Estrategia didáctica basada en aprendizaje colaborativo y modelado gráfico para la enseñanza de procesos de captura y salida de datos en fundamentos de programación

Javier Alejandro Jiménez Toledo[1], Cesar Collazos[2], Wilson Libardo Pantoja Yépez[2],
Julio Ariel Hurtado Alegría[2]
[1]Institución Universitaria CESMAG
Pasto, Colombia
[2]Universidad del Cauca
Departamento de Ingeniería de sistemas
Popayán, Colombia
jajimenez@iucesmag.edu.co, { ccollazo, wpantoja, ahurtado] @unicauca.edu.co

Abstract. Este artículo, presenta la experiencia de validación de una propuesta basada en aprendizaje colaborativo y modelado gráfico como estrategia didáctica para la enseñanza tanto de la fase de análisis como de diseño de los procesos de captura y salida de datos en la construcción de una solución de software, con estudiantes de primeros cursos de fundamentos de programación en instituciones de educación superior de modalidad presencial. La investigación se desarrolló bajo un enfoque cuantitativo con el método empírico analítico y cuyo diseño se realizó de manera cuasi experimental en tres universidades con tres grupos de control y tres experimentales de estudiantes de pregrado de primer semestre de Ingeniería quienes fueron evaluados mediante aplicación de post pruebas después de recibir o no tratamiento experimental y cuyos datos obtenidos fueron analizados con la distribución de probabilidad T de Student con la que se comprobó que la diferencia de notas entre grupo experimental y grupo de control para cada universidad participante fue estadísticamente significativa, lo que concluyó el éxito del tratamiento experimental.

Keywords—*Aprendizaje colaborativo, modelado gráfico, estrategia didáctica, captura y salida de datos.*

1. Introducción

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 [1].

Son muchos los inconvenientes que debe enfrentar un estudiante de primer curso de programación debido a la complejidad de las temáticas tratadas, por ello, el inicio en el aprendizaje de la programación siempre ha sido un proceso complicado ya que es una disciplina totalmente diferente a lo que se ha visto desde entonces por parte del programador novato porque la programación exige cambiar de manera radical el modo de pensar y analizar las cosas y aun habiendo adquirido los conocimientos teóricos necesarios se ha detectado una gran dificultad de aplicar esos conocimientos. teóricos en la resolución de problemas prácticos [2]. La enseñanza de la programación no se puede transmitir directamente desde instructores a los alumnos, debe ser adquirida activamente por los estudiantes [3].

Lo anterior forma parte esencial de un área de conocimiento que en los últimos años ha llamado el interés no solo de las área disciplinares de la ingeniería sino de otras como la educativa, filosófica, psicológica y de las mimas ciencias de la salud, dicha área forma parte del desarrollo del pensamiento computacional.

El término Pensamiento Computacional fue popularizado por Jeannette M. Wing en el 2006 [4]; 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 [5].

Es así como el pensamiento computacional 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 [6], además, el pensamiento computacional no es sinónimo de capacidad para programar un ordenador, puesto que requiere pensar en diferentes niveles de abstracción y es independiente de los dispositivos, convirtiéndose en una competencia básica que todo ciudadano debería conocer para desenvolverse en la sociedad digital, pero no es una habilidad «rutinaria» o «mecánica», ya que es una forma de resolver problemas de manera inteligente e imaginativa [7].

Existen distintas iniciativas y herramientas educativas para enseñar el pensamiento computacional [8] como ChildProgramming [9], Scratch [10], Alice [11], entre otros, lo que ha conllevado a que los sistemas educativos estén incorporando en sus currículos oficiales nuevos conocimientos relacionados con el pensamiento computacional [7].

A continuación se presenta un listado de las diversas metodologías vigentes que sirven como herramientas de apoyo en la consolidación de las competencias necesarias que un desarrollador de software debe tener en su formación: Trabajo Colaborativo como Estrategia para Mejorar el Proceso de Enseñanza-Aprendizaje [12], EDCIA [13], VPL [12], Entorno de integración de PBL y CSCL para la enseñanza de algoritmos y programación en ingeniería [14], Scratch [15], Cupi2 [16], una herramienta y técnica para la enseñanza de la programación [17], Greenfoot [18], JeCo [19], ProLearn [2], PL-Detective [20], College [21], OOP Anim [22], DPE [23], ELP [24], BlueJ [18], Alice [25], Habipro [26], Algoarena [27], Sigacle [28], entre otros.

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 [29] 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 [30].

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 [31].

Es así como Hsu determina que el objetivo del aprendizaje colaborativo es inducir a los participantes a la construcción de conocimiento mediante exploración, discusión, negociación y debate [32].

Por su parte, Brown y Atkins [33] establecen que los objetivos que persigue el aprendizaje colaborativo se centran en el desarrollo de estrategias de comprensión y explicación, de preguntas y respuestas. La discusión y el debate sirven, en primer lugar, para desarrollar las habilidades de comunicación con otros y la utilización precisa del lenguaje. En segundo lugar, genera el desarrollo de competencias intelectuales y profesionales, como por ejemplo: analizar, razonar, pensar críticamente, sintetizar, diseñar, etc. y, por último, el aprendizaje colaborativo también promueve el crecimiento personal de los estudiantes, que incluye el desarrollo de estrategias de comunicación y pensamiento, el desarrollo de la autoestima, dirige el propio aprendizaje, aprende a trabajar con otros y a conocerse a sí mismo y a los demás [34].

Existen diversas teorías sobre los modelos de enseñanza, que van desde los tradicionales, los constructivistas, hasta los investigativos [35] [25]–[30] entre otros como se citó en Romero y Rosero [42]; sin embargo, es importante aclarar que en las prácticas de enseñanza universitaria no se determina un modelo único o ideal, que copiase de manera fotográfica sus postulados, por cuanto no se puede desconocer la realidad que acontece en el aula, el contexto, la interacción entre docente y estudiantes [42].

Para Moroni y Señas [43] una estrategia de aprendizaje de programación se fundamenta en comenzar a enseñar programación utilizando los algoritmos para dar solución a un problema. Esta propuesta lleva al estudiante a realizar sus soluciones en modelos que se construyen en papel y a probarlos mediante pruebas de escritorio. A pesar de aparentemente no tener dificultades con el aprendizaje por parte del estudiante, se ha comprobado que se requiere de mucho tiempo y esfuerzo la obtención de algoritmos semánticamente correctos.

Por otro lado, el aprendizaje colaborativo para Jonhson, Jonhson y Holubec 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 [44], al mismo tiempo, Brufee considera al aprendizaje colaborativo como la composición de elementos de consenso a través de la colaboración en el aprendizaje, la participación voluntaria en el proceso, el aprendizaje no fundacional, el cambio en la relación profesor-estudiante, la importancia del trabajo y diálogo entre pares donde se discute la autoridad del profesor y la validez de los contenidos gracias al método [45].

A su vez, la didáctica se define como la técnica que se emplea para manejar, de la manera más eficiente y sistemática, el proceso de enseñanza-aprendizaje [46]. Las estrategias didácticas contemplan las estrategias de aprendizaje y las estrategias de enseñanza donde las primeras consisten en un procedimiento o conjunto de pasos o habilidades que un estudiante adquiere y emplea de forma intencional como instrumento flexible para aprender significativamente y solucionar problemas y demandas académicas. Por su parte, las estrategias de enseñanza son todas aquellas ayudas planteadas por el docente, que se proporcionan al estudiante para facilitar un procesamiento más profundo de la información [47].

Por lo anterior y ante las necesidades de un mundo cambiante que demanda el refuerzo de procesos creativos e innovadores, la formación de ingenieros se convierte en uno de los ejes estratégicos para una nación que quiera insertarse en la sociedad del conocimiento y potenciar su desarrollo [48].

Lefranc en su artículo denominado Construyendo un

modelo de enseñanza en ingeniería concluye que las principales características del nuevo modelo de enseñanza-aprendizaje son:

- ✓ Proceso centrado individualmente en el estudiante
- ✓ Mayor presencialidad en los primeros años de la carrera.
- ✓ Conocimiento de las condiciones de ingreso y diferentes ritmos del aprendizaje.
- ✓ Ascendente progresión en ciencia y tecnología, en teoría y práctica;
- ✓ Armonía en los ciclos de nivelación, formación, integración y titulación.
- ✓ Prácticas dirigidas y evaluadas; la industria en la sala de clases; innovaciones en la enseñanza de las Ciencias Básicas
- ✓ Permanente responsabilidad ante los estudiantes y las familias que los respaldan [49].

Es así como en este artículo se presenta los resultados obtenidos al combinar aprendizaje colaborativo y modelado gráfico como una estrategia didáctica en la enseñanza para el análisis y diseño de procesos de captura y salida de datos en los cursos de fundamentos de programación en programas de pre grado en ingeniería para tres universidades de la ciudad de San Juan de Pasto.

2. Metodología

El proceso metodológico llevado cabo para el presente estudio toma 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.

La metodología planteada, se basó en dar a conocer al estudiante de primer curso de fundamentos de programación, la realización sistemática de una serie de pasos necesarios para la obtención de un producto de software que tenga algunos estándares metodológicos de construcción, por ello, dicha metodología inculca al estudiante la claridad que debe tener, independiente de la metodología de software que en un futuro seleccionarán para llevar a efecto proyectos de software, en la realización de unas fases definidas que inician desde la recolección de información y terminan en el planteamiento de un modelo de construcción que más tarde se convertirá en código fuente.

La propuesta de modelado gráfico concibe algunas fases para la enseñanza de los fundamentos de programación, haciendo énfasis en que el estudiante debe identificarlas claramente ya que son transversales a muchas metodologías de desarrollo de software existentes y cuyo propósito es el de familiarizarlo con los procesos desarrollados en los futuros cursos de Ingeniería de Software que deberá cursar. Entre las fases iniciales propuestas para un primer curso de fundamentación para programación se recomienda: Fase de recolección de requerimientos, fase de análisis, fase de diseño y fase de codificación (que inicialmente no será parte del presente estudio pero es necesario que el estudiante la conozca). En la tabla 1 se presenta el formato de requerimientos propuesto para un primer curso de introducción a la programación.

Tabla 1. Formato de recolección de requerimiento principal

Nombre del sistema	**Requerimiento principal**
Se asigna un nombre al enunciado dado por el docente	Se describe brevemente el objetivo del ejemplo

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.

Fig. 1. Diagrama de actividad

Apoyados en el modelo presentado por Collazos y Mendoza [50] [51], 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.

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 [51].

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 [51].

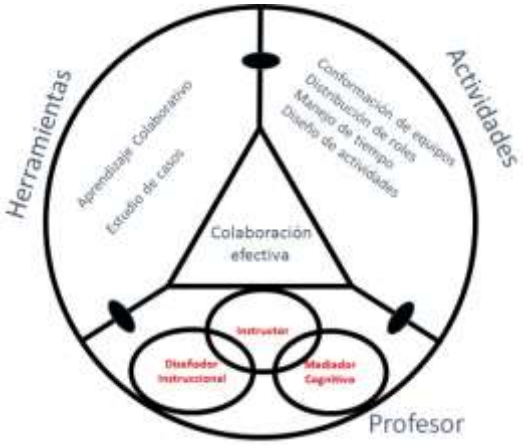

Fig. 2. Rol docente

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.

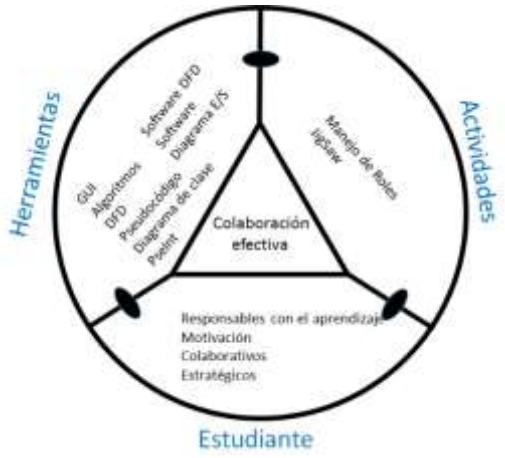

Fig. 3. Rol estudiante

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
- ✓ 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

La metodología de investigación planteada se desarrolló bajo el paradigma positivista por lo que se fundamentó en el conocimiento científico, con enfoque cuantitativo que permitió examinar datos de manera numérica, utilizando el método empírico analítico porque los datos fueron tratados con técnicas estadísticas y bajo un tipo de investigación correlacional que permitió medir el grado de relación que tuvieron las variables en estudio y con un diseño experimental como lo muestra la figura 4, donde G son los grupos de estudiantes experimentales y de control con quienes se llevó a cabo el estudio.

G_1	X	O_1
G_2	-	O_2
G_3	X	O_3
G_4	-	O_4
G_5	X	O_5
G_6	-	O_6

Fig. 4. Diseño experimental

La investigación fue realizada en seis grupos diferentes pertenecientes a tres instituciones de educación superior del municipio de Pasto (Colombia) para el primer curso de fundamentación de programación. Los grupos G_1, G_3 y G_5 corresponden a los grupos experimentales de cada institución y G_2, G_4 y G_6 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, O_1, O_2, O_3, O_4, O_5, O_6, fueron las post pruebas realizadas al final del tratamiento experimental tanto para los grupos experimentales como los de control.

El primer grupo experimental G_1 fue conformado por 33 estudiantes de la asignatura de Introducción a la programación de primer semestre de Ingeniería de Sistemas correspondientes al periodo académico II-2015 a quienes se les aplicó el tratamiento experimental X y finalmente se le aplicó una prueba posterior O_1. El grupo de control G_2 estuvo conformado por 27 estudiantes 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 O_2.

El segundo grupo experimental G_3 estuvo constituido por 11 estudiantes de la asignatura de Lógica Computacional de primer semestre de Ingeniería Electromecánica correspondientes al periodo académico II-2015 a quienes también se les aplicó el mismo tratamiento X con su correspondiente evaluación O_3 y su grupo de control G_4 fueron los 14 estudiantes del periodo académico I -2015 del mismo semestre y asignatura del grupo experimental a quienes tampoco se les aplicó tratamiento experimental, considerando también las notas obtenidas del tema en estudio O_4.

El tercer grupo experimental G_5 estuvo constituido por 29 estudiantes de la asignatura de fundamentos de programación de primer semestre de Ingeniería Electrónica correspondientes al periodo académico II-2015 a quienes también se les aplicó el mismo tratamiento X con su correspondiente evaluación O_5 y su grupo de control G_6 fueron los 33 estudiantes del periodo académico I -2015 del mismo semestre y asignatura del grupo experimental a

quienes tampoco se les aplicó tratamiento experimental, considerando también las notas obtenidas del tema en estudio O_6.

Una vez aplicado el tratamiento investigativo en cada grupo experimental (G_1, G_3 y G_5) 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 2.

Tabla 2. Promedio de notas grupos experimentales y de control

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

La tabla 3 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%.

Tabla 3. 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 Exp.	Grupo Cont.	Grupo Exp.	Grupo Cont.	Grupo Exp.	Grupo Cont.
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
Observa-cines	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	

De igual forma, en la tabla 3 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%.

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.

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, que por cierto fue sencilla de realizar dadas las características de los grupos experimentales por tratarse de estudiantes de primer semestre de pregrado en los cuales aún no existe la diferenciación de grupos marcados que se presentan generalmente con el paso de tiempo. Esto permitió que los estudiantes se acoplaran fácilmente al rol establecido el cual fue dinámico durante los diversos casos de estudio planteados.

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.

Finalmente y especialmente en el campo de la enseñanza para formación de futuros constructores de software, se hace necesario involucrar aspectos didácticos que contemplen acciones mediadas por tecnología que incluyan al estudiante de manera activa en la construcción de su lógica computacional, inculcándole la necesidad del autoaprendizaje en un ambiente colaborativo que mejore las actitudes y aptitudes del estudio y trabajo en equipo.

4. Conclusiones y trabajos futuros

El tratamiento experimental propuesto en esta investigación que consistió en proponer una estrategia didáctica de aprendizaje en un entorno colaborativo soportado por una fundamentación gráfica para la unidad de competencia denominada "captura y salida de datos" para un primer curso de fundamentos de programación demuestra que las calificaciones obtenidas por los tres grupos experimentales fueron numéricamente superiores a las obtenidas por sus respectivos grupos de control, lo cual demuestra la incidencia positiva del tratamiento experimental.

A pesar de que los tres grupos en los cuales se llevó a cabo la presente investigación eran de programas de pregrado diferentes pero adscritos a la facultad de Ingeniería, se pudo constatar que independiente del objeto de estudio de cada grupo, la estrategia didáctica planteada resultó exitosa en cada uno, concluyendo que la adecuada combinación de entornos colaborativos con entornos gráficos despiertan en el estudiante un especial interés.

En el estudio se pudo evidenciar que al mezclar la rigurosidad de unas teorías (en este caso las tres unidades de competencia) con procesos que se encuentren dentro de los esquemas de acción de los estudiantes (dinamización de actividades) 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.

Finalmente y especialmente en el campo de la enseñanza para formación de futuros constructores de software, se hace necesario involucrar aspectos didácticos que coloquen al estudiante como elemento central y activo del proceso de aprendizaje y que lo incluyan de una manera totalmente activa en la construcción de su lógica computacional, inculcándole la necesidad del autoaprendizaje en un ambiente colaborativo que mejore las actitudes y aptitudes del estudio y trabajo en equipo

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.

AGRADECIMIENTOS

Agradecimiento especial a la Institución universitaria CESMAG y a su Vicerrectora de Investigaciones PhD (c) María Eugenia Córdoba por el apoyo incondicional y de igual manera al los integrantes del grupo de investigación IDIS de la Universidad del Cauca.

Referencias

[1] J. Jiménez, C. Collazos, J. Hurtado, and W. Pantoja, "Estrategia colaborativa en entornos tridimensionales como estrategia didáctica de aprendizaje de estructuras iterativas en programación computacional" *Investigium IRE Ciencias Soc. y Humanas*, vol. 6, no. 2, pp. 80–92, 2015.

[2] M. González De Rivera Fuente and M. Paredes Velasco, "Aprendizaje con programación Colaborativa," pp. 1–33, 2008.

[3] M. Ben-Ari, "Journal of Computers in Mathematics & Science Teaching," vol. 20, pp. 24–73, 2001.

[4] C. . Selby and J. Woollard, "Computational Thinking: The Developing Definition.," 2010.

[5] L. Y. Gómez, "Competencias Mínimas En Pensamiento Computacional Que Debe Tener Un Estudiante Aspirante a La Media Técnica Para Mejorar Su Desempeño En La Media Técnica De Las Instituciones Educativas De La Alianza Futuro Digital Medellín," UNIVERSIDAD EAFIT, 2014.

[6] J. M. Wing, "Computational Thinking. It represents a universally applicable attitude and skill set everyone, not just computer scientists, would be eager to learn and use," *Commun. ACM*, vol. 49, No. 3, 2006.

[7] J. Valverde Berrocoso, M. R. Fernández Sánchez, and M. del Ć. Garrido Arroyo, "El pensamiento computacional y las nuevas ecolog{í}as del aprendizaje," *RED - Rev. Educ. a Distancia*, no. 46, pp. 1–18, 2015.

[8] E. Espino and C. González, "Estudio sobre diferencias de género en las competencias y las estrategias educativas para el desarrollo del pensamiento computacional," *Rev. Educ. a Distancia.*, no. 46, pp. 1–20, 2015.

[9] J. A. Hurtado, C. A. Collazos, S. T. Cruz, and O. E. Rojas, "Child Programming: una estrategia de aprendizaje y construcci{ó}n de Software basada en la l{ú}dica, la colaboraci{ó}n y la agilidad," *Rev. Univ. RUDIC*, vol. 1, no. 1, 2012.

[10] MIT, "Scratch." p. http://wiki.scratch.mit.edu, 2008.

[11] U. C. Mellon, "Alice," *http://alice.org/index.php*. 2003.

[12] G. Hernández, R. Jiménez, and Á. Martínez, "Creencias docentes sobre la importancia de la didáctica en la orientación de la enseñanza del primer curso de programación de computadoras," *Rev. Univ. docencia, Investig. e innovación*, vol. 2, pp. 87–103.

[13] C. Fracchia, N. Baeza, and A. Martins, "ECDIA : Entorno Colaborativo para el Diseño e Implementación de Algoritmos," 2011.

[14] J. A. Jiménez, M. A. Pavony Meneses, A. F. Álvarez, and Serna, "enseñanza de algoritmos y programación en ingeniería Integration environment of PBL and CSCL for teaching algorithms and programming in engineering," *Rev. Av. en Sist. e Informática*, vol. 5, no. 3, pp. 189–194, 2008.

[15] N. C. Colmenar, "ENTORNOS PARA ENSEÑAR PROGRAMACIÓN EN SECUNDARIA. NUEVOS ENFOQUES. Nieves Carralero Colmenar. IES Pedro Mercedes. Junta de Comunidades de Castilla-La Mancha. España.," pp. 1–13, 2011.

[16] J. A. Villalobos Salcedo, "Proyecto Cupi2 – una solución integral al problema de enseñar y aprender a programar 10°," pp. 1–37, 2009.

[17] R. Pérez Calderón, "Una Herramienta y Técnica p ara la E nseñanza de la P rogramación," 2008.

[18] M. Kölling, B. Quig, A. Patterson, and J. Rosenberg, "The BlueJ system and its pedagogy 1," vol. 13, no. 4, pp. 1–12, 2003.

[19] N. Myller, *Collaborative Software Visualization for Learning : Theory and Applications*. 2009.

[20] L. Concepts, A. Diwan, W. M. Waite, and M. H. Jackson, "PL-Detective : A System for Teaching Programming," 2005.

[21] M. Á. Redondo, "Aprendizaje en grupo de la programación mediante técnicas de colaboración distribuida en tiempo real," pp. 351–357, 2004.

[22] M. Esteves and a. J. Mendes, "A simulation tool to help learning of object oriented programming basics," *34th Annu. Front. Educ. 2004. FIE 2004.*, pp. 811–816, 2004.

[23] C. Jo and A. J. Arnold, "A Portable and Collaborative Distributed Programming Environment The architecture of DPE," 2003.

[24] N. Truong, P. Bancroft, and P. Roe, "A Web Based Environment for Learning to Program," vol. 16, 2003.

[25] U. University Carnegie Mellon, "Alice.org," 2003. [Online]. Available: http://www.alice.org/index.php?page=what_is_alice/what_is_alice. [Accessed: 22-Apr-2014].

[26] A. Vizcaíno, J. Contreras, J. Favela, and M. Prieto, "An Adaptive , Collaborative Environment to Develop Good Habits in Programming," pp. 262–271, 2002.

[27] H. Suzuki and H. Kato, "Identity formation / transformation as the process of collaborative learning through AlgoArena," *CSCL '97 Proc.*, pp. 280–289.

[28] W. S. Humphrey, *Introduction to the Personal Software Process*, Addison We. 1997.

[29] D. W. Jonhson, R. Jonhson, and E. Holubec, "Circles of learning (4th ed.)," 1993.

[30] I. T. Monterrey, "Aprendizaje Colaborativo, técnicas didácticas," *Programa de Desarrollo de Habilidades Docentes*, 2008.

[31] C. Collazos, L. Guerrero, and A. Vergara, "Aprendizaje Colaborativo: un cambio en el rol del profesor," 2012.

[32] W. Y. Hsu, "Online education on campus: A technological frames perspective on the process of technology appropriation," London, University of London, 2002.

[33] A. Escribano González, "Aprendizaje cooperativo y autónomo en la enseñanza universitaria," *Enseñanza Teach. Rev. Interuniv. didáctica*, vol. 13, pp. 89–104, 1995.

[34] G. Lavigne, M. P. Vasconcelos Ovando, J. O. Sandoval, and L. M. Salas, "Exploración preliminar del aprendizaje colaborativo dentro un entorno virtual," vol. 12, no. 3, pp. 1–20, 2012.

[35] B. . Joyce and M. Weil, *Modelos de enseñanza*. Madrid: Anya, 1985.

[36] L. S. Shulman, "Those who understand: knowledge growth in teaching," *Educ. Res.*, vol. 15, no. 2, pp. 4–14, 1986.

[37] Y. Chevallard, *Del sabio al saber enseñado*. Buenos Aires: Aiqué, 1991.

[38] R. Porlan, *Qué y cómo enseñar desde una perspectiva constructivista. En Mariño, G. (Comp.)*. Bogotá: Dimensión Educativa, 1995.

[39] G. Sacristán and G. Pérez, *Comprender y transformar la enseñanza*, 3rd ed. Madrid: Morata, 1994.

[40] P. García and F. Angulo, "Un modelo didáctico para la formación inicial del profesorado de ciencias," *Rev. Interuniv. Form. Profr.*, vol. 17, no. 001, pp. 37–49, 2003.

[41] T. Rodríguez, *Teorías y modelos de enseñanza: posibilidades y límites*. Milenio Lleida, 1999.

[42] C. Chaves and M. M. Rosero, "Procesos Metacognitivos En Programación De Teaching Approach and Its Relationship With Meta-Cognitive," *Rev. Educ. en Ing.*, vol. 9, pp. 1–12, 2014.

[43] N. Moroni and S. Perla, "Estrategias para la enseñanza de la programación," *Prim. Jornadas en Educ. en Informática y TICS. JEITICS.*, 2005.

[44] D. W. Johnson and F. P. Johnson, "Joining Together: Group Theory and Group Skills," in *Needham Heights*, MA: Allyn & Bacon, 1997.

[45] K. A. Bruffee, *Collaborative Learning Higher Education, Interdependence and the Authority of Knowledge*. Baltimore: The Johns Hopkins University Press, Second Edition, 1999.

[46] F. De La Torre, *12 lecciones de pedagogía, educación y didáctica*. Mexico: Alfaomega, 2005.

[47] F. Díaz and G. Hernández, *Estrategias Docentes para un Aprendizaje Significativo: Una interpretación constructivista*. México D.F.: McGrawHill Interamericana, 1999.

[48] K. P. Rodriguez Serrano, M. A. Maya Restrepo, and J. S. Jaén Posada, "Educación en Ingenierías: de las clases magistrales a la pedagogía del aprendizaje activo," *Ing. y Desarro.*, vol. 30, no. 1, pp. 125–142, 2012.

[49] E. Lefranc H, "Building a Model for Teaching on Engineering," *Pharos*, vol. 12, no. 1, 2005.

[50] E. Aronso, N. Blaney, C. Stephan, J. Sikes, and M. Snapp, *The Jigsaw Classroom*. Beverly Hills: Sage, 1978.

[51] C. Collazos and J. Mendoza, "Cómo aprovechar el 'aprendizaje colaborativo' en el aula," *Educ. y Educ.*, vol. 9, no. 2, pp. 61–76, 2006.