica mención Computación, docente investigador adscrito a la Facultad de Ingeniería de la I.U.CESMAG. Director del grupo de investigación Tecnofilia. Investigador asociado en Colciencias. jajimenez@iucesmag.edu.co.
- **César Collazos Ordóñez**: Ingeniero de Sistemas, Doctor en Ciencias Mención Computación, Post Doctor en Ciencias de la Computación, coordinador grupo Investigación IDIS, Integrante de CARL, Profesor Visitante Universidad de Lleida (España), Miembro Junta Directiva AIPO, Miembro Junta directiva Sociedad Colombiana de Computación, Miembro del Programa Nacional de Electrónica, Telecomunicaciones e informática de Colciencias. Profesor titular Departamento de Sistemas de la Universidad del Cauca ccollazo@unicauca.edu.co.
- **Julio Ariel Hurtado Alegría**. Ingeniero en Electrónica y Telecomunicaciones, especialista en Redes y Servicios Telemáticos, especialista en Procesos para el Desarrollo de Software, Doctor en Ciencias Mención Computación, profesor Titular Departamento de Sistemas Universidad del Cauca, Investigador grupo IDIS. ahurtado@unicauca.edu.co.
- **Wilson Libardo Pantoja Yépez**. Ingeniero de Sistemas, especialista en Redes y Servicios Telemáticos, Magister en Computación, profesor Titular Departamento de Sistemas Universidad del Cauca, Investigador grupo IDIS. wpantoja@unicauca.edu.co.
