Impacto de dos estrategias, cuentos y juegos interactivos, en Scratch Jr. en el desarrollo del pensamiento computacional, en los niños de educación inicial

Jeison Isidro Soler Correa

Universidad Pedagógica Nacional

Facultad De Educación

Departamento De Postgrados

Maestría En Tecnologías Aplicadas A La Educación

Bogotá

2025

Impacto de dos estrategias, cuentos y juegos interactivos, en Scratch Jr. en el desarrollo del pensamiento computacional, en los niños de educación inicial

Jeison Isidro Soler Correa

Director:

David Macías Mora

Universidad Pedagógica Nacional

Facultad De Educación

Departamento De Postgrados

Maestría En Tecnologías Aplicadas A La Educación

Bogotá

2025

Resumen:

Este estudio examina el impacto de la creación de cuentos y juegos interactivos en Scratch Jr. en el desarrollo del pensamiento computacional de estudiantes de educación inicial del Colegio Nueva Candelaria de Bogotá. La investigación aborda la ausencia de contenidos curriculares en tecnología e informática en los primeros grados y la escasa existencia de estudios comparativos entre narraciones digitales y juegos interactivos en un entorno virtual común. Se emplea un diseño mixto embebido cuasiexperimental con pretest y postest, complementado con un análisis cualitativo de casos, para caracterizar habilidades como el razonamiento lógico, el pensamiento algorítmico, la descomposición, la abstracción, la identificación de patrones y la evaluación. La intervención se desarrolla mediante sesiones estructuradas en Scratch Jr., organizadas en dos grupos: uno enfocado en la creación de cuentos y otro en el diseño de juegos interactivos. Los resultados cuantitativos muestran una mejora significativa en el nivel global de pensamiento computacional tras la intervención, mientras que el análisis cualitativo identifica trayectorias diferenciadas según el tipo de mediación pedagógica. En el grupo de cuentos, la narrativa facilita la comprensión de secuencias, de relaciones de causa y efecto y de abstracción. En el grupo de juegos, se resalta la importancia de las reglas, los objetivos y la depuración sistemática. Se concluye que ambas modalidades son complementarias para fortalecer el pensamiento computacional en la educación inicial y se proponen orientaciones didácticas para la integración de Scratch Jr. en el currículo.

Abstract:

This study examines the impact of creating interactive stories and games in Scratch Jr. on the development of computational thinking in early childhood students at Colegio Nueva Candelaria in Bogotá. The research addresses the absence of curricular content in technology

and computer science in the first school grades and the scarce number of comparative studies between digital storytelling and interactive games within a shared virtual environment. An embedded mixed-methods quasi-experimental design with pretest and posttest, complemented by a qualitative case analysis, is used to characterize skills such as logical reasoning, algorithmic thinking, decomposition, abstraction, pattern recognition, and evaluation. The intervention is implemented through structured Scratch Jr. sessions organized into two groups: one focused on story creation and the other on the design of interactive games. Quantitative results show a significant improvement in the overall level of computational thinking after the intervention, while the qualitative analysis identifies differentiated learning trajectories according to the type of pedagogical mediation. In the story group, narrative supports the understanding of sequences, cause-and-effect relationships, and abstraction; in the game group, the importance of rules, goals, and systematic debugging is highlighted. The study concludes that both modalities are complementary for strengthening computational thinking in early childhood education and proposes didactic guidelines for integrating Scratch Jr. into the curriculum.

Agradecimientos

A mi pareja, Yen, por su apoyo incondicional, su comprensión y su paciencia durante estos meses de trabajo, que hicieron posible sostener el esfuerzo hasta alcanzar esta meta; a mi mamá, por estar siempre pendiente de mí, por su afecto y por brindarme su ayuda en cada etapa de este proceso, convirtiéndose en un pilar fundamental en mi vida personal y académica; a los niños y las niñas, así como a sus familias, que confiaron en mí y permitieron la realización de esta investigación con su participación generosa y comprometida; y, de manera muy especial, a mi asesor, el profesor David Macías Mora, cuyo esfuerzo incansable, orientación rigurosa y confianza en mi trabajo fueron determinantes para hacer posible este proyecto de maestría.

Tabla de contenido

Lista de figuras

Lista de tablas

Lista de Abreviaturas

PC: Pensamiento computacional

Introducción

La presente investigación surge de la necesidad de incluir en educación inicial espacios pedagógicos donde se puedan abordar contenidos del área de tecnología e informática ya que se ha identificado que desde el ámbito curricular en los grados iniciales, no se cuenta con este espacio, siendo esto una posibilidad para potenciar el aprendizaje y a su vez una herramienta pedagógicas que contribuye al desarrollo de las diferentes asignaturas ya que permiten el desarrollo de habilidades de análisis, abstracción, pensamiento algorítmico, descomposición, resolución de problemas, entre otras.

> Es importante que la capacidad de pensamiento computacional se desarrolle desde un período temprano de formación. Esto redundará en empoderar a los individuos mediante su desarrollo cognitivo y en lograr su autonomía para interactuar con el mundo que les toca vivir. En este contexto, aprender a programar se considera un factor que promueve el desarrollo de una forma de pensamiento más abstracta, analítica y eficiente. En estos procesos de enseñanza y de aprendizaje la fuente de motivación es intrínseca y ayuda a promover la creatividad (Bordignon, F., & Iglesias, A. 2020, p.28)

Por ende, se pretende afianzar el desarrollo del pensamiento computacional desde la educación inicial, reconociendo esta etapa como fundamental en los procesos de desarrollo de los seres humanos, siendo los lenguajes informáticos una posibilidad para fortalecer las habilidades y destrezas de los niños, donde interactúen activamente con las tecnologías, no solo consumiéndolas, sino que las usen como un mecanismo para construir e incidir sobre el mundo que los rodea.

1 Planteamiento del problema

> El pensamiento computacional es un conjunto de saberes que merecen ser desarrollados por nuestros estudiantes, con varios fines: mejorar la habilidad para solucionar problemas y, que estas soluciones puedan traducirse para ser ejecutadas por una computadora; que dejen de ser consumidores pasivos de tecnología y que pasen a tener una relación más estrecha, activa y fructífera con recursos y herramientas tecnológicas, con el objetivo de comprender mejor el mundo que los rodea; que los nuevos lenguajes informáticos les permitan ampliar sus posibilidades de expresión y poner en juego toda su capacidad creativa (Bordignon, F., & Iglesias, A. 2020, p.29).

Es decir, el PC se enfoca en el desarrollo de habilidades de pensamiento, de manera articulada con el aprendizaje y el conocimiento que imparte la escuela. Teniendo en cuenta lo anterior en la presente investigación se pretende analizar el efecto de la creación de cuentos y juegos interactivos en Scratch Jr. en el desarrollo del pensamiento computacional de los niños de educación inicial del colegio Nueva Candelaria.

En los estudios recientes se evidencia la escasez de literatura sobre el desarrollo metódico del pensamiento computacional en la educación inicial, a pesar de su relevancia en la formación temprana de habilidades para resolver problemas, abstracción y secuenciación (Bers et al., 2015; Delgado & Prado, 2018; Zapata et al., 2021). Por otra parte, se documentan experiencias aisladas que apoyan la viabilidad de programar en el contexto de la edad preescolar, bien sea con el uso de juegos interactivos o por medio de narraciones digitales; a pesar de ello, son pocas o nulas las investigaciones que comparan explícitamente estas dos maneras de creación creativa en un mismo ambiente virtual de aprendizaje. Es allí donde la presente investigación indaga, contribuyendo de forma práctica sobre cómo la creación de

cuentos y juegos interactivos en ScratchJr. Se relaciona con el desarrollo de habilidades de pensamiento computacional en niños de educación inicial del Colegio Nueva Candelaria, exponiendo tanto el vacío curricular como la falta de estudios comparativos entre estos dos modelos de mediación pedagógica.

1.1 Pregunta de investigación

¿Cuál es el efecto de la creación de cuentos y juegos interactivos en Scratch Jr. en el desarrollo de las habilidades de pensamiento computacional en los niños de educación inicial del colegio Nueva Candelaria?

1.2 Justificación

Las TIC en la actualidad forman parte de la cotidianidad en los ámbitos social y laboral; se han convertido en una necesidad y una posibilidad para mejorar la calidad de vida de los seres humanos, por lo cual el contexto educativo es fundamental para que los estudiantes desarrollen habilidades que les permitan adquirir aprendizajes propios del pensamiento computacional.

De esta manera surge Scratch Jr. como una aplicación pensada para acercar a los niños de edad preescolar a las herramientas tecnológicas y, especialmente, al lenguaje de programación, contribuyendo al desarrollo del pensamiento computacional, entendido como "resolver problemas, diseñar sistemas y comprender el comportamiento humano, basándose en los conceptos fundamentales de la informática". (Universidad Pedagógica de Argentina, sf) Citando a Wing (2006).

En esta investigación se llevarán a cabo experiencias pedagógicas usando Scratch Jr., donde los estudiantes, desde la creación de cuentos y juegos interactivos, usan bloques de colores para ejecutar diferentes acciones de programación a los personajes, explorando de manera natural e interactiva comandos para solucionar problemas, los cuales van relacionando y aplicando a diferentes situaciones. Desde esta perspectiva, se favorecerán aprendizajes significativos en los que los estudiantes sean sujetos activos en la construcción del conocimiento, usen sus saberes previos y se atrevan a generar propuestas y soluciones ante problemáticas mediante la aplicación de algoritmos y el pensamiento computacional. "La labor del docente en esta propuesta será acompañar a los jóvenes en la resolución de estas tareas y, principalmente, generar instancias de reflexión que vinculen lo trabajado con los conceptos básicos del PC" (Bordignon, F., & Iglesias, A. 2020, p.12).

1.3 Objetivos

1.4 Objetivo General

- Analizar el efecto de la creación de cuentos y juegos interactivos en Scratch Jr. en el desarrollo de habilidades de pensamiento computacional en los niños de educación inicial del colegio Nueva Candelaria.

1.4.1 Objetivos específicos

- Analizar y caracterizar cómo la creación de cuentos en la plataforma Scratch Jr. contribuye al desarrollo de las habilidades de pensamiento computacional (razonamiento lógico, pensamiento algorítmico, descomposición, abstracción, patrones y evaluación) de los niños de educación inicial del colegio Nueva Candelaria.
- Analizar y caracterizar cómo la creación de juegos interactivos en la plataforma Scratch Jr. contribuye al desarrollo de las habilidades de pensamiento computacional (razonamiento lógico, pensamiento algorítmico, descomposición, abstracción, patrones y evaluación) de los niños de educación inicial del colegio Nueva Candelaria.
- Contrastar el efecto de la creación de cuentos y juegos interactivos en el desarrollo del pensamiento computacional en niños de educación inicial.

2 Antecedentes

Esta sección presenta un repaso sistemático que parte de la literatura internacional y regional sobre el pensamiento computacional en educación inicial, incluyendo investigaciones nacionales e internacionales realizadas entre 2018 y 2024. Se inicia revisando el trabajo sobre enfoques generales del pensamiento computacional en la primera infancia, planteando temas de robótica educativa y de interfaces tangibles (Caballero & García, 2019,

2021), y después se centra en estudios que abordan la fortaleza de ScratchJr como plataforma de programación visual apropiada en preescolar (Papadakis et al., 2024; Yang et al., 2023). A continuación, se analizan investigaciones comparativas para valorar la efectividad diferencial de diversas modalidades pedagógicas (aprendizaje basado en juegos, actividades desconectadas, métodos híbridos) (Lin et al., 2020; Dağ, 2023; Sun et al., 2024). Y finalmente, el capítulo sitúa a esta investigación en el contexto mundial y regional mediante revisiones sistemáticas que presentan la evidencia internacional y caracterizan el estado de arte en América Latina (Louka et al., 2022; Zapata-Ros et al., 2021), poniendo en evidencia el vacío investigativo que la investigación actual trata de llenar: la caracterización cualitativa de los procesos cognitivos involucrados en el desarrollo de pensamiento computacional mediante la creación de cuentos y juegos interactivos en ScratchJr en el contexto colombiano de educación inicial.

2.1 Estrategia de la indagación

Para la elaboración de estos antecedentes se procedió a realizar una busqueda sistémica apoyada en bases de datos académicas como Scopus, Web of Sciencie, ERIC y SciELO, implementando combinaciones de palabras clave en español e inglés como: “pensamiento computacional”, “educación inicial”, “early childhood education”, “ScratchJr”, “storytelling”, “game-based learning” y “computer science education”. Abarcando artículos publicados entre 2018 y 2024, en revistas arbitradas, que demostraban experiencias en el desarrollo del pensamiento computacional en la educación inicial o estudios con niños entre 4 y 7 años, dejando de lado las revisiones sin trabajo práctico, estudios con rangos de edad distintos y trabajos que no utilizaran ambientes de programación visual. A partir de allí se escogieron los antecedentes que se explican en los siguientes apartados.

2.2 Pensamiento computacional en educación inicial

2.2.1 Fortaleciendo habilidades de pensamiento computacional en educación infantil: experiencia de aprendizaje mediante interfaces tangibles y gráficas

El aumento del progreso y la evolución tecnológica que viviendo nuestra sociedad han determinado un nuevo rumbo educativo que se forma a raíz de realizar actividades de enseñanza y aprendizaje que ayuden a la apropiación de habilidades digitales vinculadas a la programación y al pensamiento computacional, como la secuenciación de movimientos. En este texto, Caballero y García (2019, 2021) nos presentan una selección de la información más relevante sobre los resultados obtenidos mediante la implementación de una experiencia de aprendizaje centrada en el pensamiento computacional aplicada a los niños en la educación infantil. Los autores llevaron a cabo un estudio cuasiexperimental con 44 estudiantes de 5 a 6 años de Salamanca (España) utilizando retos de programación de dos tipos de interfaz: una tangible (robot Bee-Bot®) y una gráfica (emulador web tipo Scratch).

Los resultados incluyen: un avance medio de 1,95 puntos entre las mediciones pretest y postest en el desarrollo del pensamiento computacional; una evaluación favorable de las actividades por parte de estudiantes y docentes (en particular, valorando positivamente la interfaz tangible); y la mejora de otras habilidades cognitivas (lateralidad, organización espacial y pensamiento crítico). Estos resultados son evidencia empírica de que el pensamiento computacional es desarrollable a estas edades a partir de contextos estructurados de programación visual.

La presente investigación se basa en estos fundamentos. En particular, este estudio traduce estos resultados a un contexto más específico: la creación de cuentos y juegos interactivos en Scratch Jr. con niños en educación inicial. No pretendemos solamente replicar los avances cuantitativos que anotan Caballero y García, sino también profundizar en los

procesos cognitivos concretos—razonamiento lógico, abstracción, depuración, descomposición—que surgen cuando los niños construyen historias digitales programadas. Así, el presente estudio complementa la preferencia previa por las interfaces tangibles, sumando la narración como andamiaje cognitivo para aprender el pensamiento computacional en educación inicial.

2.2.2 Pensamiento computacional a través de estimulación sensorial en niños de transición

Delgado & Prado (2018) desarrollaron un proyecto cuya pregunta de investigación es: ¿Cómo aprovechar la estimulación sensorial en el fortalecimiento del pensamiento computacional en niños de transición?, mediante una metodología cualitativa, donde se aplicaron de pruebas diagnósticas para identificar los conocimientos previos respecto a la pensamiento computacional que tenían los estudiantes, a partir de las cuales se diseñaron unas actividades pedagógicas para observar y analizar los beneficios que aporta la estimulación sensorial durante la ejercitación del pensamiento computacional respecto a la solución de problemas desde la aplicación de razonamiento lógico dentro de los principales resultados respecto al PC se encuentran “el desarrollo de la abstracción, la descomposición, las secuencias, los patrones y los algoritmos permitieron a los estudiantes adquirir las habilidades necesarias para resolver un problema” (Delgado & Prado, 2018, p.164).

A continuación, se resalta la importancia de cada una de ellas:

- El desarrollo de la abstracción en el proyecto permitió a los estudiantes analizar y detallar los problemas, lo que facilitó la generación de una solución adecuada aplicable a varios problemas, específicamente de la cotidianidad.

- La implementación de la descomposición les permitió dividir el problema en partes pequeñas; de este modo, los estudiantes lograron analizar cada una de ellas y así encontrar la solución más adecuada.

- La deducción de secuencias y patrones les proporcionó a los estudiantes información adicional con la cual pudieron anticiparse a los resultados o predecir el siguiente paso; sin duda, estos aspectos son de gran importancia al resolver un problema.

- La construcción de algoritmos por parte de los estudiantes les facilitó generar soluciones a los problemas mediante una secuencia de pasos organizados que siguieron para cumplir con los objetivos de cada actividad (Delgado & Prado, 2018, p.164).

De igual forma, en línea con el problema de investigación de la presente investigación, los autores corroboran que en educación inicial la enseñanza de la informática es poca o nula, y a su vez hacen hincapié en la importancia de desarrollarla en esta edad.

> Es importante ejercitar el pensamiento computacional desde temprana edad, lo que permite adquirir las habilidades necesarias para desenvolverse en la sociedad actual e introducir algunas bases de los componentes de la informática, ya que en este nivel educativo la enseñanza de esta área es casi nula (Delgado & Prado, 2018, p. 18).

2.3 Uso de Scratch Jr. en el ámbito educativo y desarrollo del pensamiento computacional

2.3.1 Scratch Jr. Aprendiendo a programar y programando para aprender

Cati Navarro es maestra de Educación Infantil en el CEIP San Agustín en Casas Ibáñez (Albacete) que en el 2020, realizo una publicación del uso de Scratch Jr. en segundo ciclo de educación infantil con edades entre los 3 y 5 años, para cual desarrollo una metodología por proyectos a partir de rincones, otorgando una tablet por parejas de

estudiantes para que realizaran retos de programación basados en la creación de personajes e historias Para lo cual cada rincón tenía con unas **tarjetas con pictogramas de instrucciones** en las que se indica el número de escenas que deben usar en su proyecto, número de fondos y personajes, bloques que deben usar por colores, y ellos van inventando su historia mientras programan siguiendo esas instrucciones, estas acciones van aumentado su nivel de complejidad. Como principales resultados, se resalta que a través de Scratch Jr. los niños no solo aprenden a programar, sino que "aprenden a organizar su pensamiento, a expresarse, a resolver problemas a través de una secuencia ordenada de instrucciones, desarrollando estrategias cognitivas de planificación, organización, análisis, representación, utilizando la lógica matemática o la lectoescritura de manera funcional y significativa" (Navarro, 2020, p.3).

Finalmente, la autora expresa que el lenguaje de programación es un elemento que contribuye a la estructuración del pensamiento, a partir de la resolución de problemas, siendo Scratch Jr. una aplicación que contribuye a desarrollar la creatividad y pensamiento computacional:

> Permite formular o resolver problemas del mundo que nos rodea mediante secuencias e instrucciones ordenadas (algoritmos) para llegar a la solución. Esto implica organizar la información, analizarla e implementar diferentes soluciones hasta llegar a la más efectiva y eficaz en términos de pasos y recursos. Por tanto, ofrece la oportunidad de desarrollar nuevas formas de pensar y expresarse (Navarro, 2020, p.4).

2.3.2 Scratch Jr.: Aprendizaje en edad temprana mediante programación

Los autores Marina Bers, Mitchel Resnick, Amanda Strawhacker, Dylan Portelance & Melissa Lee (2015), realizaron una investigación frente al aprendizaje mediante la programación en estudiantes de kínder (transición) y segundo grado, a través de Scratch Jr

donde exaltan los pocos espacios escolares destinados al uso de tecnologías en los primeros grados y el cual contribuyo desde un análisis estadístico a analizar la interacción de los estudiantes con la aplicación. Su objetivo principal fue desarrollar una versión del entorno de programación Scratch Jr. y de los recursos curriculares que lo acompañan, apropiado para los niveles de desarrollo de niños entre los 5 y los 7 años y, que además, pudiera implementarse de inmediato en entornos educativos para edad temprana para lo cual se desarrolló en 2 instituciones, la primera con 60 estudiantes y la 2 con 38 estudiantes en la cual se evaluaron los "errores" que cometían al interactuar con la aplicación, lo cual contribuyo al desarrollo y ajustes de esta aplicación final evaluando su idoneidad, frente a 203 aspectos:

- Conocimiento disciplinar específico proveniente de los marcos curriculares estatal y nacional de matemática y de alfabetización temprana.
- Estructuras fundamentales del conocimiento que se aplican de manera transversal en todas las áreas del conocimiento; habilidades como secuenciación, clasificación, símbolos, estimación y predicción.
- Habilidades complejas de solución de problemas; específicamente: 1) identificación de un) formulación de un plan; 3) desarrollo de un intento inicial para alcanzar el objetivo;) ensayar, evaluar y compartir resultados; y e) depurar y revisar el intento inicial con base en la retroalimentación (Bers et al., 2015).

A continuación, se presentan los resultados estadísticos obtenidos en la aplicación de Scratch Jr., así como los errores por curso y por género.

Medida de errores por curso

Tabla 1. **Medida de errores por curso.**

	Kindergarten	Grado 1º	Grado 2º
Niños	22.4	18.7	13.5
Niñas	19.9	15.1	14.4

Nota: Tomado de (Bers et al., 2015).

En kindergarten y en primero de primaria, los niños cometieron más errores en las evaluaciones de habilidades para secuenciar y comprender los bloques. Por otra parte, las niñas tuvieron un mejor desempeño que los niños en todos los grados, excepto en segundo de primaria, donde cometieron más errores.

Tabla 2. **Nivel de errores por institución educativa.**

Institución Educativa	Grado	Número de participantes	Media	Significancia
IE A	Kindergarten	N (niños) = 11 N (niñas) = 10	Xb = 8.73 Xg = 9.90	t(19) = 0.369, p>0.0125, ns
IE B	Kindergarten	N (niños) = 22 N (niñas) = 16	Xb = 10.68 Xg = 10.88	t(36) = 0.162, p>0.0125, ns

Nota: Xb indica errores de los niños; Xg, errores de las niñas. Tomado de (Bers et al., 2015).

Los estudiantes de kindergarten de la Institución Educativa A superaron a su contraparte de la Institución Educativa B por un porcentaje de 2 puntos por tarea del tipo “Soluciónalo” (Bers et al., 2015)

2.3.3 ScratchJr en la educación infantil

El estudio Enhancing computational thinking in early childhood education through ScratchJr integration, una investigación de Papadakis, Kalogiannakis y Zaranis (2024) de Grecia, tenía como objetivo investigar el uso de ScratchJr. como herramienta para la potenciación del pensamiento computacional y de las habilidades elementales de codificación en una muestra de preescolar. Esta investigación se caracterizó por adoptar un diseño cuasiexperimental con un grupo de 34 niños de 4 a 6 años, en el que se llevó a cabo una intervención conectada para el grupo experimental con ScratchJr., y el grupo control, mediante actividades desconectadas, todo ello en un período de intervención de tres semanas. El trabajo de investigación utilizó como instrumento el TechCheck-K, modificado para evaluar actividades con ScratchJr. Y valorar la comprensión de algunos de los conceptos básicos del pensamiento computacional, como la modularidad, las estructuras de control, la representación algorítmica, la comprensión de algoritmos y la interacción hardware/software. Los autores enunciaron como pregunta de investigación central: ¿promueve la intervención educativa a corto plazo de la aplicación ScratchJr el pensamiento computacional en niños preescolares?

Los resultados mostraron que, aunque el grupo control presentó un rendimiento inicial significativamente superior (M=7.07, SD=2.58) al grupo experimental (M=5.35, SD=1.58) en el pretest (t(22.64)=2.23, p=0.036), el análisis de los datos del postest mostró que había una mejora significativa en los participantes preescolares que habían estado expuestos a la intervención educativa con ScratchJr. Ambas propuestas formativas mostraron equipararse al desarrollar los principios de control e incluso modularidad; no obstante, el grupo experimental arrojó resultados superiores al grupo control en construcciones conceptuales poderosas como la representación, los algoritmos o la interacción hardware/software. Un hallazgo paradójico que demostró el estudio fue el hecho de que el grupo control consiguió

un mejor desempeño en la comprensión de la depuración que sus compañeros del grupo experimental, lo que indica que las actividades "desconectadas" podrían resultar útiles para ciertas habilidades del pensamiento computacional. Los autores llegaron a la conclusión de que: "los hallazgos aportan credibilidad convincente a la efectividad del entorno de programación novedoso que ofrece ScratchJr, enfatizando la efectividad de ScratchJr para el desarrollo del pensamiento computacional y las habilidades de codificación en la edad preescolar" (Papadakis, Kalogiannakis & Zaranis, 2024, p. 15).

Este estudio resulta apropiado para la investigación, debido a que presenta pruebas empíricas a favor de ScratchJr. en contextos educativos formales de forma reciente y en la franja de edad objeto de interés (4-6 años), también la comparación con las actividades desconectas indica que ambas actividades se complementan para lograr el desarrollo del PC, lo que sugiere que la integración de actividades como la construcción de cuentos y juegos en ScratchJr, se puede convertir en una ventaja que favorece beneficios sinérgicos en el desarrollo del pensamiento computacional.

2.3.4 Historias digitales y pensamiento computacional.

En el marco de su estudio titulado *The impact of story-inspired programming on preschool children's computational thinking*, Yang, Ren y Bers (2023) abordaron el impacto de la programación inspirada en relatos (Story-Inspired Programming, SIP) en el desarrollo del pensamiento computacional en niños en edad preescolar. Este estudio de diseño cuasi-experimental pretest-postest tuvo lugar en los contextos educativos de Estados Unidos y de China y abarcó la participación de niños entre 4 y 6 años que formaban parte de sesiones programadas en torno a la producción de historias digitales. Se pretendía que la utilización de historias, como tipo de andamiaje cognitivo, pudiera ayudar a los niños en la comprensión de conceptos abstractos relacionados con la programación; fue uno de los objetivos de la investigación. Los autores sustentaron que las historias permiten establecer un contexto

objetivo que fundamenta y ordena la adquisición del pensamiento computacional, aprovechando la naturalidad de los niños con respecto a la estructura narrativa (introducción, desarrollo y final).

La propuesta, en cuanto a la intervención, consistió en actividades en las que los niños debían desarrollar sus propias historias interactivas a partir de bloques de programación visual, como se hace en el entorno de ScratchJr.

Los resultados aquí presentados demostraron que los niños que estaban expuestos a la condición de programación basada en historias consiguieron incrementos significativos en los puntajes de pensamiento computacional en comparación con el grupo control. Del mismo modo es de resaltar que los resultados llegaron a la conclusión de que, no existen diferencias estadísticamente significativas ni por sexo de los participantes, ni por nivel socioeconómico familiar, lo que significa que su enfoque narrativo proporciona un andamiaje cognitivo equitativo que sirve a todos los estudiantes en la condición, sin importar cuáles son sus características demográficas. Los autores aclaran que "los escolares que han sido expuestos a la condición de programación sobre historias (SIP por sus siglas en inglés y que en este caso hace referencia a Story Inspired Programming) han conseguido aumentar los puntajes en relación con el pensamiento computacional. No se encontraron diferencias según el género ni en el nivel socioeconómico familiar" (Yang, Ren & Bers, 2023, p. 5).

Este antecedente nos interesa particularmente para el presente estudio, ya que valida de forma directa el enfoque de cuentos digitales como una de las condiciones experimentales que se está proponiendo. La evidencia empírica de Yang y sus colaboradores demuestra que la narrativa tradicional no solo puede ser una buena práctica de enseñanza, sino que es un contexto inclusivo y de sentido para abordar el aprendizaje del pensamiento computacional en educación inicial. Asimismo, se demuestra que las historias también pueden representar un

puente natural entre la experiencia de los niños y los conceptos abstractos de programación, facilitando el paso hacia la construcción de argumentos algorítmicos y resolución de problemas computacionales.

2.4 Comparativa entre Tipos de Modalidades Pedagógicas

Hasta aquí, los antecedentes mostrados respaldan el uso del pensamiento computacional en educación infantil ya sea con enfoques no digitales, bien mediante aquellas herramientas específicamente orientadas para ser empleadas por los niños en edad preescolar. Sin embargo, persiste una cuestión fundamental: ¿se producen diferencias cualitativamente importantes al implementar diferentes modalidades pedagógicas basadas en una misma herramienta? Aún más, ¿qué resultados se obtienen al sustituir las actividades digitales de juego por recursos narrativos (cuentos) o, si se elige un enfoque distinto de la tecnología, por actividades desconectadas?

La siguiente sección nos aproxima a tres estudios que abordan estas cuestiones de la forma planteada en el párrafo y nos dejan entrever cómo funcionan las diferencias entre varias modalidades pedagógicas para trabajar el pensamiento computacional.

2.4.1 Aprendizaje basado en juegos

En el estudio Enhancing computational thinking capability of preschool children by game-based smart toys (Lin, Chien, Hsiao, Hsia y Chao, 2020, Taiwán), se centraron en explorar el aprendizaje basado en juegos (Game-Based Learning, GBL) y tecnologías basadas en interfaces tangibles de usuario (Tangible User Interfaces, TUI) para elaborar habilidades del pensamiento computacional a partir de juegos en un grupo de niños de preescolar. El estudio involucró a un grupo de niños de entre los 4 y los 6 años que jugaron con juguetes inteligentes diseñados para enseñar lógica computacional y programación mediante actividades lúdicas estructuradas.

Los autores afirmaban que los niños de preescolar tienen habilidades cognitivas limitadas y requieren mayor asistencia para trasladar conceptos abstractos a un concepto del mundo real. Por consiguiente, proponen que el enfoque del aprendizaje basado en juegos con TUI, no sólo proporciona una mayor motivación para que los preescolares utilicen materiales de aprendizaje, sino que también propicia el mayor desarrollo de las habilidades de pensamiento computacional. En cumplimiento del objetivo de estudiar cómo aplicar un método de aprendizaje basado en juegos para potenciar los conceptos computacionales del estudiante, las prácticas computacionales y las perspectivas computacionales, en la investigación presentada, en el marco de los dos objetivos del estudio, se realizó: el desarrollo e implementación de una interfaz de usuario tangible que propicie el interés de aprendizaje de los preescolares y los motive a participar en las acciones de aprendizaje, y la aplicación de un método de aprendizaje basado en juegos para potenciar los conceptos computacionales del estudiante, las prácticas computacionales y las perspectivas computacionales. Los resultados de la investigación constataban que tanto el enfoque del aprendizaje basado en juegos con TUI como su aplicación mejora significativamente los resultados cognitivos en pensamiento computacional, una mayor motivación por el aprendizaje y los comportamientos de aprendizaje activo de los participantes. Los pequeños pasaron por el proceso satisfactorio de la resolución de problemas al detectar y corregir fallos en sus programas, lo cual ayudó a mejorar sus habilidades para la depuración y su perseverancia frente a los problemas. Un segundo resultado que hallaron los autores fue que la experiencia previa de los niños en relación con la tecnología contribuye en su rendimiento inicial, dicho comportamiento sugiere que es necesario exponer a los niños a experiencias tempranas de programación. Los autores concluyeron que "el enfoque de aprendizaje basado en juegos junto con aplicaciones de interfaz gráfica tangible constituye una manera de motivar a los preescolares a que participen activamente con los materiales de aprendizaje, de este modo incrementar sus

habilidades de pensamiento computacional" (Lin, Chien, Hsiao, Hsia & Chao, 2020, p. 8). El presente estudio concluyó que los juegos no sólo pueden motivar a los estudiantes, sino que también pueden estructurar el pensamiento computacional mediante mecánicas de resolución de problemas, características lúdicas y la posibilidad de experimentar el éxito, como resultado del trabajo realizado en problemas de programación superados.

Para la presente investigación, este antecedente tiene una especial relevancia, ya que se encuentra particularmente conectado con el enfoque propuesto para el segundo grupo experimental (GE2: Diseño de juegos interactivos). La evidencia aportada reitera lo que ya indicaban Lin y sus colaboradores, que el buen diseño de los juegos y la claridad en el objetivo de aprendizaje hacen que el juego sea tan válido como otro tipo de estrategias para desarrollar el pensamiento computacional. Por último, el estudio respaldó también la idea de que las interfaces tangibles y los juegos son claramente apropiados para los niños que están en las primeras etapas del desarrollo cognitivo, brindando situaciones concretas que ayudan a la comprensión de los principios abstractos de la programación.

2.4.2 Actividades desconectadas (unplugged)

El estudio The effect of unplugged coding course on primary school students' computational thinking skills, collaboration, and communication competencies llevado a cabo por Dağ en Turquía (2023) analiza el efecto de un curso de codificación desconectada (unplugged) en las habilidades de pensamiento computacional, las competencias de colaboración y comunicación en alumnos de educación primaria. El diseño de la investigación fue cuasiexperimental de un único grupo y con medidas pretest y postest, y se sumó un seguimiento a las 10 semanas. El experimento comprendió una muestra de 212 alumnos de 3º y 4º de primaria de una escuela pública turca. El estudio en cuestión tuvo como objetivo determinar la eficacia de las actividades de codificación unplugged, es decir, que no emplean ordenadores ni dispositivos digitales, para desarrollar las habilidades de

pensamiento computacional de los estudiantes de primaria. En concreto, los autores evaluaron la mejora en cinco componentes del pensamiento computacional: diseño algorítmico, abstracción, evaluación, descomposición y generalización. También se observó una mejora en las habilidades de pensamiento computacional en relación con las habilidades de colaboración y comunicación de los estudiantes, así como la influencia de variables sociodemográficas (género y nivel socioeconómico).

Los resultados del estudio mostraron que el curso que se centraba en la codificación unplugged mejoraba estadística y significativamente las habilidades de pensamiento computacional de los estudiantes en todas las dimensiones evaluadas. Resulta importante reseñarlo ya que los resultados de dicho estudio mostraron que las habilidades de pensamiento computacional no se relacionaron con las variables sociodemográficas, lo que sugiere que el enfoque resultaba equitativo y un beneficio para todos los estudiantes independientemente del género o el nivel socioeconómico. Un hallazgo del estudio que merece la pena destacar es que las habilidades de pensamiento computacional de los estudiantes se correlacionaron positiva y significativamente con sus habilidades de colaboración y comunicación, así como que el desarrollo de habilidades de pensamiento computacional esté intrínsecamente relacionado con el desarrollo de competencias socioemocionales.

Además, el seguimiento a las 10 semanas tras la intervención que realizaron los autores mostró que los efectos positivos constatados en el curso de codificación unplugged se sostenían en el tiempo, evidenciando que existe retención de conocimientos y habilidades a medio plazo. En definitiva, los autores concluyeron que “el curso de codificación unplugged mejoró estadística y significativamente las habilidades de pensamiento computacional de los participantes. Y la correlación significativa entre las habilidades de colaboración y de comunicación” (Dağ, 2023, p. 9).

Pese a que en este estudio emplearon una serie de actividades desconectadas en vez de recurrir a herramientas digitales como ScratchJr, este estudio es válido para esta investigación por dos motivos centrales. En primer lugar, demuestra que el pensamiento computacional puede desarrollarse de manera acertada a través de diferentes modalidades de enseñanza, siempre que están adecuadamente diseñadas para su utilización e implementadas con tal intención educativa. En segundo lugar, y mucho más interesante, es porque proporciona evidencia empírica sobre la relación intrínseca existente entre el pensamiento computacional y las habilidades de colaboración, incluyendo esta última como una de las variables cualitativas a evaluar en la investigación que estamos llevando a cabo. Esta evidencia contribuye, a su vez, a la hipótesis de que las interacciones entre estudiantes no surgen en las actividades de programación como subproductos de la forma en que se realiza el trabajo en grupo, sino que constituyen un componente fundamental en el desarrollo del pensamiento computacional.

2.4.3 Comparativa unplugged vs. plugged-in

El estudio investigativo Comparative experiment of unplugged and plugged-in programming activities on the computational thinking skills of lower primary school students por Sun, Zhou y Li (2024) en China, sería uno de los últimos ejemplos en el campo de la comparación entre las actividades de programación unplugged y plugged-in para el fomento de habilidades de pensamiento computacional. Este estudio experimental de tipo comparativo tuvo lugar con estudiantes de primaria y fue desarrollado durante un tiempo de intervención de 10 semanas, con el objetivo de comparar si hay diferencias significativas respecto a la presentación de las dos modalidades de instrucción cuando el contenido de aprendizaje y las evaluaciones con las que cuentan son equiparables. Los autores diseñaron dos condiciones experimentales con contenido de aprendizaje paralelo, es decir, un grupo participó en actividades de programación desconectadas mediante la utilización de materiales

manipulativos de tipo físico (tableros cuadriculados, dados numéricos y tarjetas de instrucciones); en el otro grupo se tomó como base herramientas digitales de programación visual similar a ScratchJr y Code.org. Ambos grupos aprendieron los mismos conceptos de pensamiento computacional (secuenciación, bucles, condicionales, depuración) y ambos grupos fueron evaluados con instrumentos equivalentes de evaluación para asegurar que los resultados fueran comparables.

Los resultados del estudio mostraron que ambas modalidades alcanzaron un aumento significativo en las capacidades de pensamiento computacional de los estudiantes, pero no se encontraron diferencias estadísticamente significativas entre los dos grupos en el desarrollo de las capacidades de pensamiento computacional al finalizar la intervención de las 10 semanas. Los autores destacaron que "tanto las actividades conectadas como las desconectadas lograron incrementar considerablemente las habilidades de pensamiento computacional de los más pequeños tras una intervención educativa a corto plazo, porque no se encontraron diferencias significativas entre los dos grupos" (Sun, Zhou & Li, 2024, p. 12).

Este resultado expone realidades prácticas, en el sentido de que las escuelas que tienen recursos tecnológicos limitados pueden desarrollar efectivamente el pensamiento computacional a partir de actividades desconectadas bien diseñadas. También que el tipo de modalidad (digital o no digital) no es un factor que determine el éxito en el desarrollo del pensamiento computacional, sino que la calidad del diseño instruccional, la alineación del contenido y los objetivos de aprendizaje, y la pertinencia a la hora de decidir qué actividades realizar tienen más importancia que la modalidad.

Esta investigación tiene una doble implicación para la investigación. La primera se encuentra alineada con la aceptación de la hipótesis de que la comparación entre dos modalidades digitales diferentes (cuentos digitales vs. juegos interactivos) va a dar

equivalencia estadística siempre que las dos sean correctamente diseñadas, sin restar valor pedagógico a esta afirmación, sino que sencillamente lo que pone de manifiesto es que existen diferentes caminos para llegar al mismo objetivo, a la vez que la segunda se encuentra en la aceptación de que dar valor a los resultados cuantitativos de aprendizaje no es suficiente, sino que también hay que dar valor cualitativo al proceso a través del cual los niños construyen su modo de comprender la realidad, lo que justifica que la investigación que se presenta tenga una metodología mixta embebida.

A pesar de que en esta investigación no se aborda directamente la comparación de actividades conectadas y desconectadas, lo expuesto hasta aquí permite enmarcar la propuesta de manera más amplia entre las modalidades pedagógicas para desarrollar el pensamiento computacional, demostrando que los cuentos y los juegos digitales pueden vincularse con experiencias desconectadas en medio de una didáctica mixta.

2.5 Panorama Regional y Global del Pensamiento Computacional

Las investigaciones internacionales en el área del pensamiento computacional en educación inicial permiten extraer perspectivas válidas sobre la eficacia de ambientes virtuales como ScratchJr. Además, ponen en evidencia las lagunas de investigación que aún existen, en especial en contextos latinoamericanos.

2.5.1 Revisión sistemática internacional

La revisión sistemática A systematic review of computational thinking in early childhood education de Louka, Papadakis y Kalogiannakis (2022) estudió publicaciones empíricas en revistas indexadas para detectar las mejores plataformas de programación para

trabajar el pensamiento computacional con los niños. La revisión sistemática mostró a ScratchJr. como un espacio de programación ampliamente utilizado, ya que fue revisado por diferentes autores; concretamente, siete estudios empíricos llegaron al consenso de afirmar que la aplicación para terminales táctiles se utiliza para enseñar los conceptos básicos del pensamiento computacional en preescolar. Los resultados extraídos demostraron que los niños en su etapa de preescolar son capaces de hacer un uso competente de los bloques de comandos de ScratchJr para construir proyectos completos, mostrando así aptitud para entender y usar sus capacidades de secuenciación, eventos, bucles y condicionales simples. Los autores concluyen que "ScratchJr es un entorno que favorece sobremanera la codificación y el desarrollo del pensamiento computacional en la edad inicial a partir del compromiso con el ambiente virtual de aprendizaje" (Louka et al., 2022, p. 12). Este estudio permite validar la elección de ScratchJr como herramienta de intervención y también sustenta la validez externa de la presente investigación.

2.5.2 Contexto latinoamericano

La revisión sistemática Integración del pensamiento computacional en la educación primaria y secundaria en Latinoamérica, de Zapata-Ros, González-Cabrera y Caicedo-Tamayo (2021) examina las publicaciones académicas en nueve países de Latinoamérica entre 2006 y 2020. Los hallazgos evidencian que Brasil es el país que más artículos ha publicado (13), seguido de México y Colombia, con 3 artículos cada uno de ellos. Las estrategias más trabajadas en Latinoamérica son la robótica educativa y la programación con lenguaje de bloques (Scratch y sus variantes).

Estos autores encontraban grandes limitaciones en su trabajo: la falta de infraestructura, la necesidad de formación docente y la poca investigación en la educación inicial. La conclusión que extraen es la siguiente: "En Latinoamérica son escasos los artículos que tratan el desarrollo e integración del pensamiento computacional en la educación

primaria y secundaria... Las estrategias más usadas en la región son la robótica educativa y la programación en lenguajes con bloques" (Zapata-Ros et al., 2021, p. 15).

Este antecedente sitúa a la presente investigación en el marco latinoamericano y muestra que aporta a cubrir un vacío detectado en la investigación de educación inicial en Colombia, al mismo tiempo que valida la programación con bloques (ScratchJr) como estrategia recomendada en la región.

2.6 Síntesis e Integración de Antecedentes

La indagación de antecedentes internacionales y de la región presenta patrones de hallazgos que apoyan el diseño de esta investigación. En primer lugar, existen evidencias de que niños de 4 a 6 años pueden desarrollar de forma efectiva su pensamiento computacional con una herramienta como ScratchJr (Louka et al., 2022; Papadakis et al., 2024) y, por tanto, se respalda su elección como herramienta de intervención. En segundo lugar, las actividades inspiradas en cuentos (Yang et al., 2023) o en juegos (Lin et al., 2020; Sun et al., 2024) han demostrado un efecto similar, lo que sugiere que podrían ser equivalentes si están bien enfocadas.

En tercer lugar, el rol docente se vuelve un aspecto clave en el uso de ScratchJr (Strawhacker et al., 2018), que apoya el diseño de esta investigación, en la que el investigador ejerce un rol activo como mediador pedagógico. En cuarto lugar, el pensamiento computacional está asociado al desarrollo de competencias socioemocionales (Dağ, 2023), consolidando las actividades de programación como un medio que permite atender procesos cualitativos de interacción social.

Finalmente, la revisión de antecedentes del contexto latinoamericano (Zapata-Ros et al., 2021) pone de manifiesto la escasez de investigaciones en educación inicial que examinen

o equiparen los dos modelos (cuentos vs. juegos) e incorporen análisis cualitativos. Esta falta confirma la necesidad de abordar la propuesta en el contexto colombiano.

En suma, los antecedentes validados afirman la elección del recurso (ScratchJr), las modalidades pedagógicas (cuentos y juegos), la población (4-6 años) y la metodología (mixta cuantitativa-cualitativa embebida). Aún más importante, también hacen visible la pregunta sin respuesta: ¿Cuáles son los procesos cognitivos específicos y los andamiajes diferenciados a través de los cuales la elaboración de cuentos y juegos interactivos en ScratchJr contribuye a desarrollar el pensamiento computacional en educación inicial?

3 Marco teórico

En el presente recorrido teórico se toman diferentes referentes respecto al pensamiento computacional, pero es preciso señalar que los postulados de: Wing; Bordignon & Iglesias y Computing at School son fundamentales para comprender y analizar la aplicación en el aula ya que permiten tener acercamientos a los conceptos y habilidades que potencia el PC, así mismo reconocer las características que deben tener las sesiones de clase.

3.1 Pensamiento computacional

Este concepto es el foco de investigación, el cual se encuentra relacionado con la capacidad de resolver problemas a partir de la interacción con elementos tecnológicos, entre otros, el cual se recomienda desarrollar desde edades tempranas, ya que no debe concebirse como exclusivo de los profesionales en sistemas, sino que debe ser parte de la formación educativa como posibilitador de experiencias de aprendizaje y desarrollo a partir de la resolución de problemas.

> El pensamiento computacional consiste en la resolución de problemas, el diseño de los sistemas, y la comprensión de la conducta humana haciendo uso de los conceptos

> fundamentales de la informática. (…) Son habilidades útiles para todo el mundo, no sólo para los científicos de la computación" (Wing, 2006), citado en (Zapata, 2015, p. 12).

Continuando con los postulados de Wing como referente y pionera del PC, su teoría expone la importancia de este en el siglo XXI, ya que es una habilidad que debe usarse por todos.

> Define el pensamiento computacional como la resolución de problemas, el diseño de sistemas y la comprensión del comportamiento de los seres humanos, basándose en los conceptos fundamentales de la informática; es decir, la percepción de la realidad mediante la formulación de algoritmos. Wing también agrega que dicho pensamiento será una habilidad utilizada por todos a mediados del siglo XXI." (Herrera, 2017, p. 7), citando a (Wing, 2006).

Esta afirmación contrasta con la realidad actual ya que muchos de los trabajos son remotos, los empleados de diferentes cargos, áreas y rangos, así mismo como los estudiantes han tenido que trabajar, socializar y estudiar de manera virtual, esta necesidad no está ligada únicamente a los conocimientos puntuales sobre el manejo computacional sino en la aplicabilidad para el desarrollo de habilidades y conocimientos aplicables al contexto cotidiano.

> El pensamiento computacional proporciona un marco para el estudio de la computación de gran alcance, con amplias aplicaciones que van más allá de la computación en sí. Es el proceso de reconocimiento de los aspectos de la computación en el mundo que nos rodea, así como la aplicación de herramientas y técnicas de computación para entender y razonar sobre sistemas naturales y sociales, así como sobre procesos artificiales. Permite a los estudiantes hacer frente a los problemas,

descomponerlos en partes solucionables y diseñar algoritmos para resolverlos (Computing at School, 2015, p.7).

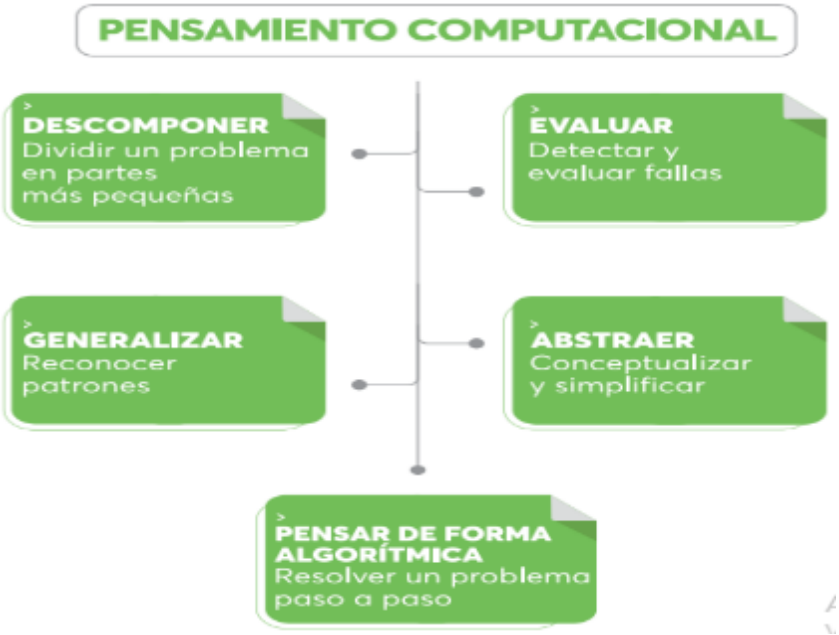
Espino y González (2015) recalcan la importancia de incluir en el sistema educativo el desarrollo del PC en edades tempranas:

> Otra de las competencias que debería incluirse en la formación de todos los niños desde las primeras etapas es la del Pensamiento Computacional, un concepto estrechamente ligado a las Tecnologías de la Información y la Comunicación (TIC). Esto se debe a que el pensamiento computacional posee una serie de características propias del pensamiento científico aplicadas al proceso de resolución de problemas (Computing at School, 2015, p. 2).

3.2 Conceptos de pensamiento computacional

A continuación, se describen los conceptos de pensamiento computacional propuestos por Computing at School (2015). Cada uno de ellos permite identificar distintos comportamientos observables en el aula.

Figura 3.1 Pensamiento computacional.



Nota: Tomado de: Computing At School (2015).

3.2.1 Razonamiento lógico

El pensamiento computacional es un proceso cognitivo que involucra el razonamiento lógico, mediante el cual se resuelven problemas, se analizan procedimientos y se comprenden los sistemas con mayor profundidad.

> La capacidad de pensar de forma algorítmica; la capacidad de pensar en términos de descomposición; la capacidad de pensar en términos de generalizaciones, identificando y haciendo uso de patrones; la capacidad de pensar en términos abstractos; la elección de buenas representaciones; y la capacidad de pensar en términos de evaluación (Computing at School, 2015, p.5).

Este concepto se relaciona con los conocimientos previos de los estudiantes para que generen ideas, hipótesis y conclusiones, los cuales contrastan con las experiencias de aprendizaje en la resolución de problemas.

> El razonamiento lógico permite a los estudiantes dar sentido a las cosas mediante el análisis y la comprobación de los hechos, con claridad y precisión. Permite a los estudiantes recurrir a sus propios conocimientos y modelos internos para formular y verificar predicciones y extraer conclusiones. Es ampliamente utilizado por los estudiantes cuando ponen a prueba, depuran y aplican algoritmos correctos. El razonamiento lógico es la aplicación de otros conceptos de pensamiento computacional para resolver problemas. (Computing at School, 2015, p.6).

3.2.2 Abstracción

Esta habilidad permite generar procesos de pensamiento frente a un problema que facilitan su comprensión; es decir, sintetizarlo sin pasar por alto los detalles que lo caracterizan.

> La habilidad de abstracción consiste en seleccionar qué detalle ocultar para que el problema se vuelva más fácil, sin perder lo importante. Una parte fundamental de la misma es la elección de una representación adecuada de un sistema. Diferentes representaciones hacen diferentes cosas fáciles de realizar (Computing at School, 2015, p.7).

3.2.3 Evaluación

Está relacionada con la capacidad de tomar decisiones asertivas, es decir, adecuadas para los estudiantes, teniendo en cuenta sus particularidades y requerimientos. Este concepto es fundamental en el desarrollo de las experiencias pedagógicas, en cuanto a la reflexión, la observación y el análisis: su pertinencia, complejidad y agilidad, entre otras.

> La evaluación es el proceso de asegurar que una solución, ya sea un algoritmo, un sistema o un proceso, es una buena solución: adecuada para el propósito. Varias propiedades de las soluciones deben evaluarse. ¿Son correctas? ¿Son lo suficientemente rápidos? ¿Son fáciles de usar para las personas? ¿Promueven una

> experiencia adecuada? Hay un enfoque específico y, con frecuencia, una atención extrema al detalle en la evaluación basada en el pensamiento computacional (Computing at School, 2015, p.7).

3.2.4 Pensamiento algorítmico

El pensamiento algorítmico es la capacidad de pensar en términos de secuencias y reglas para resolver un problema; es la capacidad de aplicar un conocimiento de manera repetitiva ante un estímulo determinado, por ejemplo, al aplicar un algoritmo a problemas comunes.

> Es una forma de llegar a una solución mediante una definición clara de los pasos. Algunos problemas se resuelven de una sola vez: se aplican soluciones y luego se aborda lo siguiente. El pensamiento algorítmico se necesita cuando problemas similares deben resolverse una y otra vez y no deben pensarse de nuevo cada hora. Se necesita una solución que funcione todo el tiempo (Computing at School, 2015, p.8).

3.2.5 Descomposición

Este concepto se relaciona con la capacidad de concretar y reducir de lo macro a lo micro, entendiendo que cada elemento es único y posee características específicas.

> La descomposición es una manera de pensar en los artefactos en términos de sus partes y componentes. Cada pieza debe entenderse, resolverse, desarrollarse y evaluarse por separado. Esto facilita la resolución de problemas complejos y el diseño de grandes sistemas. (Computing at School, 2015, p.8).

3.2.6 Generalización (patrones)

La observación, la experiencia y los conocimientos previos son fundamentales en este concepto, ya que permiten contrastar los aspectos comunes de un problema o artefacto y aplicar soluciones exitosas ya probadas previamente.

> La generalización se asocia con la identificación de patrones, similitudes y conexiones, así como con la explotación de las características. Es una forma de resolver rápidamente los nuevos problemas a partir de las soluciones de los problemas anteriores y de la construcción basada en la experiencia previa. Haciendo preguntas como: ¿Esto es similar a un problema que ya he solucionado? (Computing at School, 2015, p.8).

3.3 Técnicas para evidenciar el pensamiento computacional

Computing at School (2015) menciona la reflexión, codificación, diseño, análisis y aplicación como técnicas que permiten demostrar y evaluar el pensamiento computacional respecto a cómo opera en el aula, el lugar de trabajo y el hogar; es decir, en el marco de esta investigación, se hace referencia a los aspectos a desarrollar en los estudiantes de educación inicial del colegio Nueva Candelaria.

3.3.1 Reflexión

Es la habilidad para realizar juicios o evaluar situaciones complejas: “dentro de la informática, esta evaluación se basa en criterios que se utilizan para especificar el producto, heurísticas (o reglas de oro) y las necesidades del usuario para guiar los juicios” (Computing at School, 2015, p.10).

3.3.2 Codificación

“Un elemento esencial del desarrollo de cualquier sistema informático consiste en traducir el diseño en forma de código y evaluar de manera que garantice que funcione

correctamente en todas las condiciones" (Computing at School, 2015, p.10). Se relaciona con la aplicación de destrezas para analizar y decidir, basadas en el pensamiento lógico, a fin de anticipar un resultado.

3.3.3 Diseño

Se caracteriza por la estructura, apariencia y funcionalidad de los artefactos "incluidas las representaciones legibles por humanos, tales como diagramas de flujo, guiones gráficos, diagramas de sistemas, etc. Se trata de nuevas actividades de descomposición y de abstracción en el diseño de algoritmos" (Computing at School, 2015, p.10).

3.3.4 Analizar

Consiste en aplicar el pensamiento lógico tanto para percibir mejor las cosas, evaluar y tomar decisiones y ejecutar algoritmos:

> Consiste en dividir en partes todos los componentes (descomposición), reducir la complejidad innecesaria (abstracción), identificar los procesos (algoritmos) y buscar elementos comunes o patrones (generalización). (Computing at School, 2015, p.10).

3.3.5 Aplicar

> La aplicación consiste en la adopción de soluciones preexistentes para satisfacer las necesidades de otro contexto. Es generalización — la identificación de patrones, similitudes y conexiones — y explotar aquellas características de la estructura o la función de los artefactos (Computing at School, 2015, p.10).

Síntesis operativa del pensamiento computacional

Para comprender el concepto de PC en la investigación cualitativa de este trabajo, es necesario realizar una síntesis de las seis habilidades de PC que han sido

descritas en las secciones anteriores, precisando para cada una: a) su definición conceptual; b) las acciones observables que evidencian su desarrollo en estudiantes de educación inicial; esta síntesis será la guía de la codificación temática y el análisis en profundidad de la información cualitativa que se expondrá en el Capítulo 6.

TABLA 3: Síntesis operacional de habilidades de pensamiento computacional

Habilidad de PC	Definición Conceptual	Acciones Observables en Estudiantes de Educación Inicial
Razonamiento Lógico	Capacidad de analizar, comparar y verificar hechos a través de la lógica para solucionar problemas o extraer conclusiones	Establece relaciones particulares entre las funcionalidades de bloques con el comando
		Contrasta que lo que sucede efectivamente en el programa es lo que ha sido planeado
		Justifica la decisión tomando como base un razonamiento lógico
		Aplica los conocimientos previos para dedicarse a un determinado problema
		ealiza predicciones y posteriormente, las verifica sobre el comportamiento del código que desarrolla
Pensamiento Algorítmico	Capacidad de traducir y secuenciar acciones en un orden, en función de un estímulo determinado mediante la aplicación repetida de un algoritmo específico.	Establece acciones y las organiza en secuencias lógicas
		A partir de una secuencia propuesta predice resultados
		Repite patrones de resolución de un problema similar
		• Traduce procesos complejos en instrucciones paso a paso
		• Revisa y modifica la secuenciación si no se obtiene el efecto deseado
Descomposición	Capacidad para separar un problema o sistema complejo en partes más pequeñas y más fáciles de manejar, resolviendo éstas por separado	• Identifica las partes que componen un sistema o problema
		• Reconoce que cada parte presenta propiedades y características de forma
		• Considera cada parte como un módulo independiente
		• Resuelve cada parte por separado
		• Unifica las soluciones parciales en una solución general

Continuación TABLA 3

Habilidad de PC	Definición Conceptual	Acciones Observables en Estudiantes de Educación Inicial
Abstracción	Capacidad para sintetizar un problema, omitiendo las partes no esenciales y conservando las propiedades que se requieren para la comprensión del mismo	• Omite la información innecesaria al analizar un problema
		• Utiliza símbolos o representaciones a las ideas abstractas
		• Identifica los elementos que son esenciales en un concepto complejo
		• Generaliza de situaciones concretas a conceptos abstractos
		• Selecciona representaciones adecuadas para concepciones
Generalización / Patrones	Capacidad de señalar y discernir similitudes, conexiones y patrones recurrentes, reutilizando así soluciones anteriores para nuevos problemas.	• Reconoce patrones sistemáticos dentro de datos y/o procesos
		• Reutiliza soluciones que previamente fueron efectivas
		•Anticipa el siguiente paso, a partir de patrones observados
		• Pregunta naturalmente: "¿He visto esto antes?"
		• Aplica normas identificadas dentro de nuevas situaciones
Evaluación / Depuración	Capacidad de verificar si una solución es correcta, identificando en qué lugar está el error, para corregirlo adecuadamente	•Prueba su código mientras lo está creando
		Ubica específicamente en qué parte está el error
		Modifica el código de forma intencional para corregir el error
		•Verifica que la corrección ha dado el resultado deseado
		Emplea la depuración como parte del proceso de construcción y no solo al final

Nota. Esta tabla resume los postulados teóricos de Computing at School (2015) y Wing (2006) y sirve como referencia operacional para el análisis de los datos cualitativos que se presentan en el Capítulo 6. Cada capacidad se representa a partir de su definición teórica, y

las acciones observables mencionadas son el resultado de las conductas esperadas en niños/as de Educación Infantil en su interacción con Scratch Jr.

3.4 Educación inicial

Es la etapa inicial de la educación; comprende a los niños de la primera infancia, de 0 a 5 años. Cabe precisar que en los colegios públicos se incluyen los grados de jardín y transición.

> Para UNICEF, el primer ciclo de educación infantil es una etapa fundamental que mejora el bienestar de los niños, favorece el desarrollo físico y emocional, promueve la igualdad de género y tiene un gran impacto en la equidad y en la reducción de la desigualdad y de la pobreza infantil (UNICEF,2020).

En el 2006, el Estado colombiano estableció la política para el desarrollo integral de la primera infancia, denominada Ley 1804, la cual en su artículo 5, define la educación inicial como:

> Un derecho de los niños menores de seis (6) años. Se concibe como un proceso educativo y pedagógico intencional, permanente y estructurado, a través del cual los niños "desarrollan su potencial, capacidades y habilidades en el juego, el arte, la literatura y la exploración del medio, contando con la familia como actor central de dicho proceso. Su orientación política y técnica, así como su reglamentación, estarán a cargo del Ministerio de Educación Nacional y se elaborarán de acuerdo con los principios de la Política de Estado para el Desarrollo Integral de la Primera Infancia de Cero a Siempre (Ley 1804, 2016).

En los lineamientos de educación inicial colombiana, destacan la importancia de implementar de manera significativa con múltiples lenguajes, incluyendo el tecnológico, pero sin

enmárcalo dentro de estándares específicos de pensamiento computacional. Los estándares actuales del Ministerio de Educación Nacional descritos en la Guia 30, se presentan a partir de la educación básica, lo que evidencia un vacío curricular en la educación inicial y dificulta planear de forma sistémica espacios que fomenten la experiencia del pensamiento computacional con niños de 5 a 6 años.

3.5 Scratch Jr.

Es el programa que se utiliza para aprender programación. Fue desarrollado por el MIT (Instituto de Tecnología de Massachusetts) y se imparte de forma gratuita. Su nombre proviene de la palabra “Scratching”, que en el ámbito de la programación se refiere a trozos de código que pueden reutilizarse, combinarse fácilmente y adaptarse a nuevos usos. Esta aplicación es compatible con tablet y Android y puede concebirse como un juego que invita a los niños a generar acciones de programación.

> Scratch Jr. es un lenguaje de programación adecuado para el momento de desarrollo en el que se encuentran los niños entre los cinco y los siete años. Usando la aplicación

Scratch Jr., los niños pueden crear sus propios collages (montajes) interactivos, historias animadas y juegos (DevTech Research Group, 2016).

Figura 3.2. **Interfaz de usuario de Scratch**.

Nota. Tomado de Scratch Jr. (2021).

La creación de Scratch Jr. viene a llenar el vacío existente en tecnologías poderosas para la creación digital y la programación de computadores en la educación en edades tempranas. Scratch Jr. ofrecerá software tanto para que los infantes creen historias interactivas y animadas como para que los educadores apoyen su adopción mediante su currículo y sus recursos en línea (Flannery et al., 2015).

Se toma como referencia Scratch Jr. ya que fue diseñado para niños entre los 5 y 7 años, teniendo en cuenta las necesidades de aprendizaje y las características de esta etapa desarrollo, favoreciendo acciones exploratorias de programación y solución de problemas

acordes a su edad desde experiencias como contar historias, retos, incidir sobre personajes y escenarios virtuales mediante el "diseño de características del software que apoyen el aprendizaje y la programación, las expectativas apropiadas, generadoras de desarrollo, para el nivel de rasgos físicos, cognitivos, lingüísticos y socio-emocionales de los niños de los primeros grados escolares" (Flannery et al., 2015).

3.5.1 Características y beneficios de Scratch Jr.

- Es una aplicación diseñada acorde a la edad de los participantes de la presente investigación
- Es gratuita.
- No requiere ningún tipo de registro.
- Su interfaz es sencilla y presenta un aspecto visual atractivo.
- Despierta la curiosidad por aprender.
- Potencia la creatividad y la imaginación.
- Ofrece la posibilidad de compartir proyectos entre varios dispositivos.
- No requiere conectividad

3.6 Cuento Interactivo como andamiaje del pensamiento computacional

Dentro de esta investigación, el cuento digital comprende una secuencia narrativa programada, articulada entre personajes, escenarios y eventos, utilizando bloques (tipo LEGO), al tiempo que permite una mejor concepción, representa relaciones de causa-efecto, dentro de estructuras temporales. Así las cosas, la narración se transforma en el andamiaje perfecto para el pensamiento computacional, exigiendo a los niños organizar las acciones de manera lógica, la descomposición de la historia en escenas con sus respectivos personajes y la abstracción de reglas generales de comportamiento a partir de situaciones específicas.

3.7 Juego Interactivo como andamiaje del pensamiento computacional

Se define el juego interactivo en Scratch Jr. como un conjunto de reglas, objetivos y una serie de retroalimentaciones que acompañan la acción del jugador, enmarcada en un espacio digital controlado mediante bloques de programación. En términos de pensamiento computacional, estos juegos requieren que los niños formulen metas claras, diseñen secuencias de acciones para su logro, organizar parámetros (velocidad, repeticiones entre otros) y valorar de manera continua si las estrategias utilizadas permiten alcanzar el objetivo propuesto, lo que fomenta habilidades de descomposición, pensamiento algorítmico y de evaluación-depuración.

4 Metodología

La investigación se desarrolla como un diseño cuasiexperimental, con un enfoque mixto de diseño embebido (Creswell & Plano Clark, 2011). Hernández (2014), citando a Campbell & Stanley (2011), expresa que "Los diseños cuasiexperimentales manipulan deliberadamente al menos una variable independiente para observar su efecto sobre una o más variables dependientes" (p. 152).

4.1 Enfoque de la investigación

El enfoque de la investigación es mixto con diseño embebido, siendo el componente cuantitativo (cuasiexperimental) el primario y el cualitativo el secundario (Creswell & Plano Clark, 2011). El componente cuantitativo utiliza medidas pretest y postest que permitan observar el desarrollo del pensamiento computacional de 54 estudiantes, y para ello se realiza una lista de cotejo de 12 indicadores dicotómicos que se totalizan en una puntuación de 0 a 12 puntos y en un porcentaje de logros. Los indicadores dicotómicos son criterios que sólo toman dos valores: 0 (no logrado) y 1 (logrado), permitiendo un registro objetivo de las conductas observables; el componente cuantitativo se sirve de medidas antes (pretest) y

después (postest) y hace un análisis estadístico mediante pruebas t de Student en jamovi 2.6.4 para realizar una comparación del desarrollo de las habilidades de pensamiento computacional entre cuentos y juegos interactivos en Scratch Jr.

El componente cualitativo da prioridad a una submuestra intencionada de 8 casos, los cuales han de ser analizados en profundidad mediante rúbricas analíticas, esquemas de observación de vídeos grabados de las sesiones en Scratch Jr. El componente cualitativo también analiza las verbalizaciones, las estrategias y las manifestaciones cognitivas de estos 8 estudiantes mediante un análisis temático (Braun & Clarke, 2006), para la identificación de patrones respecto al desarrollo de las habilidades de pensamiento computacional.

4.1.1 Variables de estudio

4.1.1.1 Variable dependiente: Desarrollo de las habilidades del pensamiento computacional, construido a partir de 6 dimensiones: razonamiento lógico, pensamiento algorítmico, descomposición, abstracción, generalización (reconocimiento de patrones) y evaluación (depuración).

4.1.1.2 Variable independiente: tipo de producción creativa con Scratch Jr.: 1) producción de cuentos interactivos y 2) producción de juegos interactivos.

4.1.2 Fases

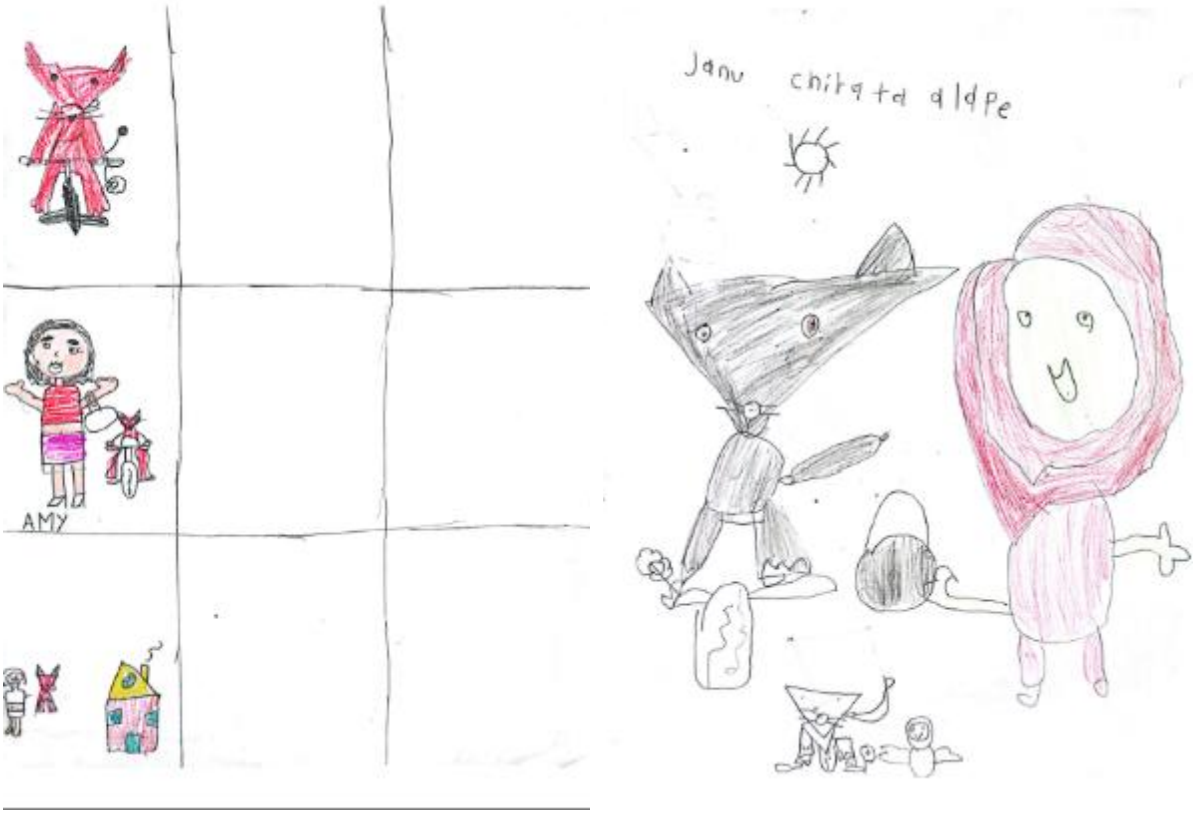
La investigación se orienta a cinco fases plenamente definidas:

4.1.2.1 Fase 1 - Valoración inicial (Semana 1): Aplicación de un pretest mediante la actividad "Robot Humano: Saliendo del Salón" (Versión 1) para evaluar el nivel inicial de las seis habilidades del pensamiento computacional. 1 sesión

4.1.2.2 Fase 2 - Reconocimiento y aprestamiento (Semana 2): Actividades de aprestamiento en direccionalidad, a través de canciones y dinámicas corporales, seguidas de una familiarización con la interfaz de Scratch Jr. y la exploración de los bloques de programación básicos. 1 sesión

4.1.2.3 Fase 3 – Etapa exploratoria actividades desconectadas diferenciadas (Semanas 3-4): Actividades sin tecnología orientadas a desarrollar conceptos de pensamiento computacional. El grupo de cuentos trabaja con secuenciación de narraciones, storyboards y análisis de las estructuras narrativas de cuentos. El grupo de juegos trabaja con algoritmos en cuadrícula. 2 sesiones (Ver Figura 4.1 y 4.2)

Figura 4.1 Storyboards en papel elaborado por una estudiante del GE1

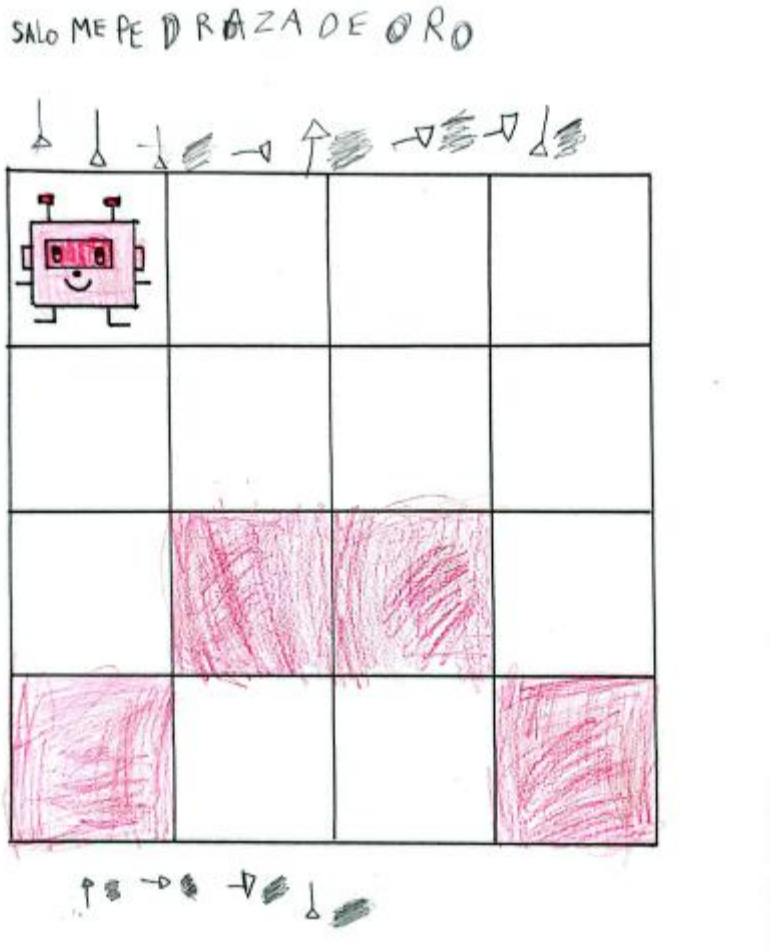


Nota. Ejemplo de planificación analógica previa a la programación en Scratch Jr.

Cada recuadro representa una escena del cuento, ya sea con personajes o con objetos y lugares. Este organizador gráfico sirve de andamiaje para la descomposición en escenas y la posterior secuenciación en bloques de código en ScrathcJr.

Fuente: elaboración propia, 2025.

Figura 4.2 Actividad desconectada "Algoritmo en cuadrícula" producida por un estudiante del GE2



Nota. El estudiante realiza el trazado de flechas direccionales indicando el camino que debe seguir el personaje en la consecución de su meta, eludiendo obstáculos. Esta representación gráfica del algoritmo sobre papel refuerza habilidades de pensamiento algorítmico, secuenciación, descomposición y planificación anticipada, que posteriormente se trasladan a la programación con bloques en las diferentes actividades de Scratch Jr.

Elaboración propia, 2025.

4.1.2.4 Fase 4 – Etapa de experimentación con actividades conectadas y diferenciadas en Scratch Jr. (Semanas 5-6): El grupo de cuentos crea cuentos digitales interactivos, programando personajes, diálogos, escenarios y transiciones narrativas. El grupo de juegos crea juegos digitales interactivos programando los controles, los objetivos, los obstáculos, los enemigos y los niveles de dificultad. Durante dichas sesiones se realizan grabaciones en video y se solicita a los estudiantes que verbalicen el proceso de razonamiento que llevan a cabo 2 sesiones

4.1.2.5 Fase 5 - Evaluación final (Semana 7): Aplicación del postest mediante la actividad "Robot Humano: Saliendo del Salón" (Versión 2, más compleja) para evaluar el desarrollo alcanzado por el alumnado en las seis habilidades de pensamiento computacional. 1 sesión

4.1.2.6 Duración total de la intervención: 7 semanas de intervención pedagógica, durante las cuales se realizaron 7 sesiones de 180 minutos cada una, para un total de 21 horas de trabajo con Scratch Jr.

4.1.2.7 Población

La población total está compuesta por 54 estudiantes de educación inicial del Colegio Nueva Candelaria, ubicado en la localidad de Ciudad Bolívar, Bogotá, D.C., Colombia. Los estudiantes tienen entre 5 y 7 años y pertenecen a los estratos socioeconómicos 1 y 2.

4.1.2.8 Muestra

La muestra es de tipo intencional (no probabilística) y está compuesta por los 54 mismos estudiantes distribuidos en dos grupos experimentales:

- **Grupo Experimental 1 (GE1) - Cuentos:** 27 estudiantes que trabajan en la producción de cuentos interactivos en Scratch Jr.

- **Grupo Experimental 2 (GE2) - Juegos:** 27 estudiantes que trabajan en la producción de juegos interactivos en Scratch Jr.

Los grupos fueron constituidos por la institución educativa, manteniendo la estructura curricular existente (grupos intactos).

4.1.2.8.1 Submuestra cualitativa: se seleccionaron 8 casos para el análisis cualitativo profundo (4 de cada grupo experimental), siguiendo criterios de variabilidad en los puntajes del pretest, de disponibilidad de registros completos en video y de riqueza de las verbalizaciones durante las actividades.

4.1.2.9 Trabajo de campo

El trabajo de campo se articula en 7 sesiones durante 7 semanas consecutivas (octubre a noviembre de 2025), enmarcado en las Fases 2, 3 y 4 de la intervención (aprestamiento, actividades desconectadas y actividades conectadas). Cada sesión de trabajo tiene un límite de 180 minutos (3 horas) y siguiendo una secuencia pedagógica que combina:

1. Introducción y activación de conocimientos previos (20-30 minutos)

2. Desarrollo de la actividad principal con acompañamiento docente (120-140 minutos)
3. Reflexión, socialización y cierre (20-30 minutos)

En la Fase 2 (aprestamiento), ambos grupos trabajan juntos en las mismas actividades de familiarización con Scratch Jr. A partir de la Fase 3, los grupos se separan y cada uno desarrolla un conjunto de actividades específicas según el tipo de producción asignado (cuentos o juegos). Las actividades desconectadas (Fase 3) pretenden la construcción de la comprensión conceptual de las estructuras de los patrones de pensamiento computacional sin ningún tipo de distracciones tecnológicas: el grupo de cuentos debe trabajar la estructura narrativa (inicio-nudo-desenlace) y la secuenciación temporal, el grupo de juegos trabajará las reglas y las condicionales, las mecánicas de juego y la resolución de problemas mediante algoritmos.

Las actividades conectadas (Fase 4) permiten transferir esos conceptos al entorno digital de Scratch Jr., donde los estudiantes pueden programar sus propias creaciones. En el transcurso de estas sesiones, se graba en video a los estudiantes de la submuestra cualitativa y se les pide que expliquen su forma de razonar durante el trabajo ("¿Qué estás haciendo?", "¿Por qué has seleccionado ese bloque?", "¿Qué crees que pasará?").

4.1.3 Instrumentos de recolección de información

La presente investigación hace uso de instrumentos diferenciados en función del componente metodológico (cuantitativo vs. cualitativo) así como de la población evaluada (N=54 vs. n=8):

Instrumentos cuantitativos (N=54 estudiantes)

1. **Lista de cotejo de habilidades de pensamiento computacional** (Anexo 1)

Instrumento cuantitativo aplicado a los 54 estudiantes (27 GE1 + 27 GE2) en la valoración pretest (semana 1) y postest (semana 7) durante la actividad desconectada "Robot Humano".

Consta de 12 indicadores de observación dicotómica (0/1)—dos por cada una de las seis habilidades de PC: razonamiento lógico, pensamiento algorítmico, descomposición, abstracto, patrones y evaluación/depuración.

Funcionalidad Operativa:

1. Registro en tiempo real (de 2 a 3 minutos por estudiante) por el observador externo
2. Puntuación total: 0-12 por estudiante convertible en porcentaje (0-100%)
3. Campo de observaciones: para registrar notas de campo breves sobre estrategias cognitivas, verbalizaciones y acciones relevantes
4. Propósito: Medida estandarizada para poder realizar un análisis estadístico comparativo pretest-postest (prueba t para muestras relacionadas)

Los 12 indicadores de la lista de cotejo se planearon a partir de la derivación de las seis habilidades de pensamiento computacional descritas en el marco teórico (razonamiento lógico, pensamiento algorítmico, descomposición, abstracción, patrones y generalización, evaluación y depuración), estableciendo 2 ítems observables para cada una de ellas. La fabricación de estos indicadores se apoyó en las rúbricas e instrumentos implementados en estudios sobre pensamiento computacional de educación infantil asistido con programación y robótica educativa (Caballero y García, 2019, 2021; Delgado y Prado, 2018, entre otros), ajustándolos al ambiente de Scratch Jr. y al grupo de educación inicial planteada en esta investigación; cada ítem se calificó de forma dicotómica (0 = no se observa la conducta, 1 = se observa la conducta) y la fiabilidad del instrumento, se estimó usando el coeficiente alfa de Cronbach, arrojando un resultado bajo en el pretest y logrando un valor aceptable en el postest ($\alpha \approx 0{,}70$), lo que sustenta su confiabilidad al final del estudio, mostrando un menor equilibrio de los valores iniciales.

Justificación metodológica:

La lista de cotejo permite un registro rápido y estandarizado de conductas observables de los 54 estudiantes, así como obtener datos cuantitativos comparables, sin la complejidad operativa y cognitiva que conlleva aplicar una rúbrica analítica en escala ordinal en tiempo real a toda la muestra.

Instrumentos cualitativos: (n=8 estudiantes -submuestra)

2. **Rúbrica analítica para la observación cualitativa del pensamiento computacional (Tabla 4)**

Es un instrumento cualitativo aplicado posteriormente a la submuestra de 8 estudiantes (4 de cada grupo experimental), mediante un análisis amplio de las videograbaciones obtenidas y de las transcripciones realizadas de las sesiones de trabajo (Semanas 1-7: pretest, actividades desconectadas, actividades conectadas en Scratch Jr. y postest).

Descripción detallada:

- Evalúa los diferentes niveles de ejecución de cada una de las 6 habilidades de PC: razonamiento lógico, pensamiento algorítmico, descomposición, abstracción, patrones y evaluación/depuración)
- Es de tipo ordinal, del 0 al 5, con descriptores cualitativos detallados para cada nivel de desempeño.
- Se compone de las siguientes columnas: Habilidad de PC | Características | Acciones observables | Escala valorativa (0-5).

Aplicación:

El análisis de los vídeos transcritos se hace en retrospectiva (no se realiza en tiempo real. El investigador realiza una revisión de los vídeos transcritos, identifica episodios significativos, extrae evidencias verbales y conductuales, y asigna niveles de desempeño justificadamente), para realizar una posterior triangulación de dichos videos y los productos digitales creados por los alumnos en Scratch Jr.+ Notas de observación de docente

Intención:

Profundización cualitativa de caracterización de procesos cognitivos, estrategias de resolución de problemas, patrones de verbalización y desarrollo del pensamiento computacional a lo largo de la intervención

Justificación metodológica:

La rúbrica analítica con escala 0-5 requiere juicios interpretativos complejos a partir de múltiples fuentes de evidencia, lo cual, desde el punto de vista metodológico, sólo es factible mediante un análisis retrospectivo pormenorizado de una submuestra pequeña, no mediante el registro en tiempo real de 54 estudiantes.

3. **Videograbaciones (N = 54 estudiantes; análisis en profundidad: n = 8).**

Registro audiovisual completo de los turnos individuales de los 54 estudiantes en el pretest (Semana 1) y en las sesiones de trabajo con Scratch Jr. (Semanas 2-7).

Doble objetivo:

- Validación de datos cuantitativos: Comprobación posterior de la adecuación de los registros en la lista de cotejo (N=54)
- Base para análisis cualitativo: Datos primarios para la transcripción literal, la codificación por temas y la aplicación de la rúbrica analítica (n=8)

4. **Transcripciones**

Registro de lo que los estudiantes de la submuestra (n=8) dicen de manera espontánea mientras trabajan en las actividades, incluyendo la petición de que expliquen qué hacen, el porqué de sus decisiones y qué es lo que esperan que ocurra, capturando así sus procesos de razonamiento explícito.

5. **Análisis de productos digitales en Scratch Jr.**

Revisión sistemática de los proyectos (cuentos o juegos) generados por los estudiantes de la submuestra (n=8): bloques de programación utilizados, estructura de las secuencias, estrategias de depuración que han implementado, grado de complejidad alcanzado.

6. **Notas de observación de docente**

Registro sistemático del docente investigador sobre comportamientos, estrategias, dificultades y logros observados en las sesiones, a través de una planilla estructurada con campos para cada habilidad de PC.

7. **Pictogramas (Figura 3)**

Representaciones visuales de las instrucciones y secuencias de programación utilizadas en las actividades desconectadas para facilitar la comprensión de conceptos abstractos a partir de símbolos concretos.

Tabla 3. Rúbrica analítica de observación cualitativa del PC

Habilidad de PC	Características	Acción ejecutar a través de las fichas o bloques de programación	Marque con una X la escala valorativa				
			1	2	3	4	5
Razonamiento lógico	El estudiante establece relaciones de causa-efecto entre las instrucciones que propone (ya sean verbales, mediante fichas físicas o a partir de bloques de programación) y los resultados esperados o ejecutados, mostrando conocimientos de las funcionalidades de cada instrucción para poder solucionar el problema.	1. Relaciona las funciones de las fichas gráficas de programación (o de los bloques digitales) con la ejecución de una instrucción específica. 2. Diferencia las funciones particulares de cada ficha de programación. 3. Anticipa el resultado que producirá una secuencia de instrucciones antes de ejecutarla. 4. Identifica las contradicciones lógicas entre lo que planea hacer y lo que es posible hacer. 5. Relaciona la noción de tiempo-espacio con las fichas de programación correspondientes.					
Pensamiento algorítmico	Aplica de manera repetitiva un algoritmo ante un estímulo determinado, desarrollando secuencias ordenadas de pasos (verbales, con fichas físicas o usando bloques de programación) para resolver un problema.	1. Elabora secuencias ordenadas de instrucciones entendiendo direccionalidad, cantidad de acciones y el orden correcto. 2. Verbaliza la secuencia que necesita para resolver un problema 3. Comprueba el orden de las instrucciones para verificar el funcionamiento de un algoritmo 4. Reusa algoritmos que le funcionaron ante problemas similares					

Descomposición	Reconoce las partes que conforman el proyecto (personajes, escenarios, acciones, secuencias) y construye una representación, dividiéndolo en partes que entiende según la meta que quiere alcanzar	1. Entiende las funciones de cada comando (ficha direccional o bloque de programación). 2. Separa y utiliza las fichas según la instrucción que desea desarrollar. 3. Divide en pasos más pequeños problemas elaborados 4. Elabora bloques de instrucciones por separado y luego los integra
Abstracción	Representa conceptos y problemas por medio de símbolos visuales (fichas, bloques, pictogramas), identificando lo que es esencial y omitiendo el resto para acceder al aprendizaje y a la solución de un problema.	1. Representa los pasos necesarios para ir a un lugar determinado por medio de fichas o bloques. 2. Representa las instrucciones complejas mediante símbolos visuales (flechas, números, color de bloques). 3. Reconoce algunos patrones visuales y los relaciona con funciones específicas. 4. Descarta aquello que no es necesario para explicar su solución, centrándose en lo más esencial. 5. Transfiere representaciones de un ámbito a otro (ej: de tableros de fichas físicas a tableros de bloques digitales).
Patrones	Identifica y usa patrones (estructuras repetitivas o similitudes entre problemas) trasladando la solución de problemas pasados a nuevos contextos.	1. Reconoce estructuras repetitivas y elige la repetición intencionadamente en acciones de programación. 2. Detecta similitudes entre un nuevo problema y otro ya solucionado que

		le permiten replicar la solución. 3. Relaciona cantidad con repetición: sabe que la misma cantidad de veces que se repite una ficha es la misma cantidad de veces que se ejecuta la acción. 4. Transfiere patrones con solución acertada a otros contextos
Evaluación	Evalúa un problema o estímulo y aplica acciones de programación según sus experiencias previas. Cuando no consigue realizar correctamente la acción deseada revisa la secuencia realizada y corrige el error sustituido, añadiendo o eliminando comandos que intervenían para conseguir la meta.	1. Usa las fichas de programación con un número determinado de acciones; cambia los comandos de acuerdo con la necesidad. 2. Detecta errores al observar que el resultado no es el esperado 3. Menciona el error detectado (ej: "se movió demasiado", "no apareció"). 4. Corrige sistemáticamente: sustituye, añade o elimina comandos que interfieren con la meta. 5. Verifica la corrección volviendo a ejecutar de nuevo la secuencia.

Nota: Autoría propia (2026)

4.1.4 Descripción de los casos

4.1.4.1 Cuantitativo:

Se llevará a cabo un análisis estadístico mediante pruebas t de Student para contrastar el desarrollo del pensamiento computacional entre el grupo de relatos (GE1) y el grupo de juegos (GE2). En concreto se utilizará:

- Prueba t para muestras relacionadas: para comparar el pretest y el postest dentro de cada grupo (GE1 y GE2) y determinar si las mejoras que se observan son estadísticamente significativas.
- Prueba t para muestras independientes: para comparar las ganancias (postest − pretest) entre GE1 y GE2 y observar si son significativas estas diferencias en la efectividad de ambos tipos de desarrollo.

Se comprobarán previamente los siguientes supuestos estadísticos para las pruebas t:

Normalidad en la distribución de los residuos (prueba de Shapiro-Wilk)

Homogeneidad de varianzas entre grupos (prueba de Levene)

Se utilizará un nivel de significación $\alpha = 0,05$ en todas las pruebas estadísticas; se llevarán a cabo en el software jamovi 2.6.4; se presentarán datos descriptivos (medias, desviaciones estándar), el valor de t, el valor de p y el tamaño del efecto (d de Cohen).

4.1.4.2 Cualitativo:

Se aplicará una estrategia específica de análisis temático (Braun & Clarke, 2006) a las transcripciones de los vídeos de la submuestra cualitativa. Para ello, el proceso de análisis es el siguiente:

1. Familiarizarse con los datos: ver los vídeos en múltiples ocasiones y leer las transcripciones completas.
2. Generar códigos iniciales: Realizar la codificación descriptiva de episodios en los que se manifiesten las seis habilidades de pensamiento computacional: razonamiento lógico, pensamiento algorítmico, descomposición, abstracción, patrones (generalización), evaluación (depuración).
3. Buscar temas: Agrupar códigos en patrones más extensos relacionados con estrategias cognitivas, procesos de razonamiento y manifestaciones de pensamiento computacional.
4. Revisar los temas: verificar que los temas identificados concuerden con los datos y que los representen adecuadamente en relación con las manifestaciones observadas.
5. Definición y dominio de los temas: Presentación al detalle de cada tema con ejemplos de episodios significativos.
6. Elaboración del informe: redacción de un análisis cualitativo con citas textuales y descripciones de episodios que ejemplifiquen los resultados.

Los productos en Scratch Jr. se analizarán a partir de la inspección del código para determinar: a) los tipos de bloques utilizados, b) la complejidad de las secuencias programadas, c) el uso de las estructuras de control (bucles, condicionales), d) la existencia o no de pruebas de depuración y de corrección, y e) el grado de integración de los distintos elementos (personajes, escenarios, interacciones). Esta inspección se llevará a cabo

manualmente, tomando capturas de pantalla del código y reproduciendo algunos proyectos guardados.

4.1.4.3 Integración de los resultados cualitativos y cuantitativos:

Los resultados cualitativos y cuantitativos se integrarán mediante la triangulación interpretativa. Los resultados de las pruebas t de Student indicarán si hay una diferencia significativa entre los grupos (cuentos vs. juegos) y en qué medida la actividad de "cuentos" facilita el desarrollo del pensamiento computacional. Los hallazgos cualitativos, por su parte, permitirán explicar cómo se manifiestan las habilidades de pensamiento computacional en cada tipo de actividad, incorporando las estrategias y los procesos cognitivos que más y mejor se emplean en cada tipo de producción creativa. La integración de las dos formas de evaluación no solo sirve para determinar si hay diferencia, sino que también da respuesta a la explicación del porqué y del cómo aparecen, así como a qué características tiene cada uno de los tipos de producción creativa (narrativa y lúdica), orientados a favorecer el desarrollo de unas u otras habilidades de pensamiento computacional.

4.1.5 Hipótesis:

Hipótesis nula (H0): Diferencias significativas nulas en las ganancias del pensamiento computacional entre los niños que elaboran cuentos interactivos y los que crean juegos interactivos en Scratch Jr.

Hipótesis alternativa (H1): Existen diferencias significativas en las ganancias del pensamiento computacional entre los niños que elaboran cuentos interactivos y los que crean juegos interactivos en Scratch Jr.

4.2 Sesiones de trabajo en Scratch Jr.

Aquí se presentan las 7 sesiones de trabajo de 180 minutos cada una (21 horas en total, desarrolladas en 7 semanas) llevadas a cabo con las 54 niñas y niños de educación inicial del colegio Nueva Candelaria y encaminadas a la creación de cuentos y juegos interactivos en la aplicación Scratch Jr., a partir de los cuales se analizaron los efectos del proceso iniciado en el desarrollo del pensamiento computacional. Las sesiones se desarrollaron en 3 etapas: reconocimiento, exploración y experimentación, con una progresión pedagógica desde la familiarización hasta la creación autónoma. Cabe señalar que las actividades estaban descritas, en su mayoría, gráficamente mediante pictogramas, dado que, en la etapa escolar en la que se encuentran los niños, su proceso lectoescritor aún no se ha consolidado.

4.2.1 Etapa de reconocimiento

4.2.1.1 Sesión 1: Pretest

Nombre: Juego del Robot Humano – Salir del Salón

Propósito: Evaluar las capacidades de pensamiento computacional de los niños de Educación Inicial del colegio Nueva Candelaria, que se utilizarán como pretest de la presente investigación mediante una actividad de programación desconectada.

Tiempo: 180 minutos.

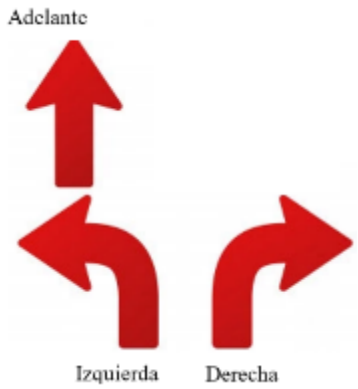
Descripción de las actividades:

Acercamiento:

Para familiarizar a los niños con el tema de programación, se decide utilizar una actividad lúdica denominada "Robot Humano: ¡Saliendo del Salón!" a partir de la cual intentaremos dar instrucciones precisas para utilizar los comandos de movimiento.

Se les presenta a los niños un conjunto de fichas de programación (tarjetas con pictogramas) que simbolizan los comandos básicos de movimiento: Adelante (avanzar un paso), Gire a la derecha (gira 90° a la derecha) y Gire a la izquierda (gira 90° a la izquierda). El profesor explica el funcionamiento de cada ficha e ilustra el tema con algunos ejemplos muy simples: "Yo quiero que una persona camine tres pasos; por eso, uso tres fichas de 'adelante'. Si quiero que gire, utilizo la ficha de 'gire a la derecha' o 'gire a la izquierda', etcétera". Se lleva a cabo un conversatorio inicial con los niños sobre qué es la programación y cómo las instrucciones ordenadas (algoritmos) permiten a una persona realizar una tarea específica (Figuras. 4.3 y 4.4).

Figura 4.1 Fichas de programación utilizadas en el pretest.



Se realizan ejercicios básicos de programación simbólica con todo el grupo, utilizando preguntas orientadoras:

- "¿Qué ficha utilizarías para hacer que tu compañero avance?"
- "¿Cuántas veces tienes que repetir la ficha “adelante” para que avance 4 pasos?"
- "Si está mirando a la puerta y tiene que girar hacia la ventana, ¿qué ficha utiliza?"

Los estudiantes se agrupan en parejas: uno da las instrucciones verbales ("adelante, adelante, gira a la derecha") y el otro las ejecuta como si fuera un robot.

Figura 4.4 Actividad Robot Humano

Nota. Elaboración propia, 2025. Durante el pretest, el niño organiza la secuencia de tarjetas con los pictogramas direccionales (hacia adelante, gira a la derecha, gira a la izquierda) para guiar al docente-robot desde la situación inicial hasta la puerta del aula.

Actividad central

El juego de roles está orientado de tal forma que el docente llega a convertirse en un robot humano que seguirá únicamente las instrucciones que cada niño le dé para intentar salir de la clase.

Procedimiento:

1. El docente adopta una posición inicial determinada en el salón (por ejemplo, a favor del tablero).
2. Cada uno de los niños debe crear, individualmente, un algoritmo (una secuencia de instrucciones en orden) usando las fichas de programación para que el docente avance desde la posición inicial hasta la puerta de salida del aula.

3. El niño verbaliza las instrucciones una a una: "Adelante, adelante, gira a la derecha, adelante, adelante, adelante.
4. El docente sigue exactamente las instrucciones que le indican; es un robot que solo puede hacer aquello que le programa (no puede añadir pasos ni mucho menos corregir errores).
5. Si el algoritmo con el que trabaja el estudiante presenta errores (por ejemplo, si al ejecutar el algoritmo el profesor choca con una mesa o no llega a la puerta), el niño debe identificar dicho error, comprobar la secuencia de fichas y corregir el algoritmo (depuración).
6. Este proceso se repite hasta que el niño consiga elaborar un algoritmo que alcance la meta de programar al profesor para que salga del salón.

Observación y Registro

Organización Operativa del pretest

Dado que la actividad sobre el "Robot Humano" tiene carácter individual (cada estudiante tiene asignado un turno de 10-15 minutos para construir su algoritmo, mientras el resto de los estudiantes realiza actividades orientadas por el docente titular del grupo), el pretest se llevó a cabo durante una semana completa (octubre de 2025) a partir de ocho subsesiones como se muestra en la siguiente tabla, una por cada uno de los grupos:

- GE1 (Cuentos – 27 estudiantes): Valorados a través de 4 subsesiones (lunes-jueves por la mañana) en grupos de 13-14 estudiantes por sesión
- GE2 (Juegos – 27 estudiantes): Valorados a través de 4 subsesiones (martes-viernes por la tarde) en grupos de 13-14 estudiantes por sesión

Esta organización permitió destinar un tiempo de 10-15 minutos de evaluación a cada uno de los 54 estudiantes.

Registro a través de lista de cotejo y videograbación

Mientras que los estudiantes desarrollan la actividad, el docente investigador (con la ayuda de otra persona externa que actúa como observador externo) registra de forma sistemática el desempeño de cada uno de los 54 estudiantes (subgrupos de 13-14 respectivamente) a través de dos estrategias:

1. Lista de cotejo en tiempo real:

El observador externo registra el desempeño a través de la lista de cotejo de pensamiento computacional (Anexo 1) mediante un instrumento cuantitativo que dispone de 12 indicadores dicotómicos (0/1) que registran (2-3 minutos por estudiante), las siguientes capacidades:

- **Razonamiento lógico:** ¿Verbaliza o demuestra comprensión de la relación entre la funcionalidad de las fichas y las acciones que se han de llevar a cabo?
- **Pensamiento algorítmico:** ¿Crea secuencias de fichas ordenadas a partir del problema?
- **Descomposición:** ¿Sabe identificar las características de cada ficha y aplicarlas de forma correcta?
- **Abstracción**: ¿Es capaz de analizar el problema y de aplicar el correspondiente algoritmo?
- **Patrones**: ¿Es capaz de repetir intencionadamente los comandos en caso de ser necesario?
- **Evaluación**: ¿Es capaz de identificar los errores y de corregir su algoritmo si este no funciona?

Cada indicador se visualiza con 0 (no logrado) o 1 (logrado), obteniendo un puntaje mínimo de 0 y un máximo de 12 puntos para cada uno de los 54 estudiantes (27 GE1 + 27

GE2), para acabar convirtiéndolo en un porcentaje (0-100%) en el análisis estadístico comparativo pretest-postest.

La lista de cotejo incluye un espacio para observaciones, donde se documenta una breve nota de campo que dará cuenta de: estrategias cognitivas observadas (ensayo-error iterativo, razonamiento anticipatorio, verbalización del proceso), calidad y riqueza de las verbalizaciones (análisis detallados, preguntas reflexivas, hipótesis), y/o comportamientos relevantes (persistencia, tipo de ayuda, manifestaciones emocionales).Las notas de campo dentro del instrumento sirven como indicadores que apoyan la selección de la submuestra cualitativa, y de esta manera detectar estudiantes con riqueza de verbalizaciones y con diversidad de estrategias sin importar el nivel de desempeño inicial.

2. **Videograbación de todos los sujetos de la muestra (N=54):**

Al tiempo de registrar la lista de cotejo, se desarrolla la videograbación de cada turno individual de los 54 estudiantes durante las ocho subsesiones del pretest. Las grabaciones recogen:

- Verbalizaciones elaboradas del alumno (construcción del algoritmo y explicación de instrucciones)
- Acciones observables (organización de las fichas, gestos, expresiones faciales de ser posible)
- Ejecución del docente-robot siguiendo las instrucciones
- Reacciones emocionales del alumno ante errores o logros en la depuración

Doble función de la videograbación:

- **Validación cualitativa**: El video permite comprobar que los datos de la lista de cotejo son correctos, permitiendo comprobar y determinar si el estudiante realmente realizó las acciones observables en el contexto correcto
- **Base para el análisis cualitativo**: Los vídeos de los 8 alumnos seleccionados para la submuestra cualitativa (ver apartado 5.6.2) constituyen el registro de datos cualitativos primarios que posteriormente serán analizados mediante la transcripción, la codificación temática y la aplicación de la rúbrica analítica.

Cierre

Al finalizar las cuatro subsesiones del pretest de cada grupo, se promueve la creación de un espacio de reunión reflexiva independiente para cada curso a la finalización de la semana (viernes, semana 1):

- 9:30-10:30 AM: Asamblea reflexiva con GE1 (27 estudiantes del grupo de cuentos interactivos)
- 11:00-12:00 PM: Asamblea reflexiva con GE2 (27 estudiantes del grupo de juegos interactivos)

En la reunión, los estudiantes comparten experiencias:

- "¿Qué fue lo más difícil al darle las instrucciones (programar) al profe?"
- "¿Qué sucedió cuando la instrucción no estaba en el orden correcto?"
- "¿Cómo sabías que tenías que cambiar algo en tu instrucción?"

El docente investigador reflexiona con cada grupo acerca de la importancia de dar instrucciones correctas y en orden, construyendo un puente conceptual con las fases posteriores: "El proceso de programación que realizamos con las fichas es muy parecido a lo

que vamos a hacer después en Scratch Jr. con los personajes digitales: tienen que pensar bien el orden de las instrucciones y revisar los errores (depuración)."

Materiales

- Fichas de programación (tarjetas plastificadas con pictogramas: adelante, derecha, izquierda)
- Lista de cotejo de pensamiento computacional (Anexo 1) – instrumento cuantitativo con campo de observaciones
- Cámara de video para grabar a los 54 estudiantes durante el pretest
- Planilla de observación para el docente investigador

Nota metodológica aclaratoria de Organización operacional: El pretest se aplicó durante una semana completa, con 8 subsesiones distribuidas entre GE1 (lunes y jueves, 13-14 estudiantes) y GE2 (martes y viernes, 13-14 estudiantes).

Sesión 2: Reconocimiento de Scratch Jr.

Propósito: Favorecer acciones de reconocimiento de las funcionalidades de los bloques de programación de Scratch Jr.

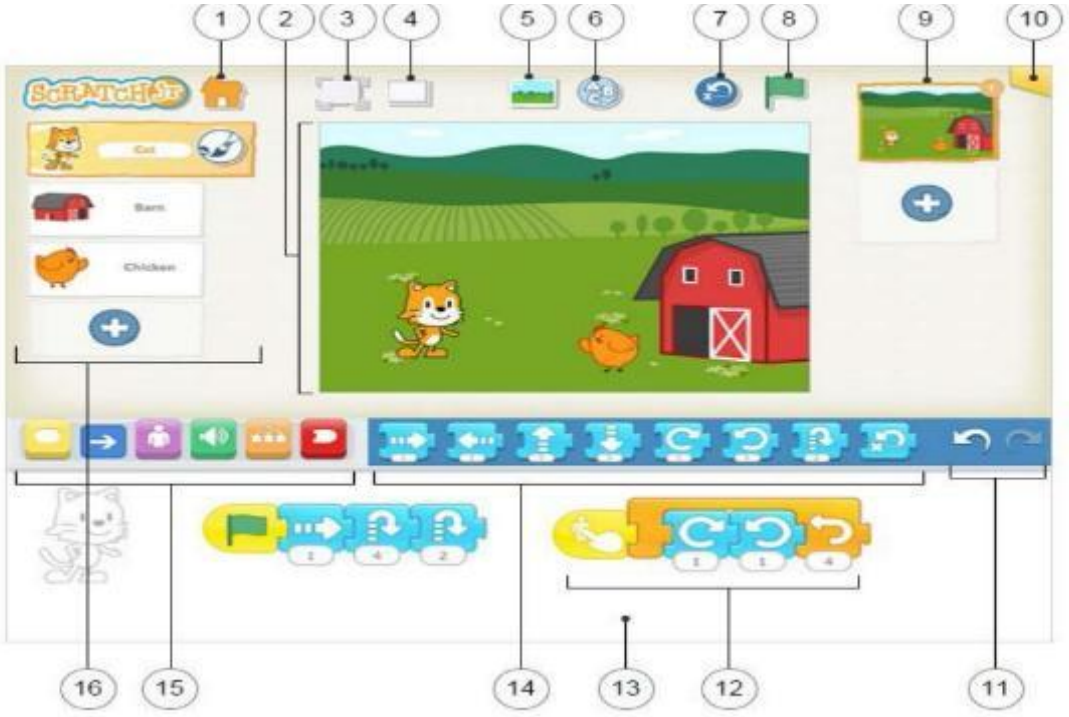
Tiempo: 180 minutos.

Descripción de las actividades:

Como medida para que los niños se familiaricen con los diferentes iconos y funciones de Scratch Jr., se compartirá un documento elaborado por el Grupo de investigación DevTech (Universidad de Tufts), el Grupo Lifelong Kindergarten (MIT Media Lab) y la compañía Playful Invention, en el que se describen las funcionalidades de cada bloque. (DevTech Research Group, Lifelong Kindergarten Group & Playful Invention Company, 2015). Además, se comparte con los estudiantes un pictograma que describe cada una de las herramientas de Scratch Jr.

1. Guardar el proyecto
2. Escenario: Lugar donde los personajes cobran vida
3. Ver pantalla completa
4. Cuadrícula: Ayuda a distribuir objetos
5. Cambiar fondo
6. Añadir texto
7. Establece o reestablece la posición de los objetos
8. Bandera verde: Inicio de los scripts de programación del bloque
9. Selección de páginas: permite añadir o eliminar páginas.
10. Permite cambiar el nombre del proyecto
11. Permite rehacer o deshacer un error
12. Área de programación donde se arrastran los bloques que serán las instrucciones para dar a los objetos. Al presionar el bloque, se ejecuta la secuencia de comandos. Permite eliminar bloques o secuencias de comandos.
13. Área de programación donde se encuentran los bloques que contienen las instrucciones que se le darán a cada objeto.
14. Paleta de bloques: Son los bloques de programación que pueden ser arrastrados para dar una instrucción a un objeto
15. Categorías de bloques: amarillos (bloques de activación), azul (movimiento), púrpura (visualización), verde (sonidos), naranja (control) y rojo (bloques de final)
16. Objetos y personajes: se seleccionan o se añaden personajes; se pueden editar y dar instrucciones de programación.

Figura 4.5**. Descripción de las funciones de Scratch Jr.**



Nota: Tomado de código 21 (sf).

4.2.2 Etapa exploratoria

Se caracteriza por una serie de sesiones en las que los niños de transición ejecutarán diferentes acciones de programación siguiendo las instrucciones del docente y los pictogramas, asociando las funcionalidades de los diversos bloques de programación para lograr un propósito o solucionar un problema.

4.2.2.1 Sesión 3. Creando espacios, personajes y acciones

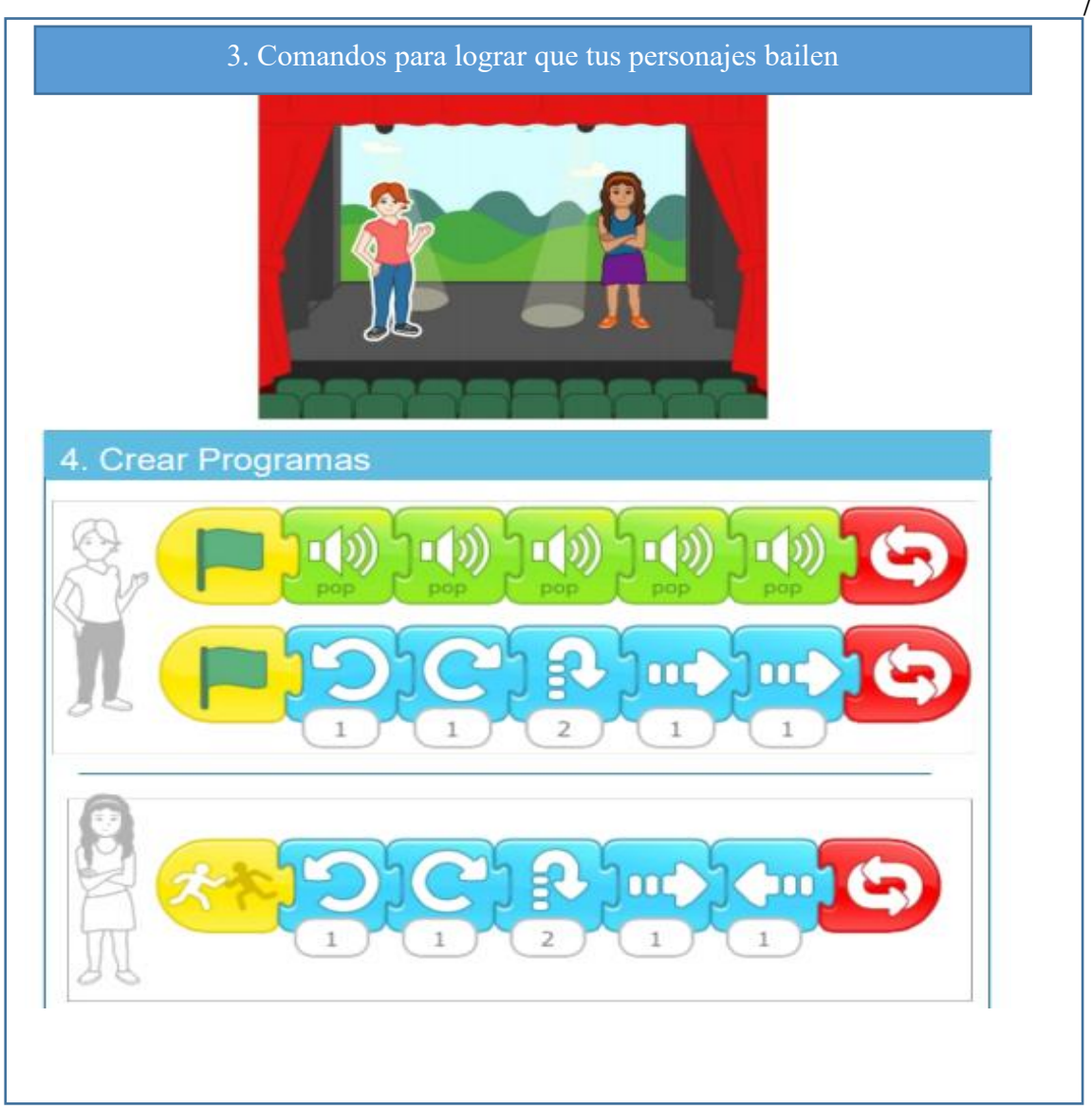
Propósito: Propiciar acciones de programación en las que los niños de educación inicial exploren los diferentes bloques de programación para ejecutar acciones en los distintos personajes, siguiendo las instrucciones y comandos dados.

Tiempo: 180 minutos.

Descripción de las actividades:

Esta actividad tiene como propósito que los niños creen fondos y personajes, y generen acciones básicas de movimiento para ellos. Para ello, se compartirán los siguientes pictogramas para que ejecuten cada una de las instrucciones y los comandos.

Figura 4.6. Explorando fondos, personajes y movimientos



Nota: Adaptada de DevTech Research Group, Lifelong Kindergarten Group & Playful Invention Company (2015).

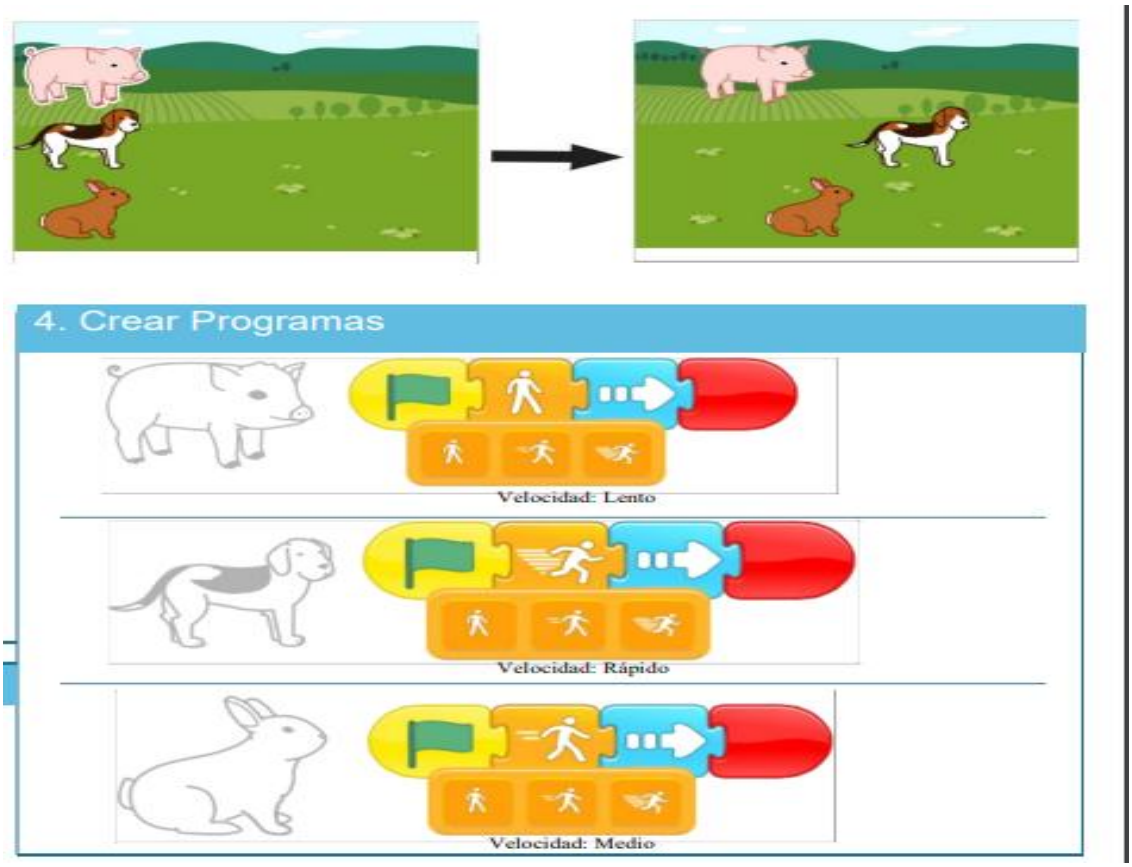
4.2.2.2 Sesión 4. Ejecutando acciones de programación

Propósito: Propiciar acciones de programación básicas en las que los niños realicen procesos de abstracción y repetición para lograr una finalidad.

Tiempo: 60 minutos

Descripción de las actividades: Seguir las indicaciones del pictograma propuesto para programar una carrera entre los personajes, seleccionando la velocidad y el ganador a partir de la ejecución de comandos de movimiento.

Figura 4.7. Programando una carrera entre mis personajes



Nota: Tomada de DevTech Research Group, Lifelong Kindergarten Group & Playful Invention Company (2015).

Nota metodológica: he de decir que las Sesiones 5 y 6 (Fase 4 - Etapa de experimentación; Semanas 5-6) se desarrollaron de forma diferenciada para cada grupo experimental: el Grupo de Cuentos (GE1) llevó a cabo la creación de cuentos interactivos complejos con muchos personajes, escenarios y transiciones de las narraciones, y el Grupo de Juegos (G2) desarrolló juegos interactivos con controles, objetivos, obstáculos y niveles de dificultad. Las sesiones mantuvieron la misma estructura pedagógica que las anteriores (presentación del reto, exploración, creación, verbalización y depuración) y fueron videograbadas para el análisis cualitativo de los 8 casos de la submuestra. La Sesión 7 (Semana 7) fue el postest, donde se puso en práctica la lista de cotejo de pensamiento

computacional con los 54 estudiantes a través de una actividad parecida a la del pretest, pero de mayor complejidad.

4.2.3 Etapa de experimentación

En esta etapa, los niños, de manera autónoma, diseñarán acciones de programación, creando sus propios cuentos y juegos interactivos, y aplicando los conocimientos adquiridos sobre cada una de las funcionalidades y los comandos de los bloques de Scratch Jr. Se proyecta que los niños pongan en práctica las habilidades de pensamiento computacional y creen sus propios escenarios, personajes, diálogos y acciones, entre otros.

4.2.4 Consideraciones éticas

La investigación fue aprobada por las directivas de la institución. Se obtuvieron los consentimientos informados escritos de los padres o tutores de todos los participantes del estudio expuestos, en donde se detallan los propósitos de la recopilación de datos de la videograbación, así como el uso académico que se daría a los datos y a las transcripciones. Además, se garantizó la confidencialidad de los participantes asignándoles un código alfanumérico (GE1EST001, GE1EST003, etc.) tanto en los informes como en los vídeos (Mason et al., 2015). Adicionalmente, se guardaron los vídeos y las transcripciones en lugares seguros a los que solo podía acceder el investigador

5 Análisis Cuantitativo de Resultados

En este capítulo se presenta el análisis cuantitativo de la intervención educativa con Scratch Jr., llevada a cabo durante siete semanas con una muestra de 54 estudiantes de grado de transición (edad: 5-6 años). En él se dará cuenta de la forma en la que la creación de cuentos y juegos interactivos en Scratch Jr. permite el avance en el desarrollo de las habilidades de pensamiento computacional en los niños de educación inicial del colegio Nueva Candelaria, buscando dar respuesta a dicha pregunta.

El capítulo se articula en ocho apartados, comenzando por describir la muestra (5.1), posteriormente se analizan los resultados del pretest (5.2) y del postest (5.3), en el apartado 5.4 se presenta la comparación estadística entre ambos momentos mediante prueba t para muestras relacionadas, el apartado 5.5 se centra en el análisis de cada habilidad que compone el pensamiento computacional, los apartados 5.6 y 5.7 dan cuenta y caracterizan la sub-muestra cualitativa seleccionada (n = 8) y finalmente, el apartado 5.8 realiza una integración entre los resultados cuantitativos e introduce el análisis cualitativo del Capítulo 6.

5.1 Características de la muestra

La muestra estuvo compuesta por 54 estudiantes de grado transición de un colegio privado de la localidad de Ciudad Bolívar en Bogotá, Colombia. La asignación de participantes fue de forma aleatoria en dos grupos experimentales:

- Grupo Experimental 1 (GE1): 27 estudiantes que crearon cuentos digitales interactivos
- Grupo Experimental 2 (GE2): 27 estudiantes que hicieron juegos digitales.

La composición de la muestra fue homogénea en relación con el género: 28 niñas (51,9 %) y 26 niños (48,1 %) con un rango de edad de 5 a 6 años.

Ningún participante tenía experiencia previa en programación ni en el uso de herramientas de pensamiento computacional.

La intervención se desarrolló a lo largo de siete semanas consecutivas (octubre-noviembre de 2025), en una sesión semanal de 180 minutos cada una, para un total de siete sesiones por estudiante. En ambas condiciones experimentales se utilizó Scratch Jr. en tabletas por grupo y la instrucción fue acompañada por los maestros titulares de grado.

Tabla 5.1. Características demográficas de la muestra (N = 54)

Variable	GE1 (Cuentos)	GE2 (Juegos)	Total
N estudiantes	27	27	54
Edad (años)	5: n=14; 6: n=13	5: n=15; 6: n=12	5: n=29; 6: n=25
Rango de edad	5-6 años	5-6 años	5-6 años
Género (niñas/niños)	14/13	14/13	28/26
Experiencia previa en programación	0	0%	0%

Nota. GE1 = Grupo Experimental 1 (cuentos); GE2 = Grupo Experimental 2 (juegos).

5.2 Resultados del pretest

5.2.1 Estadísticos descriptivos

La tarea pretest "Robot Humano: Saliendo del salón" (Sesión 1, semana 1) consistió en una actividad de programación desconectada donde cada estudiante debía construir un algoritmo a partir de las fichas de programación (tarjetas con pictogramas de "adelante", "gira derecha", "gira izquierda") para guiar al docente robot desde una posición de inicio hasta la puerta del salón. El docente ejecuta exactamente lo que le decían los estudiantes, haciendo de un robot que debía seguir las instrucciones programadas sin realizar correcciones automáticas.

Durante la actividad, los estudiantes debían:

1. Analizar el problema: observar la posición inicial del robot, la posición de la puerta, los obstáculos (mesas, sillas).
2. Construir un algoritmo: seleccionar y ordenar las fichas de programación para construir una secuencia de instrucciones.
3. Verbalizar instrucciones: comunicar al docente robot cada instrucción en el orden que corresponda: "adelante, adelante, gira derecha…".
4. Ejecutar y observar: ver qué hace el docente robot al ejecutar el algoritmo.
5. Depurar errores: si la tarea no ha funcionado (el robot se ha encontrado o ha chocado con un obstáculo o no ha llegado a la puerta), se ha de identificar el error, para luego comprobar la secuencia y finalmente corregirlo.

La actividad permitió evaluar las seis habilidades del pensamiento computacional mediante una lista de cotejo con 12 indicadores dicotómicos (0/1) correspondientes a dos de cada habilidad:

- Razonamiento lógico (RL): 2 indicadores

- Pensamiento algorítmico (PA): 2 indicadores
- Descomposición (D): 2 indicadores
- Abstracción (A): 2 indicadores
- Reconocimiento de patrones (GP): 2 indicadores
- Evaluación/depuración (EV): 2 indicadores

La puntuación se sitúa en 12 puntos máximos (lo que indica un dominio amplio de todas las habilidades del pensamiento computacional).

Tabla 5.2. Estadísticos descriptivos del pretest (N = 54)

Grupo	N	Media	DE	Mínimo	Máximo	Rango
GE1 (Cuentos)	27	6,18	2,97	190%	11,5	9,6
GE2 (Juegos)	27	6,27	2,79	290%	11,5	8,6
Total	54	6,22	2,87	190%	11,5	9,6

Nota. Escala de 0 a 12 puntos (6 habilidades × 2 indicadores dicotómicos por habilidad). DE = desviación estándar.

5.2.2 Análisis e interpretación

El rendimiento inicial promedio fue de 6,22 puntos máximo de 12 (DE = 2,87), lo que equivale a aproximadamente un 52 % del total de puntos posibles. Este resultado se corresponde con el pensamiento computacional inicial medio-bajo de este grupo de estudiantes, coherente con su inexistente experiencia previa en programación.

La elevada variabilidad (DE = 2,87), parece indicar que la habilidad de pensamiento computacional es muy heterogénea entre los participantes, ya que algunos estudiantes alcanzan 11,5 puntos (nivel muy alto) y otros apenas alcanzan 1,9 puntos (nivel muy bajo). Esta variabilidad crea la necesidad de un análisis posterior cualitativo para

caracterizar los diferentes perfiles y también justifica el análisis de los diferentes subgrupos.

Equivalencia inicial entre grupos

No se apreciaron diferencias significativas entre los grupos experimentales en el pretest; GE1 (M = 6,18; DE = 2,97) frente a GE2 (M = 6,27; DE = 2,79). Para confirmar estadísticamente la equivalencia inicial se realizó una prueba t para muestras independientes (jamovi 2.6.4):

- $t(52) = -0{,}13$; $p = 0{,}897$

Este resultado ($p > 0{,}05$) indica que no existen diferencias estadísticamente significativas entre los grupos en el inicio, lo que confirma el principio de equivalencia inicial, que necesita un diseño cuasiexperimental para tener validez interna. Ambos grupos comienzan con un nivel comparable de pensamiento computacional.

5.3 Resultados del postest

5.3.1 Estadísticos descriptivos

Transcurridas las siete semanas de duración del plan de intervención (semana 7), la actividad "Robot Humano: Saliendo del salón" se volvió a reproducir con una versión más complicada (versión 2), en donde:

- Había mayor distancia entre la posición inicial y la puerta de salida.
- Se añadían obstáculos que obligaban a realizar varias vueltas.
- Las secuencias algorítmicas eran más extensas (varios elementos, 8-12 instrucciones, frente a la versión 4-6 del pretest).

Nota. Escala de 0 a 12 puntos. GE2 tiene un máximo de 13,2 por el excelente rendimiento de algunos estudiantes en sus respuestas.

5.3.2 Análisis e interpretación

La estadística media de puntuaciones obtenidas al final de la intervención fue de 10,60 puntos (DE = 1.67), lo que implica el 88 % de la puntuación máxima. De este modo, parece constatarse que los estudiantes lograron un nivel alto de pensamiento computacional después de trabajar con Scratch Jr durante siete semanas.

La desviación estándar pasó de 2,87 en el pretest a 1,67 para el postest, que pone de manifiesto una homogeneización del rendimiento al haber reducido la intervención las diferencias que existían entre los niños con distinto punto de partida. También la amplitud de las puntuaciones pasó de 9,6 a 5,6 puntos, indicando que incluso el niño con puntuaciones más bajas ha podido realizar un avance significativo. Por último, ambos grupos han llegado a niveles altos GE1 (M = 10,22; DE = 1,79) y GE2 (M = 10,98; DE = 1,51), teniendo en cuenta que la significación estadística de esta diferencia será analizada en la siguiente parte de este análisis.

5.4 Análisis de la ganancia pretest postest

Previamente a la aplicación de las pruebas t de Student se verificó los supuestos de normalidad y homogeneidad de varianzas sobre las puntuaciones de ganancia (postest – pretest) en pensamiento computacional. Las pruebas de Shapiro-Wilk indicaron que la distribución de ganancias no se desviaba de forma significativa de la normalidad en ninguno de los grupos ($p > 0,05$) y la prueba de Levene no mostró diferencias significativas en la varianza de las puntuaciones entre el grupo de cuentos y el grupo de juegos ($p > 0,05$), por lo que se consideró pertinente el uso de pruebas t paramétricas.

5.4.1 Prueba t para muestras relacionadas

Con la intención de comprobar si el avance existente entre el pretest y el postest es estadísticamente significativo, se ha realizado una prueba t de Student para muestras

relacionadas, es decir, se ha comparado el rendimiento de los estudiantes en el pretest y en el postest. El análisis estadístico se ha hecho en jamovi 2.6.4 con α = 0,05. Adicionalmente, se realizó una prueba t para muestras independientes referente a las ganancias (postest - pretest) de GE1 y GE2.

Tabla 5.4. Comparación pretest postest mediante prueba t (N = 54)

Comparación	Media	DE	t	gl	p	d de Cohen
Total: pretest → postest	4	2,49	12,82	53	< .001	1,75
GE1: pretest → postest	4	2,3	9,12	26	< .001	1,76
GE2: pretest → postest	5	2,37	10,18	26	< .001	1,96
Diferencia de ganancias GE1 vs. GE2	−0,62	—	−1,15	**52**	**.254**	**0,27**

Nota: M = Media de puntos, DE = desviación estándar, t = estadístico de Student; gl = grados de libertad; p = significancia estadística; d de Cohen = tamaño del efecto.

5.4.2 Análisis e interpretación

La ganancia media total fue de +4,35 puntos ($t(53) = 12,87$; $p < .001$), con un efecto muy grande ($d = 1,76$), lo que permite afirmar que la intervención educativa con Scratch Jr. es altamente efectiva para el desarrollo del pensamiento computacional en educación inicial.

Por grupo:

- **GE1 (cuentos)** = ganancia media de +4,04 puntos ($t(26) = 9,12$; $p < .001$; $d = 1,76$).
- **GE2 (juegos)** = ganancia media de +4,66 puntos ($t(26) = 10,21$; $p < .001$; $d = 1,96$).

Ambas ganancias muestran un tamaño de efecto muy grande.

La diferencia de ganancias entre los grupos (0,67 puntos de los 12 máximos) no resultó ser estadísticamente significativa ($t(52) = -1{,}16$; $p = .251$), por lo que no se puede concluir que una modalidad sea mejor que la otra: crear cuentos y crear juegos fueron igualmente eficaces para desarrollar el pensamiento computacional.

5.5 Análisis diferencial por cada habilidad de pensamiento computacional

5.5.1 Desempeño según habilidad

Cada una de las seis habilidades se evaluó a partir de dos indicadores dicotómicos, permitiendo conseguir un máximo de 2 puntos por habilidad. La Tabla 5.5 muestra el desempeño inicial y final de cada habilidad, así como la ganancia observada.

Tabla 5.5. Diferencial de análisis por habilidad de pensamiento computacional (N = 54)

Habilidad de PC	Pretest (M/2)	Postest (M/2)	Ganancia (M)	% Mejora
Razonamiento lógico (RL)	1	1,29	0,72	126%
Pensamiento algorítmico (PA)	1	1,35	0,68	101%
Descomposición (D)	1	1,26	0,61	94%
Abstracción (A)	0,5	1,14	0,64	**128%**
Reconocimiento de patrones (GP)	1	138%	71%	106%
Evaluación/depuración (EV)	0	1,34	0,98	272%

Nota. Máximo 2 puntos por habilidad (2 indicadores dicotómicos). % Mejora = (Ganancia / Pretest) × 100

5.5.2 Resultados según la habilidad

Depuración: la habilidad que más aumento real alcanzó

La evaluación/depuración obtuvo el valor de incremento más significativo: +0,99 puntos (75 % de incremento). Los niños comenzaron en el nivel más bajo (M = 0,36 de 2) y terminaron en uno de los niveles más altos (M = 1,35 de 2). Esta situación resulta da a entender que la experiencia concreta de programar, ejecutar, observar los errores, de corregirlos es el mecanismo de aprendizaje que permitió obtener los mejores resultados en Scratch Jr. Los estudiantes mejoraron notablemente en la forma de detectar los errores en las secuencias de los códigos de las instrucciones y, en particular, en proponer las correcciones pertinentes. Este resultado guarda relación con la pedagogía construccionista de Papert (1980) cuando decreta que "aprender programación es aprender a pensar en uno mismo, en el propio pensamiento, sobre todo mediante la reflexión a partir de los errores". Scratch Jr. permite un entorno donde los “errores” son constantes, inmediatos y corregibles, transformando la depuración en una práctica natural y continua.

Razonamiento lógico y abstracción: porcentajes de mejora más elevados

Ambas habilidades mostraron porcentajes de mejora altos (126 % y 128 % respectivamente):

- El razonamiento lógico, base para construir relaciones de causa efecto entre bloques de código y el resultado en pantalla, pasó de 0,57 a 1,29 puntos.
- La abstracción pasó de 0,50 a 1,14 puntos, pero se mantuvo como la habilidad de menor desempeño final. Siendo así, podría decirse que la representación simbólica y la generalización son procesos cognitivos más complejos que requieren una mayor maduración en la educación inicial.

Pensamiento algorítmico, descomposición y patrones: progresos equilibrados

Estas habilidades (pensamiento algorítmico, descomposición, reconocimiento de patrones) lograron ganancias que oscilaban entre +0,61 puntos y +0,71 puntos (94 106 %), y se finalizó a niveles relativamente altos (1,26 1,38 de 2):

- El pensamiento algorítmico, necesario para secuenciar instrucciones paso a paso, llegó a 1,35 puntos.
- La descomposición, necesaria para descomponer problemas complejos en partes más manejables, llegó a 1,26 puntos.
- El reconocimiento de patrones, que ayuda a detectar estructuras que se repiten, llegó al nivel más alto (1,38 puntos).

En síntesis, todas las habilidades mostraron un progreso de importancia ($p < .001$). Asimismo, se comprobó la idea de que la depuración fue el proceso cognitivo que más se desarrolló con la intervención (275 % de mejora); mientras tanto, la abstracción, a pesar de crecer considerablemente, continúa requiriendo un mayor andamiaje pedagógico en la educación inicial.

5.6 Síntesis y conclusiones cuantitativas

De la evaluación cuantitativa se pueden extraer cinco conclusiones principales:

1. **La intervención con Scratch Jr. fue notablemente efectiva**

Los 54 estudiantes manifestaron un incremento de su pensamiento computacional del 70 % de manera muy significativa (ganancia M = 4,38 puntos de un máximo de 12; $t(53) = 12,87$; $p < .001$; $d = 1,76$). La magnitud del tamaño del efecto, contrastada con los criterios de Cohen (1988), valida la idea de que la programación visual en la educación inicial, mediada por la creación de productos digitales muy significativos (cuentos y juegos), promueve el desarrollo de las habilidades cognitivas superiores.

2. Ambas modalidades (cuentos y juegos) fueron igualmente efectivas

Las diferencias de ganancia entre el grupo que creó las historias (M = 4,04 puntos) y el grupo que creó los juegos (M = 4,71 puntos) no fueron estadísticamente significativas ($t(52) = -1,16$; $p = .251$). Ambas modalidades arrojaron tamaños de efecto de calidad muy altos ($d = 1,76$ para cuentos; $d = 1,99$ para juegos). Los resultados evidencian que el mecanismo pedagógico eficaz no reside en el contenido del proyecto en sí, sino en el procedimiento de programar secuencias de instrucciones, ejecutarlas, observar los resultados y depurarlas (prueba y error) en repetidas ocasiones.

3. La depuración es la habilidad más lograda

La evaluación/depuración es la habilidad que muestra la ganancia más alta (+0,99 de 2; 275% de mejora), lo que deja claro que el contacto directo con ciclos de error-corrección es la ganancia cognitiva más se beneficia con el trabajo en Scratch Jr. Este resultado se alinea con la pedagogía construccionista expuesta por Papert (1980), la cual sostiene que aprender a programar es "aprender a pensar sobre el propio pensamiento, especialmente a través de la reflexión de los errores".

4. La intervención acortó brechas iniciales

Bajo la intervención, la desviación estándar pasó de 2,87 (pretest) a 1,67 (postest), esto nos permite deducir que los estudiantes de menor rendimiento inicial fueron los que más elevaron su puntuación con lo que se disminuyó, la heterogeneidad. El niño con menor puntuación inicial (1,9) alcanzó 7,7 al final; el niño con mayor puntuación inicial (11,5) subió a 13,2. Todos ellos tuvieron una mejora significativa.

5. **Heterogeneidad de trayectorias**

El hecho de que haya heterogeneidad de trayectorias llevó a realizar un análisis cualitativo en el que se observó la evolución de los diferentes estudiantes. A pesar de que todos mejoraron, el tamaño de las ganancias fue muy distinto (rango: de 1,7 a 6,2 puntos de los 12 posibles). Esta variabilidad individual, no explicada por parámetros demográficos (edad, sexo, etc.) ni por el grupo experimental al que pertenecían, llevó a pensar en la existencia de distintos procesos cognitivos. El proceso de caracterización de estas diferencias se desarrolla en el Capítulo 6, donde se lleva a cabo el análisis cualitativo correspondiente.

5.7 Selección de la submuestra para el análisis cualitativo

5.7.1 Justificación del análisis cualitativo adicional

El análisis cuantitativo confirmó que la intervención funcionó (d = 1'76), pero no da respuesta al cómo ni al por qué la intervención generó mejoras. Preguntas como:

- ¿Qué procesos cognitivos específicos intervienen en el desarrollo de cada habilidad?
- ¿Cuál es la diferencia entre las estrategias de los niños que crean cuentos y las de los que crean juegos?
- ¿Por qué unos estudiantes mejoran más que otros, aunque estén bajo las mismas condiciones de intervención?
- ¿Qué rol cumple la depuración iterativa en el aprendizaje?

Son algunas de las razones que explican por qué se realizó un análisis cualitativo adicional (diseño mixto embebido QUAN→qual), que permitiera examinar más a fondo una submuestra representativa del análisis del pensamiento computacional.

5.7.2 Criterios de selección de casos

Se eligieron 8 estudiantes (4 por grupo) siguiendo estos criterios:

1. Disparidad en el rendimiento inicial: casos de bajo (≤ 4,8 puntos), medio (4,8-8,0 puntos) y alto (≥ 8,0 puntos) rendimiento inicial.
2. Equilibrio entre modalidades; 4 estudiantes de GE1 y 4 de GE2.
3. Equilibrio de género; 4 niñas y 4 niños.
4. Amplitud de los datos cualitativos; asistencia completa y grabaciones de video claras.

5.7.3 Caracterización de la submuestra cualitativa

Tabla 5.6. Caracterización de la submuestra cualitativa (n = 8)

Código	Grupo	Género	Edad	Pretest	Postest	Ganancia	Tipo de caso
GE1EST001	Cuentos	Niña	5	6,7	12	5,3	Desempeño medio-alto, mayor ganancia GE1
GE1EST003	Cuentos	Niño	5	4,3	8,4	4,1	Desempeño inicial bajo, mejora promedio
GE1EST005	Cuentos	Niña	5	1,9	7,7	5,8	Desempeño inicial muy bajo, mejora sobresaliente
GE1EST006	Cuentos	Niño	6	8,4	**12**	**3,6**	Desempeño inicial alto, mantiene nivel alto
GE2EST002	Juegos	Niño	5	6	10,1	4,1	Desempeño medio, mejora promedio
GE2EST004	Juegos	Niña	6	7,7	12	4,3	Desempeño medio-alto, alcanza nivel máximo
GE2EST007	Juegos	Niña	5	2,9	9,1	6,2	Desempeño inicial bajo, mayor ganancia GE2
GE2EST008	Juegos	Niño	6	11,5	12	0,5	Desempeño inicial más alto, menor ganancia pero nivel máximo

Nota: Códigos alfanuméricos utilizados para proteger la identidad de los participantes.

Escala: 0-12 puntos.

Representatividad de la submuestra

Para comprobar la representatividad de la submuestra cualitativa (n = 8) con la muestra total (N = 54), se compararon las estadísticas descriptivas de la submuestra con las de la muestra total. La comparativa se muestra en la Tabla 5.7.

Tabla 5.7. Validación de la representatividad de la submuestra

Indicador	Muestra total (N=54)	Submuestra (n=8)	Diferencia
Media pretest	6	5,87	−0,35
Media postest	11	10,51	−0,06
Media ganancia	4	4,06	−0,29
DE ganancia	2,49	1,47	−1,02
Rango de ganancia	0,5-6,2	0,5-6,2	0

Nota: DE = desviación estándar. Diferencia = submuestra − muestra total.

Las estadísticas de la Tabla 5.7 comprueban que la submuestra cualitativa es representativa de la muestra total, ya que las diferencias en la media pretest, postest y ganancia son únicamente menores que 0,35 puntos (aproximadamente el 2,9 % de la escala de 12 puntos). El rango de ganancia es el mismo (1,7 - 6,2 puntos), prueba de que la submuestra incluye casos de baja ganancia (efecto techo, como el caso GE2EST008) junto a casos de ganancia muy alta (mejoría sobresaliente, como el caso GE2EST007), eliminando así el sesgo de casos atípicos y asegurando que los resultados que se reportan en el Capítulo 6 provienen de estudiantes representativos del conjunto del grupo.

5.8 Transición al análisis cualitativo de los resultados

Los resultados cuantitativos permiten afirmar que la intervención ha sido exitosa, pero no nos explican el proceso mediante el cual se construye el pensamiento computacional en la práctica. Quedan abiertas, por lo tanto, las siguientes preguntas:

- ¿A qué procesos cognitivos específicos da lugar cada una de estas habilidades?
- ¿Cuál es la diferencia cualitativa entre la estrategia de los niños que hacen cuentos y la estrategia de los niños que hacen juegos?
- ¿Por qué los estudiantes que siguen la misma intervención mejoran de maneras diferentes?
- ¿Cuál es el papel que juega la depuración iterativa en el aprendizaje?
- ¿Cómo aparecen las seis habilidades del PC (razonamiento lógico, pensamiento algorítmico, descomposición, abstracción, patrones y depuración) en las verbalizaciones y acciones de los estudiantes?

El **Capítulo 6** presenta el análisis cualitativo pormenorizado de 8 estudiantes elegidos (4 de cada grupo) que caracterizan los procesos de creación de cuentos y juegos, así como la manera en que se relacionan con el desarrollo de las habilidades de pensamiento computacional. A partir de las transcripciones de los vídeos, del análisis de los artefactos digitales que se han producido y de las observaciones sistemáticas realizadas, se identificarán los modelos cognitivos que subyacen a los resultados cuantitativos que ya vimos.

6 Análisis Cualitativo del Pensamiento Computacional

6.1 Introducción al análisis cualitativo

El capítulo 5 demostró la efectividad de la intervención de forma cuantitativa en el desarrollo del pensamiento computacional en ambos grupos experimentales. Sin embargo, un análisis estadístico de los datos obtenidos por sí solo no responde a preguntas esenciales al respecto, como, por ejemplo, ¿cómo se produce el desarrollo del pensamiento computacional? ¿Qué procesos cognitivos emergen en su desarrollo? ¿Cuáles son las diferencias cualitativas entre crear cuentos interactivos y crear juegos interactivos?

Este capítulo presenta un análisis cualitativo en profundidad de 8 estudiantes (4 de cada grupo experimental) con el fin de caracterizar los mecanismos específicos a través de los cuales surge el pensamiento computacional en la educación inicial. Las preguntas que guían el presente análisis son las siguientes:

1. ¿A través de qué procesos cognitivos específicos emerge el pensamiento computacional mientras se crean cuentos interactivos?
2. ¿A través de qué procesos cognitivos específicos emerge el pensamiento computacional mientras se crean juegos interactivos?
3. ¿Cuáles son las diferencias cualitativas en los procesos de pensamiento computacional que exhiben ambas modalidades?
4. ¿Cómo se encuentran las seis habilidades de PC (razonamiento lógico, pensamiento algorítmico, descomposición, abstracción, patrones y evaluación/depuración) en las acciones y verbalizaciones de los estudiantes?
5. ¿Por qué estudiantes a los que se les aplica la misma intervención mejoran de forma distinta?

El análisis se organiza en las siguientes secciones: primero, se presenta la fundamentación metodológica (6.2); después se caracteriza el proceso cognitivo del grupo de cuentos (6.3) así como el del grupo de juegos (6.4), a partir de densas descripciones de casos ejemplares, de matrices analíticas que sistematizan la evidencia, así como de modelos emergentes que generalizan los hallazgos; prosigue el contraste entre ambas modalidades (6.5); se produce la extracción de hallazgos transversales (6.6). Finalmente, se presentan conclusiones parciales del análisis cualitativo.

6.2 Fundamentación metodológica del análisis cualitativo

6.2.1 Diseño de estudio mixto embebido (QUAN→qual)

La investigación está fundamentada en un diseño mixto embebido con prioridad cuantitativa (QUAN→qual), en el que el componente cualitativo acompaña y profundiza los hallazgos cuantitativos. El análisis cuantitativo del Capítulo 5 indicaba la efectividad global de la intervención; el análisis cualitativo del presente capítulo explicita los mecanismos cognitivos específicos que hay detrás de esa efectividad.

6.2.2 Fuentes de datos cualitativos

El análisis se apoyó en tres fuentes de datos por cada uno de los 8 estudiantes de la submuestra cualitativa:

1. Videograbaciones: Registros audiovisuales completos de todas las sesiones (Semanas 1-7, 21 horas en total), que incluyen actividades desconectadas y trabajo en Scratch Jr.
2. Transcripciones: Registros escritos de las verbalizaciones espontáneas de los estudiantes, así como de las explicaciones de razonamiento ante las preguntas estructuradas (“¿Qué estás haciendo?”, “¿Por qué has elegido ese bloque?”, “¿Qué crees que pasará?”).

3. Análisis de productos digitales: Inspección sistemática de los productos finales obtenidos en Scratch Jr., registrando bloques utilizados, secuencias programadas, estructuras de control y evidencia de depuración.

6.2.3 Estrategia de análisis temático deductivo-inductivo

Se utilizó una estrategia de análisis temático (Braun & Clarke, 2006) desglosada en dos pasos:

- **Paso 1 (Deductiva):** Se codificaron de forma previa tres categorías de habilidades de pensamiento computacional explicadas en el marco teórico: razonamiento lógico, pensamiento algorítmico, descomposición, abstracción, patrones y evaluación/depuración.
- **Paso 2 (Inductiva)**: Se extrajeron los temas que agrupaban codificaciones en patrones más amplios relacionados con estrategias cognitivas, procesos de transferencia de conocimiento y manifestaciones específicas de PC en cada modalidad.

6.2.4 Rigor y validez del análisis cualitativo

Se aplicaron cuatro estrategias para la consecución del rigor:

- Triangulación de fuentes. Cualquier conclusión proviene de al menos dos de las tres fuentes de datos (videograbaciones, transcripciones, productos digitales). Por ejemplo, si se reportaba que "depuración iterativa" se indicaba que se podía hallar evidencia en los tres tipos de fuentes.
- Descripciones densas. En las secciones de presentación narrativa de casos (6.3.2 y 6.4.2) se ofrecen transcripciones literales de gran extensión y contextualizadas para permitir al lector que pueda contrastar las interpretaciones a la luz de su credibilidad.

- Reflexividad. Se trató de un investigador externo sin formar parte del cuerpo docente, lo cual hace que se redujeran sesgos de la confirmación. Las interpretaciones fueron contrastadas con las docentes titulares en las sesiones de la retroalimentación.
- Saturación teórica. A partir de los análisis realizados de los 8 casos (4 por grupo) se contribuyó a poner al descubierto patrones recurrentes; el caso 8 propició información redundante lo cual contribuyó a evidenciar que existía suficiente saturación teórica.(Nota: La sección "Consideraciones éticas" ha sido trasladada al Capítulo 4, conforme las consideraciones del director).

6.3 Caracterización del pensamiento computacional en el grupo de cuentos (GE1)

6.3.1 Introducción

Esta subsección da respuesta al objetivo específico 1: analizar y caracterizar cómo la producción de cuentos en Scratch Jr. favorece el desarrollo de las habilidades del pensamiento computacional en educación inicial. En primer lugar, se presentan descripciones densas de cuatro niños del grupo de cuentos, que representan la diversidad de los perfiles cognitivos que han sido observados. A continuación, se hace uso de una matriz analítica, para sistematizar las evidencias de las seis habilidades del PC. Para terminar, un modelo emergente permite generalizar el desarrollo de los procesos cognitivos sin referirse a casos concretos.

6.3.2 Presentación narrativa de los alumnos del grupo de cuentos

Caso 1: GE1EST001 – Verificación iterativa como estrategia de validación

El gato perdido en el bosque

El niño elabora un cuento en el que un gato sale por el bosque a dar una vuelta, después de un rato se pierde entre los árboles y es rescatado por un hada que lo regresa a casa. De esta manera, demuestra el uso de la estrategia de verificación iterativa: programa

cada escena por separado, ejecuta las acciones usando la bandera verde para iniciar, observa el resultado y corrige antes de ir a por la escena siguiente, y así constantemente. Esta secuencia Ejecuta → Observa → Corrige se mantiene invariable en toda la intervención y pone en evidencia un acertado manejo de depuración sistemática. La estructura narrativa (salida → pérdida → ayuda → regreso) es fácilmente trasladable a la descomposición, lo cual se ratifica al conectar 3 páginas con el bloque de ir a la página siguiente, demostrando que está empezando a comprender la modularización del problema. Expresa emociones mediante propiedades visuales: el gato camina con bloques de mover a la derecha o a la izquierda y se acerca o se aleja del hada (dependiendo de la situación narrativa); en la escena final, el acercamiento al hada aparece relacionado con la resolución del conflicto al marcharse juntos, de manera que se articula espacio y estado emocional. El entrelazamiento entre los diálogos del hada (Tranquilo, yo te ayudaré) y los movimientos del gato, junto con el cambio de escena, muestra un trabajo de secuenciación y sincronización. Su progreso de 4/12 a 10/12 (+6 puntos) indica que esa forma de validar visualmente cada fragmento cumplió positivamente su propósito de servir como andamiaje para el desarrollo del pensamiento computacional (Ver Figura 6.1)

Figura 6.1. Proyecto "El gato perdido en el bosque"

Nota. Captura de pantalla del proyecto "El gato perdido en el bosque" elaborado por el estudiante GE1EST001. Se ve: (a) programación del gato con secuencias de movimiento y cambios de escenas; (b) programación del hada con bloques de aparición, diálogo y desplazamiento; (c) programación de la transición entre páginas que ordena los actos del cuento; (d) el hada se marcha con el gato. Elaboración propia, 2025.

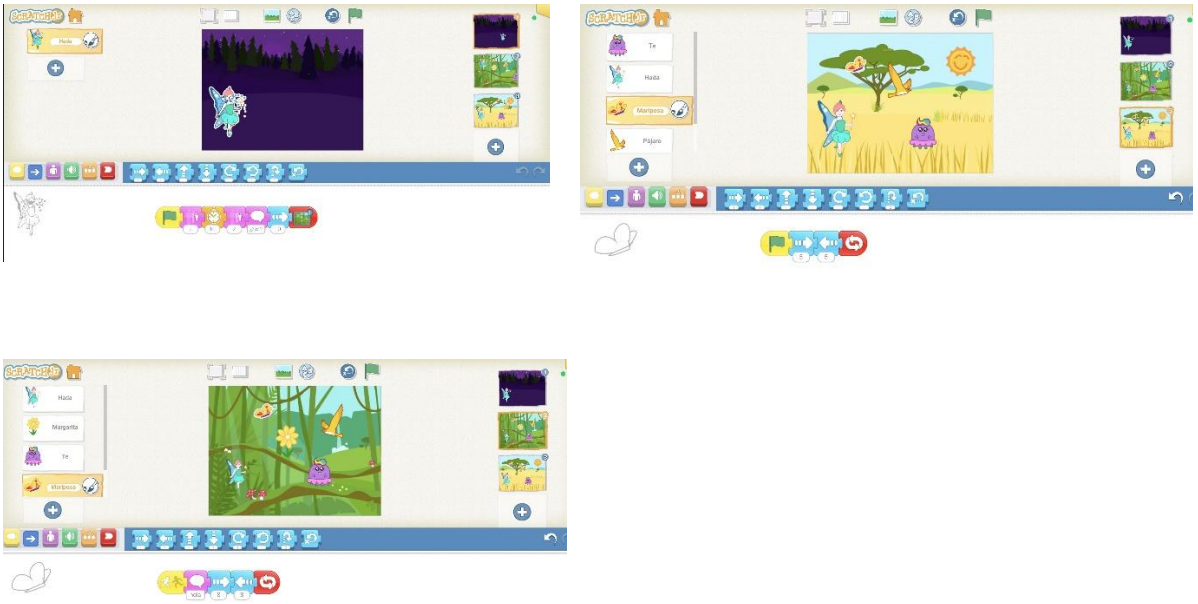
Caso 2: GE1EST005 – Abstracción simbólica con transformación tardía

El hada que buscaba el sol

Este cuento habla sobre un hada que desea encontrar el sol. Inicia en un bosque oscuro, donde aumenta su tristeza por la falta del sol, y después, en la siguiente escena, se encuentra con nuevos amigos que la acompañan a encontrar el sol en un paisaje luminoso. En lo que a pensamiento computacional se refiere, se va pasando de acciones muy literales de movimiento hacia un uso más simbólico y organizado de los bloques: en la primera escena, la hada da inicio en un fondo nocturno, pequeña y moviéndose lentamente; a partir de ahí, Inicia con bandera verde, la secuencia de instrucciones Mover, Esperar, Decir "¿Sol?" (expresando la búsqueda) y continua con el bloque de avanzar a la siguiente página. En la segunda y en la tercera escena aparecen personajes secundarios (por ejemplo, pájaros o animales del bosque), que se empiezan a activar a partir de otros eventos (Iniciar con bandera verde, Iniciar al tocar, Enviar mensaje...), mostrando la coordinación de múltiples hilos de ejecución. Utiliza el bloque repetir por siempre para manifestar acciones como el vuelo de un personaje, comportamientos reiterativos al tiempo que el Hada crece, para manifestar sus emociones con

el cambio de tamaño: el hada triste se hace menor y, al llegar al sol, aumenta tamaño sobre un fondo claro, con lo que se completa una abstracción simbólica en la que los parámetros visuales (tamaño, luminosidad del fondo) se corresponden con estados internos (triste-pequeña vs feliz-grande). Hacia la mitad de la intervención verbaliza ideas del tipo "cuando esté triste la hago pequeñita", lo que implica que la narración emocional se convierte en el soporte de una abstracción reflexiva sobre los estados de ánimo. El cambio de 2/12 a 9/12 (+7) indica una importante transformación en la forma de planear y representar la historia como un algoritmo (Ver Figura 6.2).

Figura 6.2. Proyecto "El hada que buscaba el sol



Nota. Captura de pantalla del proyecto "El hada que buscaba el sol" elaborado por el estudiante GE1EST005.Se prescribe: (a) programación del hada con bloques de movimiento, espera y búsqueda del sol por medio de diálogos; (b) programación de los personajes que acompañan a Caperucita roja con bucles de movimiento continuo; (c) programación de los

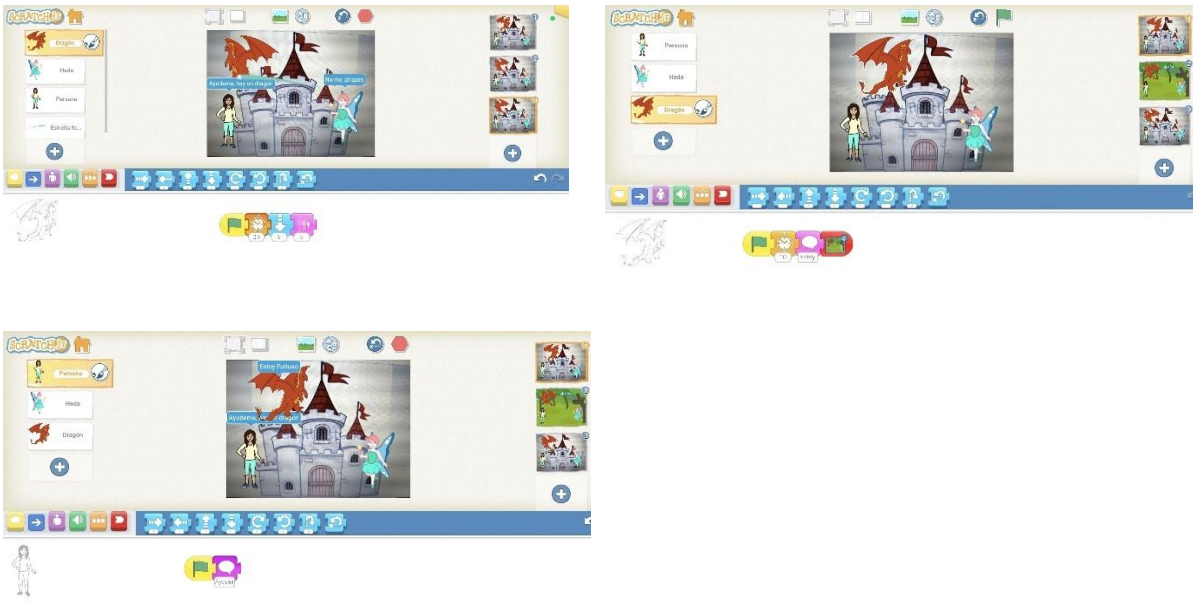
cambios de fondo, y cambios de tamaño que simbolizan el paso de la oscuridad a la luz. Elaboración propia, 2025.

Caso 3:GE1EST003 – Condicionales implícitas, descomposición

El hada y el dragón

Este estudiante desarrolló un cuento de una princesa con miedo porque había un dragón "furioso"; hasta que apareció un hada que dijo "no tengas miedo", lanzó un rayo, y finalmente el dragón se hizo pequeño y se fue. Con respecto al pensamiento computacional, la estructura narrativa clásica del conflicto → ayuda → resolución se convierte casi en secuencias de bloques, se complementan con el bloque de inicio de bandera verde en cada página: se configuran los conflictos con el bloque Decir. Después aparece el dragón desafiante, la intervención del hada, y la resolución donde el dragón pierde poder. El estudiante usa el tamaño como recurso de abstracción simbólica (el dragón grande representa amenaza; tras la intervención del hada, pasa a ser pequeño con el bloque de Encoger, combinado a veces con Esconder para representar una derrota total). El paso ordenado entre escenas mediante ir a una página, así como el encadenamiento de los mismos diálogos y los personajes que hacen aparición con la bandera verde, demuestran un entendimiento básico de la secuenciación y sincronización de eventos. El cuento de miedo y protección es el contexto significativo para organizar la lógica implícita de tipo "si aparece el hada, entonces el dragón pierde fuerza", que, aunque no sea condicional, se manifiesta en el orden y efecto del uso de los bloques. Su mejora de 3/12 a 8/12 (+5 puntos) indica que la narrativa emocional funcionó como andamiaje para una primera abstracción concreta sobre estados y cambios en el sistema (Ver Figura 6.3).

Figura 6.3. Proyecto "Hada y Dragón": Código de cada personaje en Scratch Jr.



Nota. Captura de pantalla del proyecto "Hada y Dragón" creado por el estudiante GE1EST005. Se observa: (a) programación del hada con bloques en movimiento y bloques de cambio de tamaño; (b) programación del dragón con código de aparición y desaparición; (c) programación de la princesa con código de diálogos. Elaboración propia, 2025.

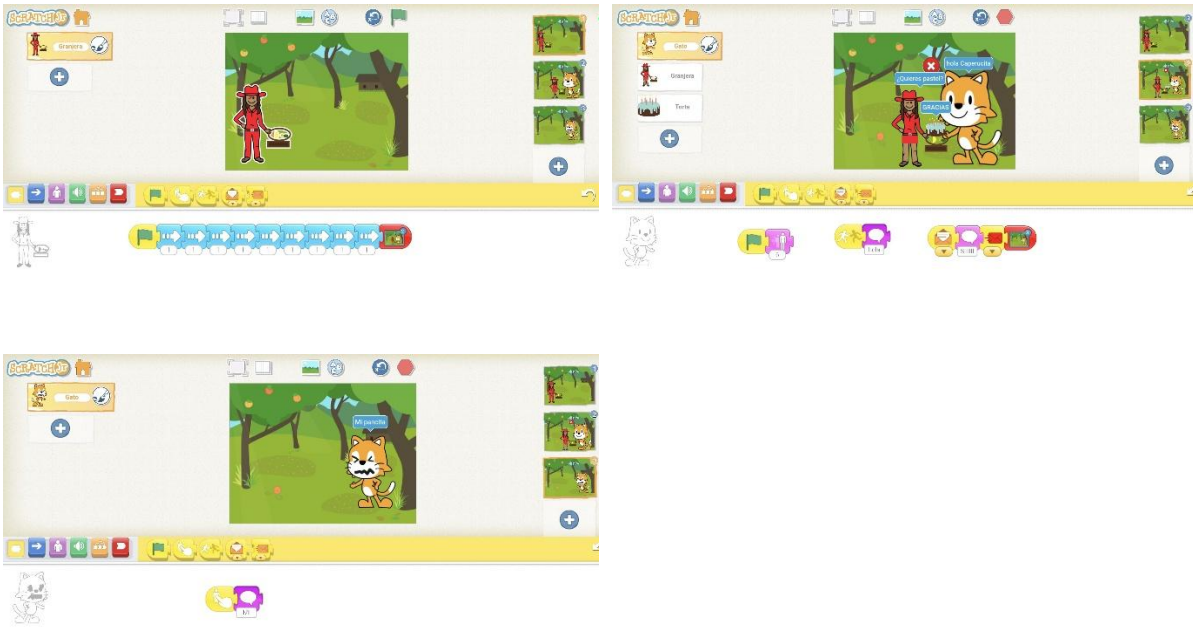
Caso 4: GE1EST006 – Causa-efecto con carga narrativa moral

Caperucita Roja Perversa

El estudiante llevó a cabo una reescritura “oscura” del cuento clásico de Caperucita. En esta versión alternativa, Caperucita lleva un pastel “envenenado” o con ingredientes horrorosos al lobito bueno, que acaba enfermo después de comerse el regalo. Desde el punto de vista relacionado con el pensamiento computacional, la historia hace resaltar la interacción de la estructura de causa-efecto, que se manifiesta en la secuencia lógica del cuento: la interacción entre Caperucita y el lobo se da a partir de bloques de (Decir) que representan las acciones importantes en la historia ("¿Quieres pastel?") y reacciones del otro personaje ("¡Mi

pancita!") coordinada con los cambios de estado visuales (rostros), y van acompañados con la codificación que iniciar con bandera verde. El pastel es un objeto con carga simbólica que media entre la acción maliciosa de Caperucita y la consecuencia en el destinatario, y eso es lo que da cuenta de su comprensión de la relación programa → efecto observable (Causa – Efecto). La estructura de páginas enlazadas de Ir a una página sirve para separar el primer acto (lugar del encuentro en el bosque) del acto de resolución (enfermedad del personaje), lo que muestra la descomposición clara del problema narrativo en sus actos constitutivos. El uso de los diálogos como forma de encadenamientos lógicos (que permite establecer la premisa "Si Caperucita da pastel, entonces el lobo se enferma") muestra una lógica condicional implícita, pero no explícita entre los bloques de ScratchJr. Esta reescritura lógica del cuento se pone de manifiesto también en que el estudiante parecía entender que la programación como lo hemos visto en otros estudiantes, va de la mano de la ruta narrativa: los mismos bloques que en otro contexto se emplean para generar un final feliz aquí dan lugar en cambio a un resultado desafortunado, hasta el punto de consolidar la idea de que los algoritmos son herramientas neutras cuya función depende del diseño intencionado. El hecho de que su mejora de 2/12 pasara a 9/12 (+7 puntos) indica que el conflicto moral implícito en la trama se convirtió en motor motivacional para crear secuencias de causa-efecto con precisión algorítmica (Ver Figura 6.4).

Figura 6.4. Proyecto "Caperucita roja perversa": Código de cada personaje en Scratch Jr.



Nota. "Caperucita roja perversa" del estudiante GE1EST006. Observamos que: (a) Caperucita está programada con un recorrido largo hasta que finalmente se encuentra con el lobo/gato; (b) el lobo/gato con una programación que incluye bloques de diálogo con los cuales expresa agradecimiento y, posteriormente, malestar; (c) el pastel, como objeto principal del proyecto, es programado junto a transiciones de escenas y el final del cuento. Elaboración propia, 2025.

6.3.3 Matriz analítica: caracterización de habilidades de PC en GE1

Habilidad PC	Descripción emergente en GE1 (Cuentos)	Mecanismo Narrativo de Apoyo
Razonamiento lógico	Los niños analizaban los eventos, las causas y los resultados de la narrativa y aplicaban esa misma lógica a los bloques de programación. Por ejemplo: "Si el personaje está triste, entonces debe llorar."	La lógica del cuento (causa→efecto) se transfiere sin problemas al pensamiento algorítmico.
Pensamiento algorítmico	La estructura de inicio-nudo-desenlace organizaba muy bien las secuencias de bloques. Los niños construían la programación en el mismo orden narrativo	La secuenciación temporal del cuento es el anadamiaje para llevar a cabo la secuenciación algorítmica.
Descomposición	Los niños descomponen los problemas en escenas narrativas (escena 1: la presentación, escena 2: el nudo o conflicto, escena 3: el desenlace o la resolución); cada escena era un módulo del código.	El acto narrativo fraccionado facilita la mejora continua del código.
Abstracción	Los niños utilizaban propiedades visuales (tamaño pequeño=tristeza, tamaño grande=alegría) entendidas como representaciones simbólicas de conceptos emocionales abstractos.	La narrativa emocional es un referente explícito del proceso de abstracción.
Patrones (Generalización)	Los niños identificaban patrones narrativos ("el personaje siempre hace lo mismo en cada escena") y simplemente copiaban el personaje de una escena a otra con la misma codificación	El proceso de la narrativa confirma que el patrón narrativo se transfiere fácilmente a los patrones del códificación
Evaluación/ Depuración	Los niños se preguntaban si el programa "contaba la historia como ellos la imaginaban" y utilizaban el cuento como juicio para validar el programa. Iteraban hasta que el código "tenía el sentido narrativo deseado".	La narración toma el sentido de ser el criterio de evaluación explícito y tangible.

6.3.3 Modelo emergente: pensamiento computacional mediado por la narrativa

El análisis del grupo de cuentos muestra un modelo determinado de progreso en el desarrollo del pensamiento computacional en el que la forma y la lógica del relato funcionan como organizadores cognitivos de los procedimientos de programación. Este modelo actúa a través de cuatro mecanismos:

1. Andamiaje temporal. La estructura del relato (introducción > conflicto > resolución) organiza de manera natural las secuencias algorítmicas.
2. Abstracción simbólica. Las propiedades visuales de los personajes (tamaño, color, posición) se convierten en representaciones metafóricas de estados emocionales en sus narraciones.
3. Descomposición natural. Los actos del cuento definen de forma orgánica de cómo perfeccionar el código fácilmente sin explicaciones complejas.
4. Criterio de validación narrativa. La coherencia del cuento se convierte en un criterio de verificación que hace la depuración más tangible y motivadora que los criterios técnicos y abstractos.

Implicación: en el grupo de cuentos, el pensamiento computacional se despliega no como una serie de habilidades técnicas, sino como una vía ampliada de los procesos de narración ya interiorizados por los niños.

6.4 Caracterización del pensamiento computacional en el grupo de juegos (GE2)

6.4.1 Introducción

Este apartado responde al objetivo específico 2: realizar un análisis y una caracterización de cómo la creación de juegos en Scratch Jr. contribuye al desarrollo de las habilidades de pensamiento computacional. Para ello, se ofrecen descripciones detalladas de

cuatro estudiantes del grupo de juegos, a fin de continuar con una matriz analítica que apoye la caracterización y, al final, se expone el modelo emergente identificado.

6.4.2 Presentación narrativa de estudiantes del grupo de juegos

Caso 1: GE2EST002 – Arquitectura modular con mensajes

El laberinto mágico

El estudiante construye un laberinto en el que un gato debe moverse hasta alcanzar una estrella, usando un mando de flechas con el que puede controlar las acciones del gato. En términos de pensamiento computacional, lo relevante fue llegar a realizar la abstracción del movimiento logrado con las flechas, desde la programación implícita de cada dirección hasta su disociación de su realidad. Cada flecha es un personaje que se programa con el bloque de tocar y enviar mensaje, acción que se repite con cada dirección, y el gato maneja cuatro subórdenes (instrucciones) que envían el mensaje de acuerdo con el color programado en cada flecha, para así poder mover el personaje. Al final del laberinto se encuentra una estrella de mar que también está programada para girar apenas el gato la alcance. Esto da cuenta de un acercamiento del estudiante a la modularización por medio de la construcción de un sistema interactivo. Las paredes del laberinto están programadas para hacer que el personaje regrese al punto de partida mediante bloques de comenzar al tocar. Toda esta complejidad de la programación funciona como andamiaje que fortalece los procesos de descomposición de un problema y de abstracción del mundo real al virtual. Esto se pone en evidencia con su paso de 3/12 a 8/12 (+ 5 puntos) (Ver Figura 6.5).

Figura 6.5. Proyecto "El laberinto mágico": Código en Scratch Jr.

Nota. Captura de pantalla "El laberinto mágico" de la estudiante GE2EST002. Aquí podemos observar: (a) instrucciones del botón de flecha de dirección con el bloque de evento "comenzar al tocar" y "envío de mensaje"; (b) instrucciones del gato jugador que recibe cada mensaje que le indica el movimiento en la pantalla; (c) instrucciones de la estrella al final del laberinto que, al ser tocada, se pone a dar vueltas. Elaboración propia, 2025.

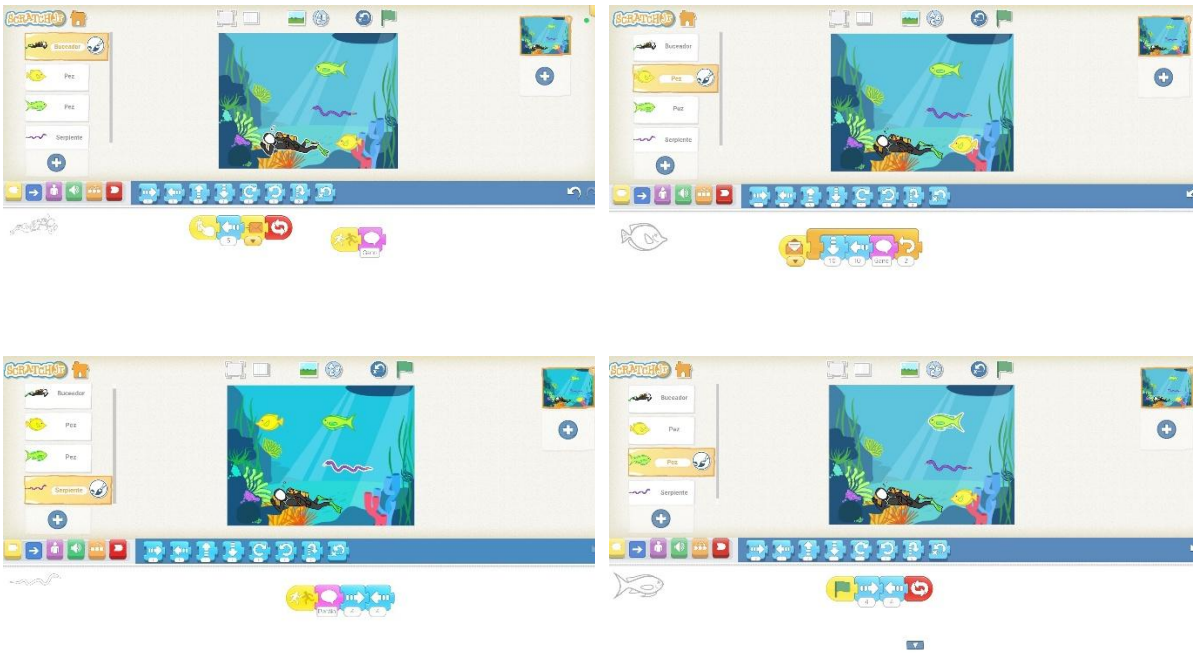
Caso 2: GE2EST004 Depuración iterativa paramétrica

El Buzo

El niño elaboró un juego de un buzo en el mar y el objetivo es que toque los peces y evite la serpiente. Cada vez que logra tocar los peces, estos le dicen “Gano”. Cada vez que choca con la serpiente, esta le dice: “Perdió”. Enmarcado en el pensamiento computacional, es clave demostrar la lógica condicional implícita a través de los movimientos programados de los diferentes personajes: El buzo se mueve cada vez que se toca; a pesar de tener una trayectoria configurada, al tocarlo en un lugar distinto, también cambia su posición original, mientras que los peces y la serpiente sí se activan con la bandera verde. A pesar de que

ScratchJr. no cuenta con bloques condicionales de “si… entonces”, la lógica condicional está explícita en las reacciones de los personajes. Este niño en particular se destacó por su actitud resiliente durante todo el proceso; probaba iterativamente muchos bloques, en conjunto con el número de cantidad y espera para ajustar la velocidad y la complejidad, aplicando y probando muchas veces hasta que lograba lo que quería. Esta preocupación por que los números funcionaran como lo deseaba demostraba su grado de conciencia sobre la importancia de los parámetros numéricos en los juegos. Su mejora de 3/12 a 8/12 (+5 puntos) da cuenta de que las fases de prueba y error orientadas por las reglas de ganar o perder fortalecen el pensamiento algorítmico y la capacidad de depuración en los niños (Ver Figura 6.6).

Figura 6.6. Proyecto "El buzo": código de los personajes en Scratch Jr



Nota. Captura de pantalla del juego "El buzo" creado por la estudiante del grupo GE2:GE2EST004. (a) Programación del buzo por bloques de comenzar al tocar; (b) Programación de los peces con bloques de diferentes direcciones y repetir siempre con evento de comenzar al tocar que activa el mensaje de ganó; (c) programación de la serpiente con

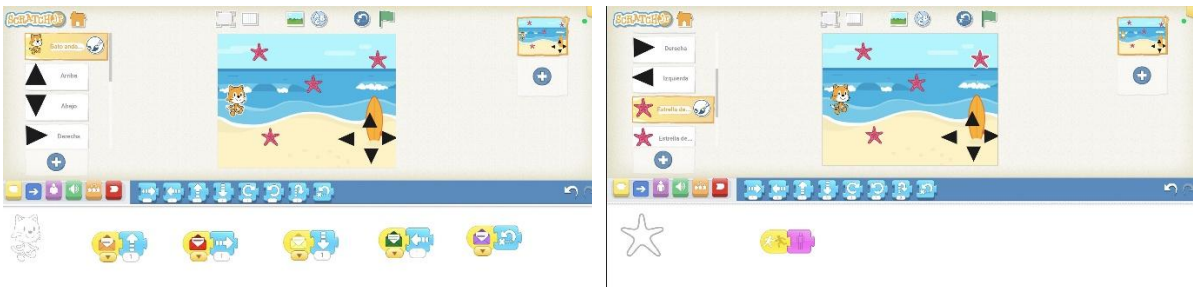
movimiento autónomo y evento de comenzar al tocar que muestra el mensaje de perdió. Elaboración propia, 2025.

Caso 3: GE2EST007 – Generalización y reutilización de esquemas

Recoge estrellas

El objetivo del juego que expone el estudiante es lograr que un personaje se pueda mover por toda la pantalla, capturando estrellas. Para ese fin y para demostrar las habilidades de pensamiento computacional desarrolladas hasta este punto, reutiliza la base del laberinto para alcanzar su meta, recordando el uso de los bloques de mensajes, que ayudan a configurar el mando direccional que permitirá el desplazamiento del personaje principal. Las estrellas están acompañadas de bloques de comenzar al tocar, lo que permite programar su desaparición cuando el protagonista las alcanza. La descomposición estuvo marcada por la destreza en la programación del mando direccional; asimismo, el hecho de poder ajustar la velocidad cambiando solamente un parámetro dentro de la configuración ilustra su capacidad de generalización y abstracción. Su ganancia de 5/12 a 11/12 (+6 puntos) demuestra que supo adaptar esquemas algorítmicos durante todo su progreso (Ver Figura 6.7).

Figura 6.7. Proyecto "Recolector de estrellas": Código de los personajes en Scratch Jr.



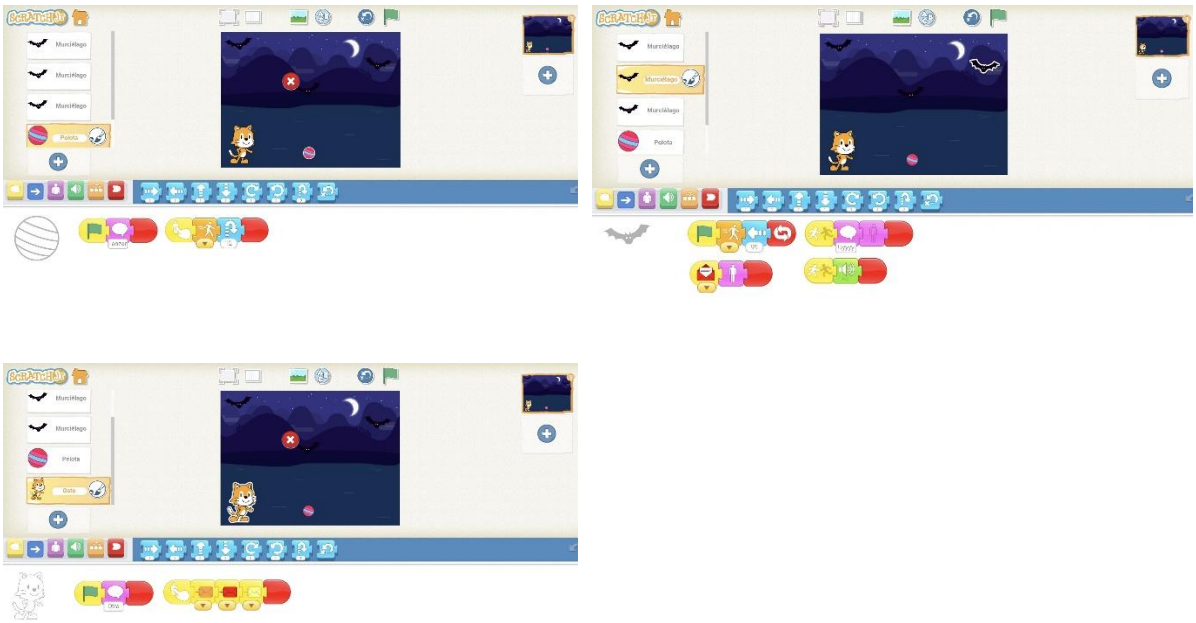
Nota. Captura de pantalla del juego "Recolector de estrellas" creado por el estudiante GE2EST007. (a) Programación del gato protagonista con 4 subprogramas; (b) programación de las estrellas con el evento de desaparecer al tocar; (c) programación de las fichas de dirección que permiten mover al personaje mediante el envío de mensajes.

Caso 4 GE2EST008 – Estrategia de Anticipación

Caza vampiros

El “Caza Vampiros”, el niño, reprograma nuevamente las acciones a partir del uso de colisiones; esta vez, la protagonista es una pelota que debe golpear unos vampiros que están moviéndose de manera constante de lado a lado en la pantalla. El juego se inicia al oprimir la bandera verde. La pelota dice: «Lánzame». Acto seguido, se puede activar tocándola para que pueda rebotar rápidamente. En términos de pensamiento computacional, la lógica de ganar o perder está organizada en las consecuencias de la colisión. Pelota murciélago. Este diseño demuestra el uso de mensajes “condicionales”: “si choca” desaparece; si logra desaparecer a los tres murciélagos, gana, enviando el mensaje, además de cada vez que sucede el choque, a un personaje que se encuentra esperando a que le lleguen todos los mensajes para decir: “Otra vez”, lo que evidencia el grado avanzado de abstracción al que llegó el estudiante. Su paso de 4/12 a 10/12 (+6 puntos) indica, además, que pensar previamente las reglas de juego y tener una retroalimentación constante e inmediata sobre varias estructuras algorítmicas aportan a la consolidación del pensamiento computacional (Ver Figura 6.8).

Figura 6.8. Proyecto "Caza vampiros": Código de los personajes en Scratch Jr.



Nota. Captura de pantalla del juego "Caza vampiros", elaborado por la estudiante GE2EST008. (a) programación de la pelota con controles de movimiento; (b) vampiros con movimiento autónomo y reacción al choque con la pelota, lo que provoca su desaparición y el envío de mensajes; (c) programación del personaje que espera a que termine la partida para decir: “Otra Vez”. Elaboración propia, 2025.

6.4.3 Matriz analítica: caracterización de habilidades de PC en GE2

Habilidad PC	Descripción emergente en grupo juegos	Mecanismo del juego que lo facilita
Razonamiento lógico	Los estudiantes analizaban las reglas del juego (condiciones para ganar/peder) y las transformaban en lógica de la programación. Ejemplo: “si toca el objeto, entonces gana”.	Las reglas explícitas a seguir en el juego permiten una lógica clara a la hora de traducirlas a código.
Pensamiento algorítmico	Los estudiantes construían secuencias de instrucciones para alcanzar los objetivos del juego (moverse, detectar colisiones, cambiar el estado). Efectuaban patrones de manera sistemática.	El propósito bien definido del juego estructura la secuenciación de acciones
Descomposición de funciones	Los niños transformaban el problema en función del significado de cada elemento (jugador, enemigos, metas y reglas) programando cada parte de forma independiente	Las mecánicas de los juegos determinan por sí mismas los elementos del sistema
Abstracción	Los niños parametrizaban el código a partir de las variables numéricas de velocidad, distancia, tiempo de espera; entendían que cambiar un número cambia el comportamiento pero no la lógica	Las variables numéricas nos otorgan un camino concreto hacia la abstracción matemática
Patrones (generalización)	Los niños identificaban patrones grupales (el mismo movimiento para varios enemigos; la misma detección de la colisión para varios objetos) y los reutilizaban (Copian)	La eficiencia del juego genera la lógica del descubrimiento y la reutilización
Evaluación/ depuración	La respuesta inmediata (ganador/perdedor) actuaba como un mecanismo de depuración.Los niños iteraban de forma sistemática hasta alcanzar el objetivo	El éxito en el juego proporciona el criterio de validación (ganas, no ganas) claro y objetivo

6.4.4 El modelo emergente: el pensamiento computacional mediado por reglas y objetivos

El análisis del grupo de juegos alcanza un modelo bien definido en el que las reglas y objetivos del juego operan como organizadores de la lógica computacional. Este modelo opera mediante 4 mecanismos:

1. Objetivos explícitos. Cada una de las metas del juego (ganar/perder, alcanzar alguna meta) otorga un sentido claro a cada línea de código.

2. Abstracción paramétrica. Modificar parámetros numéricos (velocidad, distancia) para ajustar la complejidad del juego hace que los niños comprendan relaciones abstractas de las matemáticas, como la relación entre el número y la distancia.
3. Descomposición funcional. Las mecánicas de los juegos determinan componentes funcionales (control del jugador, enemigos, detección de colisiones) que operan de forma independiente.
4. Criterio de validación objetivo. La respuesta ganador/perdedor proporciona un criterio de evaluación más objetivo y motivador frente a las evaluaciones técnicas.
6. Implicación: en el grupo de juegos, el pensamiento computacional se desarrolla como un conjunto de habilidades lógico-matemáticas que permiten modificar y optimizar el código para alcanzar objetivos definidos y verificables.

6.5 Contraste cualitativo entre cuentos y juegos

6.5.1 Introducción

Este apartado responde al objetivo específico 3: contrastar el efecto de la creación de cuentos versus juegos interactivos que desarrollen el pensamiento computacional. Este contraste no tiene lugar entre casos individuales, sino contrastando cómo cada modalidad organiza de forma diferente el desarrollo del pensamiento computacional en el grupo.

6.5.2 Comparativa de procesos (cognitivos)

Dimensión de análisis	Grupo de cuentos (GE1)	Grupo de juegos (GE2)
Principal organizador cognitivo	Base narrativa (causas-efectos temporales)	Base de reglas y de objetivos
Tipo de abstracción que predomina	Simbólica y emocional (las propiedades encontradas en lo visual representan estados de ánimo)	Paramétrica y simbólica matemática (códigos numéricos que representan conductas)
Criterio de validación	Coherencia narrativa ("¿tiene sentido la historia?")	Funcionalidad ("¿Funciona el juego de acuerdo con lo planeado?")
Tipo de iteración predominante	Verificación lineal (ir validando cada acción viendo qué ocurre antes de realizar el siguiente paso)	Experimentación cíclica (intentar una y otra vez para optimizar)
Tipo de error que impulsa la depuración	Incongruencia narrativa (el personaje no hace lo que sugiere la historia)	Fallo en el objetivo (En el juego no se pierde o se gana como se esperaba)
Patrón de razonamiento dominante	Transferencia de esquemas narrativos previos	Optimización lógica respecto a los objetivos explícitos
Relación con la complejidad del código	La narrativa invita a un código modular y legible	Las exigencias de optimización impulsan un código más conciso y parametrizado.

6.5.3 Semejanzas principales entre ambos grupos

A pesar de las diferencias cualitativas, ambos grupos mantuvieron algunos rasgos comunes en la construcción del pensamiento computacional:

1. Depuración como mecanismo central. En ambos grupos, la depuración no fue considerada una habilidad marginal, sino que ocupó un lugar destacado dentro del proceso de aprendizaje. El grupo avanzó mediante el ensayo y error, tanto inicial como posterior, así como en la reconfiguración de escenas de cuentos y en el balance entre niveles de dificultad (programación) de los juegos.

2. Requisito de andamiaje concreto. Ambos grupos necesitaron un andamiaje concreto de ideas generales para llegar a las ideas abstractas. El grupo de cuentos descubrió este andamiaje en la narrativa; el grupo de juegos, en las reglas explícitas y en las progresiones de ganar/perder.
3. Heterogeneidad de estrategias efectivas. En ambos grupos se evidenciaron múltiples vías válidas hacia una misma meta: planificación anticipada, ensayo y error resilientes, verificación reiterativa, procesamiento en lotes, etc.
4. Transferencia de saberes previos. Ambos grupos aplicaron a la nueva experiencia de programar, las estructuras cognitivas que previamente se habían interiorizado (narrativas para cuentos y estructurales basadas en reglas, para juegos).

6.5.4 Diferencias características: dos modelos complementarios

Las diferencias que emergen entre estas dos actividades no son de jerarquía (una modalidad es “mejor” que la otra), sino de tipo complementario, pues cada una de las actividades favorece una serie diferente de aspectos relacionados con el pensamiento computacional.

El grupo de cuentos desarrolló con mayor énfasis:

- Abstracción simbólica (representación emocional mediante propiedades visuales).
- Modularización natural (división en actos y escenas).
- Razonamiento causal (relación antes–después en la trama).
- Expresión creativa y emocional.

El grupo de juegos desarrolló con mayor énfasis:

- Pensamiento algorítmico explícito (secuencias para alcanzar un logro).
- Optimización sintáctica (bloques menos numerosos, parámetros más ajustados).
- Razonamiento condicional (si–entonces en reglas de juego).

- Evaluación objetiva (ganar/perder como criterio de consecución)

Conclusión: ambas variantes desarrollan el pensamiento computacional, pero mediante caminos cognitivos distintos que sostienen diferentes andamiajes: en cuentos, el narrativo y en juegos, el objetivo estratégico.

6.6 Hallazgos generales transversales

El análisis cualitativo de ambos grupos de estudio destaca tres hallazgos transversales que trascienden la distinción entre cuentos y juegos.

6.6.1 La depuración como mecanismo central, no como habilidad periférica

En contraposición a concepciones clásicas en las que la depuración se clasifica como una técnica más, el análisis de datos cualitativos muestra que la depuración se encuentra como la bisagra central de la que se articula el resto de las habilidades de pensamiento computacional.

Los ciclos de depuración en ambos grupos fueron los momentos de mayor despliegue de actividad cognitiva:

- El razonamiento lógico aparece cuando el estudiante hace un diagnóstico de por qué algo no le funciona.
- La abstracción aparece cuando el estudiante generaliza desde un error particular a un principio más general.
- La descomposición aparecía en el momento en que el estudiante era capaz de aislar cuál era el componente que generaba la dificultad.
- El reconocimiento de patrones concretos aparecía cuando el estudiante era capaz de reconocer que ya había cometido el mismo error en situaciones anteriores.

Implicación pedagógica: se pierde una oportunidad de aprendizaje si se enseña la depuración sólo al final del proceso, ya que la depuración debe estar presente desde el inicio

y debe ser presentada como el motor legítimo del aprendizaje y no como un proceso de corrección que se presenta al final del proceso de aprendizaje.

6.6.2 **Abstracción como un reto constante que funcione como un andamiaje eficaz**.

Si bien la abstracción fue conceptualizada teóricamente como una habilidad de PC más, lo cierto es que el análisis cualitativo evidencia que es un reto constante en la educación inicial y que debe ser potenciado como andamiaje de forma sistemática durante toda la intervención, no solo en las fases iniciales.

Ejemplos de estos son:

- La necesidad de observar una misma acción repetidamente con diversos personajes antes de razonar que una de las acciones puede ser generalizable.
- La dificultad para ver que cambiar un parámetro numérico (de diez pasos a cinco pasos) requiere que la relación entre los pasos se mantenga; cambiar el número no es cambiar el comportamiento.
- La tendencia a utilizar visualizaciones o listas de propiedades, o incluso las reglas o estrategias, circunstancias del problema como soporte de razonamiento para los estados emocionales, secuencias o condicionales.

Implicación pedagógica: los docentes deben ofrecer experiencias con formas de variaciones sistemáticas (cambios repetidos) y explicar inequívocamente las relaciones entre la representación (numérica, de tamaño, de color y de emoción) y el comportamiento del sistema. La paciencia hacia la abstracción es lo fundamental en la enseñanza del pensamiento computacional en estas edades.

6.6.2 La heterogeneidad de los perfiles cognitivos justifica la existencia de múltiples caminos.

Los 8 alumnos analizados mostraron al menos ocho formas distintas de llegar a desarrollar el pensamiento computacional:

1. Verificación iterativa (validaciones paso a paso).
2. Razonamiento concreto, con un fuerte apoyo narrativo.
3. Transformación tardía mediada por emociones y relatos.
4. Planificación anticipadora detallada.
5. Prueba y error repetida, experimento resiliente.
6. Procesado en lotes, en búsqueda de la economía del código.
7. Anclaje atencional con un objetivo explícito de juego.
8. Estrategias de anticipación mediante la verbalización anticipada del algoritmo.

Todas estas estrategias resultaron efectivas; todos los alumnos mejoraron significativamente en la lista de cotejo en PC entre el pretest y el postest.

Implicación pedagógica: imponer una única estrategia "válida" (es decir, "primero planifico todo y programo posteriormente") puede contribuir a invisibilizar rutas de aprendizaje respetables. La evidencia cualitativa respalda la existencia de una pedagogía que reconozca, legitime y acompañe la diversidad de caminos alternativos.

7 Interpretación Teórica de los Resultados

7.1 Consideraciones iniciales

Los resultados obtenidos en los capítulos 5 y 6, revelan que la intervención pedagógica propuesta, fundamentada en la creación de cuentos y en la construcción de juegos interactivos en Scratch Jr. favorece el desarrollo de las habilidades del pensamiento computacional en los niños de educación inicial del colegio Nueva Candelaria. Los análisis cuantitativos mostraron diferencias significativas entre pretest y postest en todas las habilidades evaluadas, además de constatar la ausencia de diferencias estadísticamente significativas en los resultados entre los grupos experimentales; el análisis cualitativo proporcionó el conocimiento necesario para entender cómo se moldea cada habilidad de acuerdo con su modalidad.

Este capítulo realiza la interpretación de estos resultados a partir de las bases y del marco teórico que sirvieron de fundamentación, con el objetivo de situar la investigación en el área del pensamiento computacional en educación infantil, contrastar las similitudes y diferencias con investigaciones precedentes y explicar las aportaciones conceptuales y metodológicas que surgen a partir del ejercicio realizado.

7.2 Desarrollo del pensamiento computacional en la educación infantil: convergencias con el estado de la investigación.

7.2.1 Posibilidades del pensamiento computacional en edades infantiles

Los resultados cuantitativos de la intervención confirman que es probable promover significativamente el pensamiento computacional en niños de 5 y 6 años y, al mismo tiempo, son coherentes con las afirmaciones que se hacen desde anteriores investigaciones llevadas a cabo con población de educación inicial. Investigaciones, como las de Caballero y García (2019, 2021) o las de Delgado y Prado (2018), demuestran que competencias, tales como la

secuenciación, la descomposición, la identificación de patrones o la construcción de algoritmos se favorecen si se establecen experiencias de aprendizaje estructuradas y contextualizadas a partir de problemas con significado para el estudiante. La mejora observada entre la puntuación obtenida en el pretest y la del postest en esta investigación, indica que los participantes no solo se acercaron a un lenguaje de programación visual y sintáctico, sino que lograron interiorizar formas de razonamiento que son propias de las prácticas que la literatura atribuye al pensamiento computacional: organizar acciones en secuencias, descomponer tareas complejas en pasos sencillos, abstraer las relaciones que establecen entre la acción, la orden a ejecutar y el comportamiento de los personajes y, por último, evaluar y depurar sus propias soluciones; un resultado que se aproxima de forma acertada con la idea ampliamente defendida en la literatura de que el pensamiento computacional no es solo un conjunto de contenidos que deban ser adquiridos específicamente en niveles superiores, por el contrario, constituye una metodología que facilita la organización de las acciones con el pensamiento, que puede ser implementada desde los niveles iniciales, siempre que orientado, y de esta manera potenciar la construcción de andamiajes efectivos de comunicación.

7.2.2 Scratch Jr. como ambiente propicio para el preescolar

La elección de Scratch Jr. como ambiente de aprendizaje cuenta con el respaldo de múltiples trabajos que han documentado su adecuado nivel de desarrollo para niños de entre 4 y 7 años. Las investigaciones de Bers et al. (2015), Navarro (2020) y Papadakis et al. (2024) documentan cómo su interfaz gráfica, propia de la programación por bloques de colores o personajes animados, permite acceder a los conceptos de programación sin necesidad de tener un dominio de la lectura o escritura convencional.

Los resultados de esta tesis confirmaron parcialmente dichas conclusiones, pero lo hicieron desde dos perspectivas diferentes. En primer lugar, una mejora del rendimiento cuantitativo,

lo que significa que los y las estudiantes lograron comprender y manipular bloques de movimiento, de eventos, de mensajes o de parámetros temporales o espaciales. En segundo lugar, el análisis cualitativo mostro que se va más allá de la simple idea de que los niños «ensamblan bloques», hasta el extremo de que llegan a manipular estructuras relativamente avanzadas, como la sincronización mediante mensajes o la manipulación paramétrica de velocidad o distancia, de la mano de la motivación por narrar o jugar. En este sentido, Scratch Jr. no es un recurso por sí mismo, sino un recurso como medio para expresar computacionalmente las historias y los juegos interactivos construidos por los niños.

7.3 Modalidades didácticas en comparación: cuentos, juegos, actividades desconectadas

7.3.1 Cuentos digitales y programación inspirada en cuentos

Una de las decisiones que definieron el trabajo de la investigación fue la de organizar un grupo de trabajo en torno a la creación de cuentos interactivos. Esta decisión se apoya en trabajos que introducen la programación mediada con cuentos como mediador del pensamiento computacional. En concreto, en la investigación sobre *story-inspired programming* en educación infantil (Yang, Ren & Bers, 2023) se concluye que la narrativa -introducción, desarrollo, fin- proporciona una estructura temporal y causal que promueve la organización de secuencias de acciones, la identificación de relaciones causa-efecto y la asociación de significados a transformaciones abstractas que emergen en el espacio de la programación. Los resultados cualitativos de esta tesis apoyan este planteamiento. En el grupo de cuentos, los bloques secuenciales se presentan cómo parte de la estructura de los actos del relato; la descomposición es un resultado natural y se constata en la fragmentación de la historia en escenas; y la abstracción simbólica se evidenciaba cuando los niños utilizaban tamaño, color o cambio de espacio —escenario— para hacer referencia a estados

emocionales o transiciones de la trama. La depuración se cumplía en buena parte preguntándose si "la historia está saliendo como la imaginé", convirtiendo la coherencia narrativa en un criterio evaluativo, pero comprensible y motivante a la vez, en línea con lo reportado por Navarro (2020) en el contexto de educación infantil con Scratch Jr.

Esta coincidencia hace suponer que los cuentos digitales no sólo operan como un recurso motivacional sino que actúan como un marco semiótico a partir del cual los niños pueden poner en circulación la lógica computacional con estructuras de un pensamiento narrativo que dominaban ya en su vida diaria, (Papadakis, Kalogiannakis & Zaranis, 2024).

7.3.2 Juegos interactivos, reglas y pensamiento algorítmico

La segunda modalidad de la intervención reunió al grupo en torno a la creación de juegos interactivos. Esta propuesta dialoga con la extensa literatura sobre aprendizaje basado en juegos, que plantea que los videojuegos pueden constituirse en escenarios privilegiados de desarrollo de habilidades lógico-matemáticas, pensamiento algorítmico y perseverancia frente al error (Lin et al., 2020; Dağ, 2023).

Los resultados cualitativos de esta investigación muestran que en el grupo de juegos el pensamiento de los estudiantes se organiza en torno a las reglas, los objetivos y los estados de ganar/perder. La lógica condicional aparece cuando los niños formulan, explícita o implícitamente, estructuras del tipo "si el jugador toca el pez, gana; si toca la culebra, pierde". La descomposición toma una forma funcional, separando el sistema en jugador, enemigos, premios y condiciones de colisión. La abstracción se refleja, especialmente, a través de la manipulación de parámetros numéricos para regular la distancia, tiempo es espera y cantidad de veces que se puede repetir una acción (bloque), lo que aproxima a los estudiantes hacia una concepción más matemática de la relación número - comportamiento.

Frente al grupo de cuentos, en la depuración, aquí la guía es un criterio de éxito objetivo: el juego ha de funcionar de acuerdo con las reglas esperadas. El error deja de ser incoherencia narrativa y se convierte en un fallo funcional (no se puede ganar; el personaje no actúa como se espera), haciendo énfasis, de este modo, en una concepción de la evaluación muy centrada en el rendimiento del sistema.

Estas consideraciones coinciden con las que informan del potencial de aprendizaje basado en juegos para estimular competencias de planificación, ensayo, ajuste y optimización de estrategias en situaciones que implican la resolución de problemas (Lin et al., 2020; Dağ, 2023).

7.3.3 Articulación con actividades desconectadas

El diseño de la investigación propuso una articulación entre actividades desconectadas (en este caso la tarea de "Robot Humano") con el trabajo posterior en Scratch Jr. Esta articulación está en sintonía con revisiones que reflejan la complementariedad de las actividades "unplugged" con experiencias de programación visual (Papadakis, Kalogiannakis & Zaranis, 2024). Las actividades desconectadas permiten el trabajo de aspectos psicomotores, espaciales y de direccionalidad que son prerrequisitos necesarios para interpretar correctamente instrucciones simbólicas, mientras que el entorno digital es el lugar donde se puede trabajar rápidamente, tener una visualización rica y ser extremadamente motivante.

Los datos cualitativos de esta tesis señalan, precisamente, que este paso de la corporeidad al símbolo es una de las críticas que conviene tener en cuenta en el desarrollo del pensamiento computacional en etapas tempranas. La dificultad inicial con la lateralidad y la necesidad de "ponerse en el lugar del robot" se va transformando en la posibilidad de anticipar movimientos y resultados dentro de un entorno digital, sin tener que validar cada

orden con el cuerpo. Desde un punto de vista teórico, se puede leer este proceso como un paso de la acción concreta a la representación internalizada, en coherencia con marcos constructivistas y socioculturales del desarrollo cognitivo (Piaget, 1975; Vygotsky, 1978).

7.4 Interpretación de los hallazgos cualitativos en el marco del pensamiento computacional

7.4.1 Razonamiento lógico y relaciones causa-efecto

El razonamiento lógico se materializó de forma transversal a ambos grupos, pero usando un andamiaje en organizadores diferentes. En el grupo de cuentos las cadenas lógicas se apoyaron en la narración: “si el personaje está triste, entonces llora"; "si aparece el hada, entonces el dragón se hace pequeño”. En el grupo de los juegos, las relaciones causas-efecto se vincularon con las reglas de la actuación: “si el buzo toca el pez, gana; si toca la serpiente, pierde”.

Desde el marco teórico del pensamiento computacional, estas expresiones de los niños pueden ser entendidas como manifestaciones puestas en lenguaje de la lógica condicional, e incluso si la herramienta Scratch Jr. no tiene bloques explícitos de “si/entonces”. El planteamiento de que los niños internalicen relaciones antes de saber formalmente la sintaxis de las estructuras condicionales refuerza la idea, planteada en la literatura (Wing, 2006; Brennan & Resnick, 2012), de que el pensamiento computacional no es sólo saber trabajar ciertos bloques, sino que requiere el desarrollo de esquemas lógicos que después puedan ser formalizados en diferentes lenguajes de programación.

7.4.2 Descomposición y modularización del problema

La descomposición es la capacidad de poder dividir un problema complejo en partes pequeñas, que puedan ser tratados según sus características (Brennan & Resnick, 2012), se encuentra en la organización de escenas narrativas, pero también en la separación de

componentes de un juego. En el grupo de cuentos, cada página del relato funciona como un módulo con su propio conjunto de scripts, y en el grupo de juegos, cada tipo de objeto (jugador, enemigo, premio) constituye en sí mismo una unidad funcional de la programación.

Teóricamente, este dato sugiere que la descomposición puede sustentarse en una determinada estructura cultural histórica: en el esquema narrativo clásico de “inicio–nudo–desenlace”, de un lado, y en las categorías “jugador–reglas–objetivos”, del otro lado. La principal contribución de la investigación en curso consiste en demostrar que estas estructuras, propias del mundo cotidiano de los niños, pueden ser aprovechadas como organizadores naturales para trabajar problemas algorítmicos (Vygotsky, 1978).

7.4.3 Abstracción: de lo corporal concreto a lo simbólico digital

La abstracción surge en esta investigación como una de las habilidades de mayor dificultad, así como lo evidencian investigaciones previas de educación inicial (Papadakis et al., 2024). Por el contrario, el enfoque cualitativo permitió observar diferentes caminos hacia ella. En el grupo de cuentos, los niños llegan a utilizar propiedades visuales como el tamaño o el color para significar los estados anímicos o en qué momento del cuento se encuentran, lo que se traduce en un proceso que toma como referencia la emoción vivida para transformarla en su representación simbólica en la pantalla. En el grupo de juegos, la abstracción se relaciona con la utilización de parámetros numéricos para regular la velocidad, así como la distancia o la duración de las esperas, al establecer relaciones que involucran cantidades y comportamientos relacionados con dicha parametrización.

Desde el marco teórico, estos procesos pueden interpretarse como instancias de abstracción semiótica y paramétrica, respectivamente. En ambos supuestos, el niño ya no ve el código como una mera sucesión de "mandatos", sino que empieza a tomar conciencia de que determinados símbolos (un número, un tamaño, un color) condensan una serie de

significados y de posibilidades de variación. La evolución de ir desde lo concreto hacia las formas de representación “abstractas” es evidencia de un progresivo desarrollo de la capacidad de operar con modelos internos del sistema, más allá de la acción in situ sobre los objetos, como bien lo plantea la teoría constructivista del desarrollo cognitivo (Piaget, 1975).

7.4.4 Evaluar y depurar como elemento articulador

El marco teórico suele presentar la evaluación y la depuración como una de las dimensiones del pensamiento computacional (Brennan & Resnick, 2012; Wing, 2006). Sin embargo, los datos que nos ha proporcionado esta investigación hacen pensar que, dentro de la educación inicial, la depuración actúa como elemento articulador del resto de las habilidades. Es en los momentos de prueba y error donde se ponen en marcha el razonamiento lógico (diagnosticar el porqué de una falla), la descomposición (aislar qué parte del código no funciona), la abstracción (generalizar a partir de una falla concreta) y el reconocimiento de patrones (detectar errores que aparecen de forma recurrente).

La lectura teórica de este fenómeno permite poner en conexión la depuración con procesos de metacognición y autorregulación (Flavell, 1979). Cuando las niñas y los niños verbalizan anticipaciones, revisan su propio código o discuten con sus pares por qué el personaje no actúa como esperaban, están ejercitando el desarrollo de la reflexión sobre su propio pensamiento. En este sentido, la depuración no es solamente un procedimiento técnico, sino un espacio privilegiado para el desarrollo de funciones ejecutivas y de autoconciencia cognitiva.

7.5 Narrativa y juego como andamiajes cognitivos complementarios

Los resultados permiten concluir que cuentos y juegos son dos andamiajes cognitivos complementarios para el pensamiento computacional en la educación inicial. La narración proporciona un marco estructurado de la temporalidad y de la causalidad, lo cual es

potencialmente adecuado para cuestiones como ordenación, la correspondencia de los estados o la representación simbólica de las emociones o las transformaciones (Yang, Ren & Bers, 2023). El juego aporta, por su parte, un marco de reglas y objetivos explícitos que pone el acento en la lógica condicional, el ajuste de las posibles estrategias y el ajuste paramétrico de los comportamientos (Lin et al., 2020).

Desde la mirada teórica, esto permite afirmar que no hay un modo cualitativamente superior sino caminos diferenciados que favorecen distintos componentes del pensamiento computacional. Esta constatación se encuentra en comunicación con las propuestas didácticas multimodales (Bers, 2020), en las que está previsto que los y las estudiantes vayan y vengan entre distintas formas de interacción (narrar, jugar, hacer una actuación corporal, programar) y que establezcan puentes entre ellas.

7.6 Aportes teóricos y metodológicos de la investigación

En concordancia con el estado del arte y con el marco conceptual, la investigación aporta varios elementos de interés:

1. Caracterización afinada de los procesos cognitivos en educación inicial. Más allá de dar cuenta de mejoras cuantitativas, el análisis cualitativo propone una descripción minuciosa de cómo emergen las habilidades de pensamiento computacional en acciones, verbalizaciones y productos digitales de niños de educación preescolar con edades entre los 5 y 6 años, pormenorizando el sentido operativo de estas habilidades en la primera infancia.
2. Articulación de modalidades narrativas y lúdicas. El análisis revela que se puede trabajar la computación a partir de cuentos o a partir de juegos en un mismo entorno de programación, mostrando diferentes, aunque igualmente válidas, rutas de acceso a la lógica algorítmica y a la abstracción.

3. La centralidad de la depuración como eje de atención. La investigación pretende entender la depuración no como un elemento más sino como el mecanismo que permite que las otras dimensiones del pensamiento computacional se integren y sistematicen, especialmente en el caso de los ambientes de formación de los niños de educación inicial.
4. Integración de lo corporal y de lo digital. Al articular actividades desconectadas y programación en Scratch Jr. el estudio apoya enfoques que consideran el pensamiento computacional un proceso encarnado, que va desde la acción física y la corporeidad hasta representaciones simbólicas cada vez más abstractas.
5. Reconocimiento de la diversidad de perfiles cognitivos. La identificación de múltiples estrategias efectivas (iteración en las verificaciones, planificación anticipada, experimentación intensiva, procesamiento en lotes, etc.) desafía visiones homogéneas del "buen programador" y sugiere la necesidad de ocupar marcos teóricos y pedagógicos que integren dicha heterogeneidad como un rasgo constitutivo en el aprendizaje del pensamiento computacional.

7.7 Proyección hacia las conclusiones e implicaciones pedagógicas

La interpretación de carácter teórico que se ha llevado a cabo en las líneas de este capítulo, sienta las bases para la obtención de conclusiones generales y de implicaciones pedagógicas. Por una parte, ha permitido enmarcar los resultados empíricos de la investigación con relación a los estudios previos y a los conceptos clave del pensamiento computacional, poniendo de manifiesto coincidencias, matices y aportaciones originales. Por la otra, le ha permitido poner de manifiesto la importancia de elaborar propuestas didácticas que combinen actividades desconectadas, narrativas y lúdicas en contextos de programación visual apropiados para la edad preescolar.

Sobre esta base, el siguiente capítulo recogerá de manera breve las conclusiones principales obtenidas a partir del análisis cuantitativo y del análisis cualitativo, y planteará implicaciones para la práctica docente en educación inicial, así como posibles líneas de trabajo venideras en torno a la integración del pensamiento computacional en el currículo escolar.

8 Conclusiones e Implicaciones Pedagógicas

8.1 Conclusiones Generales: Del Entorno Virtual a la Reestructuración Cognitiva

Al concluir esta investigación, se puede afirmar con total seguridad que la implementación didáctica sistemática de estrategias didácticas a partir del entorno de programación Scratch Jr. tiene un efecto positivo sobre el desarrollo del pensamiento computacional (PC) en la educación inicial, al tiempo que es estadísticamente significativo. El análisis de los datos obtenidos en la comparativa de los resultados del pretest y postest muestra un incremento considerable del rendimiento de los niños, aceptando la hipótesis de que el entorno virtual es más que una herramienta técnica, sino un agente que favorece de forma eficaz la activación de los procesos lógicos necesarios para plantear y resolver problemas complejos. Este avance no se limita a manifestar que los estudiantes y los docentes adquieran habilidades operativas, sino que representa un acceso hacia la reestructuración del conocimiento desde ideas intuitivas y fragmentadas hacia ideas algorítmicas completas.

Desde la dimensión cualitativa y en total acuerdo con el fenómeno del uso de los artefactos digitales producidos por los estudiantes, Scratch Jr. fue validado empíricamente como "objeto para pensar", siguiendo la teoría construccionista de Papert (1980). A partir de la intervención didáctica, se observó un tránsito evolutivo importante para la maduración de la cognición infantil: es decir, el recorrido desde una cognición corporal —a partir de la cual el niño vincula el movimiento físico y la referencia propioceptiva para validar conceptos

espaciales como la lateralidad— hacia un plano de independencia simbólica: a partir de ella emergen capacidades de orden superior como la planificación mental anticipada, la inhibición de la respuesta motora inmediata y la descentración cognitiva, ya que permite al niño anticipar la ejecución del código en su cabeza antes de plasmarla en la aplicación de programación.

8.2 Conclusiones sobre las Trayectorias de Aprendizaje: La Dualidad Pedagógica

En atención a lo dispuesto en el "Modelo de Trayectorias Convergentes", que forma parte de los resultados del modelo de análisis cualitativo profundo, se llega a considerar que el desarrollo del pensamiento computacional en la educación infantil no se realiza por una única ruta. Más bien, se ponen de manifiesto dos rutas diferentes de apropiación de la tecnología, las cuales son separadas de manera significativa por la motivación pedagógica implicada en la tarea que se propone:

1. Trayectoria Narrativa (Procesamiento Descendente): La estrategia que persigue la creación de cuentos y relatos digitales favorece la abstracción semiótica y la globalidad de la secuenciación en forma y contenido de base. En esta trayectoria, el código computacional se utiliza como un medio de expresión creativa y de comunicación. La necesidad comunicativa de los estudiantes, para elaborar diálogos coordinados y coherentes, construir secuencias temporales lógicas y ajustar la interacción entre personajes, lleva al descubrimiento orgánico y situado de conceptos de avanzada como la Sincronización de Procesos Paralelos. De esta forma, la utilización de los bloques con mensajes pasó de ser una simple instrucción técnica abstracta a una solución lógica para un problema narrativo, de tal modo que se mostró que el significado del contenido era anterior a la adquisición de la sintaxis computacional.

2. Trayectoria fundamentada en el juego (procesamiento ascendente): El diseño de retos de juego y mecánicas de juego contribuyó al progreso de otras dimensiones cognitivas, tomando como punto de partida la eficiencia algorítmica y siguiendo con el tratamiento sistemático de ellas. La naturaleza binaria (prendido y apagado) de la mecánica del juego, caracterizada por los estados de éxito y fracaso (ganar/perder, llegar y chocar), hizo que los estudiantes adoptaran una actitud analítica y rigurosa. Esta presión por realizar un rendimiento eficiente también contribuyó a la progresión en la que se daba el paso de la redundancia sintáctica (bloques simples de uso repetido) hacia la Abstracción Paramétrica, que podía manifestarse en la edición consciente de variables numéricas para optimizar las scripts. En el progreso que se plantea, el código como economía y la lógica como precisión son los valores que determinan el aprendizaje.

8.3 Acerca del Desarrollo de Habilidades Especiales: Matices y Diferentación.

El análisis detallado de las dimensiones del pensamiento computacional permite extraer conclusiones concretas respecto a la manera de enseñar las habilidades:

- Secuenciación y Eventos: Se puede concluir que la narración puede ser el andamiaje cognitivo más eficaz a la hora de comprender la causalidad asíncrona, en la medida en que los conceptos abstractos relacionados con "señales de envío y recepción" (eventos) se fueron asimilando de una manera notablemente naturalizada, anclados en la lógica social de un diálogo entre personajes o de causa-efecto en una historia. La estructura lineal y ramificada de la narración proporciona un contexto semántico que permite la asimilación de los flujos de control complejos que, de lo contrario, hubieran sido abstractos desde el nivel cognitivo de los participantes.
- Abstracción y Depuración: por el contrario, el entorno basado en retos y en el juego con metas fue más efectivo para el desarrollo de la resiliencia técnica y la depuración en

acción. En ese contexto, el error fue resignificado de forma radical: dejó de ser considerado un fracaso personal para ser considerado un dato del sistema que habla del estado del algoritmo. Y entonces, esto permitió que los niños fueran dejando lenta y progresivamente las estrategias de prueba y error basadas en la impulsividad, por una depuración que se basa en la observación de los patrones de fallo mediante la comprobación de hipótesis.

8.4 Implicaciones educativas: hacia una Didáctica de lo Integral

La principal aportación teórica y práctica que se articula en este estudio tiene que ver con la propuesta de una Didáctica Híbrida del Pensamiento Computacional en la Educación preescolar. Se sostiene que un currículo serio y fuerte no debería disociar ni jerarquizar la expresión creativa y la eficiencia técnica, sino que debe intentar unirse dialécticamente, por lo que la coherencia pedagógica sería utilizar la narrativa para dar sentido, propósito y contexto humano al código y, al mismo tiempo, utilizar las mecánicas del juego para dar estructura, desafío y rigor lógico, construyendo de esta manera una formación completa que retribuya la dimensión expresiva tanto como a la dimensión de ingeniería del pensamiento computacional.

Del mismo modo, se reconoce la condición del pensamiento computacional como una habilidad que se reconstruye socialmente. Los resultados muestran, en efecto, que el aprendizaje no se obtiene de forma aislada frente a la pantalla, sino en un ecosistema social. Las verbalizaciones entre pares, la colaboración espontánea, el andamiaje conducido por los docentes deberían orientarse como "una memoria externa" y un sistema de regulación imprescindible para el desarrollo de la autonomía particular de los estudiantes. Es por todo ello que la enseñanza del PC debería institucionalizarse como una nueva alfabetización digital que no sólo tiene como objetivo la futura empleabilidad, sino también el entendimiento crítico y la participación de los niños en el mundo contemporáneo.

8.4.1 Recomendaciones para la formación docente

La didáctica híbrida ofrece a la formación docente tres líneas de acción concretas. Primera, implementar talleres prácticos de actualización del docente de educación inicial enfocados en el ambiente virtual de Scratch Jr., integrando la creación de cuentos y juegos para acercar al estudiante al pensamiento computacional. Segundo, construir proyectos de aula que vinculen actividades "desconectadas" (juegos de papel, storyboards, dramatizaciones, canciones, dinámicas que vinculen la corporalidad) con la programación visual y de esta manera lograr que los futuros maestros comprendan de manera práctica los diferentes puntos de vista, de un mismo problema soportado por medios análogos y digitales. Tercero, fomentar espacios de reflexión entre pares para analizar videos y producciones de los niños, que les ayuden a identificar y evidenciar habilidades como la descomposición, la depuración y la abstracción, fortaleciendo sus competencias de observación y retroalimentación en términos de pensamiento computacional en la educación inicial.

8.5 Limitaciones y proyecciones a futuro

A pesar de que los resultados de esta investigación son concluyentes y significativamente estadísticos, es cierto que el tiempo de intervención resulta en una limitación metodológica para valorar la transferencia profunda de estas capacidades hacia otros campos de conocimiento académico y para valorar su sostenibilidad a largo plazo. La consolidación de estructuras cognitivas complejas requiere de procesos de maduración que superan el marco temporal de un estudio transversal.

En lo concerniente al componente cualitativo, el proceso de codificación temática fue realizado por un único investigador, sin cálculo de índices de confiabilidad entre codificadores (por ejemplo, alfa de Krippendorff o coeficientes de acuerdo). Esto forma una limitante metodológica, en tanto que la interpretación de las categorías depende en mayor grado del criterio del analista; se recomienda que en futuros estudios se incluyan al menos un

segundo codificador y procedimientos formales de estimación de la confiabilidad para mejorar la validez de los hallazgos cualitativos.

Por lo tanto, se proponen futuras líneas de investigación vinculadas al desarrollo de estudios longitudinales que permitan seguir su evolución en grados posteriores. Del mismo modo, resulta básica la aplicación y validación de este modelo híbrido para contextos de mayor heterogeneidad tecnológica y cultural, a fin de comprobar la universalidad de las conclusiones obtenidas y consolidar el pensamiento computacional como herramienta de inclusión social y de empoderamiento cognitivo universal, capaz de cerrar brechas de inequidad desde la primera infancia.

Referencias

Bers, M., Strawhacker, A., Portelance, D., & Lee, M. (2015). *ScratchJr: Aprendizaje en la edad temprana mediante programación*. (Eduketa) Obtenido de http://eduteka.icesi.edu.co/articulos/scratchjr2

Bordignon, F., & Iglesias, A. (2020). *Introducción al pensamiento computacional.* Universidad Pedagógica Nacional de Argentina. Obtenido de http://biblioteca.clacso.edu.ar/Argentina/unipe/20200414101408/introduccion-pensamiento-computacional.pdf

Braun, V., & Clarke, V. (2006). Using thematic analysis in psychology. *Qualitative Research in Psychology, 3*(2), 77-101. doi:10.1191/1478088706qp063oa

Caballero Gonzalez, Y. A., & García Valcárcel Muñoz, A. (2019). *Revista Latinoamericana de Tecnología Educativa*, 136 - 141. doi:http://dx.medra.org/10.17398/1695-288X.18.2.133

Caballero, Y., & García, A. (2019). Fortaleciendo las habilidades de pensamiento computacional en Educación Infantil: Experiencia de aprendizaje mediante interfaces tangibles y gráficas. *Revista Latinoamericana de Tecnología Educativa, 18*(2). Obtenido de https://relatec.unex.es/article/view/3577/2386

Campbell, D., & Stanley, J. (2011). *Diseños experimentales y cuasiexperimentales en la investigación social.* Amorrortu Editores.

Computing At School. (2015). *Pensamiento computacional guía para profesores.* Computing At School. Obtenido de https://es.slideshare.net/slideshow/pensamiento-computacional-gua-para-profesores/62028429#16

Cortés, M., & León, M. (2004). *Generalidades sobre la Metodología de la Investigación.* UNACAR. Obtenido de http://www.unacar.mx/contenido/gaceta/ediciones/metodologia_investigacion.pdf

Creswell, J., & Plano Clark, V. (2011). *Designing and conducting mixed methods research* (2nd ed. ed.). SAGE Publications.

Dağ, F. (2023). The effect of an unplugged coding course on primary school students' computational thinking skills, collaboration, and communication competencies. *Journal of Computer Assisted Learning, 39*(2), 372-386. doi:10.1111/jcal.12850

Delgado, J., & Prado, J. (2018). Pensamiento computacional a través de estimulación sensorial en niños de transición. Obtenido de https://sired.udenar.edu.co/6287/

Educación y aprendizaje en la primera infancia. (2019). Obtenido de https://www.unicef.org/lac/desarrollo-en-la-primera-infancia-y-educacion-inicial

Espino, E., & González, C. (2015). Estudio sobre las diferencias de género en las competencias y en las estrategias educativas para el desarrollo del pensamiento computacional. *RED- Revista de Educación a Distancia, 46*(12), 1-20. Obtenido de https://www.um.es/ead/red/46/espino_gonzalez.pdf

Flannery, L., Kazakoff, E., Silverman, B., Bers, M., & Resnick, M. (2015). *Diseñando ScratchJr: apoyo al aprendizaje en la edad temprana*. (Eduteka) Obtenido de http://eduteka.icesi.edu.co/articulos/scratchjr

Flavell, J. H. (1979). Metacognition and cognitive monitoring: A new area of cognitive-developmental inquiry. American Psychologist, 34(10), 906–911.

Guías de Scratch Junior. (s.f.). (Código 21) Obtenido de https://codigo21.educacion.navarra.es/autoaprendizaje/guias-de-scratch-junior/

Hernández, S. (2014). *Metodología de la investigación* (6ta ed. ed.). McGRAW-HILL / interamericana editores, s.a. de c.v.

Herrera, B. (2017). Propuesta de Programa Formativo en Pensamiento Computacional para Docentes de Primaria del Colegio Simón Bolívar del municipio de Dajabón, República Dominicana. Obtenido de https://repositorio.grial.eu/server/api/core/bitstreams/fa851fea-861d-441b-93f5-9ff1f4510b50/content

Ley 1804. Se establece la política de Estado para el desarrollo integral de la primera infancia, de cero a siempre, y se dictan otras disposiciones. (2016). Obtenido de https://www.suin-juriscol.gov.co/viewDocument.asp?ruta=Leyes/30021778

Lin, S., Chien, S., Hsiao, C., Hsia, C., & Chao, K. (2020). Enhancing computational thinking capability of preschool children by game-based smart toys. *Electronic Commerce Research and Applications, 44*, 101011. doi:10.1016/j.elerap.2020.101011

Louka, M., Papadakis, S., & Kalogiannakis, M. (2022). A systematic review of computational thinking in early childhood education. *Journal of Educational Computing Research, 60*(5), 1216-1250. doi:10.1177/07356331211057899

Ministerio de Educación Nacional. (2008). *Ser competente en tecnología: Una necesidad para el desarrollo. Orientaciones generales para la educación en tecnología (Guía No. 30)*. Imprenta Nacional de Colombia.

Namakforoosh, M. (2000). *Metodología de la investigación. Área de ciencias sociales.* Limusa.

Navarro, C. (2020). Scratch Jr.: Aprendiendo a programar y programando para aprender. *Observatorio de Tecnología educativa, 28*, 1-11. Obtenido de https://intef.es/wp-content/uploads/2020/11/07_Observatorio_Scratch_Jr_v2.pdf

Papadakis, S., Kalogiannakis, M., & Zaranis, N. (2024). Enhancing computational thinking in early childhood education through ScratchJr integration. *Heliyon, 10*(9), e30482. doi:10.1016/j.heliyon.2024.e30482

Patton, M. (2002). *Qualitative research and evaluation methods* (3rd ed. ed.). SAGE Publications.

Piaget, J. (1977). El desarrollo del pensamiento: Equilibración de las estructuras cognitivas. Emecé. (Obra original publicada en 1975)

ScratchJr: Bloques de programación. (2015). (Eduteka) Obtenido de https://eduteka.icesi.edu.co/articulos/scratchjr-tarjetas-imprimibles

ScratchJr: Evaluación. (2016). (Eduteka) Obtenido de http://eduteka.icesi.edu.co/articulos/scratchjr-evaluacion

Strawhacker, A., Lee, M., & Bers, M. (2018). Teaching tools, teachers' rules: Exploring the impact of teaching styles on young children's programming knowledge in ScratchJr. *International Journal of Technology and Design Education, 28*(2), 347-376. doi:10.1007/s10798-017-9400-9

Sun, L., Zhou, Z., & Li, Y. (2024). Comparative experiment of the effects of unplugged and plugged-in programming activities on the computational thinking skills of lower

primary school students. *Thinking Skills and Creativity, 51*, 101444. doi:10.1016/j.tsc.2024.101444

TIC, M. (2019). *Alianza internacional promoverá el pensamiento computacional en Colombia*. (MinTIC Colombia) Obtenido de https://mintic.gov.co/portal/inicio/Sala-de-Prensa/Noticias/101061:Alianza-internacional-promovera-el-pensamiento-computacional-en-Colombia

UNICEF. (2019). *Niños, niñas y adolescentes en línea.* Obtenido de https://www.unicef.org/chile/media/3096/file/lacro-en-linea.pdf

Universidad Pedagógica Nacional (Argentina). (2019). *Un saber de época. Proyecto de pensamiento computacional*. Obtenido de Colección de tareas para el desarrollo del Pensamiento Computacional en estudiantes de nivel secundario: https://unipe.educar.gob.ar/unipe/seccion/7/unipe

Vygotsky, L. S. (1978). Mind in society: The development of higher psychological processes. Harvard University Press.

Yang, W., Ren, Y., & Bers, M. (2023). The impact of story-inspired programming on preschool children's computational thinking. *Computers & Education, 197*, 104732. doi:10.1016/j.compedu.2023.104732

Zapata, M. (2015). Pensamiento computacional: una nueva alfabetización digital. *Revista de Educación a Distancia, 46*(4), 1-47. Obtenido de https://www.um.es/ead/red/46/zapata.pdf

Zapata-Ros, M., González-Cabrera, C., & Caicedo-Tamayo, A. (2021). Integración del pensamiento computacional en la educación primaria y secundaria en Latinoamérica: Una revisión sistemática. *Revista de Educación a Distancia (RED), 21*(66), 1-31. doi:10.6018/red.485321

ANEXOS

Anexo 1. Lista de cotejo de habilidades de pensamiento computacional

Instrumento para la evaluación de habilidades de pensamiento computacional en niños de educación inicial

Código Estudiante	Grupo	Fecha	RL-1	RL-2	PA-1	PA-2	D-1	D-2	A-1	A-2

Descripción de los criterios de evaluación:

Razonamiento Lógico (RL)

- RL-1: Identifica causa y efecto. Conecta las acciones con sus consecuencias y anticipa los resultados de las instrucciones que programa.
- RL-2: Predice resultados. Anticipa lo que ocurrirá al ejecutar una secuencia de comandos antes de probarla.

Pensamiento Algorítmico (PA)

- PA-1: Ordena instrucciones. Construye secuencias de pasos ordenados para resolver un problema específico.
- PA-2: Comunica el procedimiento. Verbaliza o explica el algoritmo que diseñó para lograr el objetivo.

Descomposición (D)

- D-1: Divide en partes. Identifica y separa las partes que forman el problema o proyecto.
- D-2: Resuelve cada parte. Trabaja cada componente del problema por separado antes de integrarlos.

Abstracción (A)

- A-1: Identifica lo esencial. Reconoce los elementos clave del problema y descarta la información que no es relevante.
- A-2: Comprende símbolos. Entiende cómo las acciones se representan digitalmente (como bloques o íconos) y las traduce a conceptos concretos.

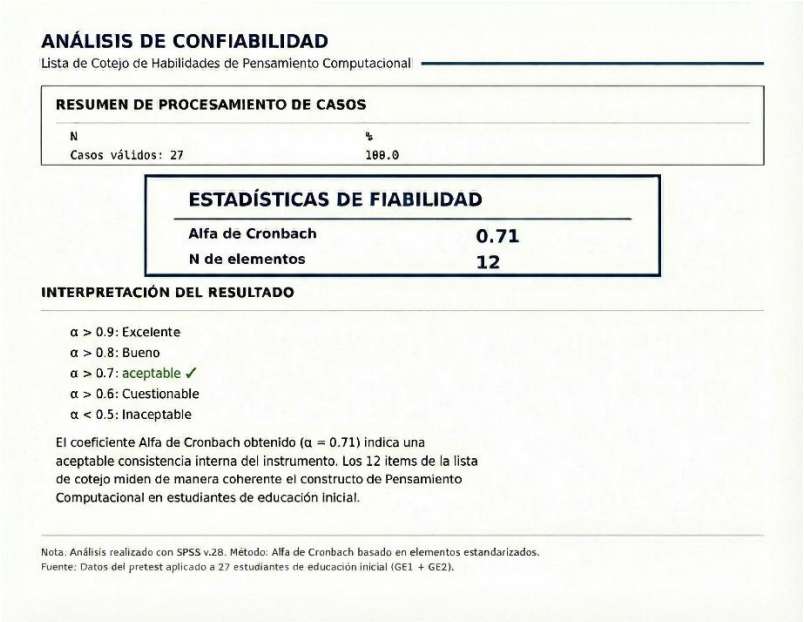
Generalización de Patrones (GP)

- GP-1: Reconoce patrones. Identifica secuencias o estructuras que se repiten en el problema.
- GP-2: Aplica patrones. Usa soluciones que le hayan funcionado antes en problemas similares.

Evaluación y Depuración (EV)

- EV-1: Prueba y detecta. Ejecuta su programa, identifica errores y nota cuando algo no funciona como esperaba.
- EV-2: Corrige. Hace cambios en el código o en la secuencia de instrucciones para solucionar el problema encontrado.

Nota. Este instrumento fue adaptado a partir de los marcos de Caballero-García (2019), Delgado-Prado (2018) y Bers et al. (2015) y validado mediante el análisis de la confiabilidad interna (α de Cronbach = 0,71). Se aplica en formato de pretest y postest durante las actividades de programación en Scratch Jr.



ANÁLISIS DE CONFIABILIDAD

Lista de Cotejo de Habilidades de Pensamiento Computacional

RESUMEN DE PROCESAMIENTO DE CASOS

	N	%
Casos válidos: 27		100.0

ESTADÍSTICAS DE FIABILIDAD

Alfa de Cronbach	0.71
N de elementos	12

INTERPRETACIÓN DEL RESULTADO

α > 0.9: Excelente
α > 0.8: Bueno
α > 0.7: aceptable ✓
α > 0.6: Cuestionable
α < 0.5: Inaceptable

El coeficiente Alfa de Cronbach obtenido (α = 0.71) indica una aceptable consistencia interna del instrumento. Los 12 ítems de la lista de cotejo miden de manera coherente el constructo de Pensamiento Computacional en estudiantes de educación inicial.

Nota. Análisis realizado con SPSS v.28. Método: Alfa de Cronbach basado en elementos estandarizados.
Fuente: Datos del pretest aplicado a 27 estudiantes de educación inicial (GE1 + GE2).