



UNIVERSIDAD NACIONAL DE COLOMBIA

Prototipo de herramienta computacional para el aprendizaje colaborativo de programación de computadores

Jhon Alexis Mendez Lara

Universidad Nacional de Colombia
Facultad de ingeniería
Departamento de Ingeniería de Sistemas y Computación
Bogotá, Colombia
2021

Prototipo de herramienta computacional para el aprendizaje colaborativo de programación de computadores

Jhon Alexis Mendez Lara

Tesis presentada como requisito parcial para optar al título de:
Magíster en Ingeniería - Ingeniería de Sistemas y Computación

Director:

Felipe Restrepo Calle, Ph.D.

Co-Director:

Jhon Jairo Ramírez Echeverry, Ph.D.

Línea de Investigación:

Computación Aplicada - Educación en Ingeniería

Grupo de Investigación:

PLaS - Programming Languages and Systems

Universidad Nacional de Colombia

Facultad de ingeniería

Departamento de Ingeniería de Sistemas y Computación

Bogotá, Colombia

2021

Dedico este trabajo a mi esposa Deysy; sin sus consejos, cuidados y ánimo no hubiese logrado esta meta personal.

Señor de los Milagros, porque te amo, me consagro a tu servicio con mi familia, mis seres queridos, mis trabajos, estudios, problemas y alegrías.

Agradecimientos

Quiero agradecer a Dios por darme la oportunidad de poder continuar con esta meta personal.

A mis maestros Felipe Restrepo Calle y Jhon Jairo Ramírez Echeverry por su paciencia, conocimiento y buenos consejos. El tiempo que invirtieron en mí se los voy a agradecer toda la vida.

A los integrantes del grupo de investigación PLaS (*Programming Languages and Systems*) del Departamento de Ingeniería de Sistemas e Industrial, por su apoyo y contribuciones durante la realización de esta tesis; especialmente agradezco a los ingenieros: Giovanni Hidalgo Suarez, Stephanie Torres Jimenez, Cristian Gonzalez Carrillo y Karen Cuervo Cely.

A la Universidad Nacional de Colombia por darme la oportunidad de cumplir el sueño de crecer profesionalmente en su Facultad de Ingeniería. Especialmente agradezco a sus estudiantes, quienes me brindaron su tiempo y colaboración en la realización del proyecto.

Resumen

El aprendizaje colaborativo asistido por computador CSCL (del inglés Computer Supported Collaborative Learning) es una propuesta educativa que busca que los estudiantes puedan alcanzar sus objetivos de aprendizaje a través de la interacción en grupo mediada por la tecnología. En el área de la programación de computadores, algunos estudios demuestran un aumento de la participación y mejores resultados en la resolución de problemas de programación por parte de los estudiantes que participan en ambientes de aprendizaje CSCL. En entornos universitarios, los cursos introductorios de programación de computadores buscan preparar a los estudiantes en la resolución de problemas a partir del uso de un lenguaje de programación; para ello, se han creado diferentes herramientas bajo el enfoque CSCL, las cuales sirven de espacios en donde los estudiantes autorregulan su aprendizaje, participan voluntariamente, discuten ideas y evalúan el trabajo de los demás. Sin embargo, según las implementaciones de software revisadas en la literatura, aún existe la necesidad de crear herramientas colaborativas no sólo en las etapas tempranas del proceso de resolución de problemas de programación, sino también durante el proceso de evaluación formativa de la resolución del problema mediante la colaboración de los compañeros. En este contexto, el objetivo de esta tesis es presentar la herramienta computacional denominada UNColab para apoyar la resolución de problemas de programación por medio de la colaboración entre estudiantes durante una etapa de evaluación por pares. Se realizó la implementación de un prototipo de UNColab, el cual fue probado durante la realización de una actividad de aprendizaje colaborativo en un curso de Programación de Computadores con 27 estudiantes de la Universidad Nacional de Colombia. Esta herramienta se desarrolló conforme a un guion de aprendizaje colaborativo, el cual permitió estructurar una dinámica de clase en dos actividades del curso; y la aplicación de la metodología de desarrollo orientada a prototipos, que facilitó la organización de cada una de las actividades de desarrollo de software. Los resultados de esta experiencia sugieren que los estudiantes perciben este tipo de herramientas como favorables en la enseñanza-aprendizaje de la programación de computadores. Además, sugirieron diferentes mejoras que podrían abrir las posibilidades a la realización de otros estudios. Por lo tanto, este trabajo contribuye al área de investigación por medio del desarrollo de una herramienta que permite apoyar la enseñanza de la programación de computadores en ambientes colaborativos. Asimismo, la validación de la herramienta en un curso de programación de computadores permite entender de una mejor manera los beneficios y retos que supone la utilización de herramientas para el aprendizaje colaborativo asistido por computador.

Palabras clave: aprendizaje colaborativo, aprendizaje colaborativo asistido por computador, programación de computadores.

Abstract

Prototype of computational tool for collaborative learning of computer programming

Computer-Supported Collaborative Learning (CSCL) is an educational proposal that seeks to enable students to achieve their learning goals through technology-mediated group interaction. In the area of computer programming, some studies show increased participation and better results in solving programming problems by students who participate in CSCL learning environments. In university environments, introductory computer programming courses seek to prepare students to solve problems through the use of a programming language; for this purpose, different tools have been created under the CSCL approach, which serve as spaces where students self-regulate their learning, participate voluntarily, discuss ideas and evaluate each other's work. However, according to the software implementations reviewed in the literature, there is still a need to create collaborative tools not only in the early stages of the programming problem solving process, but also during the formative evaluation process of the problem solving through peer collaboration. In this context, the objective of this thesis is to present the computational tool called UNColab to support programming problem solving through collaboration among students during a peer evaluation stage. A prototype of UNColab was implemented and tested during a collaborative learning activity in a Computer Programming course with 27 students from the Universidad Nacional de Colombia. This tool was developed according to a collaborative learning script, which allowed structuring a class dynamic in two activities of the course; and the application of the prototype-oriented development methodology, which facilitated the organization of each of the software development activities. Results of this experience suggest that students perceive this type of tools as favorable in the teaching-learning of computer programming. In addition, they suggested different improvements that could open the possibilities for further studies. Therefore, this work contributes to the research area through the development of a tool that allows supporting the teaching of computer programming in collaborative environments. Also, the validation of the tool in a computer programming course allows a better understanding of the benefits and challenges of using tools for computer-assisted collaborative learning.

Keywords: collaborative learning, computer supported collaborative learning, computer programming.

Esta tesis de maestría se sustentó el 15 de diciembre de 2021 a las 02:00 pm,
y fue evaluada por los siguientes jurados:

José Daniel Rodríguez Munca Msc.
Universidad Manuela Beltrán

Victor Andres Bucheli Guerrero Ph.D.
Universidad del Valle

Contenido

Agradecimientos	vii
Resumen	ix
Abstract	x
1 Introducción	2
1.1 Motivación	2
1.2 Trabajos relacionados	3
1.3 Problema de investigación	5
1.4 Objetivos	7
1.5 Alcance	7
1.6 Estructura de la tesis	8
2 Marco conceptual y trabajos relacionados	9
2.1 Marco conceptual	9
2.1.1 Aprendizaje colaborativo	9
2.1.2 Técnicas de aprendizaje colaborativo	10
2.1.3 Evaluación formativa y aprendizaje colaborativo	11
2.1.4 Aprendizaje colaborativo asistido por computador	14
2.1.5 Guiones de aprendizaje colaborativo asistido por computador	15
2.1.6 Prototipos de software	19
2.2 Trabajos relacionados	20
3 UNColab: Herramienta computacional para el aprendizaje colaborativo de programación	25
3.1 Diseño de UNColab a partir de un guion CSCL	25
3.1.1 Determinación de los objetivos de aprendizaje	27
3.1.2 Configuración del flujo de actividad	27
3.1.3 Configuración de actividades y roles	28
3.1.4 Determinación de recursos	31
3.2 Desarrollo de la herramienta UNColab mediante la metodología orientada a prototipos	34
3.2.1 Definición de especificaciones	34
3.2.2 Diseño conceptual	36

3.2.3	Desarrollo del prototipo	36
3.2.4	Pruebas	42
3.2.5	Implementación	44
4	Diseño del estudio	47
4.1	Participantes	48
4.2	Elaboración de actividades	49
4.3	Aplicación de pruebas	51
4.4	Aplicación de instrumentos de validación	52
4.4.1	Instrumentos de recolección de datos	54
4.5	Análisis de resultados	56
5	Resultados	57
5.1	Resolución de problemas de programación a través de la colaboración	57
5.2	Evaluación de la colaboración	58
5.3	Percepciones acerca del uso de UNColab	64
6	Discusión de resultados	70
7	Conclusiones y trabajos futuros	76
7.1	Conclusiones	76
7.2	Productos académicos	77
7.3	Trabajos futuros	78
	Bibliografía	80

Lista de Figuras

2-1	Clasificación de patrones en aprendizaje virtual, basada en (Hernandez-Leo et al., 2006)	16
2-2	Estructura de patrones de guiones CSCL basado en (Hernandez-Leo et al., 2006)	17
2-3	Metodología de prototipado basada en Pressman (2002)	20
3-1	Proceso de diseño de guiones CSCL con reconocimiento de evaluación, basado en (Villasclaras-Fernández et al., 2009)	26
3-2	Técnica de aprendizaje colaborativo Pensar-Par-Compartir, basado en (Hernández-leo and Asensio-pérez, 2019)	28
3-3	Técnica de asignación de roles: Programadores y Novatos, adaptado de (Hidalgo et al., 2021)	30
3-4	Diseño de herramienta computacional para el aprendizaje colaborativo de programación de computadores	33
3-5	Diseño conceptual UNColab	37
3-6	Arquitectura UNColab	38
3-7	Funcionalidad de clasificación de estudiantes: ejemplo de código fuente	39
3-8	Funcionalidad de asignación de roles: ejemplo de código fuente	40
3-9	Funcionalidad de notificar rol: ejemplo de código fuente	41
3-10	Funcionalidad de seleccionar estudiante: ejemplo de código fuente	41
3-11	Funcionalidad de escritura de comentarios: ejemplo de código fuente	42
3-12	Funcionalidad de evaluación a través de enlace a Google Forms: ejemplo de código fuente	43
3-13	Archivo de integración de componentes UNCode	43
3-14	Captura de pantalla: prueba de integración UNColab	44
3-15	Capturas de pantalla: prueba de componentes UNColab	45
3-16	Implementación UNColab en Google Cloud	46
4-1	Metodología del diseño del estudio	47
4-2	Flujo de actividades del caso de estudio	50
5-1	Distribución de cantidad de mensajes intercambiados entre estudiantes en el componente de chat durante las dos intervenciones realizadas.	59
5-2	Respuestas obtenidas del instrumento de evaluación de colaboración “Programadores” por parte de estudiantes con rol de “Programador”.	59

5-3	Respuestas obtenidas del instrumento de evaluación de colaboración “Novatos” por parte de estudiantes con rol de “Novato”.	60
5-4	Colaboración en la resolución de problemas de programación: resultado de análisis de contenido	62
5-5	Respuestas obtenidas de la encuesta de percepciones sobre la herramienta UNColab	65
5-6	UNColab: resultado de análisis de contenido	66

Lista de Tablas

2-1	Características de entornos colaborativos de evaluación formativa	12
2-2	Medios, técnicas e instrumentos de evaluación propuestos por Hamodi et al. (2018) .	13
2-3	Herramientas basadas en CSCL	18
2-4	Trabajos relacionados sobre herramientas para el aprendizaje colaborativo de programación de computadores	23
3-1	Funcionalidades y recursos para el diseño de un script CSCL en programación de computadores	31
3-2	Definición de especificaciones UNColab	35
4-1	Participantes en el estudio	49
4-2	Temáticas de la asignatura “Programación de computadores”	50
4-3	Distribución de actividades del caso de estudio	51
4-4	Cronograma de actividades de las intervenciones en el curso	52
4-5	Datos cuantitativos recolectados en UNColab	53
4-6	Instrumento de evaluación de la colaboración - Programadores	55
4-7	Instrumento de evaluación de la colaboración - Novatos	55
4-8	Encuesta de percepciones de la herramienta colaborativa UNColab	56
5-1	Resolución de ejercicios a través de la colaboración: resultados	58

1 Introducción

Esta tesis está enfocada en el estudio del aprendizaje colaborativo de la programación de computadores por medio de una estrategia de evaluación formativa entre pares soportada por una herramienta computacional. Para este fin, se desarrolló y validó un prototipo de software que sirvió como herramienta mediadora de interacción entre los participantes del estudio. Este prototipo se desarrolló con base en un guion de aprendizaje colaborativo asistido por computador, una técnica de formación de grupos y una metodología de desarrollo de software orientada a prototipos.

1.1. Motivación

Los cursos introductorios de programación de computadores tienen como principal objetivo brindar a los estudiantes las herramientas, métodos y metodologías necesarias para desarrollar soluciones de software a partir del uso de un lenguaje de programación. Estos cursos introductorios buscan preparar al estudiante en temas genéricos de la programación, tales como: declaración de variables, condicionales, bucles, funciones, entre otras. La manera más utilizada por los profesores para medir el nivel de éxito de los participantes de un curso es mediante el planteamiento de problemas o retos de programación, estos retos buscan que los estudiantes apliquen los conceptos vistos y generen soluciones que resuelvan diferentes casos o condiciones dados en el problema.

Seguidamente, durante el proceso de desarrollo de estas soluciones los estudiantes que se enfrentan a este tipo de retos de programación pueden encontrarse en diferentes situaciones de acuerdo al nivel de comprensión de los temas vistos, según Mohammed et al. (2017a) se pueden ver las siguientes: el no entendimiento del problema, lo que impide que se empiece a desarrollar una solución; frustración al intentar varias veces una solución sin éxito, lo que podría producir que el estudiante pierda motivación (Marcolino and Barbosa, 2017); o simplemente, que el estudiante pueda resolver el problema en los primeros intentos. Los factores que se dan para las primeras dos situaciones están ubicados en la literatura con diferentes categorías sobre problemas al momento de aprender a programar. Dentro de estas categorías, las que más se destacan, como limitaciones en el aprendizaje de la programación, son los conceptos básicos y la aplicación directa de éstos. Dentro de los conceptos básicos se presentan temas como recursión, sentencias de repetición, matrices, clases y los objetos (Marcolino and Barbosa, 2017).

Bajo esta perspectiva, han surgido diferentes propuestas pedagógicas que buscan mejorar el rendi-

miento de los estudiantes que presentan dificultades al momento de resolver un problema de programación. Una de estas propuestas, de la que se habla en este trabajo, es el aprendizaje colaborativo asistido por computador CSCL (del inglés *Computer Supported Collaborative Learning*), el cual propone principalmente dos ideas: la primera, que son los estudiantes quienes se deben colaborar para cumplir sus objetivos de aprendizaje personales y grupales; la segunda, que se deben adoptar tecnologías que sirvan de instrumentos mediadores en el proceso de aprendizaje (Vinagre Laranjeira, 2009). En este sentido, un escenario de enseñanza-aprendizaje de programación basado en CSCL fomentará el uso de herramientas que permitan que aquellos estudiantes que tengan dificultades en resolver un problema de programación puedan recibir ayuda de uno o varios de sus compañeros con mejor rendimiento.

Las herramientas CSCL en programación de computadores han tenido buenos resultados respecto al rendimiento de los estudiantes y el desarrollo de algunas habilidades blandas. Por ejemplo, en el estudio realizado por Serrano-Cámara et al. (2016) se reconoce una mayor eficiencia educativa y motivación en los estudiantes al usar una aplicación que facilita la comunicación en el curso. Estudios como Laakso et al. (2018); Hwang et al. (2017) reportan un aumento en la capacidad de pensamiento crítico, al involucrar a los estudiantes en el proceso de evaluación de programas mediante la revisión entre pares. Adicionalmente, existen estudios de aplicación de CSCL más formales como los vistos en De Faria et al. (2006); Tsompanoudi and Satratzemi (2014); Debdi et al. (2015); Soh et al. (2008), que se enfocan en el uso de guiones de CSCL para generar herramientas de soporte en programación de computadores, los cuales reportan facilidades en el proceso de asignar de las tareas a los estudiantes, una mejor formación de grupos de aprendizaje y un aumento en la motivación. Estas herramientas al mismo tiempo que presentan buenos resultados también sugieren experimentar más con los guiones CSCL con el fin de medir su efectividad, especialmente en la asignación de grupos y de cómo estos afectan la colaboración y contribución entre estudiantes al momento de resolver problemas de programación.

Las aplicaciones CSCL orientadas hacia la enseñanza-aprendizaje de la programación de computadores fomentan espacios en donde se pueden involucrar más a los estudiantes en su proceso de aprendizaje. A partir del uso de herramientas que faciliten la colaboración durante la resolución de problemas se puede mejorar la calidad del aprendizaje de programación al permitir que los estudiantes reconozcan errores y aprendan de ellos (Al-Khalifa and Devlin, 2020).

1.2. Trabajos relacionados

El aprendizaje colaborativo es un enfoque pedagógico el cual pretende lograr que los estudiantes puedan alcanzar sus objetivos de aprendizaje a través de la interacción en grupo. Comprende el aprendizaje como un acto social en el que los estudiantes aprenden mediante algunos elementos de comunicación como el debate, la discusión y el diálogo (Bruffee, 1983). En la actualidad, estos elementos de

comunicación se fomentan mediante el uso de tecnología, principalmente internet y las plataformas digitales, lo que ha permitido que surjan nuevas propuestas pedagógicas como el CSCL.

Tanto el aprendizaje colaborativo como el CSCL han sido aplicados en diferentes áreas del conocimiento, buscando mejorar las destrezas conversacionales, la participación e integración de los estudiantes dentro de un aula de clase; evitando así las dificultades de los ambientes de aprendizaje tradicionales en los cuales se refleja el aislamiento y la competencia (Bruffee, 1978). Estos últimos aspectos no son ajenos en los cursos introductorios de programación de computadores. El enfoque CSCL es uno de los más prometedores para la transformación de la educación en programación, debido al uso frecuente de recursos basados en TIC (tecnologías de la información y las comunicaciones) las cuales apoyan la realización de simulaciones, proyectos, resolución de problemas, entre otras.

Según Vinagre Laranjeira (2009), dependiendo de los objetivos de aprendizaje y el perfil del estudiante como factores principales, se pueden emplear diferentes herramientas TIC que pueden ser tanto síncronas como asíncronas. En función de la tarea, un docente puede incluir un amplio conjunto de herramientas dentro de las cuales están: los correos electrónicos, chats, blogs, sistemas de versionamiento, redes sociales, entre otros. En el caso de la enseñanza-aprendizaje de programación de computadores, se han utilizado herramientas colaborativas más especializadas como los sistemas de versionamiento de código, jueces virtuales o entornos de desarrollo IDE (del inglés *Integrated Development Environment*). Un ejemplo de este tipo de herramientas se puede ver en el trabajo propuesto por (Zakiah and Fauzan, 2016), en el que se propone utilizar la plataforma Github como medio de colaboración entre estudiantes de informática para el desarrollo de los proyectos de programación planteados en clase. En este estudio se evidencia la importancia de definir una dinámica de colaboración entre estudiantes, ya que presenta una propuesta de guía de colaboración que le permitió a cada estudiante conocer cómo participar dentro de la actividad de clase a través de la herramienta seleccionada.

Las herramientas computacionales establecidas como CSCL por lo general se basan en guiones de aprendizaje colaborativo los cuales permiten estructurar las actividades a realizar durante una clase. Estos guiones se enfocan principalmente en: determinar una secuencia de pasos para organizar el trabajo colaborativo, establecer qué actividades se realizarán durante la clase y asignar roles o modos de participación que tendrán los estudiantes al momento de realizar una colaboración; adicionalmente, definen el papel que desempeñará una herramienta computacional para apoyar la colaboración entre estudiantes (Hernandez-Leo et al., 2006).

Seguidamente, en la enseñanza-aprendizaje de programación de computadores se encuentran en la literatura diferentes trabajos relacionados con el desarrollo de herramientas computacionales, basadas en guiones de aprendizaje colaborativo, que apoyan las actividades de resolución de problemas de programación. En Tsompanoudi et al. (2016) se presenta un componente para el IDE Eclipse el cual permite resolver tareas de programación de manera colaborativa a partir de la configuración de un tipo

de tarea, la asignación de grupos de trabajo, la distribución de la tarea, un tipo y modo de interacción (síncrona o asíncrona) y un tiempo determinado para cada actividad. En el trabajo de Fukuma et al. (2017) se propone un componente de software para la herramienta de aprendizaje de programación *Scratch*, este componente se enfocó en la formación de grupos y en una dinámica de participación colaborativa en tiempo real, permitiendo a cada estudiante observar las soluciones planteadas por los demás integrantes del curso. En el estudio de Serrano-Cámara et al. (2016) se presenta una aplicación de escritorio y móvil que permite a los estudiantes revisar y proponer diferentes soluciones de código fuente sobre un problema de programación propuesto por el docente, encontrando buenos resultados respecto a la colaboración a través de comentarios. Estudios como el de Laakso et al. (2018) proponen integrar elementos de calificación automática de código para apoyar la resolución de ejercicios de manera colaborativa, hallando buenos resultados respecto al uso de la revisión de código entre pares. En Soh et al. (2005) se presenta un software que sirve de apoyo en los laboratorios de cursos introductorios de programación. Las funciones principales de este sistema parten también de guion de aprendizaje colaborativo y una técnica denominada *JigSaw*¹, destacan la importancia de usar un clasificador de actividades y funciones de formación de grupos como elementos indispensables en el aprendizaje colaborativo de programación. Finalmente, estudios como Debdi et al. (2015); Hwang et al. (2017) proponen herramientas que contienen funciones de evaluación entre pares las cuales permiten que cada estudiante emita una opinión y realimentación sobre el código fuente de un compañero; estos estudios encontraron buenos resultados respecto a mejoras en la escritura de código y una mejor comprensión de los ejercicios a través de la realimentación anónima.

Los estudios consultados intentan resolver diferentes problemas relacionados con el aprendizaje de la programación de computadores. En común, todos identifican dificultades en los estudiantes para comprender el contexto de un problema a resolver; adicional a ello, también identifican algunas limitaciones en conceptos básicos como las estructuras repetitivas, funciones y matrices. Asimismo, sugieren experimentar más con guiones de aprendizaje buscando medir la efectividad de los mismos en la enseñanza-aprendizaje de la programación de computadores.

1.3. Problema de investigación

El aprendizaje colaborativo asistido por computador (CSCL) como propuesta pedagógica pretende llenar vacíos en donde la educación tradicional no ha podido llegar; permitiéndole a un estudiante, de cualquier disciplina, la posibilidad de autorregular su aprendizaje a partir de las experiencias y conocimientos compartidos en grupo, adquiriendo de esta manera competencias tanto disciplinares como genéricas (Vinagre Laranjeira, 2009).

Los estudios citados en la Sección 1.2 presentaron diferentes herramientas bajo el enfoque CSCL

¹Jigsaw es un patrón pedagógico que permite organizar un flujo de aprendizaje colaborativo en clase. Consiste en formar varios grupos de estudiantes con el fin de que resuelven un mismo problema y luego discutan la solución.

que apoyan la enseñanza-aprendizaje de programación de computadores. Estos estudios evidencian buenos resultados respecto al uso de este tipo de herramientas en la resolución de problemas de programación, al permitirle a cada estudiante discutir ideas, participar y juzgar las soluciones de los demás integrantes. Sin embargo, estas implementaciones de software revisadas no consideran una realimentación exacta sobre la tarea. Generalmente, se enfocan solo en las etapas tempranas de la solución de un problema y no en todo el proceso formativo; como por ejemplo el planteamiento del mismo o la revisión de conocimientos previos. Según el estudio de Marcolino and Barbosa (2017), este es uno de los problemas que más afecta a los estudiantes de programación. En esta dirección, algunos de estos estudios han propuesto herramientas de evaluación automática de código que le permiten al estudiante recibir realimentación a partir de la identificación de diferentes errores en el desarrollo de una tarea; no obstante, autores como Ala-Mutka (2005) sugieren que es necesario no perder el juicio humano cuando se requiere realimentación de una tarea de programación, ya que las opciones de ayuda podrían ser más amplias con una persona.

La actividad de realimentar tareas de programación, en principio, es una labor del docente de una asignatura; sin embargo, esta actividad puede verse afectada debido a la cantidad considerable de estudiantes que se reflejan en estos cursos introductorios de programación. Para cada ejercicio se deben tener en cuenta aspectos como la sintaxis, la semántica, la eficiencia u otros atributos de calidad que un docente crea necesario revisar (Mohammed et al., 2017b).

Adicionalmente, autores como Debdi et al. (2015); Tsompanoudi and Satratzemi (2014); Hashim and Harun (2014); Cabrera et al. (2017); Dittmar and Forbrig (2018); Silva et al. (2020) manifiestan la necesidad de reportar más experiencias con guiones y herramientas CSCL que permitan conocer cuáles técnicas o estrategias son adecuadas para obtener una mejor distribución y seguimiento de tareas, como también una formación de grupos más efectivos. Asimismo, los autores concuerdan en que se deben evaluar de manera aislada diferentes características de las herramientas usadas para colaborar con el fin de identificar cuáles pueden beneficiar más a los estudiantes al momento de resolver un ejercicio de programación.

Por consiguiente, es importante evaluar y proponer otras opciones de herramientas de aprendizaje colaborativo que contengan opciones que le permitan al estudiante recibir una realimentación más exacta sobre la tarea a partir de actividades colaborativas que apoyen el proceso formativo dentro de los cursos introductorios de programación de computadores. Al involucrar a cada estudiante dentro de un proceso de realimentación de tareas se estaría apoyando algunas actividades que son propias de una evaluación formativa, esto según Flórez Ochoa (2005) le permitiría a cada estudiante reflexionar sobre su proceso de aprendizaje y, al mismo tiempo, apoyar el seguimiento sobre el trabajo de los demás miembros del curso. Poder conocer los efectos de este tipo de herramientas en la resolución de problemas de programación aportaría a la comunidad CSCL nuevas evidencias que demuestren los beneficios de usar el aprendizaje colaborativo mediado por tecnología.

Por lo tanto, esta tesis de maestría se enfocó en contestar a la siguiente pregunta de investigación:

¿Cuáles son los efectos en la resolución de problemas de programación de computadores ocasionados por el uso de una herramienta computacional colaborativa que apoye la evaluación formativa entre pares?

1.4. Objetivos

Objetivo general:

Desarrollar y evaluar un prototipo de herramienta computacional para el aprendizaje colaborativo de programación de computadores que apoye el proceso de resolución de problemas mediante la evaluación formativa entre pares.

Objetivos específicos:

1. Diseñar un prototipo del software para el aprendizaje colaborativo de programación de computadores con base en una técnica de aprendizaje colaborativo que permita aplicar la evaluación formativa entre pares.
2. Implementar un prototipo de software para el aprendizaje colaborativo de programación de computadores mediante la aplicación de la metodología de desarrollo de software orientada a prototipos.
3. Validar el prototipo desarrollado en un curso de programación de computadores.

1.5. Alcance

En el planteamiento de la metodología de investigación se propuso realizar el diseño de un prototipo de software cuyas funcionalidades dieran soporte al uso de un guion de aprendizaje colaborativo asistido por computador de acuerdo al modelo propuesto por Hernández-leo and Asensio-pérez (2019) en el que se proponen tres actividades principales para llevar a cabo un ejercicio de aprendizaje colaborativo asistido por computador, estas son: determinar una secuencia de pasos para organizar el trabajo colaborativo, establecer qué actividades se realizarán durante la clase y asignar roles o modos de participación. Para cumplir con cada una de estas actividades se realizaron las tareas de análisis, diseño, codificación y pruebas. La idea principal de este estudio se basa en la implementación del prototipo diseñado, el cual tiene las opciones básicas de comunicación entre estudiantes de manera sincrónica y asincrónica, formación de grupos y un instrumento que permite realizar una evaluación de la colaboración. Como resultado de esta implementación se realizó la validación de la herramienta

en un grupo de la asignatura Programación de Computadores en la Universidad Nacional de Colombia durante el segundo periodo académico de 2020. Esto incluyó la puesta a punto del prototipo, la integración del mismo en algunas actividades de la asignatura, el análisis de las interacciones entre estudiantes y la valoración final de la herramienta por parte de los estudiantes.

Adicionalmente, cabe mencionar que la implementación de la herramienta desarrollada dará soporte a una actividad de evaluación formativa entre pares. Esta actividad está concebida dentro del guion de aprendizaje colaborativo usado. La evaluación formativa según Black and Wiliam (2009) tiene como finalidad detectar las dificultades y al mismo tiempo los progresos de los estudiantes durante el desarrollo de una tarea. A partir del uso de las opciones básicas de comunicación de la herramienta propuesta se establece en este estudio que los estudiantes pueden hacer actividades de apoyo a la evaluación formativa a medida que se establece una actividad colaborativa entre pares, debido a que estas opciones están desarrolladas para que los estudiantes con buenos resultados en la resolución de un ejercicio de programación puedan detectar dificultades y hacer un seguimiento a aquellos estudiantes que presenten inconvenientes en el planteamiento de la solución.

1.6. Estructura de la tesis

Los siguientes capítulos de esta Tesis de Maestría están estructurados de la siguiente manera:

- **Capítulo 2** presenta el marco conceptual y los trabajos relacionados con la investigación.
- **Capítulo 3** presenta las actividades relacionadas con el diseño la herramienta computacional propuesta y el desarrollo del prototipo.
- **Capítulo 4** presenta el diseño del estudio realizado para validar el prototipo desarrollado en un curso de programación de computadores.
- **Capítulo 5** presenta los resultados obtenidos tras la validación.
- **Capítulo 6** presenta la discusión referente a los resultados de la investigación.
- **Capítulo 7** presenta las conclusiones y contribución del trabajo realizado, además de algunas ideas para futuras investigaciones.

2 Marco conceptual y trabajos relacionados

Este capítulo presenta una recopilación de teorías y estudios de investigación referentes a esta tesis. Inicia con una sección enfocada en mostrar los conceptos necesarios para dar un contexto sobre la temática global de la investigación: aprendizaje colaborativo, CSCL y CSCL en programación de computadores. Seguidamente, una sección que presenta un conjunto de trabajos de investigación relacionados con la temática de estudio.

2.1. Marco conceptual

2.1.1. Aprendizaje colaborativo

El aprendizaje colaborativo (AC) responde al cambio contextual sobre cómo se lleva a cabo un proceso de aprendizaje, surge como un paradigma educativo donde un grupo de personas participa de manera voluntaria para alcanzar un objetivo común, propiciando el debate, la discusión y el diálogo (Barkley et al., 2007). Teniendo en cuenta lo anterior, el AC propone cambiar los comportamientos pasivos del estudiante, la memorización, el trabajo individual y el trabajo competitivo; por la cooperación, la responsabilidad, la comunicación, el trabajo en equipo y la autoevaluación (Johnson and Johnson, 2017).

Barkley et al. (2007) caracterizan el aprendizaje colaborativo (AC) en tres aspectos:

1. Diseño intencional.
2. Colaboración.
3. Enseñanza significativa.

Como primer aspecto, el **diseño intencional** presenta la manera en cómo el docente estructura la tarea de tal manera que se tenga un objetivo o meta fija desde el principio. Esto conduce al docente a no formar grupos arbitrarios en los cuales no se tenga preparado el rol que cada miembro iría a desempeñar. Seguidamente, **la colaboración** propone que los participantes del grupo de trabajo estén comprometidos en aprender de manera colaborativa, para ello deben tener una participación activa y comprender que el trabajar con un compañero potenciará las habilidades personales y permitirá el cumplimiento de las metas grupales. Por último, la **enseñanza significativa** plantea que al finalizar el proceso de enseñanza el estudiante debe aumentar la comprensión de una temática o el incremento

de sus conocimientos.

Kennet Bruffee ha sido uno de los principales autores que ha promovido el uso del aprendizaje colaborativo en las aulas universitarias, a partir de la realización de diferentes experimentos ha demostrado que el AC fortalece las habilidades sociales tales como: la comunicación, el liderazgo, la confianza en los demás, la capacidad de intercambio de roles, sinergia, entre otras (Barkley et al., 2007). Sin embargo, ha encontrado dos inconvenientes principales en la aplicación de este enfoque de aprendizaje: el primero, que existe una fuerte resistencia cuando se solicita a un estudiante criticar el trabajo de un compañero; y el segundo, que no se pueden establecer buenos grupos colaborativos cuando no se conoce los estilos de aprendizaje de las personas que hacen parte del curso.

2.1.2. Técnicas de aprendizaje colaborativo

El diseño intencional, la colaboración y la enseñanza significativa son aspectos del aprendizaje colaborativo que solo son posibles de poner en práctica en un aula de clase mediante el uso de técnicas de aprendizaje colaborativo (TACs). Una técnica de aprendizaje colaborativo corresponde a una instrucción o un conjunto de instrucciones que un docente puede dar para estructurar el trabajo en grupo de los estudiantes. Barkley et al. (2007) proponen una clasificación de las TACs de acuerdo con las instrucciones que comparten en común, para ello las agrupan en cinco categorías:

- Técnicas para el diálogo: enfocadas en la comunicación cara a cara; algunos ejemplos son: formar una pareja y comentar una situación sobre un ejercicio, rueda de ideas, grupos de conversación, entrevista en tres pasos y debates críticos.
- Técnicas para la enseñanza recíproca: enfocadas en aprendizaje por pares; algunos ejemplos: toma de apuntes por parejas, celdas de aprendizaje, juego de rol, rompecabezas, equipos de exámenes.
- Técnicas para la resolución de problemas: resolución de problemas por parejas pensando en voz alta (RPPVA), pasar el problema, estudios de casos, resolución estructurada de problemas, equipos de análisis e investigación en grupos.
- Técnicas que utilizan organizadores gráficos de información: enfocadas en el aprendizaje visual; algunos ejemplos: agrupamiento por afinidad, tabla de grupo, matriz de equipo, cadenas secuenciales, redes de palabras.
- Técnicas centradas en la escritura: diarios para el diálogo, mesa redonda, ensayos diádicos, corrección por el compañero, escritura colaborativa, antologías de equipo, seminario sobre una ponencia.

Con respecto al uso de una TAC por parte de un docente, cabe mencionar que el AC, a diferencia de otros paradigmas pedagógicos como el aprendizaje cooperativo, propone que el docente se haga partícipe del proceso de aprendizaje como un miembro más del equipo de trabajo (Bruffee, 1983). Como

se puede observar en la clasificación por Barkley et al. (2007), una técnica de aprendizaje colaborativo puede entenderse también como diferentes dinámicas que un docente puede establecer en el aula para tratar un tema en clase.

2.1.3. Evaluación formativa y aprendizaje colaborativo

La formación comprende el significado de avanzar en la capacidad de pensar y de decidir con autonomía, como consecuencia de un proceso de aprendizaje (Flórez Ochoa, 2005). En este sentido, la evaluación es comprendida como un juicio de valor que se realiza sobre ese mismo proceso, con el fin de determinar qué tan lejos se llegó con respecto a unos objetivos de aprendizaje planteados. Lo anterior conlleva a pensar la manera en cómo los docentes establecen mecanismos o estructuras de evaluación que les permitan realizar un seguimiento a sus estudiantes con el fin de conocer si aprendieron o no; en consecuencia, existe una gran responsabilidad por parte del docente en determinar los tiempos, las actividades, las calificaciones y, por supuesto, las reprobaciones en los cursos a cargo. Esta misma responsabilidad conduce a que se realicen reflexiones como las que expresa Orozco (2014), donde pone en evidencia el protagonismo del maestro en la evaluación y un papel secundario al estudiante, siendo este último quien podría determinar también en qué nivel de aprendizaje puede encontrarse —como bien se conoce autoevaluarse— dentro de su propio proceso formativo. La autoevaluación por sí misma demanda algunas condiciones en el entorno de aprendizaje, una de ellas está determinada por la manera en cómo el estudiante autorregula su aprendizaje. En palabras de Flórez Ochoa (2005): *“La evaluación formativa no tiene otro objetivo que conseguir que los estudiantes sean capaces de construir y aplicarse un sistema efectivo de autorregulación de su aprendizaje”*. Esta autorregulación viene dada por dos factores en el proceso: motivación y colaboración. En este sentido, la Tabla 2-1 presenta un conjunto de características basadas en (Flórez Ochoa, 2005) con el fin de reconocer entornos de evaluación formativa a través de la autorregulación y cooperación.

Cabe aclarar que los términos colaboración y cooperación se refieren al conjunto de estudiantes que trabajan en grupo con el fin de cumplir un objetivo de aprendizaje (Barkley et al., 2007). Algunos autores separan estos dos conceptos en aspectos como la participación voluntaria del estudiante o el nivel de escolaridad; sin embargo, en la literatura consultada la mayoría de trabajos usan estos dos conceptos para referirse a lo mismo.

Como se puede observar en la Tabla 2-1, las características de entornos colaborativos de evaluación formativa están orientadas en conocer el papel que juega el docente al momento de implementar evaluación formativa, ciertamente podría ser el maestro el más interesado en conocer el nivel de conocimiento de sus estudiantes; sin embargo, esta forma de evaluación va más encaminada a una reflexión por parte del estudiante sobre lo que sabe y quisiera saber, con referencia a lo que conocen los demás al momento de trabajar en grupo. En consideración, el AC debe coexistir con un entorno de evaluación formativa en la manera en que cada estudiante reflexiona sobre las capacidades individuales adquiridas al trabajar en grupo y adicionalmente propicia la reflexión sobre el cumplimiento

Tabla 2-1: Características de entornos colaborativos de evaluación formativa

Factor	Descripción	Características	Estado docente	Estado estudiante
Autorregulación	Búsqueda de un equilibrio cognitivo formado por conocimientos previos y futuros.	<ul style="list-style-type: none"> ■ Identificar los motivos y objetos de aprendizaje ■ Anticipar, representar y planificar las operaciones necesarias para realizar cada proceso de aprendizaje. ■ Identificar criterios de evaluación. 	Pasivo	Activo
Colaboración	Búsqueda de generar nuevo conocimiento a través de las estructuras cognitivas de un par.	<ul style="list-style-type: none"> ■ Crítica mutua. ■ Reconocimiento de estrategias de aprendizaje. ■ Comunicación de ideas. ■ Autoevaluación permanente. 	Pasivo	Activo

de las actividades y tareas de los demás miembros del grupo. De esta manera, se podría evidenciar la principal característica del AC denominada interdependencia positiva, que no es más que un estado en el que el estudiante es consciente que su éxito depende del éxito que tengan los demás en una actividad particular.

Con respecto a la evaluación formativa es importante reconocer la diferencia entre medios, técnicas e instrumentos evaluativos. En Hamodi et al. (2018) realizan una unificación de dichos términos en un marco de referencia, como se puede observar en la Tabla 2-2.

La Tabla 2-2 presenta una clasificación de las prácticas de evaluación formativa de acuerdo con los medios, técnicas e instrumentos de evaluación que un docente puede usar en el aula de clase. Los **medios** corresponden las diferentes formas en que una actividad puede ser solicitada para ser evaluada. Las **técnicas** corresponden a las estrategias usadas para dar revisión a las actividades propuestas. Finalmente, los **instrumentos** comprenden los recursos que se pueden usar para hacer seguimiento a una actividad. Cabe destacar la clasificación de las técnicas de evaluación en el que el estudiante participa (autoevaluación, evaluación entre pares y evaluación compartida), ya que estas pueden ser utilizadas como prácticas evaluativas para potenciar el aprendizaje de los estudiantes haciendo uso de

Tabla 2-2: Medios, técnicas e instrumentos de evaluación propuestos por Hamodi et al. (2018)

MEDIOS	Escritos	<ul style="list-style-type: none"> ▪ Examen ▪ Estudio de casos ▪ Portafolio ▪ Portafolio electrónico ▪ Ensayo ▪ Control (examen) ▪ Carpeta o dossier, carpeta colaborativa 	<ul style="list-style-type: none"> ▪ Foro virtual ▪ Proyecto ▪ Memoria ▪ Prueba objetiva ▪ Monografía ▪ Recesión ▪ Trabajo escrito ▪ Póster 	<ul style="list-style-type: none"> ▪ Cuaderno, cuaderno de notas o de campo ▪ Diario reflexivo, diario de clase ▪ Cuestionario ▪ Informe ▪ Test de diagnóstico
	Orales	<ul style="list-style-type: none"> ▪ Presentación oral ▪ Discusión grupal ▪ Pregunta de clase 	<ul style="list-style-type: none"> ▪ Comunicación ▪ Mesa redonda ▪ Exposición 	<ul style="list-style-type: none"> ▪ Ponencia ▪ Cuestionario oral ▪ Debate, diálogo grupal
	Prácticos	<ul style="list-style-type: none"> ▪ Demostración, actuación o representación 	<ul style="list-style-type: none"> ▪ Práctica supervisada 	<ul style="list-style-type: none"> ▪ juego de roles
TÉCNICAS	El alumno no interviene	<ul style="list-style-type: none"> ▪ Análisis documental y de producciones (revisión de trabajos personales y grupales) ▪ Observación, observación directa del alumno, observación del grupo, observación sistemática, análisis de grabación de audio o vídeo 		
	El alumno participa	<ul style="list-style-type: none"> ▪ Autoevaluación (mediante la autoreflexión y/o el análisis documental) ▪ Evaluación compartida o colaborativa (mediante una entrevista individual o grupal entre el o la docente y los alumnos y alumnas) ▪ Evaluación entre pares (mediante el análisis documental y/o la observación) 		
INSTRUMENTOS		<ul style="list-style-type: none"> ▪ Escala de comprobación ▪ Ficha de observación ▪ Matrices de decisión ▪ Fichas de seguimiento individual o grupal ▪ Lista de control 	<ul style="list-style-type: none"> ▪ Fichas de evaluación entre iguales ▪ Escala verbal o numérica ▪ Escala descriptiva o rúbrica ▪ Fichas de autoevaluación 	<ul style="list-style-type: none"> ▪ Escala de diferencial semántico ▪ Diario del profesor ▪ Escala de estimación ▪ Informe de expertos ▪ Informe de autoevaluación

ambientes tecnológicos, al permitir que se tenga información más precisa y continua sobre el estado de formación de cada estudiante (López Pastor et al., 2004).

2.1.4. Aprendizaje colaborativo asistido por computador

El aprendizaje colaborativo asistido por computador conocido como CSCL (del inglés *Computer Supported Collaborative Learning*) es una de las propuestas educativas que pretende fortalecer la formación por competencias del actual sistema educativo. Esta propuesta de enseñanza-aprendizaje permite estudiar las diferentes estrategias sobre cómo a través de la tecnología y los pequeños grupos se pueden llegar a realizar proyectos, resolver problemas o comprender un tema en un aula de clase (Vinagre Laranjeira, 2009).

Uno de los fundamentos teóricos de la aplicación del CSCL en la educación es el nombrado por Vinagre Laranjeira (2009) al referenciar las teorías de Vigosky, Piaget y Brunner (autores principales del constructivismo), que en su conjunto establecen que el aprendizaje se potencia cuando existen interacciones de los estudiantes con el mundo social. En este contexto, una herramienta tecnológica se convierte en el mejor mediador para que se pueda proporcionar una adecuada y más eficaz interacción entre estudiantes (Carrió Pastor, 2007). Según Vinagre Laranjeira (2009), el uso de la tecnología en el proceso de aprender ha dado como resultado mejoras en el cumplimiento de los objetivos de entendimiento individual de los estudiantes en diferentes áreas del conocimiento.

A partir de la aplicación del CSCL en el aula de clase se han desarrollado diferentes herramientas tecnológicas, algunas basadas en las técnicas de aprendizaje colaborativo (TACs) y otras, simplemente, como medio de transmisión información. Teniendo en cuenta este último aspecto, cabe precisar que es necesario establecer un uso adecuado de las tecnologías aplicadas a la educación con respecto al aprendizaje colaborativo; para ello hay que entender y conocer diferentes características que nos conducen a utilizarlas de manera correcta. En (Vinagre Laranjeira, 2009) se reflejan las siguientes características:

- Entender los diferentes modelos de aprendizaje en línea.
- Conocer qué papel juega el docente en el CSCL.
- Saber cómo organizar los intercambios de CSCL.
- Poder asegurar el desarrollo de competencias lingüísticas e interculturales en CSCL.
- Distinguir qué tipo de herramienta es adecuada para llevar a cabo una determinada actividad en CSCL.

Adicional a los aspectos anteriores, existe una preocupación enfocada en determinar la mejor manera de conformar grupos de aprendizaje. Esto con el fin de cumplir el principio de equidad que propone

el aprendizaje colaborativo (Barkley et al., 2014), en el cual los grupos de estudio deben aportar conocimientos de la misma manera, evitando que unos grupos estén más adelantados que otros, tal y como se expone en (Castro-Bleda et al., 2013). Por otro lado, es importante reconocer qué método de evaluación es representativo para soportar el aprendizaje colaborativo asistido por computador; si bien la evaluación puede tornarse algo abstracta según las competencias o disciplinas a evaluar, cabe mencionar que en los enfoques pedagógicos constructivistas la evaluación depende de un proceso de reflexión, el cual conduce al docente a medir una serie de eventos antes de llegar a un resultado final de un estudiante (Flórez Ochoa, 2005).

Es necesario aclarar el rol que emplea la tecnología en ambientes de CSCL, ya que a menudo se ha relacionado este concepto principalmente con *e-learning*. Aunque existen características que comparten, vale la pena exponer las siguientes ideas vistas en (Stahl et al., 2006):

- Los contenidos de aprendizaje suministrados en una plataforma tecnología tales como diapositivas, textos y videos no conllevan a dar una instrucción a un estudiante.
- El esfuerzo que un docente realiza en una clase presencial debe ser igual o superior en ambientes virtuales, debe orientar y guiar al estudiante en cuanto al material disponible.
- El CSCL con respecto a *e-learning* debe fomentar la colaboración como el núcleo de desarrollo de las actividades, el soporte computacional que se le dé a ello es lo más aproximado que tiene un concepto del otro.
- El soporte computacional en el aprendizaje no tiene como finalidad plantear escenarios totalmente online, ya que CSCL no es usar tecnología sino aprender de manera colaborativa por medio de herramientas que faciliten esa colaboración.

De acuerdo con lo anterior, es importante reconocer que el CSCL puede estar implícito dentro de *e-learning* como un elemento adicional. Si bien en las primeras versiones de *e-learning* el aprendizaje fue concebido en solitario y de modo asíncrono, a partir de la aparición de la web 2.0 –junto a los desarrollos tecnológicos que se propiciaron con ello– se estructuraron las plataformas con opciones de conexión más síncronas y sociales, todo esto con el objetivo de buscar maneras de poder diseñar actividades que se pudieran resolver de manera colaborativa (Clark, 2011).

2.1.5. Guiones de aprendizaje colaborativo asistido por computador

Las diferentes opciones de colaboración que son proporcionadas en ambientes de aprendizaje virtual (*e-learning*) son conocidas como *scripts* CSCL, los cuales establecen escenarios predefinidos en donde se promueve la interacción productiva entre estudiantes (Alharbi et al., 2015). Como se puede observar en la Figura 2-1, basada en Hernandez-Leo et al. (2006), CSCL es clasificado como un elemento dentro del tipo de patrones de diseño pedagógicos –que son los dedicados al diseño de sistemas de *e-learning*– que permiten establecer soluciones que puedan aplicar la colaboración por medio de un

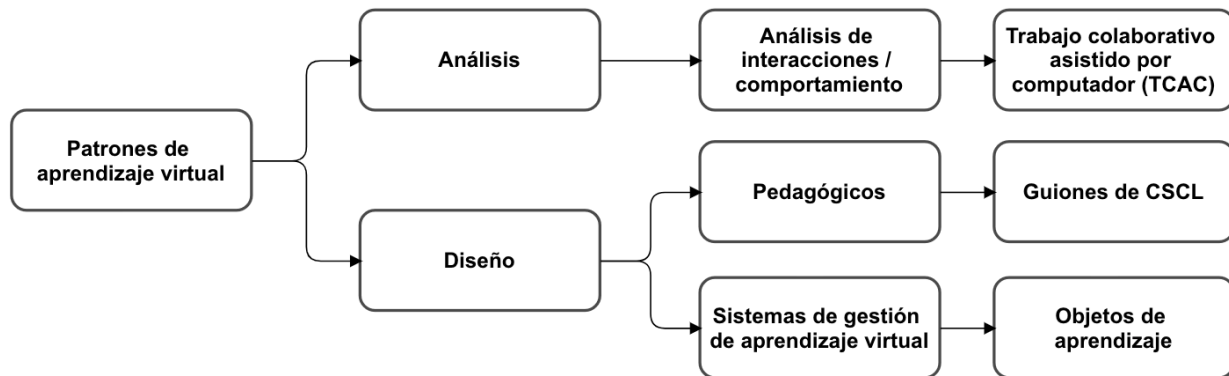


Figura 2-1: Clasificación de patrones en aprendizaje virtual, basada en (Hernandez-Leo et al., 2006)

elemento tecnológico.

Es oportuno decir que estos *scripts* establecen formas y organizaciones dentro de actividades colaborativas; en principio, los *scripts* podrían asimilarse a las TACs vistas anteriormente, ya que se estructuran a través de procesos en los cuales se fomenta el AC; sin embargo, existe un concepto denominado patrón (*pattern*), que es el utilizado en CSCL para referirse a los elementos que están contemplados dentro de la estructura de un *script*, que en un nivel de flujos de trabajo colaborativo sí podrían contener algún tipo de TAC.

Las aplicaciones que tengan un componente colaborativo basado en un *script* CSCL contemplan tres niveles principales de especificación (Hernandez-Leo et al., 2006). El primer nivel, denominado flujo de aprendizaje colaborativo comprende la secuencia de actividades que conforman un proceso de aprendizaje. Cabe mencionar que en este nivel pueden estar incluidas las TACs vistas anteriormente. El segundo nivel, establecido como actividad, permite fijar las tareas o actividades a desarrollar. Por último, existe un nivel de recursos el cual comprende el uso de herramientas y materiales para soportar las actividades colaborativas. Adicionalmente, existe un cuarto nivel, que es transversal a los niveles anteriores, correspondiente a la formación de grupos o asignación de roles; este nivel adicional es visto en (Alharbi et al., 2015) como el primer acercamiento a la tarea colaborativa luego que un docente especifique un objetivo de aprendizaje sobre la misma. La Figura 2-2 presenta la estructura de los niveles escritos anteriormente.

Los niveles de un *script* CSCL están compuestos de un conjunto de patrones asociados a dicha clasificación: patrones para formación de grupos (PFG), patrones para niveles de flujo de trabajo colaborativo (PFTC), patrones para actividades (PA) y patrones para recursos (PR). Los patrones de CSCL en algunos casos coinciden con los nombres de las TACs nombradas al inicio, esto debido a que en términos generales cumplen la misma función dentro del AC. La Figura 2-2 se interpreta de forma jerárquica; siendo el nivel de flujo de trabajo colaborativo el primer aspecto a tener en cuenta en plataformas

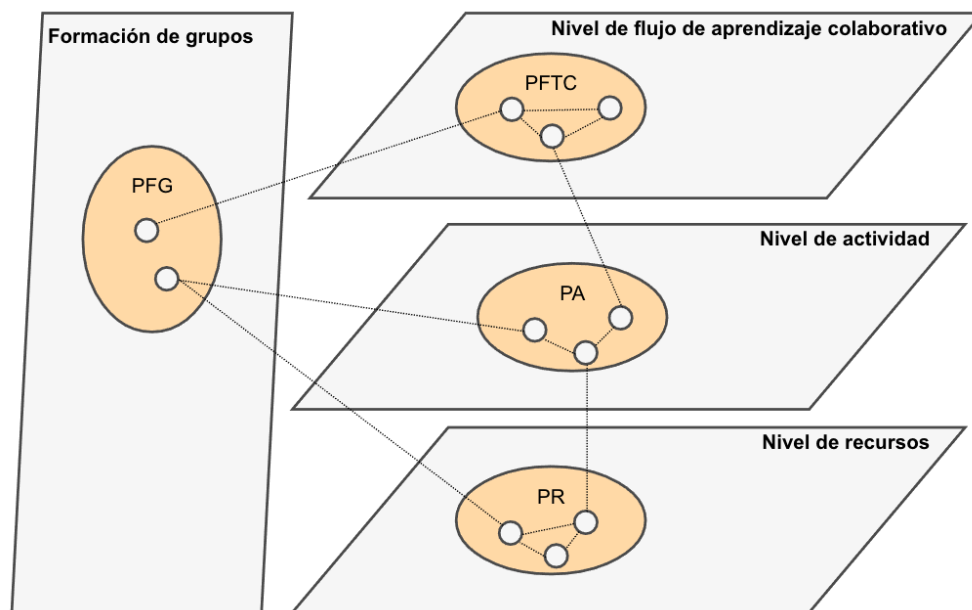


Figura 2-2: Estructura de patrones de guiones CSCL basado en (Hernandez-Leo et al., 2006)

CSCL, seguido del nivel de actividad y recursos; el nivel de formación de grupos está relacionado directamente con los demás niveles, ya que los patrones asociados al él determinan los roles que los estudiantes pueden tener al conformar grupos colaborativos y de esta manera conocer qué papel desempeñarán en los demás patrones asociados a cada nivel.

Con el fin de conocer qué patrones o técnicas de evaluación (según la Tabla 2-1) son utilizados en diferentes tipos de herramientas computacionales que se basan en el aprendizaje colaborativo, se presenta en la Tabla 2-3 algunas herramientas basadas en CSCL que muestran una primera aproximación a los tipos de herramientas que se pueden usar dentro de un aula de clase para apoyar la colaboración entre estudiantes. Dentro de estas aplicaciones podemos encontrar herramientas móviles, web, videojuegos, redes sociales, entre otras. Todos estos tipos de herramientas indican que cualquier elemento tecnológico puede ser usado como recurso para implementar actividades de aprendizaje colaborativo.

Existen otras aplicaciones que pueden servir de soporte en el AC. Por ejemplo, los sistemas de sincronización de archivos como Dropbox o Google Drive, o los denominados *workshop*; sin embargo, no se encontraron trabajos que referenciaran a este tipo de aplicaciones en CSCL o AC. Con relación a las técnicas de evaluación, se evidencia en las herramientas consultadas una fuerte inclinación hacia una participación pasiva por parte de los estudiantes en el proceso evaluativo. En este caso la Observación se presentó como uno de los métodos más frecuentes, ya que es el docente el principal responsable de coordinar actividades y hacer los seguimientos correspondientes al nivel de aprendizaje de los participantes.

Tabla 2-3: Herramientas basadas en CSCL

Tipo de herramienta facilitadora CSCL	Descripción	Trabajos relacionados	Nivel CSCL	Patrones o TACs	Técnica de evaluación
Blogs	Sitio web para escribir comentarios.	(Michailidis et al., 2013)	Trabajo colaborativo, actividad y recursos	Escritura colaborativa, rueda de ideas	Observación
Conferencias virtuales	Aplicación para compartir multimedia	(Szedmina and Pinter, 2010)	Actividad, recursos	Diálogo	
Chats	Dos o más participantes se comunican por medio de texto (Clark, 2011)	(Tirado et al., 2009; Caballe et al., 2014)	Actividad, recursos	Corrección por el compañero	
E-mail	Dos o más participantes se comunican por medio de correo electrónico (Clark, 2011)	(Zhao and Okamoto, 2009)	Actividad	Corrección por el compañero, Diálogo	
Pizarra de mensajes	Dos o más participantes discuten sobre un tema de manera asíncrona por medio de texto	(Tirado et al., 2009)	Actividad, recursos	Diálogo, Escritura colaborativa	
Redes sociales	Los participantes publican páginas con diversos elementos de medios y vinculan sus páginas con otros seleccionados	(Judele et al., 2014)	Actividad	Diálogo	Observación
Wikis	Sitio web para editar contenidos (Clark, 2011)	(Tirado et al., 2009; Liu et al., 2004; Lampe et al., 2012)	Actividad, recursos	Escritura colaborativa	
Libro virtual	Texto electrónico (libros, artículo, memoria)	(Tseng et al., 2012)	Actividad, recursos	Debates críticos	
MOOCs	Es una forma de ofrecer cursos en línea. Entre otras prestaciones, proporcionan foros de discusión para que los estudiantes hagan preguntas y discutan temas relacionados con el curso (Revelo-Sánchez et al., 2018).	(Ramírez-Donoso et al., 2017)	Trabajo colaborativo, actividad, recursos	Jigsaw	Observación de grupo
Aplicación móvil	Aplicación móvil de soporte para compartir o acceder a contenidos.	(Ramírez-Donoso et al., 2017; Zhao and Okamoto, 2009; Zamfirache et al., 2013; Tseng et al., 2009; Serrano Cámara et al., 2012)	Actividad	Enseñanza recíproca	Autoevaluación, pares
Aplicaciones multimedia	Aplicaciones orientadas al entretenimiento	(Fukuma et al., 2017; Frank-Bolton and Simha, 2018)	Actividad	Gamificación	Observación

La mayoría de las aplicaciones mencionadas en la Tabla 2-3 son de uso comercial y fueron aplicadas en CSCL con el fin de experimentar su uso en la educación. Por otro lado, también se encuentran diferentes herramientas con desarrollos de software a la medida, cuyas puestas en marcha se pudieron llevar a cabo a partir de la definición de diferentes prototipos que permitieron conocer las necesidades requeridas en un aula de clase específica. Esto último apoya la idea de Pressman (2010), la cual establece que la manera más efectiva de conocer un contexto es mediante la propuesta de prototipos, ya que estos nos permiten fijar los objetivos de un software cuando no se tienen claros los requisitos del mismo.

Las propuestas de desarrollo de prototipos para el aprendizaje de programación de computadores han permitido poner a prueba diferentes herramientas que soportan procesos de aprendizaje colaborativo. Dentro de estos se encuentran: la formación de grupos, la asignación de roles, la organización de las actividades y el apoyo en el desarrollo de las tareas. Estos procesos involucrados en herramientas que apoyen la enseñanza en los cursos de programación se vuelven relevantes, ya que se convierten en componentes claves al momento de garantizar que los estudiantes adquieran habilidades necesarias para trabajar de manera efectiva en entornos laborales, sobre todo cuando se enfrentan a grandes equipos de diseño y desarrollo de software (Soundarajan et al., 2015).

2.1.6. Prototipos de software

Las metodologías de software actuales han adoptado el desarrollo de prototipos debido a la simplicidad y facilidad de construir un primer sistema. La comunicación, el diseño rápido y la realimentación son los aspectos más importantes al momento de proponer un prototipo (Pressman, 2010). Estos aspectos permiten que un prototipo pueda mejorar sus funciones con el tiempo o “evolucionar”.

En contextos educativos la puesta en marcha de prototipos cuyas funciones estén terminadas en su totalidad es muy poco frecuente, debido a que se usan como medios que permiten ayudar a la construcción del conocimiento en la enseñanza-aprendizaje de un área específica (Hashim and Harun, 2014).

El desarrollo de software orientado a prototipos implica, al igual que otros mecanismos, usar una metodología adecuada que permita coordinar todo el conjunto de actividades que se requieren para poder cumplir con el objetivo de un proyecto. En Pressman (2002) se establece un paradigma básico para trabajar prototipos, el cual establece tres actividades principales: escuchar al cliente, construir una maqueta y probar la maqueta. Este paradigma propone realizar diferentes iteraciones para cada una de las versiones que se quieren mejorar sobre el sistema. Cabe resaltar que una de las ventajas de este paradigma es no contar con versiones finales sino versiones cada vez más robustas de un software; razón por la cual es muy adoptado en metodologías ágiles.

En la Figura 2-3 se puede observar el conjunto de actividades adicionales que contempla el desarrollo orientado a prototipos. Los siguientes describen cada una de las fases:

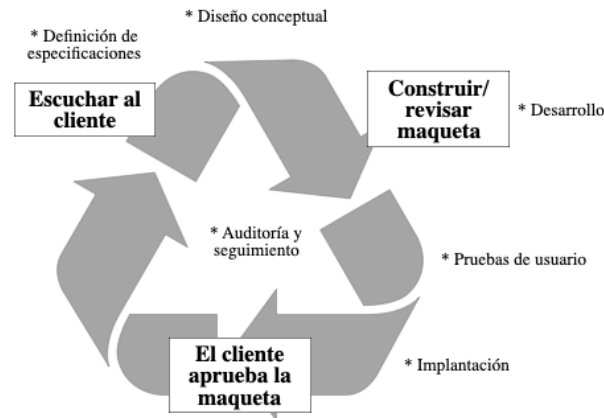


Figura 2-3: Metodología de prototipado basada en Pressman (2002)

- Definición de especificaciones: se describen las principales funciones que el prototipo debe realizar para dar cumplimiento a una idea inicial o la evolución de la misma.
- Diseño conceptual: contempla un diseño rápido sobre las principales funciones que el prototipo debe realizar.
- Desarrollo del prototipo: realizar la codificación según la definición de las especificaciones.
- Pruebas de usuario: esta fase permite realizar las pruebas funcionales del prototipo, con el fin de corregir o mejorar la conexión lógica de los datos trabajados por el usuario.
- Implementación: esta fase contempla la puesta en marcha del prototipo en un ambiente similar al usado por el usuario.
- Auditoría y seguimiento: esta fase permite realizar un control sobre cada una de las fases anteriores, con el fin de corregir los errores lo más pronto posible.

La elaboración o desarrollo de un software es un proceso completamente social, en donde el constante aprendizaje del contexto hace que sea posible comprender la naturaleza del problema a resolver y poder ofrecer una aplicación de calidad (Pressman, 2010). A medida que se ponen a prueba prototipos de software en un aula de clase se van conociendo aspectos y detalles del contexto que pueden conducir al desarrollo de un producto final. Un ejemplo de ello se ve en (Serrano-Cámara et al., 2016) en el que se propone una herramienta CSCL para aprender programación de computadores y mediante el desarrollo de varios prototipos se logró la implementación de un sistema más completo y usable; todo a través de la constante realimentación por parte de estudiantes y profesores.

2.2. Trabajos relacionados

A partir del surgimiento del aprendizaje colaborativo en los años 50's se han realizado diferentes estudios, dentro de los cuales se encuentran: (Judele et al., 2014; Zakiah and Fauzan, 2016; Dorneles

et al., 2010; Cisar et al., 2013; Maksimenkova and Alexey, 2015; Arora et al., 2017; Serrano-Cámara et al., 2016; Laakso et al., 2018), que pretenden demostrar cómo la colaboración entre estudiantes, profesores y otros profesionales puede potenciar el rendimiento académico de los estudiantes de programación. En algunos casos, han hecho uso de herramientas populares tales como GitHub (Zakiah and Fauzan, 2016) o Facebook (Judele et al., 2014); en otros, se han desarrollado herramientas de software a la medida para apoyar el proceso de enseñanza-aprendizaje en una institución en particular.

Las propuestas pedagógicas orientadas al uso de software como elemento mediador en la enseñanza-aprendizaje de programación de computadores, por lo general, se encuentran soportadas por diferentes tipos de TACs, dentro de estas se pueden destacar: el aprendizaje basado en problemas (Knutas et al., 2013), las comunidades de aprendizaje de código (Zakiah and Fauzan, 2016), el aprendizaje basado en proyectos (Hoffbeck et al., 2016), la programación por pares (Gehring et al., 2005), entre otras. Todas estas propuestas se caracterizan por hacer uso de la tecnología como elemento mediador en el proceso de resolución de problemas de programación a través de la colaboración; adicional a ello, poseen un diseño centrado en el estudiante y hacen un gran énfasis en lograr un aprendizaje significativo que le permita al estudiante participar en contextos lo más reales posibles al ámbito laboral (Vinagre Laranjeira, 2009).

El aprendizaje colaborativo establece que un grupo de personas participe de manera voluntaria para alcanzar un objetivo común, propiciando el debate, la discusión y el diálogo (Bruffee, 1983); estas características apoyadas en tecnología es lo que se ha establecido como CSCL. Teniendo en cuenta esta premisa, en la enseñanza de programación de computadores se ha cuestionado sobre las diferentes formas tradicionales de enseñanza-aprendizaje, ya que es una actividad que profesionalmente se desarrolla en grupo y en su proceso de enseñanza se propicia la competencia y el aislamiento (Deitrick et al., 2016). Adicionalmente, existe cierta preocupación sobre el papel de docente como único actor en el proceso de evaluación; aunque es el mismo docente quien conoce las competencias a desarrollar en un curso y orienta en la actividad pedagógica, autores como Hamodi et al. (2018); López Pastor et al. (2004); Bretones Román (2008) coinciden en el manejo de prácticas educativas en donde se tenga participación de todos los sujetos que estén involucrados en un proceso de enseñanza-aprendizaje al momento de evaluar. En este sentido, los beneficios de involucrar al estudiante en un proceso de evaluación, basado en la revisión literaria de Hamodi et al. (2018), son los siguientes:

- Desarrollo del pensamiento crítico, autonomía y toma de conciencia.
- Desarrollo de competencias para el futuro profesional.
- Desarrollo de trabajo en equipo.
- Democratización del conocimiento.
- Fortalecimiento de valores.

Teniendo en cuenta todos los aspectos descritos anteriormente sobre evaluación formativa (entornos, medios, técnicas, instrumentos) y patrones o técnicas de aprendizaje colaborativo, se realiza una revisión literaria sobre herramientas y estudios en donde se aplique CSCL orientado al aprendizaje de programación de computadores.

En la Tabla 2-4 se mencionan algunos trabajos que presentan aplicaciones o herramientas desarrolladas bajo el modelo de AC, encaminadas a apoyar el proceso de enseñanza-aprendizaje de programación de computadores. En primer lugar, se reconocen las técnicas de aprendizaje colaborativo (TACs), que se definen como las diferentes formas de interacción colaborativa entre estudiantes y docentes para desarrollar una actividad (Barkley et al., 2014). En segundo lugar, se identifican los medios, técnicas, instrumentos y entornos correspondientes al uso de algún elemento de evaluación formativa.

Tabla 2-4: Trabajos relacionados sobre herramientas para el aprendizaje colaborativo de programación de computadores

TAC	Herramientas de aprendizaje colaborativo	Factores de evaluación formativa			Entornos	
		Medios	Técnicas	Instrumentos	A	C
Pirámide	<ul style="list-style-type: none"> ■ 3DVirCSLab (Pullan et al., 2013) 	<ul style="list-style-type: none"> ■ práctica supervisada ■ juego de rol 	<ul style="list-style-type: none"> ■ observación sistemática 	<ul style="list-style-type: none"> ■ escala de comprobación ■ fichas de seguimiento grupal/individual 		X
Rompecabezas	<ul style="list-style-type: none"> ■ Pex4Fun (Cisar et al., 2013) ■ I-MINDS (Soh et al., 2008) ■ Jigsaw-adventure (Burch, 2009) ■ Mobile Campus (Pérez-Sanagustín et al., 2012) 	<ul style="list-style-type: none"> ■ práctica supervisada ■ diálogo 	<ul style="list-style-type: none"> ■ observación de grupo 	<ul style="list-style-type: none"> ■ escala de estimación 		X
Lluvia de ideas	<ul style="list-style-type: none"> ■ Collaborative Session Conceptual Schema (Caballé et al., 2011) 	<ul style="list-style-type: none"> ■ práctica supervisada 	<ul style="list-style-type: none"> ■ evaluación entre pares 	<ul style="list-style-type: none"> ■ fichas de evaluación entre iguales 		X
Juego de rol	<ul style="list-style-type: none"> ■ Kammy (Shabalina et al., 2013) ■ MeLoISE (Vosinakis et al., 2014) ■ Co-Learning-Master (Fukuma et al., 2017) 	<ul style="list-style-type: none"> ■ práctica supervisada 	<ul style="list-style-type: none"> ■ observación 	<ul style="list-style-type: none"> ■ fichas de seguimiento ■ escala de estimación ■ escala numérica 		X
Pensar en voz alta la solución de un problema	<ul style="list-style-type: none"> ■ VILLE (Laakso et al., 2018) ■ 3D virtual worlds (Fetemban et al., 2016) 	<ul style="list-style-type: none"> ■ juego de rol 	<ul style="list-style-type: none"> ■ observación de grupo 	<ul style="list-style-type: none"> ■ informe de expertos 	X	X
Grupo de discusión	<ul style="list-style-type: none"> ■ MoCAS (Serrano Cámara et al., 2012) ■ Intelligent Tutoring Systems (Virvou and Sidiro-poulos, 2012) 	<ul style="list-style-type: none"> ■ práctica supervisada 	<ul style="list-style-type: none"> ■ análisis documental y producciones 	<ul style="list-style-type: none"> ■ escala de comprobación 	X	X
Formación libre de grupos	<ul style="list-style-type: none"> ■ AES-CS (Triantafyllou et al., 2002) ■ Memetic Algorithm (Yannibelli and Amandi, 2012) 	<ul style="list-style-type: none"> ■ debate ■ diálogo grupal ■ discusión grupal 	<ul style="list-style-type: none"> ■ observación de grupo 	<ul style="list-style-type: none"> ■ escala de comprobación 		X
Formación controlada de grupos	<ul style="list-style-type: none"> ■ Algoritmo de retroceso (Sancho-Asensio et al., 2014) 	<ul style="list-style-type: none"> ■ práctica supervisada 	<ul style="list-style-type: none"> ■ evaluación entre pares 	<ul style="list-style-type: none"> ■ escala de comprobación 		X
Aprendizaje semipresencial	<ul style="list-style-type: none"> ■ Diseño de aprendizaje combinado (Yongxing, 2008) 	<ul style="list-style-type: none"> ■ demostración ■ práctica supervisada 	<ul style="list-style-type: none"> ■ autoevaluación 	<ul style="list-style-type: none"> ■ informe de autoevaluación 		X
Legó	<ul style="list-style-type: none"> ■ Robots LEGO (Lykke et al., 2014) 	<ul style="list-style-type: none"> ■ práctica supervisada 	<ul style="list-style-type: none"> ■ observación de grupo 	<ul style="list-style-type: none"> ■ lista de control 	X	X

Entornos: (A) Autorregulación, (C) Colaboración

El análisis de la revisión literaria sobre los estudios basados en CSCL, en programación de computadores, gira entorno a las técnicas de aprendizaje colaborativo TAC identificadas en cada estudio consultado, ya que éstas establecen las bases teóricas sobre las cuales están desarrolladas las herramientas vistas en la columna Plataformas de la Tabla 2-4. En cada estudio se identificaron los medios, técnicas e instrumentos de evaluación utilizadas según la clasificación de (Hamodi et al., 2018) mostradas en la Tabla 2-2. De igual modo, se distinguió en cada estudio si se implementan factores de evaluación formativa dirigidos a la autorregulación o colaboración.

Los trabajos expuestos en la Tabla 2-4 se centran en técnicas y herramientas que permiten compartir recursos de aprendizaje, gestionar actividades y usar rúbricas de evaluación de acuerdo a diferentes parámetros dados por un docente. Estos estudios demuestran buenos resultados respecto al rendimiento de los estudiantes en los cursos de programación y el desarrollo de algunas habilidades blandas. Por ejemplo, en (Serrano Cámara et al., 2012) se concluye una mayor eficiencia educativa y motivación en los estudiantes al usar una aplicación que facilita la comunicación en el curso. En (Laakso et al., 2018; Hwang et al., 2017) reportan un aumento en la capacidad de pensamiento crítico, al involucrar a los estudiantes en el proceso de evaluación de programas mediante la revisión por pares. Adicionalmente, se identifican estudios de aplicación de CSCL más formales como los vistos en (De Faria et al., 2006; Tsompanoudi et al., 2016; Debdi et al., 2015; Soh et al., 2005), que se enfocan en el uso de guiones de CSCL para generar herramientas de soporte en programación de computadores, los cuales reportan facilidades en el proceso de asignar de las tareas a los estudiantes, una mejor formación de grupos de aprendizaje y un aumento en la motivación.

Los estudios consultados también sugieren experimentar más con los guiones de CSCL con el fin de medir su efectividad, especialmente en la asignación de grupos y de cómo estos afectan la colaboración y contribución entre estudiantes. Respecto a la aplicación de evaluación formativa, estos estudios no reflejan técnicas que apliquen procesos de coevaluación o evaluación entre pares, tan necesarios cuando se busca que un trabajo en grupo supere las deficiencias que se dan cuando el resultado del trabajo final es menor al esperado, teniendo en cuenta el potencial individual de los miembros que lo conforman (García, 2003). Las ausencias de este tipo de evaluación pueden estar basadas en los prejuicios que se le da al estudiante frente a la responsabilidad que tiene sobre su propio aprendizaje, ya que es visto como alguien que moralmente no puede decidir y poner en una escala lo que sabe y lo que no, aún más cuando se cree que la autoevaluación o la crítica es una calificación (Orozco, 2014).

Como se expuso en la Sección 2.1.6 la mejor forma de conocer o experimentar en un contexto es mediante el desarrollo de prototipos, esto permitirá aterrizar las ideas que se tengan sobre el problema a resolver y su puesta en marcha dará recomendaciones para realizar aplicaciones más robustas. En la siguiente sección se presentará el prototipo denominado UNColab, una herramienta CSCL propuesta para facilitar la evaluación formativa entre pares, la cual fue desarrollada para experimentar el aprendizaje colaborativo en un curso de programación de computadores.

3 UNColab: Herramienta computacional para el aprendizaje colaborativo de programación

Este capítulo presenta el conjunto de actividades relacionadas con el desarrollo de la herramienta computacional para el aprendizaje colaborativo de programación de computadores denominada UNColab. Inicialmente, se expone el diseño de la herramienta computacional, basado en el uso de un marco de trabajo que permite el diseño de guiones (*scripts*) de CSCL, los cuales sirven para estructurar una actividad colaborativa en clase. Como resultado, se obtiene una selección de diferentes TACs que soportan el diseño de la herramienta para cada uno de los niveles de un guion CSCL, incluyendo la evaluación. Por último, se presenta la aplicación de la metodología de desarrollo de software orientada a prototipos, que permitió estructurar y controlar la construcción e implementación de la herramienta propuesta.

3.1. Diseño de UNColab a partir de un guion CSCL

Los guiones de aprendizaje colaborativo permiten estructurar la colaboración deseada entre estudiantes con el fin de obtener una participación efectiva en una clase. Estos han sido usados para proponer flujos o procesos de aprendizaje que permiten aplicar diferentes técnicas de aprendizaje colaborativo. Proponer este tipo de guiones puede ser un trabajo de investigación completo. Por ello, Villasclaras-Fernández et al. (2009) propone usar patrones de diseño de guiones de CSCL o marcos de trabajo cuyas capturas de buenas prácticas estén documentadas y puedan facilitar la propuesta de herramientas que permitan organizar la colaboración y la actividad en un escenario propio de aprendizaje.

Teniendo en cuenta lo anterior, se ha tomado como referencia el marco de trabajo propuesto por Villasclaras-Fernández et al. (2009) para el proceso de diseño de guiones CSCL. En la Figura 3-1 se pueden observar los diferentes pasos para realizar una propuesta de guion para CSCL basado en evaluación. A continuación se describen cada uno de los pasos propuestos:

1. Determinar los objetivos de aprendizaje: indican las metas o logros que se quieren alcanzar con la actividad colaborativa.

2. Configurar flujo de actividad: permite identificar la secuencia de actividades del proceso de aprendizaje.
3. Configurar actividades y roles: permite proponer las actividades o tareas a realizar de manera colaborativa como también la asignación de los roles de trabajo.
4. Determinar recursos: estos corresponden a las herramientas que servirán de soporte a los aspectos anteriores.

También, en la Figura 3-1 podemos ver que la selección de patrones está relacionada directamente con los procesos de aprendizaje y evaluación para cada uno de los pasos del diseño de un guion CSCL. Estos pasos coinciden con los niveles de estructuración de guiones CSCL vistos en la Sección 2.1.5. Cabe recordar que los patrones CSCL cumplen la misma función que las técnicas de aprendizaje colaborativo (TAC), la cual implica reutilizar prácticas exitosas ya probadas en el aula de clase. En este sentido, “la selección de patrón” hace referencia a la elección de una técnica de aprendizaje colaborativo que permita soportar cada una de las actividades de un guion CSCL.

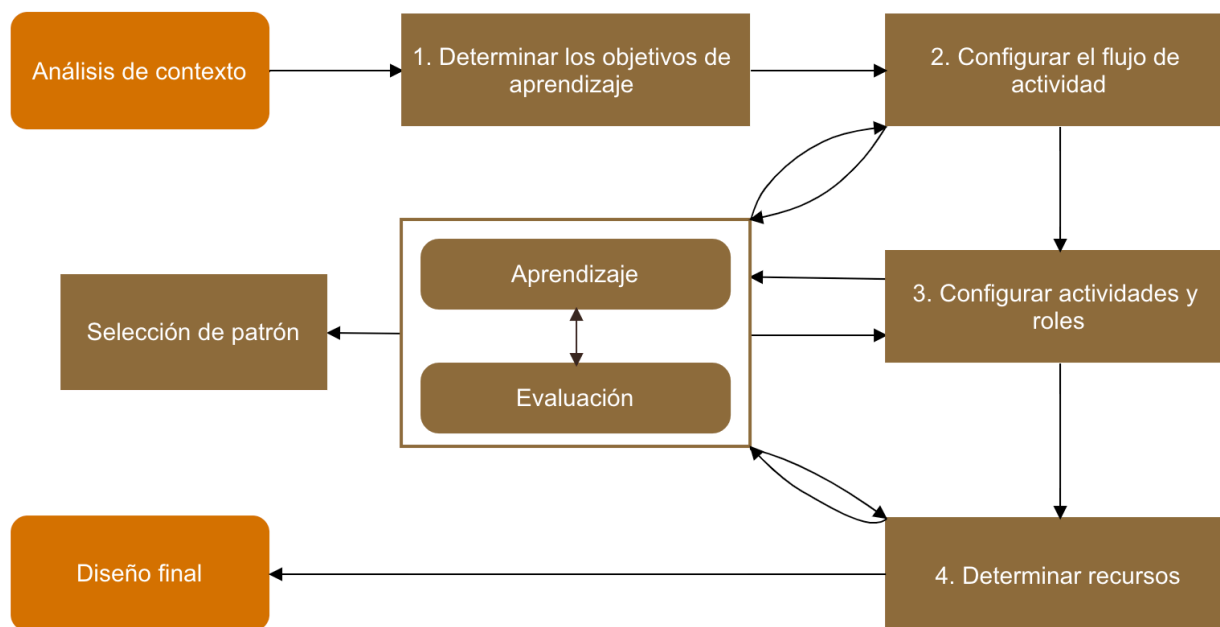


Figura 3-1: Proceso de diseño de guiones CSCL con reconocimiento de evaluación, basado en (Villasclaras-Fernández et al., 2009)

El paso preliminar propuesto en el marco de trabajo usado para el diseño del guion CSCL refiere un **análisis de contexto**. En este aspecto es importante considerar las características que debe tener el escenario de aprendizaje de acuerdo al tipo de actividad que se quiere realizar (Villasclaras-Fernández

et al., 2009). En este sentido, considerando que este estudio se enfoca en apoyar la actividad de resolución de problemas de programación por medio de la colaboración, se establecieron tres características principales para un escenario de aprendizaje colaborativo en programación de computadores: por un lado, es importante contar con un curso de programación introductorio que establezca dentro de su contenido conceptos relacionados con ciclos, sentencias de repetición y matrices. Por otro lado, dentro del curso se debe trabajar una herramienta de apoyo en la resolución de ejercicios que permita a los estudiantes proponer soluciones a problemas de programación de computadores y de esta manera practicar los temas vistos en la teoría. Finalmente, es necesario que el curso trabaje una metodología teórico-práctica, ya que el escenario colaborativo está enfocado en el aspecto práctico de la resolución de ejercicios de programación. A partir de la definición de este contexto fue posible realizar los pasos presentados en el marco de trabajo propuesto por Villasclaras-Fernández et al. (2009) para el diseño de un guion CSCL. A continuación se presenta el desarrollo de cada uno de ellos.

3.1.1. Determinación de los objetivos de aprendizaje

El paso #1 del marco de trabajo usado establece **determinar los objetivos de aprendizaje** para las actividades a realizar de manera colaborativa. Establecer estos objetivos al inicio de una actividad colaborativa le permite a un docente fijar un propósito respecto al uso de la colaboración dentro de una clase. En este estudio se estableció un objetivo de aprendizaje de acuerdo a las características de entornos colaborativos basados en evaluación formativa vistas en la Sección 2.1.3. De acuerdo con esto, el propósito que persigue la actividad colaborativa en este estudio está enfocada en brindar un espacio en el cual los estudiantes puedan aprender las temáticas de *estructuras de control*, *ciclos* y *matrices* de manera colaborativa. A partir de este paso se deben definir las técnicas de aprendizaje colaborativo que permitirán estructurar el trabajo colaborativo. En este estudio se ha seleccionado una TAC para apoyar las actividades del paso #2 y el paso #3 del marco de trabajo, cabe mencionar que estas TACs harán parte de la herramienta propuesta.

3.1.2. Configuración del flujo de actividad

La Figura 3-2 muestra la técnica de aprendizaje colaborativo Pensar-Par-Compartir (PPC), seleccionada para desarrollar el paso #2 denominado **“Configurar el flujo de actividad”** en el marco de trabajo usado. La técnica PPC permite plantear el trabajo colaborativo en las siguientes 3 fases:

- Pensar: esta fase busca que el estudiante trabaje un problema propuesto de manera individual, de tal manera que se intente reconocer las posibles soluciones e inconvenientes que se puedan dar durante el proceso de resolución.
- Par: esta fase consiste en agrupar los estudiantes en parejas, buscando que se discutan los problemas y soluciones analizados en la fase “Pensar”.

- **Compartir:** esta fase consiste en compartir las experiencias adquiridas durante la fase “Par” con otros miembros de curso.

Al aplicar esta técnica se busca que los estudiantes con mejores resultados en la resolución de ejercicios de programación puedan apoyar y guiar a aquellos estudiantes con dificultades. Esta técnica representa el flujo general de actividades que se espera que el estudiante realice durante un trabajo en clase. Según Hernández-leo and Asensio-pérez (2019) PPC se usa para remediar situaciones en las que la colaboración libre no conduce al aprendizaje y se hace necesario planificar las tareas básicas de colaboración entre estudiantes cuando se tiene un problema base por resolver.

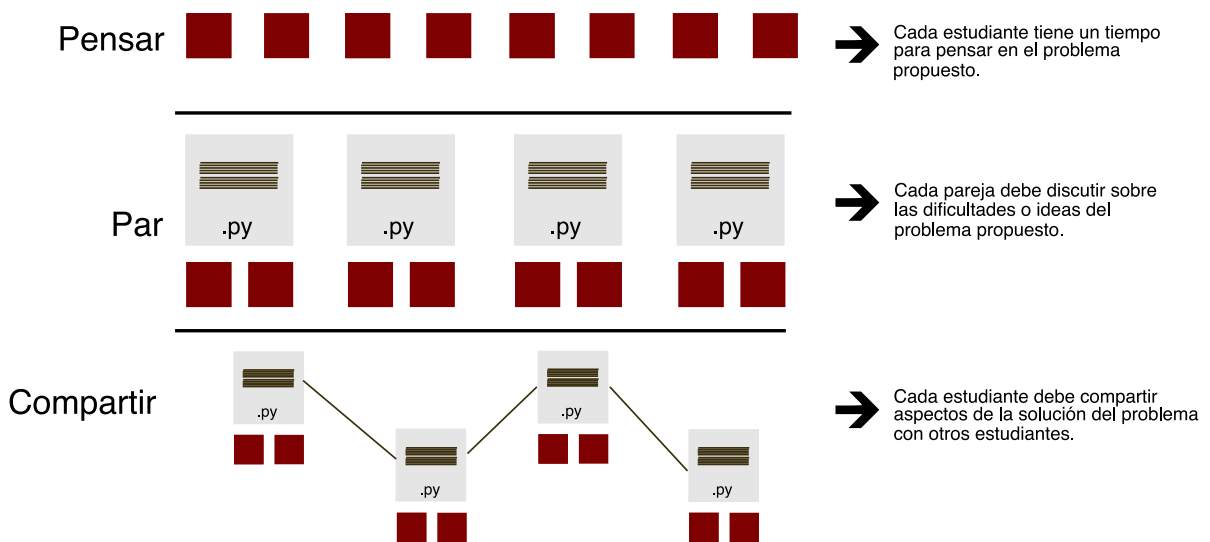


Figura 3-2: Técnica de aprendizaje colaborativo Pensar-Par-Compartir, basado en (Hernández-leo and Asensio-pérez, 2019)

3.1.3. Configuración de actividades y roles

En el paso #3 denominado “**Configurar actividades y roles**” se tuvo en cuenta tres técnicas de aprendizaje colaborativo: dos para la configuración de actividades y una para la configuración de roles. En cuanto la configuración de actividades, se han seleccionado las técnicas “Actividad introductoria” y “discusión en grupo”; respecto a la configuración de roles, se ha seleccionado la técnica “Programadores y novatos”.

La TAC “Actividad introductoria” permite mostrar los logros que el estudiante alcanzará una vez resuelva la tarea propuesta, esta TAC contempla un conjunto de explicaciones que se deben realizar a

los estudiantes antes de realizar una actividad colaborativa; estas explicaciones buscan hacer efectivas las colaboraciones que se establezcan en una clase (Hernández-leo and Asensio-pérez, 2019). La aplicación de esta técnica se basa en los siguientes aspectos:

1. Explicar el diseño de aprendizaje (guion) antes de iniciar las actividades.
2. Explicar los objetivos de la actividad colaborativa para asegurar que los resultados de cada estudiante sean los esperados de acuerdo al objetivo de aprendizaje.
3. Explicar las reglas que los estudiantes deben tener en cuenta al momento de desarrollar la actividad colaborativa.
4. Dar a conocer estrategias de colaboración, de modo que cada estudiante pueda usarlas al momento de interactuar con un par.
5. Explicar los procedimientos que debe seguir el estudiante para completar la actividad colaborativa.
6. Responder a las preguntas que se tengan sobre la expectativas y organización del trabajo colaborativo.

La técnica “discusión en grupo”, propone que el estudiante discuta con un compañero para contribuir en la resolución de la tarea, según (Hernández-leo and Asensio-pérez, 2019) los grupos de discusión son la forma más común de organizar la actividad en entornos de aprendizaje en red. Esta técnica permite abordar tres elementos principales:

1. Una discusión activa y sustancial, con muchas contribuciones en la tarea.
2. Los estudiantes al finalizar la discusión deben tener una buena comprensión de las contribuciones hechas.
3. Las contribuciones deben ser realizadas por todos los miembros de una clase.

En este estudio propone que los estudiantes con mejores resultados en la resolución de problemas de programación puedan discutir el desarrollo de un ejercicio con un par que tenga dificultades en la realización del mismo.

Respecto a la configuración de roles, se menciona “Programadores y novatos”. Esta es una técnica basada en la estrategia propuesta en (Hidalgo et al., 2021), la cual permite apoyar los procesos de asignación de roles y colaboración a través de la división entre estudiantes que logran resolver un problema de programación (programadores) y los que presentan dificultades (novatos).

La técnica de Programadores y Novatos es uno de los elementos principales del diseño del guion CSCL para este estudio, por medio de su aplicación en una actividad de resolución de problemas de programación se podrá estructurar el trabajo colaborativo a partir de la asignación de dos roles principales:

un “Programador”, correspondiente a los estudiantes que logren buenos resultados en un ejercicio de programación y un “Novato”, el cual está relacionado a aquellos estudiantes que presenten dificultades en la resolución de ejercicios.

La Figura 3-3 presenta una adaptación de la estrategia propuesta por (Hidalgo et al., 2021) en la cual se propone una fase de formación de grupos que consiste en clasificar a cada estudiante de acuerdo con los resultados obtenidos de un ejercicio de programación. En este estudio se adoptó esta misma idea buscando que los estudiantes tengan un rol dentro de una actividad en clase y que estos roles permitan generar grupos de colaboración más efectivos, de tal manera que los estudiantes con dificultades puedan encontrar y solicitar una ayuda a quien en realidad puede hacerlo.

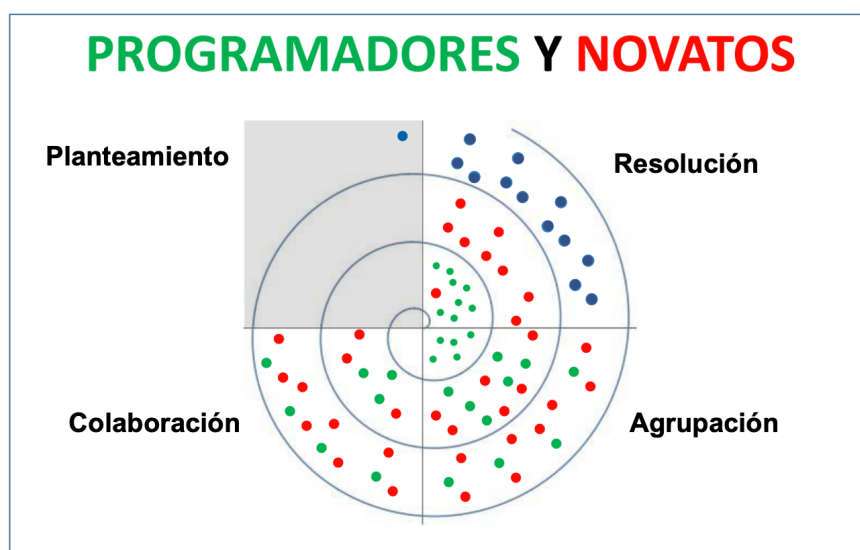


Figura 3-3: Técnica de asignación de roles: Programadores y Novatos, adaptado de (Hidalgo et al., 2021)

Asimismo, como se puede observar en la Figura 3-3 se plantean cuatro actividades principales para hacer uso de la técnica. Inicialmente, en la actividad de “Planteamiento” se establece que un docente debe proponer un problema de programación a resolver en un curso. “La resolución” es la segunda actividad de la técnica en la cual cada estudiante debe proponer una solución al problema planteado en la primera actividad. Durante las actividades de “Planteamiento” y “Resolución” los estudiantes no tienen ningún rol, ya que estos se definirán una vez cada estudiante proponga una solución al problema planteado por el docente. Seguidamente, una vez se conocen los resultados de cada ejercicio, esta técnica en la actividad de “Agrupación” busca que los estudiantes se clasifiquen en dos roles principales: los que obtienen buenos resultados (Programadores) y los que presentan dificultades (Novatos). Finalmente, la actividad de “Colaboración” busca que por medio de la clasificación descrita anteriormente se pueda establecer una comunicación entre los estudiantes con rol de “Programador” y los estudiantes con rol de “Novato”.

Esta técnica está pensada en ser cíclica, de tal manera que aquellos estudiantes que fueron clasificados con rol “Novato” puedan resolver el ejercicio propuesto y cambiar de rol a “Programador” en la medida que reciben una colaboración, buscando que a su vez puedan apoyar a otros compañeros con dificultades.

3.1.4. Determinación de recursos

La determinación de recursos hace parte del paso #4 del marco de trabajo visto en la Figura 3-1 y también en uno de los niveles de estructuración de guiones CSCL. De acuerdo con las TACs identificadas para el guion CSCL se ha decidido usar dos recursos computacionales: el desarrollo propuesto UNColab y la herramienta UNCode¹. UNCode es entorno educativo propuesto en (Restrepo Calle et al., 2018) para la práctica frecuente y la evaluación automática de ejercicios de programación de computadores, esta plataforma sirve de apoyo en cursos de programación en la Universidad Nacional de Colombia y cuenta con diferentes funciones de administración académica como la gestión de estudiantes, clases, tareas y entregas. Los estudiantes usan UNCode para presentar sus soluciones de código fuente, ya que les permite tener una realimentación automática sobre cada una de las tareas, además de diferentes componentes que apoyan la resolución de ejercicios de programación.

El diseño final de la herramienta de aprendizaje colaborativo UNColab se realizó buscando que su desarrollo funcione como un componente de la plataforma UNCode, ambas herramientas permitirán soportar las TACs descritas en la sección anterior. Basado en ello, se determinaron un conjunto de funcionalidades que deben contemplar las herramientas para hacer una práctica de CSCL en programación de computadores. UNColab junto a la plataforma UNCode soportan el nivel de recursos de acuerdo con los niveles de estructuración del guion CSCL. Como se puede observar en la Tabla 3-1 cada una de las funciones están soportadas por las herramientas establecidas.

Tabla 3-1: Funcionalidades y recursos para el diseño de un script CSCL en programación de computadores

Funcionalidades	Nivel de Recursos
Planteamiento de la tarea	UNCode
Resolución de la tarea	UNCode
Agrupación de estudiantes	UNCode, UNColab
Colaboración entre estudiantes	UNColab
Evaluación de la colaboración	UNColab , Google Forms

La funcionalidad “Planteamiento de la tarea” busca que un docente pueda proponer un reto o problema de programación de computadores a resolver. La “Resolución de la tarea” involucra un espacio en

¹ Más información en: <https://juezun.github.io/index.html>

Código fuente disponible en: <https://github.com/JuezUN/INGInious>

donde el estudiante pueda resolver el ejercicio. La “Agrupación de estudiantes” establece el uso de un mecanismo que permita asignar roles. La “Colaboración entre estudiantes” comprende un momento en el cual se suministre un medio de comunicación para establecer una ayuda. Finalmente, la “Evaluación de la colaboración” propone un instrumento que permita valorar la colaboración establecida en el ejercicio por medio de *Google Forms*².

Como se puede observar en la Tabla 3-1, algunas funciones que se requieren en el proceso ya son cubiertas por la plataforma UNCode. Por lo tanto, UNColab funciona como un recurso complementario para apoyar los procesos de agrupación y colaboración entre estudiantes y evaluación de la colaboración. En consecuencia, los requerimientos para UNColab se definieron a partir de los escenarios de uso descritos a continuación:

- Clasificar estudiante: establecer una división entre estudiantes de acuerdo al porcentaje de éxito de la tarea.
- Asignar rol: asignar un rol de “Programador” o “Novato” de acuerdo con la clasificación del estudiante.
- Notificar rol: cada estudiante debe conocer su rol durante la actividad colaborativa.
- Seleccionar compañero: cada estudiante debe poder seleccionar un compañero para establecer una colaboración.
- Escribir comentario: se debe proporcionar un mecanismo de comunicación para entablar una colaboración.
- Evaluar colaboración: cada estudiante debe evaluar la colaboración.

Teniendo en cuenta el escenario de uso propuesto para UNColab, la Figura 3-4 presenta el diseño de la herramienta de acuerdo a los principales procesos y TACs que se llevarán a cabo para lograr una actividad colaborativa en programación de computadores.

Se propuso que el diseño de la herramienta tomara como base los procesos principales de la plataforma UNCode para las funciones de **Planteamiento** y **Resolución** de tareas, para cada uno de los procesos asociados a estas funciones la plataforma usa un componente de software (*course administrator, students, task, submissions*). En el proceso **Proponer tarea**, la plataforma le permite a un profesor plantear un ejercicio de programación con diferentes configuraciones, dentro de las que están: casos de prueba, tiempo límite, lenguaje de programación, número de intentos, entre otras. A su vez, estos programas como tarea pueden ser resueltos dentro de la misma plataforma gracias a las funciones de edición de código y calificación automática³, para cada uno de los intentos que un estudiante realice;

²Google Forms es un software de administración de encuestas que funciona como un servicio en la nube dentro del conjunto de aplicaciones *workspace* de Google.

³La plataforma UNCode cuenta con un calificador automático de código fuente, el cual permite al estudiante obtener un porcentaje de 0 a 100 de acuerdo a la cantidad de casos de prueba resueltos.

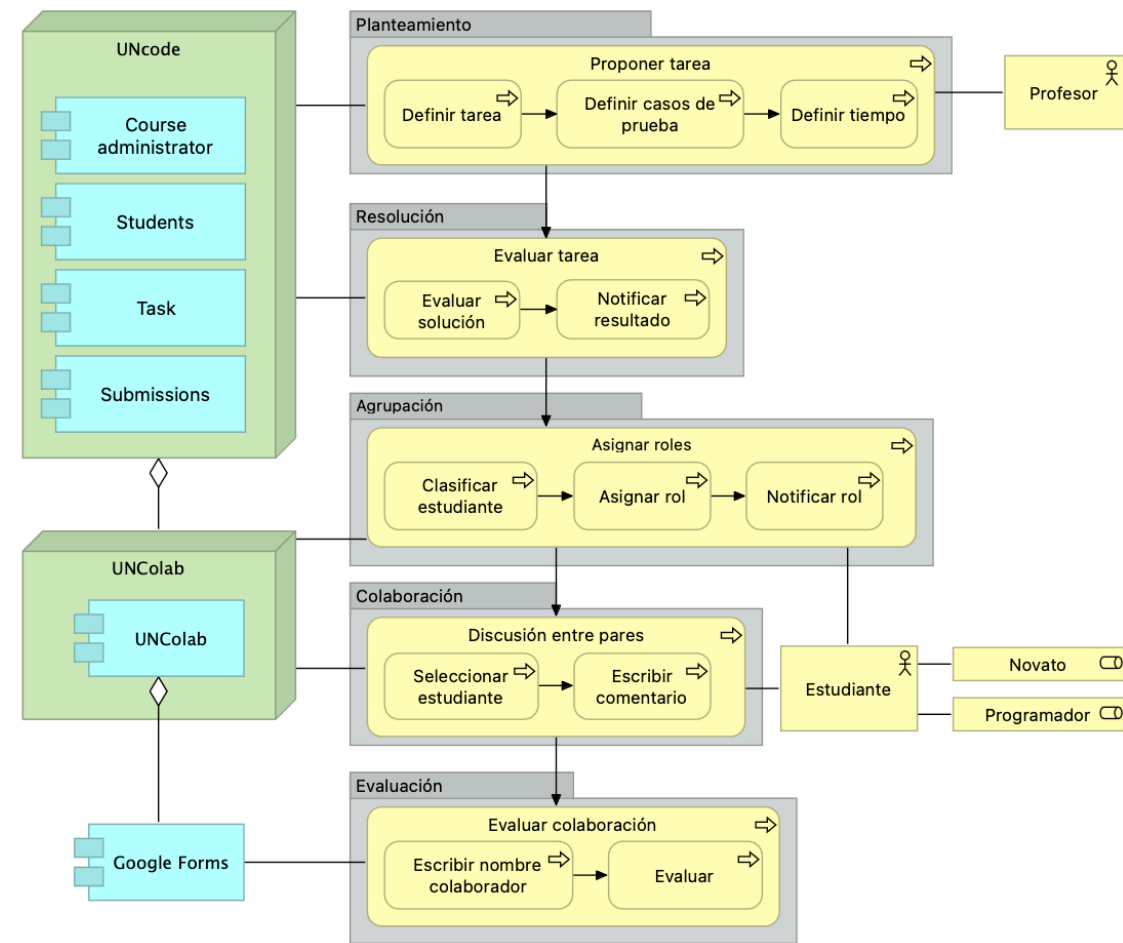


Figura 3-4: Diseño de herramienta computacional para el aprendizaje colaborativo de programación de computadores

esto se establece en el proceso **Evaluar tarea**.

La propuesta de diseño inicia a partir de la función **Agrupación**, para ello es necesario que se agregue un nuevo componente a UNCode, en este caso UNColab. Este componente soporta el proceso **Asignar roles** y cumple las actividades de clasificar al estudiante, asignarlo en un determinado rol y notificarle en qué rol le corresponde trabajar; en este caso los roles corresponden a **Programadores y Novatos**, los cuales son dos roles que el estudiante puede ejercer de acuerdo al porcentaje de resolución de su tarea. Un rol de “Programador” se asigna a un estudiante que ha resuelto el problema de programación de manera satisfactoria.

La plataforma UNCode permite conocer si un estudiante resolvió un ejercicio de manera satisfactoria mediante la cantidad de casos de prueba exitosos o aprobados —los casos de prueba corresponden a las salidas o respuestas esperadas sobre la solución de un problema de programación; estos son propuestos por el docente al momento de plantear un ejercicio de programación. En este sentido, si

una solución enviada por un estudiante cumple con todos los casos de prueba este estudiante se le asignará un rol de “Programador”. De igual forma, si un estudiante no logra cumplir la totalidad de los casos de prueba se le asignará un rol de “Novato”.

La función **Colaboración** incluye el proceso de **Discusión entre pares**, las actividades asociadas comprenden: Seleccionar estudiante, buscando que un estudiante pueda escoger a un par para establecer colaboración; y Escribir comentario, lo que permite que los estudiantes de ambos roles puedan enviar mensajes de manera instantánea.

La colaboración entre estudiantes tiene como objetivo que los estudiantes con dificultades en la resolución de un ejercicio de programación puedan solicitar ayuda a un compañero que haya resuelto el problema planteado. La función de colaboración está diseñada para capturar el rol que tiene asignado cada estudiante y de esta manera solo permitir colaboración entre “Programadores” y “Novatos”. En esta función cada estudiante puede elegir a quién colaborar o a quién solicitar ayuda, dependiendo del rol asignado. Adicionalmente, posibilita que cada estudiante pueda entablar comunicación con varios compañeros en diferentes instancias de tiempo; por ejemplo, un estudiante con rol “Programador” podrá colaborar a más de un estudiante con rol “Novato”.

Finalmente, la función **Evaluación** tiene asociado el proceso de **Evaluar colaboración**, el cual le permite a un estudiante ejecutar la actividad de seleccionar un par colaborador —ya sea programador o novato— y valorar su ayuda durante el desarrollo de la tarea. Este proceso es soportado por la aplicación *Google Forms*, la cual se agregará como complemento en el componente UNColab.

3.2. Desarrollo de la herramienta UNColab mediante la metodología orientada a prototipos

El desarrollo de la herramienta computacional UNColab se basó en la aplicación de la metodología de prototipado propuesta por Pressman (2002) la cual se presentó en la Figura 2-3.

3.2.1. Definición de especificaciones

Las especificaciones de la herramienta computacional UNColab se establecieron a partir de la definición de diferentes escenarios de uso. Estos permitieron profundizar en los diferentes procesos asociados a la colaboración entre estudiantes. La Tabla 3-2 presenta una descripción de los escenarios de la herramienta propuesta.

Por otra parte, hay que tener en cuenta que UNColab se desarrolló como un componente adicional de UNCode, para ello se asume que el siguiente flujo de eventos previos ya está incluido dentro de esta plataforma:

1. El Docente propone un programa como tarea a un grupo de estudiantes.
2. El Docente parametriza el tiempo de resolución de la tarea para el grupo.
3. El Estudiante resuelve el ejercicio propuesto.
4. El Estudiante envía la solución del ejercicio propuesto.
5. El Sistema registra los resultados de la resolución de la tarea.

Tabla 3-2: Definición de especificaciones UNColab

Escenario	Descripción	Proceso asociado
Clasificar estudiante	Se debe clasificar a un estudiante de acuerdo al porcentaje de éxito que haya obtenido sobre la resolución de una tarea de programación. Si un estudiante obtuvo 100 % en su tarea, la herramienta deberá clasificarlo como <i>Programador</i> ; de lo contrario, lo deberá clasificar como <i>Novato</i> .	Asignar roles
Asignar rol	En la interfaz de usuario la herramienta debe mostrar un listado de estudiantes de acuerdo con la clasificación de los dos roles (<i>Programadores</i> y <i>Novatos</i>). Si un estudiante fue clasificado como <i>Programador</i> solo debe poder ver estudiantes <i>Novatos</i> y viceversa.	Asignar roles
Notificar rol	La herramienta debe notificarle a un estudiante a qué rol pertenece una vez decida usar el componente de colaboración. Para poder observar esta clasificación debe realizar al menos dos intentos en la resolución de la tarea.	Asignar roles
Seleccionar estudiante	La herramienta debe permitirle a un estudiante seleccionar a un compañero con el cual quiere establecer una colaboración. Para ello, se debe mostrar el nombre de los estudiantes (<i>Programadores</i> o <i>Novatos</i>) dentro de un listado y abrir una interfaz de comunicación entre pares. El estudiante debe poder cambiar de compañero si así lo requiere.	Discusión entre pares
Escribir comentario	La herramienta debe proporcionar una interfaz de comunicación que permita enviar mensajes entre <i>Programadores</i> y <i>Novatos</i> a través de texto. Adicionalmente, debe permitir notificar cuando se tengan nuevos mensajes.	Discusión entre pares
Evaluar	Se debe proporcionar a cada estudiante (<i>Programador</i> o <i>Novato</i>) un formulario en que pueda escribir el nombre del compañero con el que estableció la discusión y evaluar su colaboración a partir de un conjunto de criterios. Adicional, la herramienta debe notificarle al estudiante cuándo realizar esta evaluación una vez se haya establecido algún tiempo de actividad colaborativa.	Evaluar colaboración

La Tabla 3-2 deja ver a nivel global el comportamiento que tendrá la herramienta para cumplir cada uno de los procesos principales. A nivel de prototipos no se requiere definir al detalle los requerimientos —como si se realiza en otras metodologías de desarrollo de software—, ya que el desarrollo inicial del prototipo dará la especificación completa de estos para otras iteraciones incrementales o evolutivas de la herramienta.

3.2.2. Diseño conceptual

El objetivo de un diseño conceptual es transformar las necesidades de un prototipo en ideas o modelos que permitan conocer qué se quiere poner a prueba a nivel de desarrollo. Existen diferentes técnicas y herramientas para realizar diseños conceptuales; en este caso se utilizaron *wireframes*, los cuales permiten estructurar un boceto del sistema que se va a construir (Hamm, 2014). En la Figura 3-5 se pueden observar los *wireframes* diseñados para la herramienta propuesta.

Como se puede observar en la Figura 3-5, el diseño conceptual fue dividido en 4 etapas de navegación. En la primera etapa, se representa la plataforma UNCode a partir de dos escenarios: un estudiante que no logra resolver una tarea de programación, definido como *novato* (color rojo) y uno que si lo hace, definido como *programador* (color verde). En la segunda etapa, se presenta la herramienta UNColab – representada por un botón de color vinotinto –, en esta etapa se busca que la herramienta pueda notificar a un estudiante si es *programador* o *novato*, para ello se presenta un cuadro de diálogo que indica esta asignación de un rol. En la tercera etapa, se diagrama el escenario en el que cada estudiante puede establecer una discusión con un par; el estudiante *programador* puede ver un listado de estudiantes con rol *novato* y viceversa. Finalmente, la cuarta etapa representa la evaluación de la colaboración para los estudiantes de ambos roles; una notificación en la interfaz de usuario es la principal función de esta etapa, la cual indica el momento de dirigirse al formulario que contiene el esquema de evaluación de la colaboración.

3.2.3. Desarrollo del prototipo

La plataforma UNCode está basada en el juez virtual INGINIOUS⁴, el cual permite realizar pruebas automatizadas de código fuente; este sistema está escrito en el lenguaje de programación Python y usa un *backend* con contenedores livianos (Docker) para ejecutar el código del estudiante en diferentes lenguajes de programación; de manera independiente, maneja un *frontend* al cual se le permite agregar diferentes componentes o *plugins* para extender las funcionalidades principales.

Con base en lo anterior, la herramienta propuesta UNColab⁵ se desarrolló como un componente que se permite agregar a la plataforma UNCode. En la Figura 3-6 se puede observar la arquitectura de software utilizada para el desarrollo. Para cada una de las funciones descritas en la Tabla 3-2 se usó el lenguaje de programación Python y JavaScript con la librería ReactJs⁶.

Como se puede observar en la Figura 3-6 la herramienta propuesta está contenida en un componente principal denominado UNColab, este componente a su vez incluye dos componentes denominados *api* y *Chat*. El componente *api* se encarga de manejar las solicitudes que se envían desde el componente

⁴ <https://inginius.org/>

⁵ El código fuente de UNColab está disponible en: https://github.com/jhonmendex/un_colab.git

⁶ ReactJs es una librería especializada en la creación de interfaces de usuario (<https://es.reactjs.org/>)

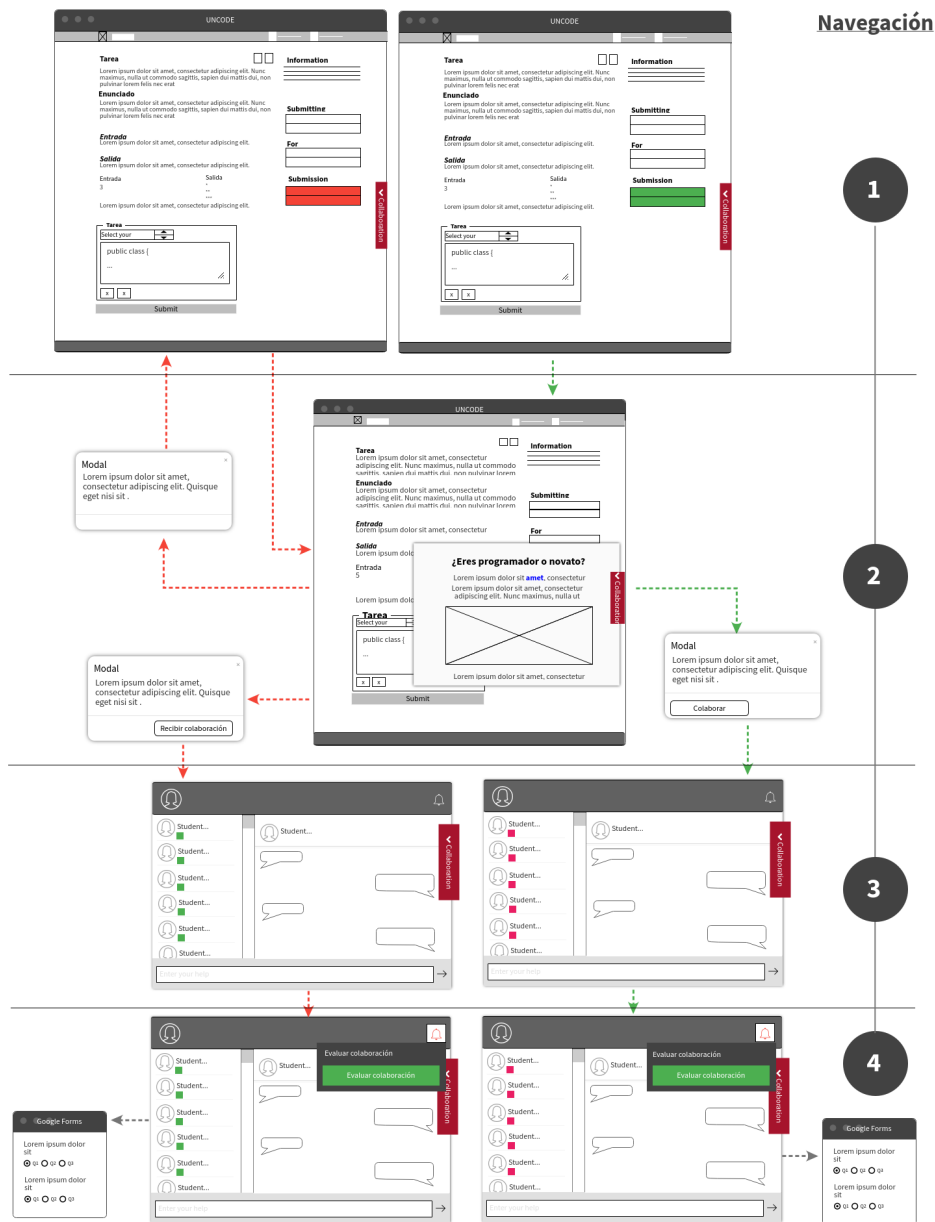


Figura 3-5: Diseño conceptual UNColab

Chat a la plataforma UNCode, principalmente consultas a la base de datos del sistema base INGenious. Por otro lado, en esta arquitectura se establece usar la base de datos Firestore del servicio en la nube Firebase⁷ Chat.

⁷ Firebase es una plataforma ubicada en la nube que usa un conjunto de herramientas para la creación y sincronización de proyectos orientados a la web.

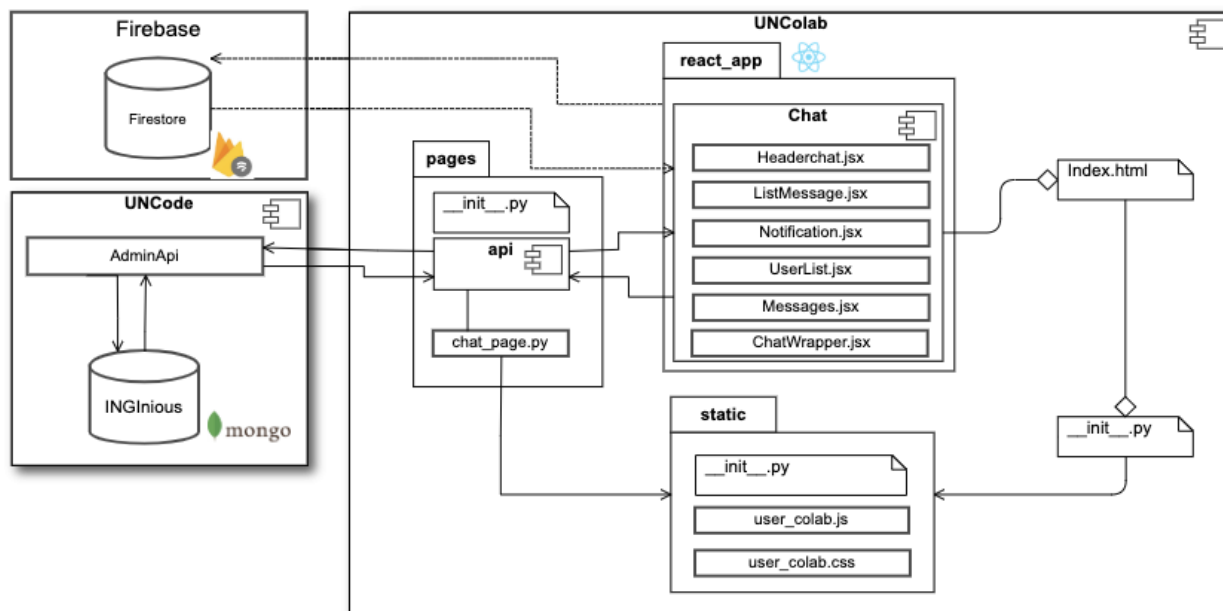


Figura 3-6: Arquitectura UNColab

Los archivos incluidos dentro de los componentes de la arquitectura de la herramienta propuesta están enfocados en realizar cada uno de los escenarios vistos en la Tabla 3-2. A continuación se detallan estos escenarios y su relación con la arquitectura de UNColab.

Clasificar estudiante

Para esta primera funcionalidad de la herramienta se utilizó el paquete *pages* presentado en la Figura 3-6, este paquete contiene un componente de API (del inglés *Application Programming Interface*) que permite recibir solicitudes de las vistas (componente *Chat*) y conectarlas con la base de datos de UNCode. La clasificación del estudiante en el rol de *programador* o *novato* depende del porcentaje de éxito de su tarea, por ello se usan los datos de envío (*submission*). En la Figura 3-7 se puede observar un fragmento del código desarrollado para esta función.

Asignar rol

Esta función permite que la herramienta muestre un listado de estudiantes de acuerdo a la clasificación anterior. Para ello se utilizó el paquete *react_app* y el componente *UserList.jsx* (ver Figura 3-6), este componente reconoce el usuario registrado en UNCode e inmediatamente realiza una conexión al servicio en la nube *Firebase* y un registro en la base de datos *Firestore*. Posteriormente, se realiza

```
class ScoreTaskUser(AdminApi):
    def API_GET(self):
        parameters = web.input()
        course_id = get_mandatory_parameter(parameters, "course_id")
        task_id = get_mandatory_parameter(parameters, "task_id")
        user = self.user_manager.session_username()
        db_results = self.database.submissions.find({
            "username": user,
            "courseid": course_id,
            "taskid": task_id,
            "status": "done"},
            {"username": True, "_id": False, "taskid": True,
            "result": True, "submitted_on": True}).sort([("submitted_on", pymongo.ASCENDING)])
        success = 0
        fails = 0
        for submission in db_results:
            if submission['result'] == "success":
                success += 1
            else:
                fails += 1
```

Figura 3-7: Funcionalidad de clasificación de estudiantes: ejemplo de código fuente

una solicitud al API de UNCode para verificar si el usuario registrado tiene algún envío (*submission*) de la tarea en la que está ubicado; en caso de que exista un envío, el componente verifica su porcentaje de éxito y muestra una interfaz diferente dependiendo del rol. En la Figura 3-8 se puede observar un fragmento del código fuente de esta clasificación.

Notificar rol

La función de notificar rol permite indicar al estudiante a qué rol fue asignado (*programador* o *novato*). Esta función fue desarrollada usando el paquete *react_app*, y mediante una función del archivo principal se realiza una petición a la API de UNCode. Esto permite conocer si el estudiante ha enviado alguna tarea y, si es el caso, su porcentaje de éxito. Una vez obtenidos estos datos, se despliega una ventana emergente con la indicación al estudiante; dentro de estas indicaciones están: si debe realizar otro intento para solicitar colaboración, si ya puede recibir colaboración de un compañero y si ya puede brindar colaboración a un compañero. En la Figura 3-9 se puede observar un fragmento de la petición que se realiza al API.

Seleccionar estudiante

El paquete *react_app* mediante el componente *UserList.jsx* (ver Figura 3-6) permite que se listen los estudiantes y a partir de una función de JavaScript se da la opción a cada estudiante de seleccionar un compañero para establecer una discusión; cabe mencionar que esta selección depende del rol al que fue asignado el estudiante. La Figura 3-10 muestra un fragmento del código fuente de la función de selección.

```

const userList = (props) => (
  <React.Fragment>
    <div id="uncolabChat_leftbar">
      <div className="tab-content" id="nav-tabContent">
        <div id="home" role="tabpanel" aria-labelledby="nav-home-tab">
          {props.users.map((usr, index) =>
            props.taskState === "true"
            ? props.currentUserId !== usr.id &&
              !usr.data.programmer && (
                <label
                  key={index}
                  htmlFor={usr.id}
                  className="uncolabChat_users"
                  onClick={() => props.getPairUser(usr.id, usr.data)}
                >
                  <div
                    className={
                      props.currentPairUser &&
                      props.currentPairUser.id === usr.id
                      ? "uncolabChat_userlist_focus"
                      : "uncolabChat_userlist"
                    }
                  >
                    <span className="uncolabChat_userscontentavatar">
                      <img
                        alt="imagen"
                        className="uncolabChat_userscontentavatarimage"
                        src={
                          usr.data.programmer
                            ? AppString.PROGRAMMER_IMAGE
                            : AppString.ROOKIE_IMAGE
                        }
                      />
                    </span>
                  </div>
                </label>
              )
            : null
          )}
        </div>
      </div>
    </div>
  </React.Fragment>
)

```

Figura 3-8: Funcionalidad de asignación de roles: ejemplo de código fuente

Como se puede observar, la propiedad *taskState* verifica el estado de la tarea, mostrando un listado de estudiantes de acuerdo al porcentaje de éxito que se tenga en esta. Los datos de la tarea ya han sido cargados en funciones anteriores, por lo que no es necesario solicitarlos a la API de nuevo.

Escribir comentario

Esta función permite realizar una discusión o diálogo entre pares a través de mensajes escritos; una vez se usa la función anterior, cada estudiante podrá escribirles a cuantos compañeros desee; siempre y cuando no pertenezcan al mismo rol. En la Figura 3-11 se puede observar un fragmento del código implementado para esta función. Se utilizan componentes *ListMessage* y *Messages* (ver Figura 3-6) los cuales contienen una conexión al servicio de *Firebase* y mediante el uso de la base de datos *Firestore* se almacenan todos los mensajes de los estudiantes a través de las funciones *Snapshot()*, que permiten la detección de lectura de datos en tiempo real; esto permite que se tenga una comunicación sincrónica entre estudiantes.

```
fetchTaskStatus = async () => {
  this.setState({ isLoading: true, error: false });
  try {
    let url =
      "/plugins/un_colab/api/score_task_user?" +
      $.param({ course_id: this.state.courseId, task_id: this.state.taskId });
    let taskStatusUser = await fetch(url);
    let data = await taskStatusUser.json();
    this.setState({
      isLoading: false,
      data: data.programmer,
      fails: data.fails,
    });
  } catch (error) {
    this.setState({ isLoading: false, error: true });
  }
};
```

Figura 3-9: Funcionalidad de notificar rol: ejemplo de código fuente

```
const userList = (props) => (
  <React.Fragment>
    <div id="uncolabChat_leftbar">
      <div className="tab-content" id="nav-tabContent">
        <div id="home" role="tabpanel" aria-labelledby="nav-home-tab">
          {props.users.map((usr, index) =>
            props.taskState === "true"
            ? props.currentUserId !== usr.id &&
              !usr.data.programmer && (
                <label
                  key={index}
                  htmlFor={usr.id}
                  className="uncolabChat_users"
                  onClick={() => props.getPairUser(usr.id, usr.data)}
                >

```

Figura 3-10: Funcionalidad de seleccionar estudiante: ejemplo de código fuente

Evaluar

La evaluación de la colaboración entre estudiantes se realizó usando una aplicación externa llamada Google Forms; sin embargo, es necesario que cada estudiante pueda ser notificado del momento adecuado para realizar este proceso. Para esta notificación se usa el componente *HeaderChat.jsx* el cual contiene una función que permite contabilizar el tiempo desde que se estableció alguna comunicación entre estudiantes, una vez se cumple algún tiempo parametrizado se renderiza una alerta en la cabecera del chat. En la Figura 3-12 se puede ver un fragmento del renderizado condicional.

```

getListHistory = () => {
  if (this.removeListener) {
    this.removeListener();
  }
  this.listMessage.length = 0;
  if (
    this.hashString(this.props.currentUser.id) <=
    this.hashString(this.props.currentPairUser.id)
  ) {
    this.groupChatId = `${this.props.currentUser.id}-${this.props.currentPairUser.id}`;
  } else {
    this.groupChatId = `${this.props.currentPairUser.id}-${this.props.currentUser.id}`;
  }

  this.removeListener = myFirestore
    .collection(AppString.MESSAGES)
    .doc(this.groupChatId)
    .collection(this.groupChatId)
    .onSnapshot(
      (snapshot) => {
        snapshot.docChanges().forEach((change) => {
          if (change.type === AppString.DOC_ADDED) {
            this.listMessage.push(change.doc.data());
            this.child.scrollToMyRef();
          }
        });
      },
      (err) => {
        console.log(err);
      }
    );
};

```

Figura 3-11: Funcionalidad de escritura de comentarios: ejemplo de código fuente

3.2.4. Pruebas

De acuerdo con las especificaciones vistas en la Sección 3.2.1 y el diseño propuesto en la Sección 3.1, se obtiene un prototipo funcional el cual fue sometido a una prueba de integración y de componentes. La primera prueba, permitió conocer la compatibilidad del componente en la plataforma UNCode; la segunda, fue necesaria para verificar la funcionalidad de los módulos conforme el diseño y especificaciones del prototipo.

Integración

UNCode proporciona un sistema de extensión de funcionalidades denominada *hooks*, estas deben usarse dentro de un archivo Python en un directorio con el nombre del componente; posteriormente, debe registrarse el directorio en un archivo de configuración tal como muestra en la Figura 3-13. Como se puede observar, en este archivo se habilitan los componentes adicionales de UNCode; para

```
{evaluate && taskState === "false" && (  
  <ListItem className={classes.myListItem}>  
    <ListItemText  
      className={classes.myListItemText}  
      primary="Evalúa la colaboración"  
      secondary="El siguiente botón te llevará al formulario de evaluación."  
    />  
  </ListItem>  
  <ListItem className={classes.myListItem}>  
    <Button  
      className={classes.buttonNotification}  
      variant="contained"  
      target="_blank"  
      rel="noopener"  
      href="https://docs.google.com/forms/d/1LzpWxbLYPxEJlpFKsfn0m8A_pnzC71oL9NxdKcv0eCk/edit?usp=sharing"  
      startIcon={<AssignmentIcon />}  
    >  
      Evaluar colaboración  
    </Button>  
  </ListItem>  
</>  
)}
```

Figura 3-12: Funcionalidad de evaluación a través de enlace a Google Forms: ejemplo de código fuente

```
cbackend: local #tcp://127.0.0.1:2000  
backup_directory: ../backup  
local-config: {}  
  
mongo_opt:  
  database: INGINIOUS  
  host: localhost  
superadmins:  
- superadmin  
tasks_directory: ../tasks  
tmp_dir: /tmp  
use_minified_js: true  
plugins:  
- plugin_module: inginius.frontend.plugins.statistics  
  use_minified: false  
- plugin_module: inginius.frontend.plugins.multilang  
  show_tools: true  
  use_minified: false  
- plugin_module: inginius.frontend.plugins.task_cache  
- plugin_module: inginius.frontend.plugins.problem_bank  
- plugin_module: inginius.frontend.plugins.grader_generator  
- plugin_module: inginius.frontend.plugins.custom_input  
- plugin_module: inginius.frontend.plugins.UN_template  
#use_minified: false  
- plugin_module: inginius.frontend.plugins.UNCode  
- plugin_module: inginius.frontend.plugins.register_students  
- plugin_module: inginius.frontend.plugins.un_colab
```

Figura 3-13: Archivo de integración de componentes UNCode

las pruebas de integración no se habilitaron todos los complementos de UNCode, ya que esto se realizó en la implementación con algunos usuarios de prueba. Esta prueba se realizó sobre una máquina virtual de Ubuntu 18.0 en donde se instaló una versión de UNCode similar a la encontrada el ambiente de producción. Como resultado de esta prueba, se verificó que la plataforma funcionara de manera correcta al agregar el nuevo complemento; la Figura 3-14 muestra una vista del funcionamiento de

UNCode y UNColab.



Figura 3-14: Captura de pantalla: prueba de integración UNColab

Componentes

Cada una de las funcionalidades del prototipo UNColab fueron probadas mediante la creación de usuarios simulados en la plataforma UNCode y algunas tareas de programación que ya habían sido usadas en el ambiente de producción. Adicionalmente, se simuló una resolución de ejercicio por parte de dos usuarios con el fin de verificar los procesos de Asignación de roles, Discusión entre pares y Evaluación. Las pruebas realizadas para estos procesos están representadas en la Figura 3-15.

3.2.5. Implementación

Una vez finalizado el desarrollo y las pruebas de funcionalidad, se procedió a implementar una versión de UNCode en servidor en la nube con el fin de evaluar la herramienta UNColab en un ambiente real con estudiantes de programación de computadores. Se selecciona Google Cloud como plataforma de nube al permitir configurar ambientes virtualizados mediante sus funciones *Compute Engine*.

Compute Engine es un componente de infraestructura como servicio que permite instanciar o lanzar diferentes sistemas operativos virtualizados ⁸. Como se puede observar en la Figura 3-16 el sistema operativo seleccionado para implementar UNCode fue CentOS 7, dentro de este sistema operativo se configuró e instaló la plataforma INGenious, que es el sistema base sobre el cual está desarrollado UNCode. Dentro de la configuración del sistema base se encuentran diferentes componentes (*plugins*) que permiten extender funciones de *frontend* y *backend*, estas funciones fueron usadas para agregar el prototipo propuesto UNColab como una extensión de UNCode mediante un manejador de *plugins*

⁸<https://cloud.google.com/compute/>

3.2 Desarrollo de la herramienta UNColab mediante la metodología orientada a prototipos 45



Figura 3-15: Capturas de pantalla: prueba de componentes UNColab

que viene dentro de INGenious. Todos los componentes de UNCode pueden acceder a los datos de los usuarios mediante el uso de la base de datos MongoDB⁹; adicionalmente, también se puede tener acceso a los contendores Docker¹⁰, los cuales permiten evaluar automáticamente el código fuente enviado por los usuarios.

⁹<https://www.mongodb.com/es>

¹⁰[docker.com](https://www.docker.com)

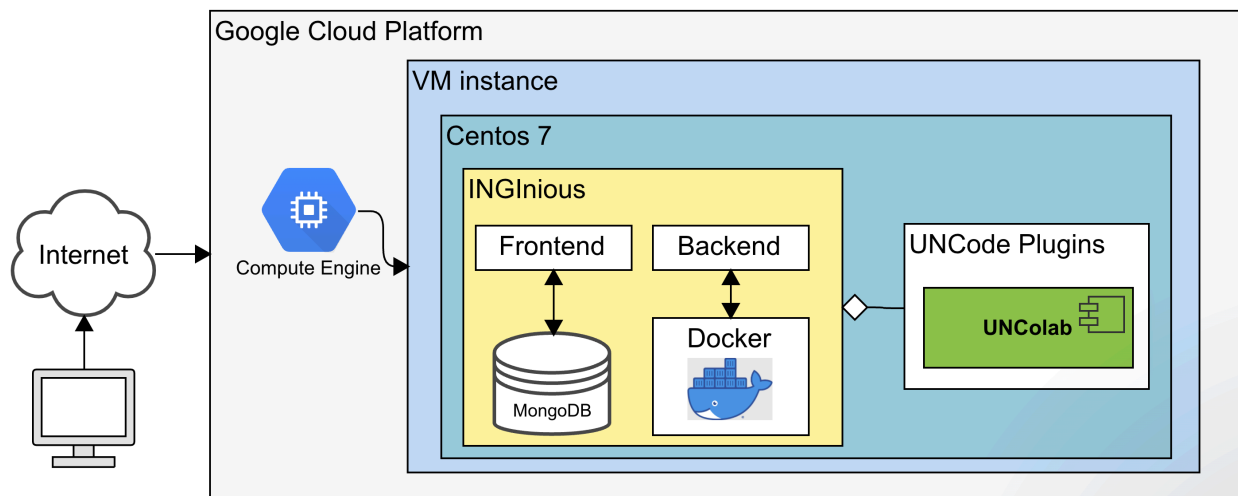


Figura 3-16: Implementación UNColab en Google Cloud

4 Diseño del estudio

Este estudio está enfocado en conocer los efectos en la resolución de problemas de programación de computadores de manera colaborativa a partir de utilizar una herramienta computacional que facilite la evaluación formativa entre pares. Para este fin, se propuso desarrollar la herramienta vista en el **Capítulo 3**, cuyos elementos y funciones fueron diseñados acorde a la teoría del aprendizaje colaborativo asistido por computador, descrita en el **Capítulo 2**. Para lograr conocer estos efectos y validar la herramienta desarrollada se eligió un diseño de investigación de estudio de caso. Según Bryman (2015), este tipo de estudio implica analizar hechos, acontecimientos o situaciones sobre un caso en particular buscando comprender su naturaleza o complejidad. En consecuencia, se eligió un grupo de estudiantes de programación de computadores para el estudio de caso en el cual se realizaron actividades de resolución de problemas de programación utilizando la herramienta UNColab.

Como estudio de caso se propone realizar dos intervenciones al curso de programación de computadores seleccionado en el cual se implemente la herramienta UNColab en la resolución de ejercicios de programación. La Figura 4-1 presenta las diferentes fases que se llevaron a cabo para la realización de dos intervenciones en el curso. La configuración de estas intervenciones comprende el uso por parte de los estudiantes de las herramientas UNCode y UNColab con el fin de realizar un proceso de resolución un ejercicio de programación y posteriormente un proceso de colaboración entre estudiantes.

La fase I establece la selección de los participantes del estudio, para ello se ha usado la técnica de selección de muestreo por conveniencia o selección de grupos intactos mencionada en (Hernández

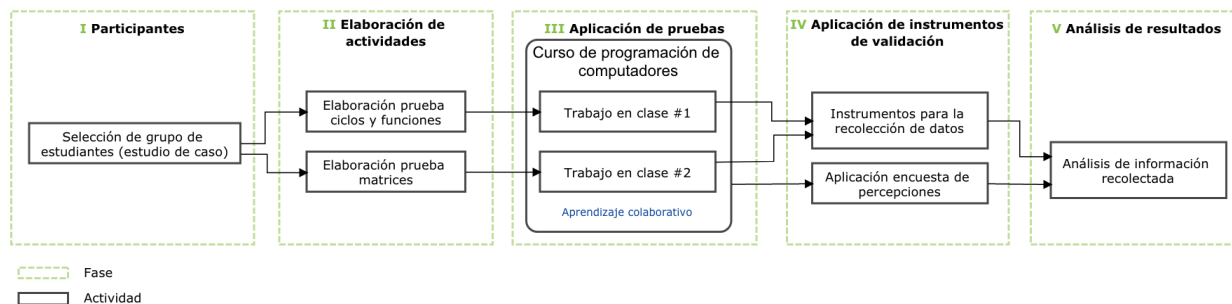


Figura 4-1: Metodología del diseño del estudio

Sampieri et al., 2006). Esta técnica se ha utilizado debido a la facilidad de acceso a los participantes y la disposición de los mismos. En este sentido, esta fase comprende la selección de un grupo de estudiantes inscritos en una asignatura introductoria de programación de computadores que dentro de su temática de estudio se trabajen conceptos de condicionales, funciones y matrices. De acuerdo al cronograma de estas temáticas en el curso se programaron las intervenciones con la herramientas UNCode y UNColab.

La fase II comprende una propuesta de ejercicios de programación y de actividades para abordar las temáticas de estudio condicionales, funciones y matrices. Las temáticas fueron seleccionadas de acuerdo con la consulta literaria realizada, en donde se evidencia que son estos conceptos básicos en donde se tienen problemas en la enseñanza-aprendizaje de programación de computadores (Marcolino and Barbosa, 2017; Piteira and Costa, 2012). En este estudio se elaboraron dos actividades para abarcar las temáticas propuestas: la primera, buscando trabajar los conceptos de ciclos y funciones; la segunda, el concepto de matrices.

La fase III, denominada “Aplicación de pruebas”, se compone de dos actividades de trabajo en clase para el grupo de estudiantes. Tanto en la primera como en la segunda actividad el grupo utiliza la herramienta UNCode y el complemento UNColab para resolver los ejercicios de programación propuestos en la fase II. La única diferencia entre estas actividades son las temáticas de los ejercicios que se trabajan y las instancias de tiempo en que se realizan. Cabe mencionar que la herramienta UNCode es usada en el curso durante gran parte del periodo académico. En esta etapa propuesta se realizan ejercicios de aprendizaje colaborativo al usar el complemento UNColab.

La fase IV establece aplicar al grupo de estudiantes dos instrumentos de validación a partir de dos actividades: la primera, denominada “Instrumento para la recolección de datos”, busca que por medio del complemento UNColab los estudiantes puedan evaluar la colaboración recibida y brindada en las actividades de la fase III; la segunda, denominada “Aplicación encuesta de percepciones”, permite conocer algunas apreciaciones sobre la herramienta UNColab en la resolución de los ejercicios de la fase III. El instrumento de evaluación de la colaboración se incluyó en cada una de las actividades de la fase III, mientras que la encuesta de percepciones se incluyó solo al finalizar la segunda actividad de esa misma fase.

Finalmente, la fase V propone un análisis cuantitativo y cualitativo de los datos obtenidos durante las fases III Y IV.

4.1. Participantes

De acuerdo con la aplicación de la técnica de selección por conveniencia, fue seleccionado un curso de “Programación de computadores” de la Universidad Nacional de Colombia, adscrito a la Facultad

de Ingeniería, conformado por estudiantes de diferentes programas académicos que cursaban la asignatura en el segundo periodo académico del año 2020. Contando con la colaboración del docente titular, se pudo establecer un cronograma de trabajo con el fin de realizar las dos intervenciones en el curso propuestas en la sección anterior.

La conformación del grupo de estudiantes se puede observar en la Tabla 4-1. Cabe mencionar que no se intervino en la organización de los estudiantes en el grupo, debido a que este ya estaba formado con los estudiantes inscritos en la asignatura “Programación de computadores”. En total participaron 27 estudiantes distribuidos de la siguiente manera: 55.5 % de Ingeniería Electrónica; 11.1 % de Ingeniería Mecatrónica y 33.3 % de Ingeniería Agrícola.

Tabla 4-1: Participantes en el estudio

Grupo	Programa académico	Cantidad participantes
Programación de computadores	Ingeniería agrícola	9
	Ingeniería mecatrónica	3
	Ingeniería electrónica	15

La asignatura “Programación de computadores” en la Universidad Nacional de Colombia es una asignatura de introducción a la programación y estudia los elementos básicos para resolver problemas a través de un lenguaje de programación; en este curso principalmente se trabaja con el lenguaje de programación Python. La Tabla 4-2 muestra el conjunto de temas que aborda la asignatura durante un periodo de 16 semanas; la dinámica de trabajo dada por los docentes por lo general establece dos sesiones de dos horas divididas en teoría y práctica, respectivamente.

4.2. Elaboración de actividades

En el desarrollo de los temas del curso “Programación de computadores” gran parte de las tareas propuestas por los docentes a cargo son ejercicios de programación; estos buscan que el estudiante afiance los conceptos vistos en la teoría y pueda mejorar sus habilidades en la resolución de problemas de programación mediante la práctica continua. La plataforma UNCode soporta la elaboración, resolución y evaluación de estas tareas, por ello se ha tomado como punto de partida en este estudio, ya que precisamente esta función de evaluación automática de tareas permite conocer el estado de cada una de las entregas realizadas por un estudiante. Por consiguiente, se propuso la integración del componente UNColab con el fin de conocer los efectos que puede tener la colaboración entre estudiantes en la resolución de estas tareas. La Figura 4-2 presenta un flujo de actividades generales que son realizadas en por el grupo de participantes de este estudio.

El flujo de actividades inicia con el ingreso de un estudiante a la plataforma UNCode (#1) con su usuario y contraseña respectivos —la versión de UNCode usada por los estudiantes es la expuesta en

Tabla 4-2: Temáticas de la asignatura “Programación de computadores”

Semana	Tema
1	Problemas, algoritmos y programas
2	Introducción a la Programación de computadores
3	Variables, tipos de datos, E/S
4	Variables, tipos de datos, E/S
5	Estructuras de control condicionales
6	Estructuras de control iterativas 1
7	Estructuras de control iterativas 2
8	Examen parcial I
9	Funciones
10	Listas
11	Arreglos y Matrices
12	Ordenamiento, diccionarios y conjuntos
13	Archivos y diccionarios
14	Gráficos y recursión
15	Programación Orientada a Objetos
16	Examen parcial II

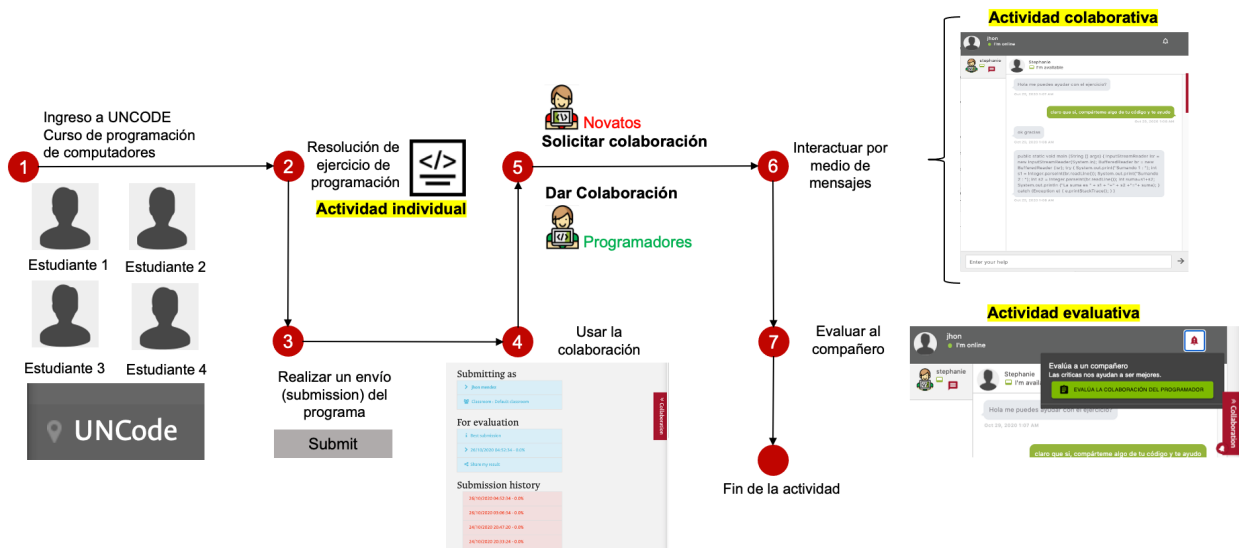


Figura 4-2: Flujo de actividades del caso de estudio

la Sección 3.2.5. La **Actividad individual** (#2) comprende un trabajo individual correspondiente a la resolución de una tarea de programación de computadores a través del uso de UNCode. Se espera que el estudiante realice un envío de solución y que esta sea evaluada por la plataforma UNCode (#3). Una

vez enviada la solución, el estudiante puede hacer uso del componente de colaboración UNColab (#4) y, dependiendo del resultado de la tarea evaluada, tendrá un rol dentro de la actividad (“Programador” o “Novato”); de acuerdo a ello, se podrá dar o recibir ayuda (#5). A partir de allí se da inicio a la **Actividad colaborativa** (#6) la cual consiste en el intercambio de mensajes entre Programadores y Novatos a través de la herramienta UNColab. Finalmente, se espera que cada estudiante pueda realizar la **Actividad evaluativa** (#7) respecto a la valoración de la colaboración dada o recibida.

4.3. Aplicación de pruebas

Las actividades presentadas en la sección 4.2 fueron configuradas para realizarse en dos intervenciones de la clase en diferentes fechas, para cada una se preparó un ejercicio de programación de acuerdo al tema visto en la teoría. La Tabla 4-3 muestra una distribución de las actividades establecidas y los aspectos de participación en estas intervenciones de acuerdo con la dinámica *Programadores* y *Novatos* expuesta en la Sección 3.1.

Tabla 4-3: Distribución de actividades del caso de estudio

	Intervención #1	Intervención #2
Temáticas de trabajo	Estructuras de control y funciones	Matrices
Distribución de actividades	<ul style="list-style-type: none"> ■ 20 min. de actividad individual ■ 25 min. de actividad colaborativa ■ 15 min. de actividad evaluativa 	<ul style="list-style-type: none"> ■ 20 min. de actividad individual ■ 25 min. de actividad colaborativa ■ 15 min. de actividad evaluativa ■ 15 min. Encuesta de percepción
Aspectos de participación	<ul style="list-style-type: none"> ■ Para pedir colaboración un estudiante debe tener al menos dos intentos significativos de resolución del programa. ■ Una vez el estudiante use el botón de colaboración el sistema notificará si es <i>Programador</i> o <i>Novato</i> de acuerdo al porcentaje de éxito del ejercicio. ■ El sistema notificará cuándo se debe hacer la evaluación de la colaboración a un compañero. ■ Un <i>programador</i> puede ayudar a varios <i>Novatos</i>. ■ Un <i>novato</i> puede recibir colaboración de varios <i>Programadores</i> ■ La actividad finaliza una vez se realice la actividad evaluativa y la encuesta de percepción. 	

La primera intervención se configuró con un ejercicio que involucra los temas “estructuras de control” y “funciones”; adicional a los 60 minutos de la actividad, se tuvo en cuenta un tiempo de 15 minutos al inicio de la sesión de clase con el fin de explicar la dinámica *Programadores* y *Novatos*, la distribución del tiempo para cada una de las actividades y los aspectos de participación vistos en la Tabla 4-3. En la

segunda intervención, se propuso un ejercicio que aborda el tema “Matrices”, se tiene la misma configuración de la anterior intervención, sumando 15 minutos al final en los cuales el estudiante debe diligenciar la encuesta de percepciones sobre las actividades realizadas en la herramienta UNColab. Para programar estas intervenciones se tuvo en cuenta el cronograma del curso y la dinámica usada por el docente a cargo, esto con el fin de asegurar que los estudiantes previamente hayan trabajado los temas en una sesión de clase teórica y, adicionalmente, contaran con experiencia en el manejo de la plataforma UNCode.

La ejecución de cada una de las actividades de este estudio se realizó con base en el cronograma establecido en la asignatura “Programación de computadores” en un curso de la Universidad Nacional de Colombia. La Tabla 4-4 presenta las semanas correspondientes a cada intervención en el curso de programación. Inicialmente, en las semanas 5, 9 y 11 los estudiantes reciben una sesión de clase teórica en los temas Estructuras de Control (TEC), Funciones (TF) y Matrices (TM), respectivamente. En la semana 13 y 14, en el curso se ejecuta el sistema UNCode y el complemento UNColab sobre el despliegue establecido en la Sección 3.2.5, en este despliegue se cargan los ejercicios relacionados a los temas nombrados. Con respecto a los instrumentos de medición, en la semana 13 y 14 en cada actividad del curso aplica el instrumento de evaluación de la colaboración por pares. Finalmente, en la misma intervención de la semana 14 se aplica encuesta de percepciones sobre las actividades realizadas en UNColab.

Tabla 4-4: Cronograma de actividades de las intervenciones en el curso

	Actividad	Recursos	S5	S9	S11	S12	S13	S14	S15	S16
Curso de programación	Intervención #1	UNCode, UNColab	TEC	TF			X			
	Intervención #2	UNCode, UNColab			TM			X		

4.4. Aplicación de instrumentos de validación

Este estudio se realizó con la finalidad de conocer los efectos en la resolución de problemas de programación de computadores a partir del uso de una herramienta computacional colaborativa que facilite la evaluación formativa entre pares. Para ello se propuso recolectar datos cuantitativos y cualitativos en las intervenciones realizada al grupo de estudio.

Datos cuantitativos

La resolución de problemas de programación comprende un conjunto de estrategias que un estudiante utiliza para responder a los requerimientos planteados, estas son utilizadas tanto para comprender el contexto del problema como para dar la solución en un lenguaje de programación. En esto último, las diferentes herramientas que un estudiante utiliza para resolver el problema le pueden indicar si

lo hizo de manera correcta. Para este estudio se utilizó la plataforma UNCode, esta herramienta tiene una forma práctica de conocer si un estudiante pudo resolver un problema de programación de manera correcta, lo hace a través del cumplimiento de diferentes casos de prueba que el docente incluye dentro de cada ejercicio como también de otros factores como la cantidad de intentos, el tiempo, los recursos de máquina u otros elementos que el docente crea necesarios evaluar; en consecuencia, el porcentaje de éxito sobre la resolución de un ejercicio dependerá de la cantidad de casos que el estudiante apruebe de acuerdo a las condiciones establecidas.

La implementación de UNColab en UNCode permite obtener datos relacionados con la resolución de un problema de programación a través de la colaboración. De acuerdo con las técnicas de aprendizaje colaborativo aplicadas al grupo de estudio se captura el número de programadores, y así mismo, el número de novatos, el número de cambios de rol, el número de colaboraciones y el número de mensajes. Las descripciones de estos datos se pueden observar en la Tabla 4-5.

Tabla 4-5: Datos cuantitativos recolectados en UNColab

Datos cuantitativos	Descripción
Número de programadores	De acuerdo con la TAC Programadores y Novatos, el número de programadores corresponde a la cantidad de estudiantes que logran resolver los casos de prueba de un ejercicio de programación de computadores.
Número de novatos	De acuerdo a la TAC Programadores y Novatos, el número de novatos corresponde a la cantidad de estudiantes que no logran resolver los casos de prueba de un ejercicio de programación de computadores.
Número de cambios de rol	De acuerdo a la TAC Programadores y Novatos, el número de cambios de rol corresponde al número de estudiantes que inicialmente fueron asignados como Novatos y eventualmente cambiaron de rol a Programadores.
Número de colaboraciones	De acuerdo a la TAC Programadores y Novatos, el número de colaboraciones corresponde a la cantidad de parejas Programador-Novato que se logran conformar durante la resolución de un ejercicio de programación de manera colaborativa.
Número de mensajes	De acuerdo a la TAC Discusión en grupo, el número de mensajes corresponde a la cantidad de interacciones escritas que una pareja Programador-Novato logran realizar durante una etapa de colaboración en la resolución de un ejercicio de programación de computadores.

Datos cualitativos

Respecto a los datos cualitativos se establece recolectar información relacionada con la evaluación de la colaboración y las percepciones de la herramienta UNColab. La evaluación de la colaboración comprende una valoración de la ayuda tanto recibida como ofrecida por un estudiante respecto a la aplicación de la TAC Programadores y Novatos durante una etapa de colaboración en la resolución de un ejercicio de programación de computadores. Esta información permitirá conocer si este tipo

de técnicas puede apoyar la resolución de ejercicios en clase. Por otra parte, las percepciones de los estudiantes respecto al uso de la herramienta propuesta UNColab permitirá conocer la importancia, impresiones o apreciaciones sobre la experiencia de usar elementos de aprendizaje colaborativo asistido por computador para apoyar la resolución de problemas de programación.

Para conocer las valoraciones de la colaboración y las percepciones generales de UNColab se crearon tres encuestas en Google Forms correspondientes a: valoración de la colaboración ofrecida por el rol “Programador”, valoración de la colaboración recibida por el rol “Novato”, y finalmente, percepciones generales acerca de UNColab. Estos instrumentos fueron anclados a la aplicación UNColab con el fin de que los estudiantes una vez realizaran la etapa colaborativa pudiesen realizar la evaluación al instante. La descripción de estos instrumentos se presenta en la siguiente sección.

4.4.1. Instrumentos de recolección de datos

Los datos necesarios para iniciar el análisis de resultados de este estudio provienen de dos aspectos principales: la evaluación de la colaboración entre estudiantes y las percepciones acerca de la herramienta UNColab. Para el primer aspecto, se crean dos instrumentos de evaluación que le permiten a cada estudiante evaluar la colaboración de sus compañeros; para el segundo aspecto, se plantea una encuesta de percepciones con el fin de conocer las apreciaciones de los estudiantes respecto a la herramienta UNColab.

Instrumentos de evaluación de la colaboración

Según el flujo de actividades visto en la Figura 4-2 de la Sección 4.2, la actividad evaluativa se realiza una vez los estudiantes se colaboran para solucionar el problema de programación. La TAC planteada en el estudio, llamada *Programadores y Novatos*, indica que los estudiantes que logren solucionar o realizar los casos de prueba de un ejercicio pueden colaborar a quienes no lo han podido resolver. En este contexto, se realizaron dos instrumentos de evaluación que le permiten a cada estudiante valorar la utilidad de la colaboración tanto recibida como ofrecida.

El primer instrumento de evaluación de la colaboración fue denominado: “Evaluación de la colaboración - Programadores”, permite que el estudiante clasificado como *Programador* pueda valorar si la colaboración brindada a un estudiante *Novato* fue útil para ayudarlo a resolver el problema de programación. Este instrumento puede ubicarse en el siguiente enlace de Google Forms: <https://forms.gle/A69c4YabqHsTEq9e8>. La Tabla 4-6 presenta los elementos asociados al instrumento. Como se puede observar, está compuesto por cuatro preguntas con opciones de una escala Likert de 6 opciones y una pregunta abierta; adicionalmente, se incluye un campo para que cada estudiante escriba el nombre del compañero a evaluar. Este mismo esquema se utilizó para el segundo instrumento de evaluación.

El segundo instrumento fue denominado: “Evaluación de la colaboración - Novatos”, permite que el

Tabla 4-6: Instrumento de evaluación de la colaboración - Programadores

#	Enunciado	Opción de respuesta
1	Considero que la colaboración dada a mi compañero fue útil para ayudarlo a mejorar su propuesta de solución del problema de programación.	1. Totalmente en desacuerdo 2. En desacuerdo 3. Algo en desacuerdo 4. Algo de acuerdo 5. De acuerdo 6. Totalmente de acuerdo
2	Considero que la colaboración dada a mi compañero le ayudó a identificar algunos errores de su propuesta de solución.	
3	Considero que la colaboración dada a mi compañero le ayudó a corregir algunos errores de su propuesta de solución.	
4	Considero que la colaboración dada me permitió afianzar mis conocimientos sobre la temática del problema.	
5	¿Qué opinión tienes acerca de la colaboración ofrecida a tu compañero?	Abierta

estudiante *Novato* pueda valorar si la colaboración recibida de un estudiante *Programador* le fue útil para resolver el problema de programación. Este instrumento está disponible en el siguiente enlace de Google Forms: <https://forms.gle/h8hdwc33WaoizbjH9>. La Tabla 4-7 presenta los elementos que contiene el instrumento.

Tabla 4-7: Instrumento de evaluación de la colaboración - Novatos

#	Enunciado	Opción de respuesta
1	Considero que la colaboración recibida me permitió mejorar mi propuesta de solución del problema de programación.	1. Totalmente en desacuerdo 2. En desacuerdo 3. Algo en desacuerdo 4. Algo de acuerdo 5. De acuerdo 6. Totalmente de acuerdo
2	Considero que la colaboración recibida me ayudó a identificar algunos errores que tenía mi propuesta de solución.	
3	Considero que la colaboración recibida me ayudó a corregir algunos errores que tenía mi propuesta de solución.	
4	Considero que la colaboración recibida me dio mayor confianza para intentar resolver el problema de programación.	
5	¿Qué opinión tienes acerca de la colaboración recibida?	Abierta

Encuesta de percepciones

Este instrumento busca principalmente que se puedan valorar las apreciaciones respecto a la utilidad de la herramienta UNColab para el aprendizaje colaborativo de programación de computadores. Las preguntas asociadas a esta encuesta están basadas en tres aspectos destacados del aprendizaje colaborativo: la utilidad de las herramientas CSCL, la comunicación y la participación. Esta encuesta fue elaborada en Google Forms y puede ser consultada en el siguiente enlace: <https://forms.gle/z8JcByGVTcbCnwgh7>. En la Tabla 4-8 se pueden observar las preguntas formuladas en el cuestiona-

rio, para los aspectos nombrados se tienen tres preguntas con opciones de tipo escala Likert y aparte una pregunta abierta. Cabe mencionar que de manera anticipada se presenta una sección que contiene un consentimiento informado y acuerdo de confidencialidad, en el que cada estudiante indica si está o no de acuerdo con participar libre y voluntariamente en la investigación.

Tabla 4-8: Encuesta de percepciones de la herramienta colaborativa UNColab

#	Enunciado	Opción de respuesta
1	Considero que la herramienta UN_COLAB fue útil para la resolución de problemas de programación de manera colaborativa.	1. Totalmente en desacuerdo 2. En desacuerdo 3. Algo en desacuerdo 4. Algo de acuerdo 5. De acuerdo 6. Totalmente de acuerdo
2	Considero que la herramienta UN_COLAB facilitó la comunicación con mis compañeros para resolver el problema de programación de manera colaborativa.	
3	Considero que la herramienta UN_COLAB fomentó mi participación en la resolución del problema de programación de manera colaborativa.	
4	En general, ¿qué opina de la herramienta de colaboración UN_COLAB?	Abierta

4.5. Análisis de resultados

Esta fase comprende el análisis de los datos cualitativos y cuantitativos capturados por medio de los instrumentos de recolección de datos establecidos en la sección 4.4. El análisis de los resultados tiene como finalidad conocer las percepciones de los estudiantes sobre el uso de la herramienta UNColab, como también las valoraciones respecto a la colaboración establecida entre los participantes al momento de recibir o dar apoyo en la resolución de un problema de programación. Por un lado, los datos cuantitativos recolectados a través de UNColab permiten tener una visión general sobre las interacciones realizadas entre estudiantes al momento de resolver el problema de manera colaborativa. Por otro lado, los datos cualitativos obtenidos mediante la aplicación de los instrumentos de recolección de datos permiten reconocer los posibles efectos en la resolución de problemas de programación de manera colaborativa el usar herramientas CSCL. En el próximo capítulo se presentan los resultados obtenidos de este estudio a partir del análisis de los datos recolectados.

5 Resultados

De acuerdo con el estudio de caso propuesto en la Sección 4.2, las dos intervenciones propuestas al grupo de estudio plantearon tres actividades principales: una actividad individual, una actividad colaborativa y una actividad evaluativa. La actividad individual fue soportada por la herramienta UNCode y permitió conocer qué estudiantes lograban resolver el ejercicio propuesto en los primeros intentos y cuáles no lograban hacerlo. Esta información posteriormente fue utilizada por la herramienta UNColab para que los estudiantes pudiesen hacer la actividad colaborativa, que consistió en la asignación de roles, formación de grupos e intercambio de mensajes. Finalmente, la actividad evaluativa permitió recolectar las impresiones y valoraciones referentes a la colaboración entre estudiantes mediante el uso de UNColab y la herramienta *Google Forms*. De esta forma, tanto los datos cuantitativos como cualitativos se obtuvieron de las plataformas UNCode, UNColab y *Google Forms*.

UNCode permitió recolectar la información relacionada con la resolución de los ejercicios presentados por los estudiantes, mientras que UNColab permitió obtener datos relacionados con la colaboración entre estudiantes. En cuanto a los datos cualitativos, UNColab facilitó la evaluación de la colaboración a través de la implementación de dos encuestas desarrolladas en *Google Forms* que fueron ancladas al sistema. Al finalizar las dos intervenciones, se solicitó a los estudiantes diligenciar la encuesta de percepciones que capturó las apreciaciones y comentarios referentes al uso de la herramienta desarrollada. A continuación se presentan los resultados obtenidos en este estudio.

5.1. Resolución de problemas de programación a través de la colaboración

La resolución de los ejercicios propuestos en las dos intervenciones realizadas implicaba, en general, que cada estudiante se tomara un tiempo específico de 20 minutos en proponer una solución en la plataforma UNCode y, dependiendo del resultado obtenido, el componente UNColab lo clasificaba como “Programador” o “Novato”. Posterior a este tiempo, los estudiantes podían tanto solicitar como brindar ayuda a través de la función de chat de la herramienta desarrollada. Los datos capturados durante los procesos de resolución de problemas y colaboración de cada ejercicio propuesto en las intervenciones realizadas se muestran en la Tabla 5-1.

Los datos muestran que en las dos intervenciones realizadas aproximadamente la mitad de los estudiantes presentaron dificultades en resolver los ejercicios durante la etapa de resolución individual;

Tabla 5-1: Resolución de ejercicios a través de la colaboración: resultados

Datos recolectados	Resultados	
	Intervención 1	Intervención 2
Número de programadores	13	12
Número de novatos	14	15
Número de cambios de rol	6	2
Número de colaboraciones	12	39

a estos estudiantes la herramienta UNColab los clasificó como “Novatos”. Durante la actividad colaborativa, se obtuvo que en la primera intervención 6 estudiantes lograron cambiar de rol de “Novato” a “Programador”; es decir, estudiantes que lograron resolver el ejercicio a través de la colaboración brindada por un compañero. En la segunda intervención en este aspecto se ve una disminución a 2 cambios de rol; sin embargo, cabe resaltar que para cada intervención se trabajaron temáticas diferentes: *Estructuras de control y funciones* para la intervención 1; y *Matrices* para la intervención 2. A pesar de que en la segunda intervención se reflejaron menos cambios de rol, fue una en la que los estudiantes interactuaron más entre ellos a través de la herramienta al generar más grupos de colaboración. Como se presenta en la Tabla 5-1, para la primera intervención se formaron 12 parejas Programador-Novato y 39 en la segunda, esto indica que al menos cada estudiante del curso pudo establecer comunicación con un par para discutir el ejercicio a resolver.

Otro de los datos capturados durante el proceso de resolución de ejercicios fue el intercambio de mensajes entre estudiantes durante la actividad colaborativa de ambas intervenciones. La Figura 5-1 muestra un diagrama de cajas para representar la distribución de las cantidades de mensajes intercambiados por cada pareja Programador-Novato durante la etapa de colaboración en cada una de las intervenciones.

En la primera intervención se registró un promedio de 3,0 mensajes y una desviación estándar de 2,9, mientras que en la segunda intervención se registró un promedio de 6,6 mensajes y una desviación estándar de 11,0. Estos datos demuestran que la herramienta UNColab fue usada por los estudiantes durante la etapa de colaboración en las dos intervenciones. Se evidencia que en la segunda intervención hubo un mayor número de intercambio de mensajes y algunos datos atípicos producto de grupos o pares de estudiantes que establecieron una comunicación constante durante toda la actividad colaborativa, a diferencia de la mayoría de los grupos cuya comunicación fue esporádica.

5.2. Evaluación de la colaboración

Los datos obtenidos sobre las valoraciones de la colaboración entre estudiantes, al momento de resolver un ejercicio de programación de computadores, corresponden a la aplicación de los instrumentos

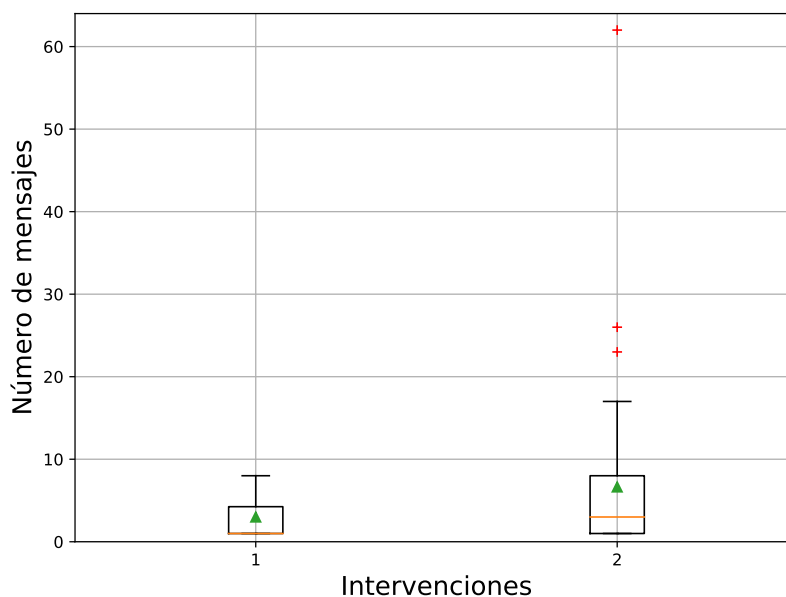


Figura 5-1: Distribución de cantidad de mensajes intercambiados entre estudiantes en el componente de chat durante las dos intervenciones realizadas.

vistos en las Tablas 4-6 y 4-7 de la Sección 4.4.1. La evaluación de la colaboración se llevó a cabo en las dos intervenciones realizadas al grupo de estudio, en las cuales los participantes contaban con un tiempo de 15 minutos para evaluar la colaboración tanto recibida como ofrecida dependiendo del rol asignado por la herramienta UNColab.

La Figura 5-2 presenta los resultados obtenidos de acuerdo a las cuatro primeras preguntas del instrumento de evaluación para “Programadores”. De igual forma, la Figura 5-3 presenta los resultados del instrumento de evaluación para “Novatos”.

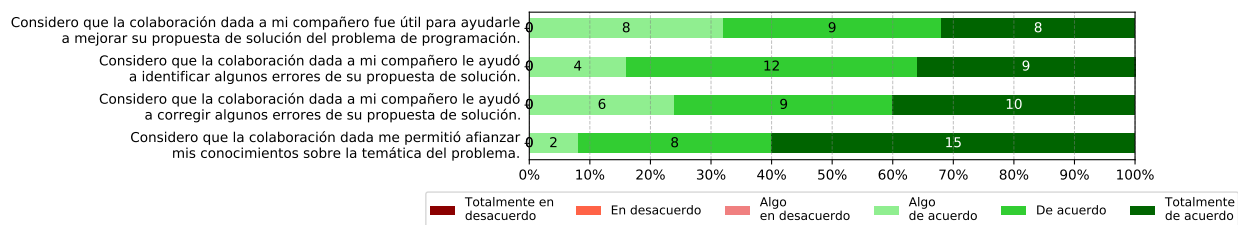


Figura 5-2: Respuestas obtenidas del instrumento de evaluación de colaboración “Programadores” por parte de estudiantes con rol de “Programador”.

Cabe recordar que para cada intervención que se realizó al grupo de estudio se solicitó a los estudiantes evaluar la colaboración de acuerdo con los diferentes roles que se podían dar al momento de resolver el ejercicio. Las respuestas que se muestran en la Figura 5-2 y la Figura 5-3 fueron compi-

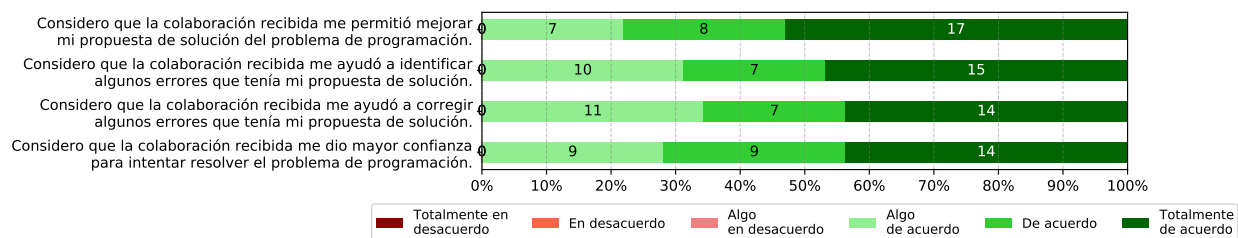


Figura 5-3: Respuestas obtenidas del instrumento de evaluación de colaboración “Novatos” por parte de estudiantes con rol de “Novato”.

ladas a partir de los datos recolectados de ambas intervenciones. Esta compilación se realizó debido a que el principal objetivo de cada instrumento consistió en conocer las valoraciones de la colaboración realizada por parte de cada rol (Programador y Novato) al momento de resolver un ejercicio de programación; independiente de la temática del problema a resolver. Como resultado, se recolectaron 25 respuestas para la evaluación de “Programadores” y 32 para la evaluación de “Novatos”. La cantidad de respuestas obtenidas se refiere a las valoraciones que cada estudiante podía hacer sobre uno o varios compañeros dependiendo del rol en el que se encontraba.

En cuanto al análisis de las respuestas obtenidas, el 100 % de los estudiantes coincidieron con algún nivel de acuerdo en que la colaboración dada o recibida les había sido útil para mejorar las soluciones de los problemas de programación, para identificar y corregir errores en la propuesta de solución, para consolidar el conocimiento en la temática evaluada y mejorar la confianza al solucionar el problema de programación.

Los datos recolectados en ambos instrumentos también comprenden una respuesta abierta respecto a la opinión acerca de la colaboración ofrecida y recibida. A partir de la recolección de estas opiniones se buscó recolectar características o aspectos relevantes que no fueron contemplados en las preguntas cerradas del mismo instrumento. Para realizar el análisis de estas respuestas se utilizó la técnica de *análisis de contenido* presentada en (Hernández Sampieri et al., 2006). Esto permitió identificar indicadores, categorizar y codificar cada uno de los comentarios dados por los estudiantes de acuerdo a la interpretación y sentido que el estudiante expresaba en el contexto del rol que desempeñaba (“Programador” o “Novato”).

El proceso de análisis de contenido se realizó de manera iterativa mediante la revisión y acompañamiento del director y codirector de este proyecto. Se realizaron diferentes encuentros sincrónicos en donde se propusieron y ajustaron diferentes indicadores, categorías y temas para cada una de las respuestas recolectadas en los instrumentos de evaluación de la colaboración.

El análisis de contenido inició con la recolección de las respuestas abiertas dadas por los estudiantes

mediante los instrumentos de evaluación de la colaboración para ambos roles (Programador y Novato). Se reunieron las respuestas de ambos instrumentos con el fin de identificar características que ambos roles lograran manifestar respecto a su opinión sobre la colaboración en general.

Una vez se recolectaron las opiniones, cada una de estas respuestas se agrupó en diferentes indicadores de acuerdo al grado de afinidad de los comentarios dados por los estudiantes. Cabe mencionar que durante este proceso se tuvo en cuenta el rol que desempeñaba el estudiante (Programador o Novato) al momento de responder la pregunta, con el fin realizar una interpretación adecuada sobre la respuesta recolectada.

Posteriormente, buscando caracterizar cada uno de los indicadores identificados, se establecieron diferentes categorías que permitieron agrupar y etiquetar en diferentes aspectos más amplios las opiniones dadas por los estudiantes.

Finalmente, la aplicación de la técnica de análisis de contenido permitió identificar un tema general que agrupó las categorías e indicadores que se reconocieron en las respuestas anteriormente descritas para los instrumentos de evaluación. A continuación se presenta el análisis de las categorías encontradas de acuerdo al tema principal denominado: *Colaboración en la resolución de problemas de programación*.

Colaboración en la resolución de problemas de programación

La pregunta final de cada uno de los instrumentos de evaluación buscaba indagar más acerca de las percepciones de los estudiantes frente a la colaboración ofrecida y recibida por un compañero al momento de resolver un problema de programación. La Figura 5-4 presenta las cinco categorías obtenidas a partir de análisis de contenido realizado, estas categorías están asociadas a un tema general identificado como “Colaboración en la resolución de problemas de programación”. Como se puede observar, cada una de estas categorías tiene asociado uno o varios indicadores, los cuales permitieron reconocer y agrupar diferentes características sobre los comentarios y opiniones dadas por los estudiantes.

En la categoría denominada **identificar errores de código fuente** se identificó un indicador a partir de 5 respuestas relacionadas con la identificación de errores en el código fuente. Aquí los estudiantes manifestaron, desde sus diferentes roles (programador/novato), la utilidad de la ayuda de un compañero para identificar posibles fallos de una solución propuesta. En atención a este indicador se pueden ver respuestas como por ejemplo: “...me pareció que le fue muy útil para identificar fallos que tenía y en mi caso aprender mas” [sic] (respuesta de un programador), “...es agradable que compañeros que ya lo lograron puedan orientarlo a uno con los errores que tiene” (respuesta de un novato).

Solución de problemas de programación es una categoría que recopila 3 indicadores que relacionan

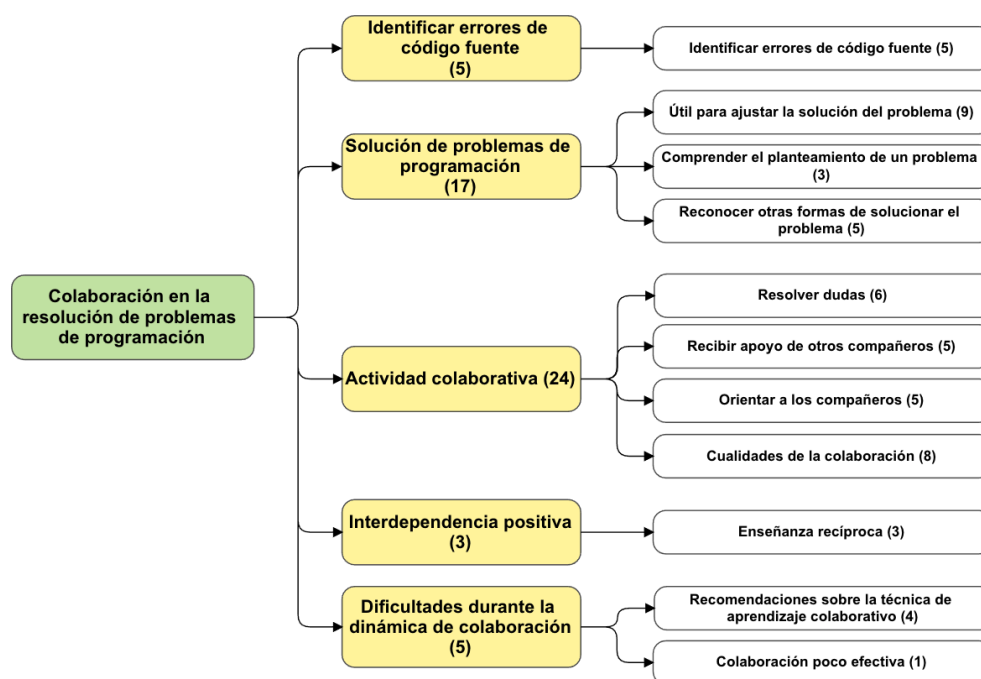


Figura 5-4: Colaboración en la resolución de problemas de programación: resultado de análisis de contenido

aspectos de algunas actividades comunes que se dan al momento de resolver un problema de programación; por ejemplo, la revisión del planteamiento del problema, el ajuste del mismo o la consulta a otras soluciones ya establecidas. El indicador nombrado **útil para ajustar la solución del problema** recopiló nueve opiniones referentes a estudiantes cuyas propuestas de solución para los ejercicios requerían de solo una pequeña colaboración para dar con la solución del problema. En cuanto a este indicador se pueden encontrar respuestas como: *“Implementó casi la totalidad del programa y verifico algunos conocimientos para asegurarse de su solución”* [sic] (respuesta de un programador), *“...gracias a la ayuda pude completar el ejercicio”* (respuesta de un novato).

El segundo indicador relacionado con la categoría **Solución de problemas de programación** reúne 3 opiniones relacionadas con el uso de la colaboración para **comprender el planteamiento de un problema**. En este indicador, los estudiantes reconocen la colaboración como una ayuda para aclarar dudas sobre lo que se espera que el programa requerido resuelva. Algunos de los comentarios vistos en este indicador se muestran a continuación: *“Tenía un enfoque diferente en su planteamiento, por eso considero que la falta de conocimientos de programación no era el factor por el cual no podía realizar su ejercicio”* (respuesta de un programador), *“Me ayudó a replantearme el problema de una manera más sencilla y práctica”* (respuesta de un novato).

El tercer indicador identificado para la categoría **Solución de problemas de programación** establece el uso de la colaboración como forma de **reconocer otras formas de solucionar un problema**. Este

indicador abarcó 5 comentarios en donde los estudiantes con rol de novato expresaron el uso de la colaboración como una manera de conocer otros modos de resolver un problema y no solo como un medio de resolución de dudas. Los siguientes corresponden a algunos comentarios dados por los estudiantes: *“Con la colaboración se logran ver otras formas de solucionar los problemas...”* [sic] (respuesta de un novato), *“Es necesaria porque no todas las ideas que uno tiene para resolver un código son las adecuadas...”* (respuesta de un novato).

La **actividad colaborativa** es una categoría identificada de acuerdo a la recolección de opiniones dirigidas a expresar algunas facilidades, ventajas y cualidades de la colaboración al momento de resolver un problema de programación; a partir del análisis realizado se agruparon estas opiniones en 4 indicadores. El primer indicador relaciona los comentarios cuya percepción de colaboración está encaminada en la actividad de **resolver dudas**, los estudiantes en sus diferentes roles reconocen la ayuda por parte de un compañero como un momento para preguntar y aclarar aspectos del código fuente que no se tenían claros. En este sentido, se pueden evidenciar dentro de esta agrupación los siguientes ejemplos de comentarios: *“Mi compañero pudo resolver algunas dudas, pero tuvo muchos inconvenientes en su código...”* (respuesta de un programador), *“...asi que me respondia concretamente y asi me aclaro las dudas que tenia”* [sic] (respuesta de un novato).

El segundo indicador correspondiente a la **actividad colaborativa** agrupa 5 comentarios que expresan la utilidad que un estudiante percibe al **recibir apoyo de otros compañeros**, en especial aquellos que requieren o reciben una colaboración. Adicionalmente, los estudiantes distinguen una facilidad al momento de solicitar la ayuda debido a que la asignación de roles les permite conocer qué compañeros pudieron solucionar el problema y están prestos a ayudar. Dentro de este indicador se reflejan comentarios como: *“...pienso que el concepto de tener el respaldo y la ayuda de alguien es bastante agradable”* (respuesta de un novato), *“...la ayuda es personalizada y no necesariamente hay que ir al profesor sino que los compañeros que ya lo lograron pueden colaborarnos”* [sic] (respuesta de un novato).

El tercer indicador de la categoría **actividad colaborativa** fue denominado **Orientar a los compañeros**. Agrupa 5 comentarios relacionados con percepciones de estudiantes que lograron resolver los ejercicios propuestos (programadores) y ven en la colaboración un mecanismo que les permite acercarse a aquellos estudiantes con dificultades en la resolución de un problema. Algunos ejemplos de los comentarios pueden verse a continuación: *“...para los próximos cursos de programación va a hacer que las personas que manejen mejor un tema puedan orientar a los que les cuesta o están un poco mas perdidos”* [sic] (respuesta de un programador), *“Me pareció que intente hacer la mejor colaboración posible, intentando explicar cada paso que realizamos para la generación de nuestra solución...”* [sic] (respuesta de un programador).

Finalmente, en la categoría **actividad colaborativa** se presenta el indicador **Cualidades de la colaboración**. Este indicador asoció diferentes comentarios relacionado con cualidades y ventajas que los estudiantes encontraron al realizar la actividad colaborativa y, en general, de la colaboración ofrecida

y recibida. Estas cualidades se pueden observar en los siguientes ejemplos de comentarios: “*fue objetiva y practica*” [sic] (respuesta de un novato), “*Buena colaboración y rápida*” (respuesta de un novato), “*fue bastante rápida*” (respuesta de un programador), “*...es mas fácil ya que la ayuda es personalizada y no necesariamente hay que ir al profesor...*” [sic] (respuesta de un novato).

Una de las categorías identificadas dentro del análisis de contenido y que hace parte fundamental de la teoría del aprendizaje colaborativo es la **interdependencia positiva**. Este concepto hace referencia a un conjunto de elementos que involucra un reconocimiento por parte de los estudiantes sobre la importancia de la colaboración para el cumplimiento de objetivos comunes (Barkley et al., 2014). Dentro de estos elementos existe la **enseñanza recíproca**, un concepto sobre el cual se ha planteado un indicador que reunió 3 opiniones en donde los estudiantes con rol de *programador* expresaron haber aprendido algo de manera mutua al ofrecer la colaboración a un compañero. El siguiente ejemplo de respuesta soporta este indicador: “*...pienso que ambos pudimos aprender mutuamente, ayudándonos unos a otros...*” (respuesta de un programador).

Por último, se identificó una categoría que reunió algunas **dificultades durante la dinámica de colaboración** por parte de los estudiantes. Estas dificultades fueron agrupadas en dos indicadores que relacionan opiniones acerca de la configuración de la técnica de aprendizaje colaborativo utilizada para este estudio y una percepción de ineficacia en la colaboración. El primer indicador se ha definido como **recomendaciones sobre la técnica de aprendizaje colaborativo** ya que en 4 opiniones los estudiantes manifiestan no contar con el tiempo suficiente para poder ayudar o recibir ayuda; a pesar de ello, se percibe en los estudiantes que lograron cumplir con el objetivo del ejercicio una buena disposición para colaborar a quienes no lograron hacerlo. Dentro de estas opiniones se encuentran los siguientes ejemplos: “*Esta bien, pero aun me falta tiempo para expresar mis ideas junto con las que me digan.*” [sic] (respuesta de un novato), “*...había disposición para entender y responder las preguntas planteadas, a pesar de que el tiempo fuera algo corto.*” (respuesta de un novato). Finalmente, el segundo indicador recolectó la opinión de un único estudiante cuya respuesta relaciona una **colaboración poco efectiva** debido a la dificultad de solucionar el ejercicio, a pesar de haber recibido la ayuda por parte de un compañero. La respuesta dada en este indicador se cita a continuación: “*Luis me respondía pero no creo que hubiera sido muy útil para la solución*” [sic] (respuesta de un novato).

5.3. Percepciones acerca del uso de UNColab

La herramienta UNColab descrita en la Sección 3 se desarrolló con el fin de servir como elemento mediador en la resolución de ejercicios de programación de manera colaborativa a partir de funciones que permitieron apoyar una actividad de evaluación formativa entre pares. Una vez se logró realizar las dos intervenciones en el curso de programación, se invitó a los estudiantes a evaluar UNColab buscando conocer las valoraciones sobre la utilidad de la herramienta durante el proceso de resolución de ejercicios de programación a través de la colaboración entre estudiantes. La Figura 5-5 presenta los

resultados relacionados con las tres primeras preguntas del instrumento de percepciones presentado en la Sección 4.4.1.

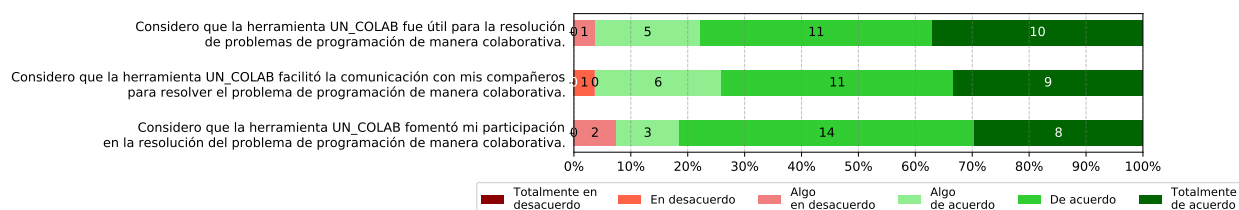


Figura 5-5: Respuestas obtenidas de la encuesta de percepciones sobre la herramienta UNColab

Gráficamente se evidencia que el 96 % de los estudiantes coincidió con algún nivel de acuerdo sobre la utilidad de la herramienta para resolver problemas de manera colaborativa. Así mismo, el 96 % de los participantes estuvieron de acuerdo con las valoraciones más altas de la escala Likert al establecer que la herramienta UNColab facilita la comunicación entre compañeros al momento de resolver un ejercicio de manera colaborativa. Finalmente, en cuanto a la utilidad de la herramienta para fomentar la participación en la resolución de problemas de programación de manera colaborativa, el 93 % de los estudiantes coincidieron con algún nivel de acuerdo respecto a la escala Likert.

Al igual que la evaluación de la colaboración vista en la Sección 5.2, los datos recolectados del instrumento de percepciones sobre UNColab contienen respuestas abiertas acerca de la opinión de los estudiantes sobre la herramienta una vez fue usada durante las dos intervenciones realizadas. Para examinar las respuestas dadas por los estudiantes se utilizó de igual modo la técnica de análisis de contenido, buscando estudiar, clasificar y codificar cada comentario obtenido.

Para el análisis de las respuestas obtenidas se realizó una agrupación de las mismas en diferentes indicadores de acuerdo al grado de afinidad de las ideas expresadas por el estudiante. Esto permitió definir diferentes categorías para cada grupo de respuestas debido a que varias de las respuestas podían estar asociadas a diferentes ideas. Las categorías resultantes permitieron reunir un conjunto de características que fueron asociadas a un tema general. A continuación, se presenta el análisis de las categorías encontradas de acuerdo al tema principal denominado: Herramienta de aprendizaje colaborativo para la resolución de problemas de programación UNColab.

Herramienta de aprendizaje colaborativo para la resolución de problemas de programación UNColab

El análisis de las respuestas obtenidas mediante la encuesta de percepciones permitió identificar diferentes categorías respecto a las valoraciones sobre la herramienta UNColab. Algunas de estas valoraciones fueron recolectadas a partir del análisis de contenido de la evaluación de la colaboración,

vista en la Sección 5.2, debido a que varios estudiantes aprovecharon estos instrumentos para dar una opinión acerca de la herramienta usada. La Figura 5-6 presenta 5 categorías identificadas en relación a las percepciones de los estudiantes sobre la herramienta UNColab.

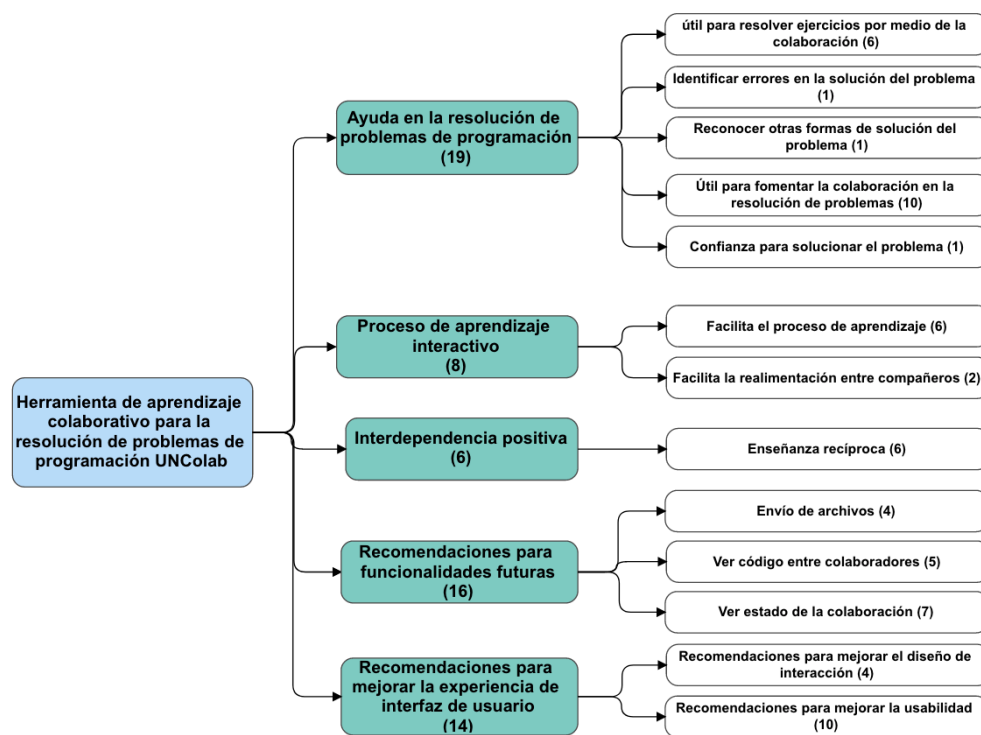


Figura 5-6: UNColab: resultado de análisis de contenido

La categoría **Ayuda en la resolución de problemas de programación** relaciona, por medio de 5 indicadores, las opiniones acerca de la utilidad de la herramienta como un elemento facilitador al momento de resolver un problema de programación de manera colaborativa. Dentro de estos indicadores, en 6 respuestas los estudiantes han valorado la herramienta como **útil para resolver ejercicios por medio de la colaboración**, resaltando la necesidad de incluirla dentro de los demás talleres o ejercicios de la clase. Lo anterior, se puede evidenciar en las siguientes respuestas: “...*amplia los métodos de solución que usa una persona porque facilita conocer diferentes formas para resolver un mismo problema.*” [sic], “...*En si, es una herramienta muy buena, por el hecho de que permite la interacción para ayudar a determinar una solución*” [sic]. Así mismo, se agrupó una respuesta en un indicador que reconoce que UNColab puede ayudar a **identificar errores en la solución de un problema** a través de interacción con un compañero; esta respuesta se refleja a continuación: “*Es una herramienta muy útil porque en primera instancia permite reconocer los errores propios con ayuda de una persona con percepción diferente...*”. Otro indicador identificado hace referencia al uso de la herramienta como un elemento que permite **reconocer otras formas de solución de un problema**, aquí se capturó una respuesta en donde se manifiesta la utilidad de UNColab en la ampliación de los métodos de solución que un estudiante puede tener al momento

de resolver un problema. El siguiente es un fragmento de la opinión analizada: “...*amplia los métodos de solución que usa una persona porque facilita conocer diferentes formas para resolver un mismo problema*” [sic]. En esta misma línea se analizaron 10 respuestas en las cuales se percibe que la herramienta puede ser **útil para fomentar la colaboración en la resolución de problemas**. Este es un importante hallazgo dentro del estudio ya que dentro de la teoría del aprendizaje colaborativo se enfatiza en que el estudiante reconozca la colaboración como parte de su proceso formativo e identifique mecanismos que permitan facilitar la colaboración; en este aspecto, los estudiantes expresan la facilidad que les brinda la herramienta y la técnica de “Programadores y Novatos” al momento de dar o recibir una colaboración. Se destacan las siguientes opiniones que apoyan esta idea: “*es muy útil y facilita la ayuda entre compañeros...*”, “*...atenúa la inhibición a la hora de dar o pedir ayuda*”, “*Es una buena herramienta cuando decides pedir ayuda a quien en realidad puede hacerlo, como es el caso de los programadores*”. Finalmente, se agrupó dentro de esta categoría el indicador **Confianza para solucionar el problema** el cual asoció una respuesta respecto al apoyo que da la herramienta cuando un estudiante tiene poca confianza en los conocimientos o requiere de una ayuda cuando ha intentado varias veces una solución sin éxito. Se relaciona a continuación la respuesta analizada: “*Es una propuesta excelente, ya que sin ella, al intentar solucionar un problema y no lograrlo se sentía una frustración horrible, en cambio con la herramienta de colaboración se siente uno más confiado y el aprendizaje es más rápido*”.

Proceso de aprendizaje interactivo es una categoría identificada en la que se recolectaron los comentarios referentes al reconocimiento de la herramienta como ayuda en el proceso de aprendizaje. Estos comentarios dieron lugar a la definición de dos indicadores: **Facilita el proceso de aprendizaje** y **Facilita la realimentación entre compañeros**. El primer indicador hace mención a 6 comentarios en el que los estudiantes perciben en el uso de la herramienta una forma más fácil de aprender programación al contar con elementos interactivos dentro del curso. Los siguientes representan algunas opiniones dadas: “*Es una herramienta que permite mejorar en el aprendizaje de la programación...*”, “*Gracias a esta herramienta, pienso que ambos pudimos aprender mutuamente, ayudándonos unos a otros, así haciendo el proceso de aprendizaje mas sencillo e interactivo*” [sic]. Respecto al segundo indicador, se incluyeron 2 respuestas concernientes a la facilidad que da la herramienta a los estudiantes al poder opinar y apoyar el trabajo de un compañero. Se percibe en estas respuestas una disposición en los estudiantes en no competir entre ellos, y en vez de ello colaborar; tal y como se refleja en el siguiente ejemplo de comentario: “*...la idea es buena ya que tiene una parte de retroalimentación con compañeros que hace que todos de alguna manera entendamos y podemos ayudarnos en temas*”.

La **Interdependencia positiva** se definió como una categoría que reúne 6 respuestas en donde se evidencia un reconocimiento de UNColab como elemento mediador para la **enseñanza recíproca**. Vale la pena resaltar que este concepto ya había sido identificado como un indicador dentro del análisis de contenido en la evaluación de la colaboración entre estudiantes. Al encontrarlo nuevamente dentro de este análisis, se suma a los elementos distintivos de la teoría del aprendizaje colaborativo que se han encontrado en este estudio. En estas respuestas los estudiantes han expresado haber aprendido de manera mutua al interactuar entre ellos a través de la herramienta UNColab. A continuación se

presentan algunos ejemplos textuales de las respuestas brindadas: “...ayuda a afianzar los conocimientos a la persona que explica, y ayuda a entender un problema o una duda para al que se le explica. En si, es una herramienta muy buena...” [sic], “...por medio de esta comunicación se afianzan conceptos para aquel que colabora, y le permite al que recibe la ayuda hacerse una idea del modo de pensamiento que hay que implementar en programación.”, “Gracias a esta herramienta, pienso que ambos pudimos aprender mutuamente...”.

Otras de las categorías identificadas en este análisis hacen mención a recomendaciones sobre diferentes funcionalidades que los estudiantes manifestaron para mejorar la herramienta. Los indicadores presentados a continuación permitirán a futuros desarrolladores realizar aplicaciones de aprendizaje colaborativo que vayan acorde a las necesidades de estudiantes de programación de computadores. Por otra parte, se refuerza la idea expresada por Pressman (2010) en el que establece el uso de prototipos de software como una manera eficaz para analizar las necesidades y definir las especificaciones de un software en un contexto específico.

De acuerdo con lo anterior, la categoría **Recomendaciones para funcionalidades futuras** captura opiniones acerca de ideas sobre nuevas características que los estudiantes manifestaron que podrían mejorar la herramienta. El primer indicador hace mención a 4 comentarios en los cuales se expresa la necesidad de incluir el **envío de archivos** dentro de la funcionalidad de chat de UNColab, dentro de estas recomendaciones está el envío de imágenes, audio y vídeo con el fin de mejorar la colaboración. Algunas de las respuestas son las siguientes: “sería muy útil alguna opción para mostrar el código o ver imágenes y así facilitar la asesoría.”, “...muchas veces las palabras no son suficientes para realizar una explicación y me gusta utilizar graficas para desglosar el problema y lograr comprenderlo mejor” [sic]. Así mismo, el segundo indicador establece 5 comentarios relacionados a la falta de una funcionalidad que permita **ver el código entre colaboradores** y de esta manera darle al colaborador una forma de realizar la ayuda más rápida, las siguientes respuestas soportan esta recomendación: “Debería haber una facilidad para mostrar partes de código de el novato al programador, para que éste comprenda mucho mejor el desarrollo que uno le da a las cosas” [sic], “La Herramienta me parece buena, pero sería increíble si se pudiera mostrar código también...” [sic]. Por último, los estudiantes sugieren una opción de **ver el estado de la colaboración**, esta sugerencia reunió 7 respuestas concernientes a la posibilidad de que en los diferentes roles se notifique si existe una colaboración en curso y de esta manera ampliar las ayudas dentro de la clase. Las siguientes corresponden a algunas de las respuestas obtenidas para este último indicador: “Inicialmente, seria bueno que cada novato pudiese poner un estado en el que defina si ya fue ayudado o no, esto debido a que pues un programador le puede escribir y pues no serviría de nada porque el usuario ya no necesitaría ayuda...” [sic], “...podria tener opciones como escoger si un novato ya fue ayudado por un programador...” [sic].

Finalmente, la categoría **Recomendaciones para mejorar la experiencia de interfaz de usuario** relaciona dos indicadores respecto a los aspectos que se pueden mejorar para que la herramienta sea más fácil de usar y tenga una mejor interacción. Como primer indicador se definen las **recomendaciones para mejorar el diseño de interacción**, estas recomendaciones se encuentran direccionadas a la eva-

luación de la herramienta al momento de dar respuestas a las acciones que los estudiantes esperan que realice. Algunas de las valoraciones identificadas se pueden ver en los siguientes comentarios: “Es muy buena y llamativa, solo me parece mal que aunque los programadores no esten en linea se pueda escribirles...” [sic], “...que el sistema de notificación sea un poco mas explicito me refiero a un numero o símbolo que indique si tiene mensajes recibidos sin leer” [sic]. El segundo indicador reunió **recomendaciones para mejorar la usabilidad**, dentro de éstas se encuentran percepciones de estudiantes que identificaron en la herramienta algunos elementos que no les permitieron usar la aplicación de una manera más fácil y sencilla. La mayoría de estudiantes coincidieron con la sugerencia expresada en el siguiente ejemplo de comentario: “...Se podria revisar el hecho de retomar los chat de apoyo desde el ultimo punto...” [sic].

6 Discusión de resultados

El desarrollo de la herramienta UNColab y su posterior evaluación en un curso de programación de computadores permitieron responder a la pregunta de investigación establecida: ¿Cuáles son los efectos en la resolución de problemas de programación de computadores ocasionados por el uso de una herramienta computacional colaborativa que apoye la evaluación formativa entre pares? Mediante el análisis de los datos obtenidos por la herramienta y las valoraciones de los estudiantes, se da a conocer una experiencia adicional, en relación con las propuestas vistas en la literatura, sobre el uso de herramientas CSCL en programación de computadores. Adicional a ello, se suple la necesidad de poder contar en los cursos de programación con elementos que permitan involucrar más a los estudiantes dentro de actividades de aprendizaje colaborativo que permitan apoyar el proceso de realimentación de tareas.

Los datos cuantitativos obtenidos mediante el uso de la herramienta colaborativa desarrollada y su posterior valoración por parte de estudiantes de un curso de programación, a través de la encuesta de percepciones sobre la herramienta, evidenciaron diferentes efectos positivos en la resolución de problemas de programación de computadores. Los siguientes aspectos presentan detalles de los resultados obtenidos.

En primer lugar, en las dos intervenciones realizadas se evidenció una mejora en algunas propuestas de solución de estudiantes que presentaban dificultades en la resolución de los problemas planteados. En este aspecto, UNColab registró al menos un cambio de rol de estudiantes clasificados como “Novatos” al rol de “Programador” luego de recibir la colaboración.

En segundo lugar, la cantidad de grupos formados y los mensajes intercambiados por parte de los estudiantes demostraron que cada participante dentro del curso logró establecer comunicación con un compañero y, en algunos casos, mantener una comunicación constante con un colaborador al momento de resolver el ejercicio. En este punto, es interesante mostrar cómo la herramienta permitió promover la participación de los estudiantes clasificados como “Programadores” en aras de ayudar a sus compañeros con dificultades en la resolución de los problemas de programación propuestos.

En tercer lugar, los datos cuantitativos recolectados en la encuesta de percepciones sobre UNColab permitieron conocer algunas apreciaciones sobre el uso de la herramienta en la resolución de problemas de programación. Los resultados señalaron que la mayoría de estudiantes coincidieron en que la herramienta es útil para resolver problemas de programación de manera colaborativa. Asimismo,

consideraron que es un recurso que permite facilitar la comunicación entre compañeros y fomentar la participación. Estos resultados permitieron verificar elementos encontrados en la literatura como beneficios de usar herramientas tecnológicas en actividades de aprendizaje colaborativo. Un ejemplo de ello, es lo expuesto por Vinagre Laranjeira (2009), cuando menciona que el uso de tecnología en actividades colaborativas permite estimular la comunicación interpersonal y facilitar la resolución de problemas.

Por otra parte, los resultados cuantitativos relacionados con las valoraciones sobre la colaboración entre estudiantes, en sus diferentes roles, puso en evidencia otros beneficios del uso de la colaboración en la resolución de ejercicios de programación de computadores. Por un lado, los estudiantes indicaron un alto nivel de acuerdo en que la colaboración en programación de computadores es útil cuando se quiere resolver dudas e identificar errores. En el estudio realizado por Shambhavi (2017), en el que se evaluó la efectividad de una estrategia de aprendizaje colaborativo para enseñar programación, se coincide con los resultados presentados en este estudio en estos aspectos. Los estudiantes ven en los compañeros una alternativa de apoyo adicional a la ayuda y orientación ofrecida por el docente a cargo. Por otro lado, se observó en los datos recolectados que los participantes del estudio ven en la colaboración una forma de ganar confianza para proponer soluciones a los problemas propuestos. Este hallazgo se vincula al estudio realizado por Altebarmakian and Alterman (2017) en el que se menciona que los alumnos que carecen de confianza pueden participar menos en una actividad. En este sentido, el uso de herramientas basadas en guiones de aprendizaje colaborativo puede promover la participación de los estudiantes durante el proceso de aprendizaje de la programación de computadores.

Los datos cualitativos obtenidos mediante la encuesta de percepciones sobre la herramienta UNColab y los instrumentos de recolección de datos relacionados con las valoraciones de la colaboración establecida entre estudiantes mostraron efectos favorables en el proceso de resolución de problemas de programación de computadores.

En las opiniones recolectadas, los estudiantes le atribuyeron a la colaboración de un compañero una forma de poder ajustar una solución al problema planteado, además de conocer otras maneras de resolver el mismo. Autores como Virvou and Sidiropoulos (2012) afirman que el aprendizaje colaborativo puede alentar a los estudiantes con menores aptitudes a aumentar las posibilidades de resolución de un problema mediante la indagación de diferentes formas de soluciones. En esta dirección, se puede decir que las técnicas de aprendizaje colaborativo, como por ejemplo, “Programadores” y “Novatos” pueden apoyar la mejora de una propuesta de solución de un problema, ya que le permite a los estudiantes con dificultades interactuar con compañeros que les comparten su experiencia y forma de solucionar un problema de programación.

Comprender el planteamiento de un problema de programación y la sintaxis básica de un programa son algunas de las dificultades identificadas en la enseñanza-aprendizaje de la programación de

computadores, según el estudio realizado por Mohammed et al. (2017a). Durante el análisis de los datos recolectados en esta tesis, se lograron reconocer dos indicadores relacionados con estas dificultades. Por un lado, los estudiantes expresaron en los comentarios haber comprendido el contexto del ejercicio a resolver gracias a la colaboración recibida durante la actividad colaborativa realizada en las dos intervenciones. De igual modo, otras opiniones estudiadas relacionaban la colaboración como un apoyo en la identificación de errores de código fuente. El reconocimiento de estos indicadores permitieron establecer una conexión con los resultados expuestos en (Shui Ng, 2012; Giordano and Maiorana, 2014; Vosinakis et al., 2014), en donde mencionan que el desempeño de un curso de programación puede mejorar significativamente si entre los participantes de un curso se colaboran para comprender los ejercicios propuestos y existe un apoyo en el reconocimiento de errores de código básicos.

Adicionalmente, el análisis de los datos cualitativos acerca de las valoraciones de la colaboración permitieron también identificar que los estudiantes ven en la actividad colaborativa una oportunidad de recibir y dar apoyo cuando se tienen dudas o se desea orientar en el desarrollo de un problema. Para los participantes fue interesante y llamativo el poder tener un rol dentro de la actividad, ya que en comparación con otros estudios, como por ejemplo (Tsompanoudi and Satratzemi, 2014), la colaboración entre estudiantes no se realizó de manera libre, en vez de ello se utilizó la técnica “Programadores y Novatos” para formar grupos de colaboración. Esta técnica implementada en la herramienta jugó un papel diferenciador, debido a que muchos de los estudiantes manifestaron la utilidad de saber quién puede dar u ofrecer una ayuda en un momento determinado. En consecuencia, estas percepciones hacen destacar la importancia de diseñar aplicaciones basadas en guiones CSCL que permitan realizar una participación y colaboración más productiva por parte de los estudiantes. Esta idea es apoyada en la revisión literaria sobre herramientas CSCL propuesta en Alharbi et al. (2015), en el que se menciona que en este tipo de herramientas no es suficiente tener un sistema de comunicación entre estudiantes si el trabajo colaborativo no está estructurado.

En cuanto a los resultados cualitativos relacionados con el uso de la herramienta UNColab, se dieron a conocer percepciones positivas respecto a la utilidad de esta herramienta en la resolución de ejercicios de programación de manera colaborativa, como también en facilitar la comunicación entre estudiantes y fomentar su participación. Como aspectos principales, dentro de estos resultados se evidenció que los estudiantes le atribuyen a la herramienta una manera de apoyar la identificación de errores y el conocer otras formas de solucionar un ejercicio. La identificación de estos aspectos puede conducir al uso de herramientas de aprendizaje colaborativo como un apoyo dentro de los cursos de programación que usen herramientas de calificación automática de código, como por ejemplo UNCode. El estudio presentado por Lancaster et al. (2019) reflexiona sobre la carencia de elementos de juicio humano dentro de la automatización al momento de evaluar tareas de programación debido al poco alcance que se tiene en la realimentación de las actividades. En este estudio se ha evidenciado, según las percepciones analizadas, que cuando un estudiante presenta dificultades en el desarrollo de un problema, la evaluación sumativa de una herramienta de calificación automática no le es su-

ficiente y requiere de una colaboración de un compañero para conocer sus errores y aprender de ellos.

Dentro de los resultados de este estudio se destaca que los estudiantes percibieron en UNColab una herramienta que sirvió como un espacio en el que ambos roles (*Programador* y *Novato*) pudieron aprender de manera mutua a través de la interacción realizada. De igual modo, también le atribuyeron a la herramienta hacer el proceso de aprendizaje más interactivo y facilitar la realimentación entre pares. Estos resultados coinciden con las ideas expresadas por Vinagre Laranjeira (2009), quien expone algunas características para asegurar el éxito de un ejercicio de aprendizaje colaborativo, incluyendo: la contribución individual, la interdependencia positiva (enseñanza recíproca) y la interacción entre estudiantes. No obstante, en la característica de interdependencia positiva, el estudio presentado por Ashenafi (2017) manifiesta que se pueden presentar dificultades cuando se tienen actividades entre pares debido al poco criterio que podría tener un programador “novato” respecto a lo que le aportaría a un compañero con más habilidades. A pesar de ello, en este estudio de caso los estudiantes con rol de “Programador” expresaron haber aprendido algo al colaborar con los compañeros. Este hallazgo concuerda con las ideas expresadas en el estudio realizado por Shadiev et al. (2013), en que se manifiesta la necesidad de implementar la enseñanza recíproca en las clases de programación, ya que le otorga a estudiantes novatos la posibilidad de resolver un problema cuando se presentan dificultades y, adicionalmente, ayuda a estudiantes programadores a afianzar sus conocimientos.

Asimismo, un hallazgo importante con respecto a las percepciones de los estudiantes acerca de UNColab se refiere a las recomendaciones de nuevas funcionalidades futuras y mejoras en la experiencia de la interfaz de usuario. Esto permitirá a futuros desarrolladores e investigadores implementar herramientas que se adapten a actividades de aprendizaje colaborativo en los cursos introductorios de programación de computadores. Gracias a las sugerencias recibidas por parte de los participantes de este estudio se pueden obtener nuevos requerimientos que ayuden a ajustar mejor las aplicaciones de tipo CSCL encaminadas en apoyar el aprendizaje de la programación, como también poder mejorar las funcionalidades de UNColab. Esto último permite apoyar la idea expuesta en Pressman (2010) la cual establece que el uso de prototipos es el mejor mecanismo para identificar necesidades del software y así poder desarrollar herramientas más robustas. Dentro de las mejoras sugeridas por lo estudiantes se destacan: opciones de envío de archivos, como por ejemplo audio y vídeo; la revisión de código, buscando que entre colaboradores se pueda ver el código fuente; un manejo de estados, sugerido especialmente para que el rol de “Programador” pueda ver si un compañero ya recibió colaboración.

Los resultados mostrados en este estudio aportan evidencias empíricas al estudio de los efectos de uso de herramientas CSCL en la resolución de ejercicios de programación de computadores. En el caso estudiado, se evidenció que UNColab tuvo una buena aceptación por parte de los estudiantes en las intervenciones realizadas en el curso de programación. Tanto los datos cuantitativos como cualitativos mostraron que una herramienta de este tipo puede facilitar el aprendizaje de la programación de computadores en los cursos introductorios al permitirle a los estudiantes tener un medio para recibir una orientación y apoyo por parte de los demás compañeros; adicionalmente, fomentando la partici-

pación, confianza y compañerismo entre los participantes.

Sumado a lo anterior, los resultados se constituyen como evidencias empíricas que dan soporte a futuros desarrollos e investigaciones. La implementación del prototipo de software con la técnica de aprendizaje colaborativo “Programadores” y “Novatos” aporta a la comprensión de los beneficios de esta técnica y que requiere aún más estudios en el contexto de los cursos de programación. La propuesta de la técnica mencionada, junto a otras consultadas en la literatura, permitió proponer una configuración de guion CSCL que facilitó la dinámica propuesta en el curso de programación de computadores tomado como estudio de caso. Este guion podría implementarse en cursos cuyos objetivos sean mejorar la comunicación entre estudiantes, la participación y las propuestas de solución de estudiantes con dificultades en la resolución de problemas. Asimismo, los resultados permitieron dar a conocer elementos que permitirán desarrollar futuras aplicaciones colaborativas ajustadas a la enseñanza de la programación de computadores.

Esta tesis de maestría aporta una experiencia respecto al uso de herramientas CSCL en la resolución de problemas en los cursos introductorios de programación de computadores. Tanto la herramienta como los efectos reportados de la misma servirán de base para otros estudios similares. Adicionalmente, durante el planteamiento del estudio se aportó un análisis comparativo de diferentes herramientas CSCL que usan medios, técnicas e instrumentos de evaluación formativa para apoyar la enseñanza de la programación de computadores. Cabe mencionar que, durante el desarrollo de este proyecto, la pandemia del COVID-19 obligó a muchas instituciones educativas a ofrecer clases de manera remota o “asistidas por computador”. En su momento se cuestionó si era posible poner a punto la herramienta desarrollada debido a que en un entorno virtual se dificultaba tener un control directo sobre la participación de los estudiantes, o por lo menos, no se había pensado en ello en la metodología. No obstante, se logró organizar junto al docente del curso las dos intervenciones propuestas que permitieron realizar la actividad de aprendizaje colaborativo sin ningún contratiempo. A pesar de no poder controlar la participación y tampoco el uso de otras herramientas convencionales que los estudiantes frecuentan para establecer comunicación, los datos mostrados en este documento evidenciaron que la herramienta fue usada por los participantes y sirvió de apoyo a los estudiantes en el proceso de resolución de ejercicios de manera colaborativa.

Finalmente, es necesario mencionar algunas limitaciones de este trabajo. Para empezar, los hallazgos obtenidos en este estudio no se pueden generalizar para otros cursos de programación debido a que se hizo énfasis en examinar un caso en particular. Posteriormente, este estudio no contempló temáticas como la *recursividad*, *arreglos* y *programación orientada a objetos* —que son otros de los temas en los cuales se presentan dificultades al momento de aprender a programar— ni se hizo énfasis en determinar en cuáles de los temas trabajados (ciclos, funciones y matrices) se favorece más en su aprendizaje con la colaboración. Lo anterior debido a que el estudio se enfocó en analizar la herramienta CSCL propuesta como un medio de comunicación entre estudiantes en el cual se pudiese abordar cualquier temática. Por último, el prototipo UNColab tuvo un solo despliegue de prueba, lo que implica que

debe someterse a más demostraciones con diferentes tamaños de grupos que permitan validar sus funcionalidades y al mismo tiempo proponer mejoras que permitan robustecer la herramienta.

7 Conclusiones y trabajos futuros

7.1. Conclusiones

Este trabajo presentó la herramienta computacional UNColab, que permitió apoyar la resolución de problemas de programación a través de la colaboración entre estudiantes. Como se mostró en la Sección 3.1 del Capítulo 3, esta herramienta se diseñó conforme a un guion CSCL que permitió estructurar y organizar un conjunto de técnicas de aprendizaje colaborativo que facilitaron la realización de una dinámica de clase. A partir de esto, se logró implementar un prototipo de la herramienta diseñada mediante la aplicación de la metodología de desarrollo orientada a prototipos, como se presentó en la Sección 3.2 del Capítulo 3. Buscando validar la herramienta y analizar los efectos de su uso en la resolución de problemas de programación a través de la colaboración, en la Sección 4.2 del Capítulo 4 se expuso el desarrollo de un estudio de caso en el que participaron 27 estudiantes de la Facultad de Ingeniería de la Universidad Nacional de Colombia inscritos en la asignatura Programación de Computadores. Dentro de este estudio de caso se propusieron dos intervenciones en el curso que permitieron a los estudiantes interactuar con la herramienta a partir de la definición de dos problemas de programación de computadores. Al finalizar cada intervención la herramienta le sugería a cada estudiante, por medio de un instrumento de recolección de datos, valorar la colaboración establecida de acuerdo al rol que desempeñaba. Así mismo, en la segunda intervención se aplicó la encuesta de percepciones sobre el uso de la herramienta. De esta manera, las intervenciones realizadas en el curso permitieron recolectar datos cuantitativos y cualitativos relacionados con las percepciones de la actividad de aprendizaje colaborativo y la herramienta UNColab. Estos datos indicaron efectos positivos en la resolución de problemas de programación, dentro de ellos se destacan: mejoras en las propuestas de solución de los estudiantes con dificultades, fomento de la participación y la comunicación, facilidad en la identificación de errores y la comprensión del planteamiento de un problema.

El análisis de los datos cuantitativos obtenidos a partir de la información que UNColab almacenó durante las dos intervenciones permite concluir que el uso de la herramienta UNColab favoreció la enseñanza-aprendizaje de la programación de computadores en el curso tomado como estudio de caso. Los datos cuantitativos recolectados evidenciaron algunos cambios de rol e interacciones de estudiantes que presentaban dificultades al momento de resolver un ejercicio de programación. De igual modo, los datos cualitativos obtenidos en las evaluaciones de la colaboración mostraron valoraciones positivas respecto al uso de estas dinámicas dentro de las herramientas trabajadas en clase. Lo anterior se reflejó en las percepciones relacionadas con el poder aprender de una manera más interactiva, el poder solicitar ayuda a quién en realidad la puede dar y en reconocer un aprendizaje mutuo cuando

se colabora. Adicionalmente, se le atribuye a la herramienta un medio por el cual se puede tener más confianza al momento de resolver un problema. Los estudiantes demostraron un gusto e interés por la herramienta desarrollada y propusieron diferentes mejoras y sugerencias esperando que éstas se implementen a futuro dentro de las herramientas que se trabajan en el curso.

Asimismo, el prototipo desarrollado UNColab es un aporte al área de investigación sobre el uso de herramientas CSCL en los cursos introductorios de programación. A pesar de haber examinado en esta investigación un caso particular, las valoraciones dadas por los estudiantes sobre UNColab podrían usarse como requerimientos a tener en cuenta en aplicaciones que permitan apoyar la enseñanza-aprendizaje de la programación de computadores de manera colaborativa.

Por otro lado, es importante señalar que la revisión de literatura realizada al inicio de este trabajo permitió recopilar una serie de teorías y experiencias sobre el uso de herramientas CSCL y técnicas de aprendizaje colaborativo en programación de computadores. Esto facilitó la estructuración de un guion CSCL que permitió realizar la actividad colaborativa en el curso de programación de computadores. Respecto a esta actividad colaborativa, en este estudio se lograron identificar resultados comunes con los estudios consultados, como por ejemplo indicadores de enseñanza recíproca, confianza en la resolución de ejercicios, apoyo en la resolución de dudas y la orientación entre compañeros. Por otro lado, respecto al uso de herramientas CSCL en los cursos de programación, los hallazgos de este estudio coinciden en la utilidad de usar guiones de aprendizaje colaborativo para realizar interacciones más productivas entre estudiantes y así apoyar su proceso formativo.

Por último cabe mencionar que, a pesar de haber elaborado esta tesis durante la pandemia del COVID-19, se considera que no tuvo una influencia sobre el desarrollo y la validación de la herramienta UNColab. Por un lado, las herramientas CSCL están enfocadas en facilitar la comunicación e interacción entre estudiantes mediante el uso de tecnología (Vinagre Laranjeira, 2009); desde el inicio del proyecto se contempló que la herramienta debía usar una conexión a Internet para que esta interacción fuese posible. Por otro lado, los datos recolectados en relación con el uso de UNColab evidenciaron que los estudiantes interactuaron con la herramienta y ésta les permitió realizar un ejercicio de aprendizaje colaborativo que apoyó la resolución de los ejercicios de programación propuestos. Finalmente, en la literatura consultada al inicio del proyecto, no se mencionan eventos adversos como lo es una pandemia como una limitación o impedimento de realización de estudios que implementen herramientas CSCL en cursos de programación de computadores.

7.2. Productos académicos

Como resultados de esta investigación se presentan los siguientes productos académicos:

- Software: UNColab¹ – prototipo de una herramienta computacional para el aprendizaje cola-

¹https://github.com/jhonmendex/un_colab.git

borativo de programación, dirigido a cursos introductorios de programación de computadores que usen como apoyo la plataforma educativa UNCode. 2021.

- Artículo en conferencia internacional: *J. Mendez-Lara, J.J. Ramírez-Echeverry, F. Restrepo-Calle (2021) UNCOLAB: A COMPUTATIONAL TOOL FOR COLLABORATIVE LEARNING OF COMPUTER PROGRAMMING, EDULEARN21 Proceedings, pp. 9542-9553. (Mendez-Lara et al., 2021)*. Este artículo contiene los hallazgos finales de esta investigación.

7.3. Trabajos futuros

El aprendizaje colaborativo asistido por computador en programación de computadores es un enfoque educativo reciente en comparación con otras formas tradicionales de enseñanza, por lo tanto, se requieren de más reportes de experiencias y estudios de caso con varios tamaños de muestra que permitan identificar más requerimientos y valoraciones sobre el uso de CSCL en cursos de programación. Lo anterior buscando obtener herramientas colaborativas que cada vez más se ajusten a las necesidades de la enseñanza-aprendizaje de la programación de computadores.

En términos de los resultados obtenidos, se evidenció, conforme a la cantidad de interacciones analizadas, que los estudiantes usaron la herramienta desarrollada para realizar colaboración al momento de resolver un ejercicio de programación; sin embargo, en este estudio no se tuvo en cuenta la influencia directa de estas interacciones en la resolución de los ejercicios propuestos por parte de los estudiantes con rol de “Novato”. En este sentido, se propone como trabajo futuro realizar un análisis de la calidad de las interacciones realizadas durante las etapas de colaboración. La mayoría de aplicaciones basadas en CSCL contienen elementos de comunicación como por ejemplo foros, chats y vídeos. Creemos que la información almacenada en estos medios de comunicación puede servir para evaluar el tipo de ayuda que se establece cuando interactúan los estudiantes y su impacto en el éxito de la resolución de un problema. Conocer estos tipos de colaboración permitiría saber si es una ayuda enfocada a dar pistas, orientaciones, consejos, o simplemente, limitada a enviar las soluciones de los ejercicios propuestos. Encontrar cuáles son las formas más efectivas de colaboración impulsaría a los docentes de programación a usar otro tipo de estrategias para enseñar programación.

Asimismo, algunos estudiantes sugirieron mejoras y funcionalidades adicionales a UNColab, lo que motivará el desarrollo de nuevos prototipos de software que incluyan funciones acordes a las necesidades de estudiantes de programación de computadores. Estas sugerencias también permitirán realizar estudios futuros sobre los efectos de usar la herramienta UNColab una vez se desarrollen e implementen las funcionalidades y mejoras recomendadas.

Más adelante, podría resultar interesante estudiar los efectos de usar herramientas CSCL en casos en los cuales se involucren cursos de programación avanzados. Si bien en este estudio se evidenciaron buenos resultados en el curso introductorio estudiado, puede que estos efectos se repitan para otros

niveles de enseñanza usando la misma configuración de estudio de caso.

En este estudio se ha utilizado el marco de trabajo propuesto por Hernández-leo and Asensio-pérez (2019) para la construcción de guiones CSCL. El guion usado en este trabajo facilitó la estructura de una actividad colaborativa en un curso de programación. Durante la configuración de este guion CSCL se consultaron y propusieron diferentes TACs para organizar la colaboración de los estudiantes. Esta estructura de guion puede someterse a diferentes estudios en donde se pueda validar su uso en diferentes cursos introductorios de programación de computadores. Adicionalmente, puede resultar interesante conocer otras formas de configuración de técnicas que permitan usarse para apoyar la resolución de ejercicios de programación de manera colaborativa.

Finalmente, se pueden obtener más evidencias sobre la utilidad de la herramienta UNColab en la resolución de problemas de programación, al someterla a estudios experimentales o cuasi-experimentales que usen grupos de control y grupos experimentales, a partir de los cuales sea posible contrastar aspectos como el rendimiento académico, la participación o la calidad de las soluciones entregadas por los estudiantes.

Bibliografía

- Al-Khalifa, A. K. and Devlin, M. (2020). Evaluating a peer assessment approach in introductory programming courses. *ACM International Conference Proceeding Series*, pages 51–58.
- Ala-Mutka, K. M. (2005). A Survey of Automated Assessment Approaches for Programming Assignments. *Computer Science Education*, 15(2):83–102.
- Alharbi, N. M., Athauda, R. I., and Chiong, R. (2015). A survey of CSCL script tools that support designing collaborative scenarios. *2014 International Conference on Web and Open Access to Learning, ICWOAL 2014*, pages 1–8.
- Altebarmakian, M. and Alterman, R. (2017). A Study of Engagement and Collaborative Learning in a Virtual Environment. *IEEE Transactions on Learning Technologies*.
- Arora, R., Goel, S., and Mittal, R. K. (2017). Supporting collaborative software development in academic learning environment: A collaborative pair and quadruple programming based approach. In *2017 Tenth International Conference on Contemporary Computing (IC3)*, pages 1–7.
- Ashenafi, M. M. (2017). Peer-assessment in higher education – twenty-first century practices, challenges and the way forward. *Assessment and Evaluation in Higher Education*, 42(2):226–251.
- Barkley, E. F., Cross, K. P., and Major, C. H. (2007). *Técnicas de aprendizaje colaborativo : manual para el profesorado universitario*. Ministerio de Educación y Ciencia.
- Barkley, E. F., Major, C. H., and Cross, K. P. (2014). *Collaborative learning techniques: a handbook for college faculty*. Wiley.
- Black, P. and Wiliam, D. (2009). Developing the theory of formative assessment. *Educational Assessment, Evaluation and Accountability*, 21(1):5–31.
- Bretones Román, A. (2008). Participación del alumnado de Educación Superior en su evaluación Participation of the High Education students in their assessment. *Revista de Educación*, pages 181–202.
- Bruffee, K. (1978). The brooklyn plan: Attaining intellectual growth through peer-group tutoring. *Liberal Education*, 64(4):447–468.

- Bruffee, K. A. (1983). Teaching writing through collaboration. *New Directions for Teaching and Learning*, 1983(14):23–29.
- Bryman, A. (2015). *Social Research Methods (4th Edition)* .
- Burch, C. (2009). Jigsaw, A Programming Environment for Java in CS1. *Journal of Computing Sciences in Colleges*, 24(5):37–43.
- Caballe, S., Britch, D., Barolli, L., and Xhafa, F. (2014). A Methodological Approach to Provide Effective Web-Based Training by Using Collaborative Learning and Social Networks. *2014 Eighth International Conference on Complex, Intelligent and Software Intensive Systems*, pages 64–71.
- Caballé, S., Mora, N., Dunwell, I., and Gañan, D. (2011). CC-LO: A new type of learning object for the virtualization of live collaborative sessions. *Proceedings - 3rd IEEE International Conference on Intelligent Networking and Collaborative Systems, INCoS 2011*, pages 818–823.
- Cabrera, I., Villalon, J., and Chavez, J. (2017). Blending Communities and Team-Based Learning in a Programming Course. *IEEE Transactions on Education*, 60(4):288–295.
- Carrió Pastor, M. L. (2007). Ventajas del uso de la tecnología en el aprendizaje colaborativo. *Revista Iberoamericana de Educación*, 41:10.
- Castro-Bleda, M. J., Bernabucci, I., Goffredo, M., Hassinen, E., Hollas, B., Lleida, E., Ortega, A., Pastor-Pellicer, J., Salopuro, A., Søndergaard, H., Mulder, R. A., Turcotte, S., Basher, M., Burd, L., Munro, M., Baghaei, N., Ioannou, A., Mama, M., Demetriou, S., Georgouli, K., Sgouropoulou, C., Skalkidis, I., Tsetsekas, C., Van Leeuwen, A., Janssen, J., Erkens, G., Brekelmans, M., Raes, A., Schellens, T., De Wever, B., Kollar, I., Wecker, C., Langer, S., Fischer, F., Tissenbaum, M., Slotta, J. D., Peters, V. L., Songer, N. B., Dillenbourg, P., Villasclaras-Fernández, E., Hernández-Leo, D., Asensio-Pérez, J. I., and Dimitriadis, Y. (2013). Computer-Supported Collaborative Inquiry on Buoyancy: A Discourse Analysis Supporting the "Pieces" Position on Conceptual Change. In *Computer-Supported Collaborative Learning Conference, CSCL*, volume 2, pages 221–226.
- Cisar, S. M., Pinter, R., and Cisar, Petar and Radosav, Dragica (2013). Teaching computer science in a web-based environment. *SISY 2013 - IEEE 11th International Symposium on Intelligent Systems and Informatics, Proceedings*, pages 415–418.
- Clark, R. C., editor (2011). *eLearning and the Science of Instruction*. John Wiley and Sons, Ltd, New Jersey.
- De Faria, E. S. J., Adan-Coello, J. M., and Yamanaka, K. (2006). Forming groups for collaborative learning in introductory computer programming courses based on students' programming styles: An empirical study. In *Proceedings - Frontiers in Education Conference, FIE*, pages 6–11. IEEE.

- Debdi, O., Paredes-Velasco, M., and Velázquez-Iturbide, J. (2015). GreedExCol, A CSCL tool for experimenting with greedy algorithms. *Computer Applications in Engineering Education*, 23(5):790–804.
- Deitrick, E., Shapiro, R. B., and Gravel, B. (2016). How do we assess equity in programming pairs? *Proceedings of International Conference of the Learning Sciences, ICLS*, 1:370–377.
- Dittmar, A. and Forbrig, P. (2018). A Case Study on Supporting Teachers' Collective Reflection in Higher Education: Reflection on Modeling Sessions in Software Engineering Education.
- Dorneles, R. V., Picinin Jr., D., Adami, A. G., Jin, H., Rahman, S. M., Juell, P. L., Mendonça, A., De Oliveira, C., Guerrero, D., Costa, E., Huelsman, L. P., Peng, W., What, I., Narasimhamurthy, U., Shawkani, K. A., Lu, L., Li, L. L., Song, W., and Xu, Y. (2010). The teaching method about programming design language. *2010 5th International Conference on Computer Science & Education, 2006(Iceit)*:170–172.
- Felemban, S., Gardner, M., Callaghan, V., Vasileva, T. K., Tchoumatchenko, V. P., and Stefanova, S. A. (2016). An event detection approach for identifying learning evidence in collaborative virtual environments. *2016 8th Computer Science and Electronic Engineering Conference, CEEC 2016 - Conference Proceedings*, pages 3–6.
- Flórez Ochoa, R. (2005). *Pedagogía del conocimiento*. MCGRAW-HILL, Bogotá, 2 edition.
- Frank-Bolton, P. and Simha, R. (2018). Docendo Discimus: Students Learn by Teaching Peers Through Video. *Proceedings of the 49th ACM Technical Symposium on Computer Science Education*, pages 473–478.
- Fukuma, Y., Tsutsui, K., Takada, H., and Piumarta, I. (2017). A scratch-based collaborative learning system with a shared stage screen. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 10397 LNCS, pages 84–98.
- García, S. (2003). *Aprender cooperando: el aprendizaje cooperativo en el aula*. Investigación educativa. Dirección General de Ordenación, Renovación y Centros.
- Gehring, E. F., Chinn, D. D., Pérez-Quñones, M. A., and Ardis, M. A. (2005). Using peer review in teaching computing. *ACM SIGCSE Bulletin*, 37(1):321.
- Giordano, D. and Maiorana, F. (2014). Use of cutting edge educational tools for an initial programming course. *IEEE Global Engineering Education Conference, EDUCON*, (April 2014):556–563.
- Hamm, M. J. (2014). *Wireframing Essentials*. Birmingham.
- Hamodi, C., López Pastor, V. M., and López Pastor, A. T. (2018). Medios, técnicas e instrumentos de evaluación formativa y compartida del aprendizaje en educación superior. *Perfiles Educativos*, 37(147):103–116.

- Hashim, S. and Harun, J. (2014). CSCL learning approach for students' knowledge construction process in School Based Assessment (SBA) environment: An observation. pages 139–144.
- Hernández-leo, D. and Asensio-pérez, J. I. (2019). Generating cscl scripts. 64.
- Hernandez-Leo, D., Villasclaras-Fernandez, E. D., Asensio-Perez, J. I., Dimitriadis, Y. A., and Retalis, S. (2006). Cscl scripting patterns: Hierarchical relationships and applicability. In *Sixth IEEE International Conference on Advanced Learning Technologies (ICALT'06)*, pages 388–392.
- Hernández Sampieri, R., Fernández Collado, C., and Baptista Lucio, P. (2006). *Metodología de la investigación*.
- Hidalgo, C. G., Bucheli Guerrero, V. A., Restrepo Calle, F., and González Osorio, F. A. (2021). Estrategia de enseñanza basada en la colaboración y la evaluación automática de código fuente en un curso de programación cs1. *Investigación e Innovación en Ingenierías*, 9(1):50–60.
- Hoffbeck, J. P., Dillon, H. E., Albright, R. J., Lu, W., and Doughty, T. A. (2016). Teaching programming in the context of solving engineering problems. *Proceedings - Frontiers in Education Conference, FIE, 2016-Novem*.
- Hwang, G. J., Liang, Z. Y., and Wang, H. Y. (2017). An Online Peer Assessment-Based Programming Approach to Improving Students' Programming Knowledge and Skills. *Proceedings - 5th International Conference on Educational Innovation through Technology, EITT 2016*, pages 81–85.
- Johnson, D. and Johnson, F. (2017). *Joining Together: Group Theory and Group Skills*. Pearson Education.
- Judele, R., Tsovaltzi, D., Puhl, T., and Weinberger, A. (2014). Collaborative learning in facebook: Adverse effects of individual preparation. *Proceedings of the Annual Hawaii International Conference on System Sciences*, pages 1616–1624.
- Knutas, A., Ikonen, J., and Porras, J. (2013). Communication patterns in collaborative software engineering courses. In *Proceedings of the 13th Koli Calling International Conference on Computing Education Research - Koli Calling '13*, pages 169–177, New York, New York, USA. ACM Press.
- Laakso, M.-J., Kaila, E., and Rajala, T. (2018). ViLLE – collaborative education tool: Designing and utilizing an exercise-based learning environment. *Education and Information Technologies*, pages 1–22.
- Lampe, C., Obar, J., Ozkaya, E., Zube, P., and Velasquez, A. (2012). Classroom wikipedia participation effects on future intentions to contribute. In *Proceedings of the ACM 2012 Conference on Computer Supported Cooperative Work, CSCW '12*, page 403–406, New York, NY, USA. Association for Computing Machinery.
- Lancaster, T., Robins, A. V., and Fincher, S. A. (2019). *Assessment and Plagiarism*.

- Liu, Y., Lin, F., and Wang, X. (2004). Using agents in web-based constructivist collaborative learning system. *Tsinghua Science and Technology*, 9(2):189–196.
- Lykke, M., Coto, M., Mora, S., Vandell, N., and Jantzen, C. (2014). Motivating programming students by problem based learning and lego robots. In *2014 IEEE Global Engineering Education Conference (EDUCON)*, pages 544–555.
- López Pastor, V. M., González Pascual, M., and Barba Martín, J. J. (2004). La participación del alumnado en la evaluación : la autoevaluación, la coevaluación y la evaluación compartida. *Tándem: didáctica de la educación física*.
- Maksimenkova, O. and Alexey, N. (2015). Blended learning in software engineering education: The application lifecycle management experience with computer-supported collaborative learning. In *Proceedings of 2015 International Conference on Interactive Collaborative Learning, ICL 2015*, pages 655–662. Institute of Electrical and Electronics Engineers Inc.
- Marcolino, A. S. and Barbosa, E. (2017). A survey on problems related to the teaching of programming in brazilian educational institutions. In *2017 IEEE Frontiers in Education Conference (FIE)*, pages 1–9.
- Mendez-Lara, J., Ramírez-Echeverry, J., and Restrepo-Calle, F. (2021). Uncolab: A computational tool for collaborative learning of computer programming. In *EDULEARN21 Proceedings, 13th International Conference on Education and New Learning Technologies*, pages 9542–9553. IATED.
- Michailidis, N. P., Nathanailidis, C., Papadopoulos, I., and Tsiatsos, T. (2013). Teachers’ support using interaction analysis and visualization tools in educational group blogging. *IEEE Global Engineering Education Conference, EDUCON*, pages 790–797.
- Mohammed, S. A., Elhaddad, M. E., and Mohammed, A. O. (2017a). Proposing a solution for the problem of teaching programming to novice students using soft systems methodology. *2017 International Conference on Engineering & MIS (ICEMIS)*, pages 1–6.
- Mohammed, S. A., Elhaddad, M. E., Mohammed, A. O., Hoffbeck, J. P., Dillon, H. E., Albright, R. J., Lu, W., and Doughty, T. A. (2017b). Proposing a solution for the problem of teaching programming to novice students using soft systems methodology. *2017 International Conference on Engineering & MIS (ICEMIS)*, (1):1–6.
- Orozco, G. L. (2014). *Docencia universitaria: sentidos, didácticas, sujetos y saberes*. Vol. 6. U. de La Salle, Bogotá.
- Pérez-Sanagustín, M., Ramirez-Gonzalez, G., Hernández-Leo, D., Muñoz-Organero, M., Santos, P., Blat, J., and Delgado Kloos, C. (2012). Discovering the campus together: A mobile and computer-based learning experience. *Journal of Network and Computer Applications*, 35(1):176–188.
- Piteira, M. and Costa, C. (2012). Computer programming and novice programmers. *ACM International Conference Proceeding Series*, pages 51–53.

- Pressman, R. S. (2002). *Ingeniería del software. Un enfoque práctico.*(5: edición).
- Pressman, R. S. (2010). *Ingeniería del software: Un enfoque práctico.* McGraw-Hill.
- Pullan, W., Drew, S., and Tucker, S. (2013). An integrated approach to teaching introductory programming. *2013 Second International Conference on E-Learning and E-Technologies in Education (ICEEE)*, pages 81–86.
- Ramírez-Donoso, L., Rojas-Riethmuller, J. S., Pérez-Sanagustín, M., and Neyem, A. (2017). Enhancing collaborative learning in higher education online courses through a mobile game app. In *2017 IEEE 21st International Conference on Computer Supported Cooperative Work in Design (CSCWD)*, pages 103–108.
- Restrepo Calle, F., Ramírez-Echeverry, J. J., and Gonzalez, F. A. (2018). Uncode: Interactive System for Learning and Automatic Evaluation of Computer Programming Skills. *EDULEARN18 Proceedings*, 1(July):6888–6898.
- Revelo-Sánchez, O., Collazos-Ordóñez, C. A., and Jiménez-Toledo, J. A. (2018). El trabajo colaborativo como estrategia didáctica para la enseñanza / aprendizaje de la programación : una revisión sistemática de literatura. *Tecnológicas*, 21(41):115–134.
- Sancho-Asensio, A., Sole, X., Montero, J., Navarro, J., Canaleta, X., and Vernet, D. (2014). Support tool for the formation of working groups in collaborative learning environments. *2014 9th Iberian Conference on Information Systems and Technologies (CISTI)*, pages 1–6.
- Serrano Cámara, L. M., Paredes Velasco, M., and Velázquez-Iturbide, J. Á. (2012). Evaluation of a Collaborative Instructional Framework for Programming Learning. *Proceedings of the 17th ACM Annual Conference on Innovation and Technology in Computer Science Education*, pages 162–167.
- Serrano-Cámara, L. M., Paredes-Velasco, M., Velázquez-Iturbide, J. A., Alcover, C.-M., and Castellanos, M. E. (2016). MoCAS: A Mobile Collaborative Tool for Learning Scope of Identifiers in Programming Courses. *International Journal of Engineering Education*, 32(2):969–981.
- Shabalina, O., Sadovnikova, N., and Kravets, A. (2013). Methodology of teaching software engineering: Game-based learning cycle. *Proceedings - 2013 IEEE 3rd Eastern European Regional Conference on the Engineering of Computer Based Systems, ECBS-EERC 2013*, pages 113–119.
- Shadiev, R., Hwang, W. Y., Yeh, S. C., Yang, S. J., Wang, J. L., Han, L., Huang, Y. M., and Liu, C. J. (2013). Applying unidirectional versus reciprocal teaching strategies in web-based environment and their effects on computer programming learning. *Proceedings - 2013 IEEE 13th International Conference on Advanced Learning Technologies, ICALT 2013*, pages 5–9.

- Shambhavi, B. R. (2017). Effective collaborative activities and active learning in engineering education: A case study. In Ramachandran M. Garg D., K. V. A., editor, *Proceedings - 2016 IEEE 4th International Conference on MOOCs, Innovation and Technology in Education, MITE 2016*, pages 137–139. Institute of Electrical and Electronics Engineers Inc.
- Shui Ng, W. (2012). The Impact of Peer Assessment and Feedback Strategy in Learning Computer Programming in Higher Education. *Issues in Informing Science and Information Technology*, 9:017–027.
- Silva, L., Mendes, A. J., and Gomes, A. (2020). Computer-supported collaborative learning in programming education: A systematic literature review. *IEEE Global Engineering Education Conference, EDUCON*, 2020-April:1086–1095.
- Soh, L.-K., Khandaker, N., and Jiang, H. (2008). I-MINDS: A multiagent system for intelligent computer-supported collaborative learning and classroom management. *International Journal of Artificial Intelligence in Education*, 18(2):119–151.
- Soh, L.-K., Khandaker, N., Liu, X., and Jiang, H. (2005). Computer-supported structured cooperative learning. In *Proc. Int. Conf. on Computers in Education 2005: "Towards Sustainable and Scalable Educational Innovations Informed by the Learning Sciences Sharing Research Results and Exemplary Innovations, ICCE*, pages 426–433.
- Soundarajan, N., Joshi, S., and Ramnath, R. (2015). Collaborative and Cooperative-Learning in Software Engineering Courses. volume 2, pages 319–322.
- Stahl, G., Koschmann, T., and Suthers, D. (2006). Aprendizaje Colaborativo Apoyado por Computador: Una perspectiva histórica. *Cambridge handbook of the learning sciences [Computer-supported collaborative learning: An historical perspective](C. Collazos Trans.)*, pages 409–426.
- Szedmina, L. and Pinter, R. (2010). Experiences from using Skype in language teaching. *SIISY 2010 - 8th IEEE International Symposium on Intelligent Systems and Informatics*, pages 449–452.
- Tirado, F., Bustos, A., Miranda, A., and Del Bosque, A. E. (2009). Inducing student interaction in a virtual environment. *ACM SIGCSE Bulletin*, 41(3):378.
- Triantafillou, E., Triantafillou, E., Pomportsis, A., and Georgiadou, E. (2002). AES-CS: Adaptive Educational System based on Cognitive Styles. In *2nd International Conference on Adaptive Hypermedia and Adaptive Web Based Systems. Workshop Proceedings on Adaptive Systems for Web-based Education*.
- Tseng, J. C., Hsu, S. Y., and Hwang, G.-J. (2009). A collaborative ubiquitous learning platform for computer science education. *ACM SIGCSE Bulletin*, 41(3):368.
- Tseng, K. H., Liu, C. C., and Liu, B. J. (2012). Scaffolded participatory and collaborative learning: Enhancing children reading with E-book readers. *Proceedings 2012 17th IEEE International Conference on Wireless, Mobile and Ubiquitous Technology in Education, WMUTE 2012*, pages 142–146.

- Tsompanoudi, D. and Satratzemi, M. (2014). A web-based authoring tool for scripting distributed pair programming. *Proceedings - IEEE 14th International Conference on Advanced Learning Technologies, ICALT 2014, (Section IV):259–263.*
- Tsompanoudi, D., Satratzemi, M., and Xinogalos, S. (2016). Evaluating the Effects of Scripted Distributed Pair Programming on Student Performance and Participation. *IEEE Transactions on Education, 59(1):24–31.*
- Villasclaras-Fernández, E. D., Hernández-Leo, D., Asensio-Pérez, J. I., and Dimitriadis, Y. (2009). Incorporating assessment in a pattern-based design process for CSCL scripts. *Computers in Human Behavior, 25(5):1028–1039.*
- Vinagre Laranjeira, M. (2009). *Teoría Y Práctica Del Aprendizaje Colaborativo Asistido Por Ordenador.* Editorial Síntesis.
- Virvou, M. and Sidiropoulos, S. (2012). Collaborative tools in learning a programming language. *e-Learning and e-Technologies in ...*, pages 162–165.
- Vosinakis, S., Koutsabasis, P., and Anastassakis, G. (2014). A Platform for Teaching Logic Programming Using Virtual Worlds. *2014 IEEE 14th International Conference on Advanced Learning Technologies*, pages 657–661.
- Yannibelli, V. and Amandi, A. (2012). A memetic algorithm for collaborative learning team formation in the context of software engineering courses. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 7547 LNCS, pages 92–103. Springer, Berlin, Heidelberg.
- Yongxing, W. (2008). Blended learning design for software engineering course design. In *Proceedings - International Conference on Computer Science and Software Engineering, CSSE 2008*, volume 5, pages 345–348.
- Zakiah, A. and Fauzan, M. N. (2016). Collaborative learning model of software engineering using github for informatics student. In *2016 4th International Conference on Cyber and IT Service Management*, pages 1–5.
- Zamfirache, V., Olteanu, A. C., and Tapus, N. (2013). Collaborative learning assistant for Android. *Proceedings - RoEduNet IEEE International Conference.*
- Zhao, X. and Okamoto, T. (2009). An email-based discussion learning system with ubiquitous device support. *Proceedings of 2009 4th International Conference on Computer Science and Education, ICCSE 2009*, pages 1420–1424.