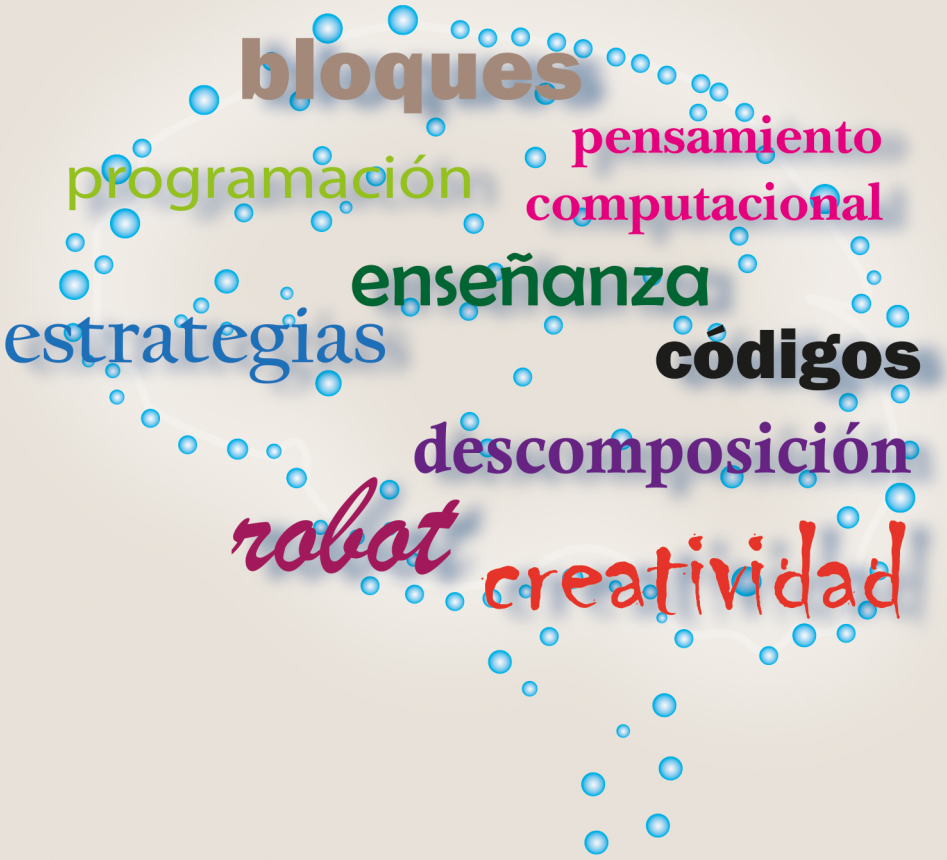
Pautas teóricas de la descomposición

Fundamentos, estrategias y aplicaciones en el aula

Luis Daniel Lozano Flores



Universidad de Guadalajara

Pautas teóricas de la descomposición

Fundamentos, estrategias y aplicaciones en el aula

AGRADECIMIENTOS

Extiendo mi agradecimiento más profundo y sincero al Dr. Antonio Ponce por su orientación experta, su rigor intelectual y su paciencia inagotable durante el proceso de esta obra. Sus valiosos consejos y su visión crítica fueron esenciales para estructurar y dar forma a este libro, guiando el proyecto hasta su feliz culminación.

A Caro y Regina, cuya luz me inspira a creer que todo sueño tiene un horizonte posible si se persigue con pasión.

Pautas teóricas de la descomposición

Fundamentos, estrategias y aplicaciones en el aula

Luis Daniel Lozano Flores

Centro Universitario de Los Altos
2025

Esta publicación ha sido arbitrada favorablemente por pares académicos; mediante arbitraje doble ciego, el expediente que lo respalda se conserva en la Coordinación de Investigación del Centro Universitario de los Altos de la Universidad de Guadalajara.

Se publica con el aval de la institución dictaminadora.

Primera edición, 2025

Centro Universitario de Los Altos
Av. Rafael Casillas Aceves 1200
47620, Tepatitlán, Jalisco, México
Consulte nuestro catálogo en: www.cucsh.udg.mx

ISBN: 978-607-581-838-2

Editado y hecho en México
Edited and made in Mexico

Índice

Introducción

El pensamiento computacional (PC) en el ámbito educativo no debe entenderse únicamente como una habilidad técnica vinculada a la programación, sino como un marco conceptual que posibilita el análisis, la estructuración y la resolución de problemas complejos. Este enfoque cobra especial relevancia al ofrecer una perspectiva metodológica que conecta la comprensión teórica con la acción práctica, a través de procesos como la descomposición. Dicho proceso permite identificar y reorganizar componentes de problemas, estableciendo un puente entre las dinámicas del pensamiento abstracto y la acción concreta, uniendo así dimensiones clave del aprendizaje y la resolución de problemas.

Sin embargo, la integración del PC en el ámbito educativo enfrenta desafíos considerables, que van más allá de la disponibilidad de recursos tecnológicos. Su implementación exige una comprensión profunda de los procesos cognitivos y pedagógicos que fundamentan prácticas como la descomposición, donde se fomenta no solo la resolución eficaz de problemas, sino también el desarrollo de habilidades de reflexión y análisis crítico. La descomposición, como proceso central del PC, promueve un aprendizaje basado en la fragmentación estratégica de problemas, lo que permite abordarlos de manera más comprensible y efectiva, conectando directamente con las competencias cognitivas necesarias para la educación actual.

El PC no constituye una solución única para los problemas educativos, pero representa una propuesta que, correctamente contextualizada, puede enriquecer las prácticas pedagógicas al articular teorías y estrategias que fomenten aprendizajes significativos. Este libro se posiciona como una contribución para comprender y aprovechar el potencial del PC, con un énfasis en la des-

composición, al tiempo que reconoce las complejidades inherentes a su implementación en diferentes contextos educativos. A través de una aproximación crítica y reflexiva, esta obra busca no solo proveer herramientas prácticas, sino también invitar a un diálogo más amplio sobre cómo el PC puede integrarse en los marcos pedagógicos existentes de manera consciente y transformadora.

La descomposición, entendida como un proceso inherente al pensamiento computacional, se constituye en un eje articulador entre la teoría y la práctica educativa. Al descomponer un problema en partes manejables, no solo se fomenta la comprensión de su estructura interna, sino que se facilita el diseño de soluciones adaptadas a las particularidades del contexto. Este enfoque encuentra resonancia en la pedagogía contemporánea al priorizar el aprendizaje significativo y contextualizado, donde los estudiantes no son meros receptores de conocimiento, sino actores activos que interactúan con su entorno para construir soluciones coherentes y fundamentadas.

En este marco, la relevancia del PC y, en particular, de la descomposición, radica en su capacidad para dialogar con teorías pedagógicas fundamentales como el constructivismo y el socio-constructivismo, al tiempo que responde a las necesidades de un entorno educativo en constante transformación. La descomposición no solo permite abordar problemas con mayor claridad, sino que promueve un aprendizaje que trasciende las barreras disciplinares, integrando herramientas tecnológicas y estrategias cognitivo-procedimentales que enriquecen la experiencia educativa. Así, se convierte en un medio para fortalecer competencias como la abstracción, el razonamiento lógico y la evaluación crítica.

La implementación de estas ideas en el aula representa un desafío significativo, pero también una oportunidad inigualable para replantear los objetivos y métodos de la enseñanza. Este libro busca no solo describir los fundamentos teóricos del pensamiento computacional y la descomposición, sino también proponer estrategias concretas que orienten a los docentes en su práctica diaria. La intención no es simplificar la complejidad del aprendizaje, sino ofrecer un marco que permita abordarlo con mayor claridad y eficacia, considerando las particularidades de cada contexto educativo y la diversidad de experiencias que en ellos se generan.

En síntesis, la descomposición como proceso central del pensamiento computacional no solo aporta una metodología para enfrentar problemas complejos, sino que también redefine la relación entre estudiantes, docentes y conocimiento en el aula. Su inclusión en la práctica educativa no debe entenderse como una simple incorporación de tecnología, sino como un replanteamiento profundo de las dinámicas de aprendizaje, donde se promueva la reflexión, la creatividad y la capacidad de conectar ideas aparentemente dispares. Este enfoque, lejos de ser una moda pedagógica, se configura como una respuesta teórica y práctica a las demandas de una educación que busca formar sujetos críticos, autónomos y preparados para un mundo en constante transformación.

El propósito de este libro es, precisamente, resaltar la importancia de este proceso, no como un fin en sí mismo, sino como un medio para enriquecer las experiencias educativas. Al explorar la descomposición desde sus fundamentos teóricos hasta sus aplicaciones prácticas, se busca dotar a los lectores de herramientas que les permitan repensar su papel como agentes de cambio en el aula. Así, el pensamiento computacional y la descomposición no se limitan a ser habilidades técnicas, sino que emergen como perspectivas integradoras que contribuyen a una educación más inclusiva, reflexiva y orientada hacia la construcción colectiva del conocimiento.

Esta obra trasciende la mera transmisión de conocimientos sobre el pensamiento computacional y la descomposición; busca construir un puente entre la teoría y la práctica, ofreciendo a los lectores un marco analítico que no solo explique, sino también inspire. En lugar de abordar la tecnología educativa como una herramienta estática, este libro la conceptualiza como un entorno dinámico donde los procesos cognitivos, las estrategias pedagógicas y las interacciones humanas convergen para dar forma a nuevas maneras de aprender y enseñar.

Al enfocarse en la descomposición como eje articulador, el texto plantea una visión integrada que permite a los educadores y estudiantes comprender el pensamiento computacional no como una habilidad aislada, sino como una metodología que se enriquece con la reflexión crítica y la acción contextualizada. A través de este enfoque, se pretende que los lectores desarrollen no solo una comprensión técnica, sino también una perspectiva que valore la capaci-

dad de adaptación y de generar conexiones significativas entre conocimientos diversos.

Además, a ser una invitación para la innovación pedagógica este libro aspira a combinar fundamentos teóricos sólidos con propuestas prácticas concretas, busca empoderar a los docentes como agentes de cambio, capaces de transformar el aula en un espacio donde el pensamiento computacional se traduzca en oportunidades para el desarrollo de habilidades transversales, como el trabajo colaborativo, el pensamiento crítico y la resolución creativa de problemas. En este sentido, el propósito del libro no solo reside en ampliar los horizontes del pensamiento computacional, sino también en contribuir a la construcción de una educación más equitativa y pertinente.

La estructura de este libro ha sido diseñada cuidadosamente para ofrecer una visión progresiva y profunda del pensamiento computacional, con énfasis en el proceso de descomposición. Cada capítulo establece los fundamentos necesarios para abordar el siguiente, asegurando que el lector construya un entendimiento integral del tema. Este enfoque permite no solo explorar los conceptos desde una perspectiva teórica, sino también reconocer sus aplicaciones y retos en contextos educativos concretos.

El Capítulo 1 introduce al lector al pensamiento computacional (PC) como una metodología fundamental para enfrentar los retos del aprendizaje en el siglo XXI. En este apartado, se analiza el concepto de PC desde sus componentes esenciales: la descomposición, la abstracción, el razonamiento algorítmico y la evaluación. Además, se ofrece una reflexión sobre cómo estos procesos contribuyen al desarrollo de habilidades cognitivas avanzadas que trascienden el ámbito de la programación. También establece las bases epistemológicas de la obra, posicionando el PC como un enfoque no solo técnico, sino profundamente ligado al pensamiento crítico y creativo, haciendo énfasis en su relevancia para la educación básica y su potencial como herramienta formativa en múltiples contextos.

El Capítulo 2 se dedica exclusivamente al proceso de descomposición, presentado como el eje central del pensamiento computacional. Aquí, el lector encontrará un análisis exhaustivo de las diversas formas en que la descomposición puede manifestarse, desde su tipología hasta las estrategias que la sustentan. Este capítulo detalla cómo la descomposición permite desglosar

problemas complejos en componentes más manejables, facilitando su comprensión y resolución. Además, se profundiza en conceptos como los patrones puente, los objetos de anclaje y la descolocación, proporcionando un marco teórico que no solo clarifica el proceso, sino que también lo sitúa dentro de un contexto educativo y cognitivo. El capítulo culmina con una reflexión sobre cómo este proceso puede ser utilizado como una herramienta pedagógica transversal, capaz de enriquecer las prácticas docentes en el aula.

Estos dos capítulos iniciales forman la base conceptual del libro, estableciendo un lenguaje común y una comprensión compartida que será fundamental para explorar los aspectos más aplicados en los capítulos subsecuentes. Con un equilibrio entre rigor teórico y claridad expositiva, buscan preparar al lector para adentrarse en los capítulos posteriores, donde las aplicaciones prácticas y los hallazgos metodológicos enriquecen la visión integral del pensamiento computacional y la descomposición.

El Capítulo 3 amplía la discusión teórica hacia un terreno práctico al explorar el uso del pensamiento computacional en la robótica educativa. Este apartado ofrece un análisis detallado sobre cómo herramientas como el *Mbot* y el *software Mblock* pueden integrarse en contextos educativos para promover un aprendizaje significativo. A través de ejemplos concretos y casos representativos, se demuestra cómo el pensamiento computacional, en particular el proceso de descomposición, permite a los estudiantes enfrentarse a retos tecnológicos y creativos. Además, analiza las interacciones entre los conceptos teóricos presentados anteriormente y su materialización en el diseño y resolución de problemas a través de robots educativos. Aquí, el lector encontrará recursos, recomendaciones y reflexiones sobre el potencial pedagógico de la robótica como un medio para vincular lo abstracto con lo tangible.

El Capítulo 4 se centra en los hallazgos teóricos y metodológicos que sustentan esta obra. En este capítulo, se sistematizan las observaciones y análisis realizados durante el desarrollo de los conceptos clave del pensamiento computacional y la descomposición. Se profundiza en las relaciones entre los conceptos teóricos, su coherencia interna y su aplicabilidad en diversos contextos educativos. Este apartado también aborda las limitaciones del estudio, subrayando la necesidad de seguir investigando sobre el tema, pero destaca, a su vez, la solidez del marco teórico construido. Este capítulo funciona como un

espacio de reflexión crítica, donde el lector puede confrontar los planteamientos del libro con las realidades y retos del ámbito educativo.

Finalmente, el Capítulo 5 ofrece una propuesta pedagógica que sintetiza los aprendizajes y reflexiones de los capítulos anteriores. Este capítulo está diseñado como una guía práctica para docentes interesados en implementar el pensamiento computacional y la descomposición en sus aulas. Incluye ejemplos de planeaciones, actividades didácticas y estrategias que pueden adaptarse a diferentes niveles educativos y contextos. Con un enfoque en la aplicabilidad, este capítulo proporciona herramientas claras para traducir las teorías y conceptos en prácticas educativas que promuevan un aprendizaje significativo. Además, se reflexiona sobre cómo estas propuestas pueden ser el punto de partida para futuras adaptaciones y experimentaciones pedagógicas, alineadas con las necesidades de cada comunidad educativa.

En conjunto, estos capítulos estructuran un camino que lleva al lector desde los fundamentos teóricos del pensamiento computacional y la descomposición hasta su implementación práctica y reflexión crítica. Este enfoque progresivo asegura que el lector no solo comprenda los conceptos, sino que también los visualice como herramientas dinámicas y adaptables al complejo panorama educativo actual.

La audiencia objetivo de esta obra está cuidadosamente delimitada, con un énfasis particular en quienes trabajan directamente en el ámbito educativo. En primer lugar, este libro está dirigido a docentes de educación básica que buscan integrar herramientas innovadoras en sus prácticas pedagógicas. En un contexto donde las tecnologías emergentes ganan relevancia, los docentes enfrentan el desafío de transformar estas herramientas en instrumentos efectivos de aprendizaje. Les proporciona un marco conceptual y práctico que facilita la incorporación del pensamiento computacional y la descomposición en sus planeaciones y actividades diarias. Más que una guía técnica, esta obra busca convertirse en un aliado pedagógico que les permita abordar la complejidad del aprendizaje en un entorno tecnológico.

Asimismo, será de gran utilidad para formadores de docentes y estudiantes de pedagogía. Para quienes están en formación o buscan actualizar sus enfoques educativos, ya que ofrece un puente entre la teoría y la práctica. En ella se presentan fundamentos sólidos, estrategias didácticas y aplicaciones

concretas que pueden integrarse en cursos, talleres y proyectos de formación inicial y continua. Este enfoque asegura que los futuros docentes comprendan no solo los principios del pensamiento computacional, sino también cómo implementarlos eficazmente en contextos reales, fortaleciendo su preparación para las demandas actuales del aula.

Esta obra también se dirige a investigadores y diseñadores curriculares interesados en el estudio y aplicación del pensamiento computacional en la educación básica. Al proponer conceptos teóricos específicos, basados en un análisis riguroso, el libro abre caminos para nuevas investigaciones que profundicen en el impacto de la tecnología y la programación en el desarrollo cognitivo de los estudiantes. Además, los diseñadores curriculares encontrarán en esta obra una referencia valiosa para desarrollar planes de estudio que integren de manera efectiva el pensamiento computacional y la robótica educativa en los programas escolares. En suma, esta obra se posiciona como un recurso versátil, adaptado a las necesidades de distintos actores clave en el sistema educativo.

Este libro no pretende ser un compendio definitivo ni un manual rígido sobre el pensamiento computacional o la descomposición. Más bien, se concibe como una invitación abierta a la reflexión, la experimentación y el diálogo sobre el papel que estas habilidades pueden desempeñar en la educación básica. A lo largo de sus páginas, se busca provocar en el lector no solo la curiosidad por explorar nuevas herramientas y enfoques pedagógicos, sino también la voluntad de cuestionar y enriquecer las ideas aquí expuestas. En este sentido, la obra aspira a convertirse en un punto de partida para la innovación educativa, más que en un destino final.

En un mundo donde las soluciones prefabricadas rara vez son efectivas, esta obra propone al lector un modelo flexible y adaptable, construido sobre principios sólidos pero abiertos a reinterpretaciones y ajustes. Cada capítulo ofrece un marco que permite integrar los conceptos en prácticas concretas, sin imponer un enfoque único. De esta manera, se invita a los docentes y formadores a tomar lo que sea útil para sus contextos particulares, adaptarlo a sus necesidades y construir a partir de ello. En este proceso, el lector se convierte no solo en usuario de las ideas aquí desarrolladas, sino también en co-creador de nuevos significados y estrategias.

Finalmente, esta invitación al lector no se limita al aula. Más allá de los espacios educativos formales, los conceptos de pensamiento computacional y descomposición tienen el potencial de transformar la manera en que abordamos los problemas cotidianos, fomentando una mentalidad crítica, analítica y orientada a soluciones. Al sumergirse en esta obra, se espera que el lector no solo encuentre herramientas útiles para su labor profesional, sino también una perspectiva renovada sobre la educación, el aprendizaje y las posibilidades que ofrece el siglo XXI. En última instancia, el mayor deseo del autor es que este libro inspire no solo a enseñar de manera diferente, sino a pensar de manera diferente.

1. El Pensamiento Computacional como habilidad clave en el siglo XXI

En los últimos años, se ha debatido acerca del papel que desempeña la tecnología en el desarrollo de habilidades para el siglo XXI, es decir, este conjunto de competencias que se consideran esenciales en diversas situaciones del mundo globalizado y que además, requieren de demandantes habilidades tecnológicas. En este sentido, la creatividad, innovación, alfabetización digital, adaptabilidad, productividad, uso de Tecnologías de la Información y Comunicación (TIC), liderazgo, entre otros conceptos, son regularmente mencionados y analizados por la comunidad académica en congresos educativos, así como artículos de investigación (Partnership for 21st Century Learning, 2019).

El uso de inteligencia artificial y la repercusión de la robótica educativa han estado cada vez más presentes en nuestra vida cotidiana, un ejemplo común son, los robots que ayudan en quehaceres del hogar, incluso aquellos dedicados al entretenimiento, como la reproducción de contenido audiovisual por medio de comandos de voz. No obstante, la aparición de este tipo de tecnología, ha desencadenado una serie de debates acerca de cómo sacarle provecho en el ámbito educativo. Dichos debates se han situado desde propuestas pedagógicas, críticas teóricas, entre otros que, se ven reflejados en estados del arte, revisiones documentales, investigaciones aplicadas, así como seminarios y congresos organizados por expertos en el tema. Una de las líneas que cuenta con mayor crecimiento actualmente en la comunidad científica, es el concepto de "Pensamiento Computacional" (PC).

Este concepto ha surgido como una respuesta teórica, que busca resolver la problemática de la inclusión de la programación, los robots y dispositivos inteligentes en la escuela. Principalmente, en nivel primaria, ya que es ahí, donde se han posicionado el mayor número de estudios. Para ello, ha sido necesario utilizar una serie de componentes para llevarlo a la práctica, a éstos se

les conoce como procesos centrales o habilidades clave del PC, los cuales se muestran a continuación:

- Descomposición.
- Generalización.
- Pensamiento algorítmico.
- Abstracción.
- Razonamiento lógico.
- Evaluación[1].

Estos seis procesos centrales son herramientas teóricas de las cuales se sirve el PC, con el propósito de explicar lo que sucede en el campo. Sin embargo, concretamente, ¿Qué es el PC? A pesar de que todavía no existe un consenso con respecto a su definición, uno de los debates más enérgicos ha llegado a la conclusión de que se trata de un "conjunto de habilidades y destrezas (…) habituales en los profesionales de las ciencias de la computación, pero que todos los seres humanos deberían poseer y utilizar para resolver problemas, diseñar sistemas, y sorprendentemente comprender el comportamiento humano" (Adell, et al., 2019, p. 173). Asimismo, este tipo de pensamiento se basa en las ciencias de la computación, con el propósito de desarrollar las habilidades propias de dicha disciplina, en estudiantes desde nivel preescolar, en adelante.

Uno de los supuestos que Papert (1980) sostuvo en sus obras, fue el cuestionamiento y crítica a que los sujetos utilizan la tecnología sin antes haber desarrollado los conocimientos básicos acerca de cómo funciona. Desde esta perspectiva, difícilmente se podrán aprovechar todas las ventajas que ofrece dicha tecnología, sin importar si se trata de un dispositivo móvil, una base de datos, un robot, una inteligencia artificial, etc. Por ello, el PC brinda las pautas pedagógicas para desarrollar las habilidades necesarias para utilizar dichas tecnologías.

El avance tecnológico en cuestión de las aplicaciones en dispositivos inteligentes, el fenómeno del consumo audio-visual por demanda, el *streaming*, videoconferencias virtuales, el uso de la Inteligencia Artificial (IA), entre otros,

1. Los procesos centrales se atribuyen a Wing (2017), sin embargo, fueron las guías didácticas del Reino Unido las que los catapultaron en la comunidad científica y académica.

han tenido como consecuencia, la necesidad de comprender cómo es que funciona y se programa cada una de estas herramientas. El PC se encuentra justo dentro de este debate, que busca que los estudiantes desarrollen habilidades computacionales específicas que les permitan reconocer las herramientas tecnológicas como una extensión de sí mismos[2], ya sea a través del análisis, automatización y abstracción de procesos.

Se ha debatido acerca de si los procesos centrales del PC se tratan de procesos cognitivos, habilidades, o características de dicha propuesta pedagógica. Para esta obra, y dado el análisis exhaustivo que se llevó a cabo en investigaciones pasadas, se referirá a los procesos centrales como procesos cognitivos que se reflejan en decisiones y acciones procedimentales relacionadas o no con el uso de la tecnología. A continuación, se explican brevemente y de manera introductoria, dado que capítulos posteriores tienen el propósito de hacerlo más a profundidad.

Por lo tanto, *la descomposición* como proceso cognitivo, hace énfasis en segmentar un problema u objeto, con el propósito de convertirlo en una situación más concreta y manejable, por ejemplo, cuando un sujeto intenta comprender un suceso histórico, una receta de cocina, una situación problemática aritmética, un código de programación, entre otras situaciones. Este proceso central toma el papel de eje central del PC, dado que éste guarda los nodos relacionales de los otros cinco procesos cognitivos. En otras palabras, sin la descomposición, difícilmente ocurrirían.

La generalización tiene la característica de identificar los elementos en común y patrones de un procedimiento u objeto. Un ejemplo claro se observa cuando se utiliza una aplicación de un dispositivo móvil, para después utilizar otra similar, es decir, se relacionan ambas interfaces para comprender la lógica con la que funcionan. Lo mismo sucede al enfrentarnos a un problema matemático que no es totalmente nuevo, sino que se cuente con conocimientos previos al respecto, en ese caso se aplican generalidades para realizar inferencias acerca de los procedimientos propios para resolverlo dicho problema.

2. La "Extended-Mind" se trata de un concepto atribuido a Clark y Chalmers (1998), sin embargo, con el avance tecnológico cobra mayor sentido al relacionarlo con el PC.

El pensamiento algorítmico hace referencia a una serie de procedimientos y pasos estratégicos que se llevan a cabo a la hora de resolver un problema, por ejemplo, al elaborar un ensayo sobre algún tema, el estudiante puede organizarse sobre las actividades que llevará a cabo de manera inicial, para después estructurar el texto y determinar el orden de cada uno de los apartados.

La abstracción es el proceso cognitivo en donde el sujeto es capaz de comprender un objeto, problema o artefacto sin necesidad de llevar a cabo procedimientos concretos, sino que las relaciones pueden ser difusas o invisibles. Por ejemplo, al resolver un problema mediante el cálculo mental o darle un significado propio a un nuevo concepto dentro de sus propios esquemas mentales.

El razonamiento lógico pareciera que mantiene similitudes con la abstracción o el pensamiento algorítmico, no obstante, uno de los procesos centrales que lo diferencian es la automatización que, dentro de este proceso cognitivo, utiliza una serie de pasos con el propósito de que una estructura lógica externa lleve a cabo acciones que el sujeto necesito, por ejemplo, al utilizar una plataforma de inteligencia artificial estaríamos hablando de razonamiento lógico y automatización.

Por último, pero no menos importante, *la evaluación*, que se caracteriza por un ejercicio de recuperación de procedimientos y la depuración que necesite cada uno de sus pasos. Por lo que involucra elementos de seguimiento y mejora de los otros cinco procesos centrales. Generalmente la evaluación ocurre de manera dinámica en la aplicación de cualquiera de los procesos cognitivos del PC.

Dicho lo anterior, resulta problemático pensar que el PC no se encuentre dentro de las habilidades propuestas para el desarrollo integral de un individuo en el siglo XXI, dejando fuera los seis procesos cognitivos expuestos, además de que se ignore por completo su implementación en contextos computacionales y tecnológicos. Tal es el caso del Sistema Educativo Mexicano, que ha hecho inversiones en políticas públicas que implementan robots y plataformas educativas[3], pero la propuesta pedagógica se ha quedado al margen

3. Por ejemplo, con el programa Aprende 2.0, como un primer esfuerzo por incluir el PC en México.

de la replicación de otros programas como los de Reino Unido, Estados Unidos, España, entre otros.

Apenas en 2024, han surgido estudios que plantean la idea de que el PC se trata de una competencia, más que de una habilidad, debate que se abordará más adelante en capítulos posteriores, sin embargo, el hecho de que exista esta confusión epistemológica tiene relación directa con la falta de solidez teórica por parte de dicho concepto. Es decir, es necesario profundizar en los procesos centrales del PC, para poder diferenciarlo de una habilidad o de una competencia.

Dicho esto, pensar computacionalmente no sólo repercute en situaciones que impliquen el uso de la tecnología, sino que impacta en contextos de educación formal, informal y no formal, así como situaciones cotidianas en el siglo XXI. Es tarea de los investigadores apuntalar los supuestos teóricos que brinden solidez epistemológica, por esto, existe un proceso cognitivo o también llamado desde este enfoque "proceso central", que toma el papel de eje articulador y funciona como elemento mediador, sin caer en una jerarquización innecesaria, se trata del proceso de descomposición (PD).

La descomposición: Un concepto clave en el Pensamiento Computacional

El PD desde el enfoque del PC se trata de una habilidad, aunque no se ha especificado claramente si también puede definirse como un proceso central. Desde la perspectiva de Czismadia (2015) y Cárdenas (2017) sus características elementales apuntan más a definirse como aquella habilidad en donde se piensa a un objeto con base en sus propios componentes, con el propósito que puedan ser entendidos, resueltos, desarrollados y en ocasiones evaluados de forma separada, y por consecuencia, menos compleja.

En esta obra, se plantea una idea distinta del PD, ya que uno de los problemas principales, por los cuales el PC no ha logrado posicionarse dentro del debate teórico, pedagógico y didáctico sobre la robótica educativa, tiene relación directa con la falta de análisis y creación de pautas teóricas sobre sus propios procesos cognitivos centrales. De esta manera, convertirse en una competen-

cia en donde convergen habilidades o procesos cognitivos específicos, sería una realidad.

En este caso, el argumento en torno al PD se posiciona dentro del marco del PC, por lo tanto, pueden existir excepciones que aquí mismo se expondrán. Éstas tienen que ver, por ejemplo, con la aritmética y problemas de la vida cotidiana, que dotan de características específicas al PD (González, 2016). No es lo mismo tratar la descomposición al resolver problemas algebraicos, que problemas de programación al programar un robot.

La descomposición no se caracteriza por ser un proceso que nunca haya sido estudiado, por lo tanto, la expresión de "vacío" no hace énfasis en el desconocimiento total de dicho proceso, sino que, a partir de lo que se ha investigado, todavía existen elementos problemáticos que imposibilitan su completa comprensión. Sin embargo, para profundizar en el tema, es preciso comenzar por su definición.

El PD se define como "una forma de pensar artefactos en términos de sus propios componentes. Estos componentes pueden ser entendidos, resueltos, desarrollados y evaluados de manera separada" (Czismadia et al., 2015). Uno de los ejemplos comunes utilizados en la literatura por Cárdenas (2017), es el proceso que se lleva a cabo para una receta de cocina, se tienen ingredientes que son componentes separados, no obstante, el proceso completo de cocinar un platillo debe de descomponerse en distintas actividades y conjuntos de componentes, por lo tanto, licuar es un proceso, al igual que, hornear y mezclar. Al final de cuentas, se debe de descomponer en sub-procesos, más pequeños y manejables, en ocasiones en un orden específico, para finalmente componerse en la estructura original de la receta.

En ejemplos como el anterior, el PD parece sencillo, no así en contextos aritméticos, algebraicos o de programación. Selby (2015) evidenció la complejidad del desarrollo de la descomposición, catalogándola en su estudio como una de las habilidades más complicadas del PC. Asimismo, la Royal Society (2012) estableció que, a pesar de que existen otros cinco procesos centrales, que como se mencionó anteriormente, poseen su propia importancia y relevancia, es necesario descomponer problemas antes de desarrollar las otras habilidades clave, ya que, permite comprender el problema y aplicarlo en nue-

vas situaciones, por lo tanto, sin la descomposición, difícilmente podrían desarrollarse los otros procesos (Cheng, 2012).

González-González (2019) resalta la importancia que tienen las estrategias de descomposición dentro del PC, no obstante, sólo lo mencionó y no profundizó en ellas. En este sentido, la investigación de Cheng (2012) ilumina el camino en torno a estas estrategias. Este autor consideró no solamente la descomposición, sino la reagrupación del problema como un tipo de procesamiento que requiere de mayor madurez, es decir, no solamente se trata de descomponer.

La descomposición de problemas posee diversas características, una de las más importantes, es la necesidad de observar la situación problemática como una parte, relacionada a otras partes y a un todo. Además de esto, se necesita llevar a cabo procedimientos de descomposición mentales, con el propósito de explorar diversas combinaciones, por lo que algunos autores relacionan la tarea de descomposición con la de clasificación (Cheng, 2012; Piaget, 1971).

Por otro lado, los conocimientos previos juegan un papel clave en la descomposición. Se ha evidenciado que, en este tipo de tareas, los estudiantes tienden a buscar sub-problemas conocidos, por ejemplo, si se tiene dominadas las adiciones con el número 10 y posteriormente se enfrenta a una nueva situación (7+8), se comenzará a descomponer el número 7, en 5 y 2; posteriormente el 2 se sumará al 8 para obtener 10. Una vez que se llega al número o situación conocida, será más sencillo resolver el problema, simplemente con 10+5. Sin embargo, el proceso de descomposición se torna complejo, cuando se toma en cuenta las diferentes posibilidades para resolver problemas, siguiendo con el ejemplo anterior, el estudiante también pudo descomponer el número 8, en 3 y 5, o 5 y 3, posteriormente sumar el 3 al 7 para volver al 10 y sumar 5 (Cheng, 2012)[4].

Lo expuesto por Cheng (2012), González-González (2019), Selby (2015) y la Royal Society (2012) es útil para caracterizar al PD, sin embargo, existen diversas problemáticas en torno a este proceso, que han sido documentadas

4. Es necesario seguir profundizando en este proceso, dado que Cheng (2012) exploró las estrategias de descomposición de problemas dentro de un contexto aritmético, es decir, brindó insumos básicos para su futura comprensión dentro del marco del PC.

en algunos estudios. Tales como, la descomposición nula y extremadamente fina, la reagrupación del problema, la descomposición incompleta, la transversalidad e implicación del PD con los cinco procesos centrales, y, por último, el debate en torno a la relación entre el PD y la programación.

Los problemas antes mencionados, cobran mayor sentido cuando se llevan al contexto de la programación, ya que, es ahí donde se han documentado. Sin embargo, Rich, Egan y Ellsworth (2019) han logrado identificar los tipos, estrategias y pasos genéricos de la descomposición:

- *Tipos de descomposición*: Estructural, funcional, secuencial y dependiente.
- *Pasos genéricos:* Identificación del eje, evaluación proactiva, aceptación o rechazo, ejecución de la descomposición y la evaluación retroactiva.
- *Estrategias específicas:* Final significativo, de abajo hacia arriba, multi-variable, multi-nivel y, por último, la comparativa.

Los componentes anteriores son los más recientes y cercanos a una caracterización de la descomposición en contextos de programación, sin embargo, los ejemplos se llevaron a cabo incipientemente, sin brindar detalles con respecto a las relaciones que ocurren en el interior. En otras palabras, a pesar del esfuerzo teórico por describir cada uno de los elementos básicos del PD, existió un vacío en torno a la estructura interna del mismo, y que en su momento se le denominó como "caja negra del PD". En esta obra, las pautas teóricas que se brindarán acerca de este proceso cognitivo serán los elementos clave de su estructura interna, y que permitió transparentar esta metáfora teórico-metodológica.

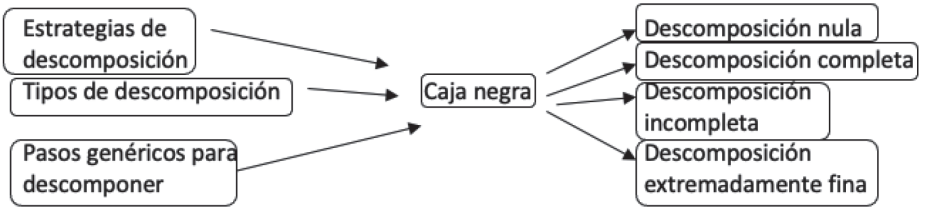
La caja negra del Proceso de Descomposición

Las cajas negras, se definen como aquellos artefactos en donde es posible observar los elementos de entrada y posteriormente los de salida, en ocasiones se podrá observar una evolución en dichos elementos, incluso algunos no saldrán y desaparecerán como si se tratara de un agujero negro. El problema radica en que se desconoce la estructura y las relaciones que ocurren en el interior de dicha caja.

Con respecto a la descomposición, es posible identificar su tipología, las estrategias que utilizan los estudiantes, incluso los pasos genéricos no lineales durante el proceso, como se mencionó anteriormente, Cheng (2012) logró documentar el proceso desde los conocimientos previos y una metodológica específica para resolver problemas aritméticos, de ahí en adelante, los siguientes esfuerzos se han enfocado en explorar incipientemente a la descomposición, algunos limitándose a su simple definición, sin entrar en detalles, puesto que en estos está la complejidad.

No basta con nombrarle "caja negra" a cualquier artefacto desconocido, sino que es necesario argumentarlo y por supuesto, demostrarlo. Para ello, fue necesario codificar algunos de los elementos de entrada y de salida, posteriormente llevar a cabo algunas relaciones al respecto, y por último analizar si efectivamente existen elementos faltantes que dificulten develar la caja negra únicamente desde la teoría ya existente (dichos elementos pueden observarse en la figura 1).

Figura 1. Componentes de la caja negra.



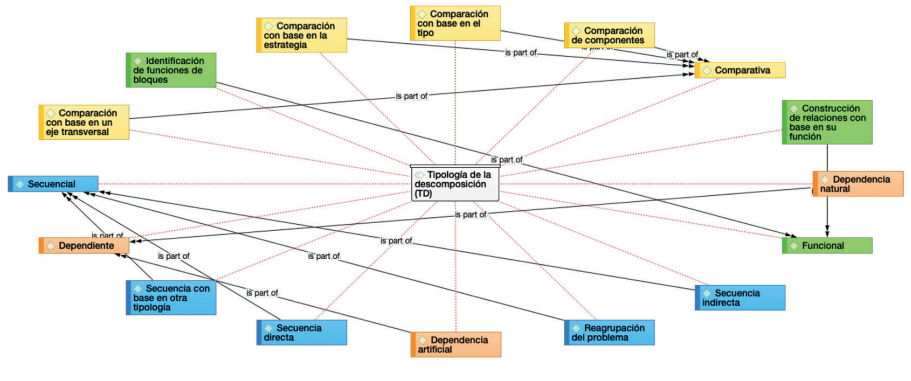
Fuente: Elaboración propia con base en Rich, Egan y Ellsworth (2019).

La tipología sobre la descomposición es conocida, aunque todavía no existe una basta teoría al respecto. En este sentido, Rich, Egan y Ellsworth (2019) fueron los primeros investigadores en denominarle "caja negra de la descomposición". Su argumentación principal estuvo basada en la falta de profundización en dicho proceso, es decir, existen diversos estudios que hablan sobre la descomposición, pero en su mayoría se aplican con base únicamente en su definición principal desde la perspectiva del PC. De tal forma que, ¿Dónde que-

dan su tipología? ¿Dónde quedan sus características esenciales? ¿Dónde quedan sus relaciones con la programación?

Un aspecto que Rich, Egan y Ellsworth (2019) dejaron pendiente, fue el cuestionamiento sobre qué sucede con la descomposición en contextos de programación. Al respecto, si el PD es complejo per se, llevarlo a un escenario de programación, en donde se debe respetar un orden específico de los componentes, lo convierte en una verdadera caja negra. Como ejemplo inicial, se muestra a continuación las relaciones que existen sobre la tipología de la descomposición.

Figura 2. Relaciones que existen sobre la tipología de la descomposición.



Fuente: Elaboración propia.

La descomposición puede organizarse con base en su tipología, es decir, la estructural, funcional, secuencial y dependiente. Cada uno de estos tipos, posee características particulares, el propósito del presente capítulo no es describirlas, eso será en capítulos posteriores, sin embargo, la figura 2 es apenas la punta de *iceberg*, ya que, se trata de una sola categoría de entrada (posteriormente abordaremos las otras 2).

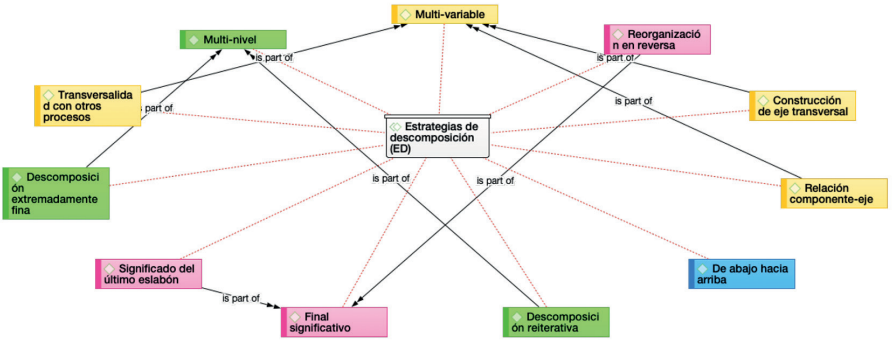
Es posible llegar a la conclusión de que, descomponer estructuralmente puede funcionar al descomponer una receta de cocina, incluso sería posible hacerlo de manera secuencial, funcional y dependiente, hasta este punto, en problemas de la vida cotidiana, descomponer no se trata de una caja negra, ya que perfectamente es posible observar el proceso directamente, no obstante,

en términos de programación, el hecho de seguir una lógica de orden descendente, incluso atender a los bloques de un pseudo-lenguaje, en donde algunos de ellos, encierran, secuencian y dependen de otros, en ese justo momento, todo se complejiza.

En otras palabras, no es lo mismo reagrupar un problema desde la dependencia de las partes de una receta de cocina, a reagruparlo considerando las funciones de cada bloque de programación, la secuencialidad de cada uno de ellos, la lógica descendente, la abstracción de componentes, la simplificación de códigos, incluso, la misma interfaz, a veces desconocida en algunos estudiantes.

Aunque no todo se trata de la tipología, existen, por ejemplo, otras categorías como los pasos genéricos y las estrategias de descomposición, ambas se muestran en las siguientes figuras 3 y 4.

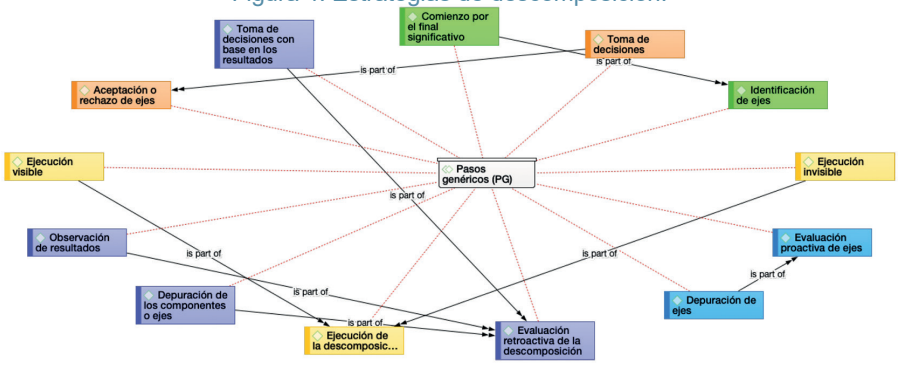
Figura 3: Tipologías existentes.



Fuente: Elaboración propia.

En ambas figuras se puede observar una red semántica, que mantiene diversas relaciones, algunas de ellas podrán ser visibles, ya que las flechas apuntan directamente, mientras que muchas otras, podrían existir e ir en paralelo, porque existen indicios para pensar en ello, sin embargo, la teoría todavía no alcanza a explicar dichas relaciones. Al final de cuentas, Rich, Egan y Ellsworth (2019), Csizmadia (2015), Bocconi et al. (2016), entre otros, han fortalecido sus argumentos, con base en ejemplos hipotéticos de lo que podría suceder

Figura 4. Estrategias de descomposición.



Fuente: Elaboración propia.

en el campo. Ninguno de ellos, ha creado teoría a partir de los datos (teoría anclada).

Existen diversos argumentos que ejemplifican la complejidad de la descomposición en contextos de programación (la mayoría de estos se expone en el marco teórico), no obstante, en el presente apartado sólo se presentará uno, en donde intervienen diversos elementos. Resulta interesante pensar, en que un estudiante necesite descomponer un código específico, en donde involucre bloques de evento y acción del pseudo-lenguaje de *Mblockly*, para ello, no bastará con hacerlo de manera estructural, sino que necesitará analizar la funcionalidad de cada bloque, además de probablemente utilizar el paso genérico de evaluación, y, por último, seleccionar la estrategia de "final significativo".

Al llevar a cabo la estrategia de final significativo, el estudiante tendrá que hacer uso de un pensamiento reversible. Posteriormente, podrá identificar que algunos bloques de eventos y de control no podrán separarse, porque en ese caso caería en una descomposición extremadamente fina, en donde se perderá el sentido totalmente. Por lo tanto, es posible utilizar la descomposición dependiente.

En el ejemplo anterior, se mencionaron los siguientes elementos de la descomposición:

- Descomposición estructural.
- Descomposición funcional.

- Paso genérico de evaluación (retroactiva y proactiva).
- Estrategia de descomposición de final significativo.
- Pensamiento reversible.
- Descomposición extremadamente fina.

El problema teórico radica en cómo es posible explicar las relaciones que ocurren en el PD, con base en todos estos elementos. Partiendo de la premisa que, dichos elementos son las entradas, y la salida podrá ser una descomposición eficaz o una descomposición extremadamente fina, incluso nula. Cualquiera que sea la salida, la estructura que tome cada uno de los elementos de descomposición es, al menos hasta el momento, desconocida.

La manera en la que es posible ejemplificarla, sería crear ejemplos ficticios, tal y como se ha estado haciendo en la comunidad científica. Sin embargo, mediante este tipo de ejemplos, es imposible encontrar una regularidad que permita crear nueva teoría al respecto. Por lo tanto, es necesario acercarse al campo y comenzar a develar la caja negra de la descomposición, ya que lo anterior es sólo un ejemplo, de decenas que podrían existir.

A manera de cierre del presente apartado, la caja negra se trata de una metáfora teórico-metodológica. Que si bien, se sabe que puede ser develada al utilizar estrategias de análisis de datos sistemáticamente, en términos de problematización es necesaria e incluso primordial. Esto se debe a que permite pensar el Proceso de Descomposición como una estructura con elementos difusos, por lo que la tarea principal de la presente obra, es identificarlos y relacionarlos entre sí. De esta forma, la caja negra pasará de ser una metáfora, a una herramienta metodológica que incluso será el eje principal de una posible teoría de salida.

2. El Pensamiento Computacional y su relación con la robótica educativa: una revisión de la literatura

Es imprescindible una revisión literaria del PC para comprender de dónde surgen sus procesos centrales, además de su relación con los robots educativos. A continuación, se presenta un análisis comparativo de cómo se han implementado diversos robots en el campo educativo, con el propósito de comprender el debate en torno al PC y posicionar este documento en las tendencias y vacíos en el conocimiento reconocidas.

La búsqueda se llevó a cabo con las palabras clave en inglés y en español, debido a que cerca del 90% de los estudios indexados se encuentran en inglés (Bitetti y Ferreras, 2017). Por otra parte, se seleccionaron estudios realizados a partir del 2011, ya que en el 2012 se creó *Scratch* y otros *softwares* de programación con bloques, no obstante, a pesar de este límite cronológico, también se utilizaron fuentes anteriores a este año, por ejemplo, obras clásicas de Seymour Papert que sustentan lo que actualmente es el PC. Asimismo, fue necesario limitar la información a investigaciones que utilizaran robots y *software* de programación modernos, tales como las últimas versiones de *Scratch*, *Mblock*[5] y el *Mbot*.

En una primera etapa, las búsquedas se llevaron a cabo en *Science Direct*, *Scopus*, *Web of Science*, *ProQuest*, entre otras bases de datos, dado que concentran el mayor número de publicaciones científicas internacionales, así como aquellas que son de libre acceso, tales como *Dialnet*, *Redalyc* y *ScieLo*, que concentran la mayoría de publicaciones científicas iberoamericanas (Hernández, 2020). Es importante señalar que, una vez que los artículos se encontraban

5. *Scratch* y *Mblock* se tratan de pseudolenguajes de programación que ayudan a los estudiantes más jóvenes a familiarizarse con la lógica computacional al programar un robot, videojuego, entre otros productos.

almacenados en la base de datos, se realizó una depuración con base en aquellas revistas indexadas, que tuvieran principalmente nivel de impacto Q1 o Q2, ya que esto garantizaba que fueran estudios de mayor impacto dentro de la comunidad científica.

La segunda etapa consistió en la sistematización de la información, en donde se creó una tabla comparativa, que permitiera el análisis transversal de los estudios, con base en apartados como los propósitos, referentes teóricos, marco metodológico y principales hallazgos y/o resultados de los estudios. Por último, en la tercera etapa se construyeron las unidades de análisis, con base en un mapa de la literatura, que Creswell (2014) recomienda como una herramienta de síntesis visual, en donde se presenta un orden jerárquico, con el propósito de que el investigador adquiera una visión general de la literatura existente acerca de un tema.

Por lo anterior, se llevó a cabo un análisis con *Iramuteq* sobre el debate en torno al PC, ya que se trata de un debate con densidad teórica, por lo que el procesamiento de la información con *Iramuteq* permitió identificar semejanzas, diferencias, así como relaciones y construcción de comunidades entre los estudios analizados. Por lo anterior, se elaboró un mapa que contiene todos los artículos seleccionados, de tal forma que las unidades de análisis surgieron a partir de la identificación, sistematización y relación entre los estudios.

La enseñanza de la programación y el PC son temas que, como otros, pudieran ser complejos, ya que, existen diversas estrategias y técnicas de enseñanza, robots educativos, edades distintas de los sujetos estudiados, referentes teóricos, contenidos y habilidades matemáticas que desarrolla, así como diferentes tipos de *software* de programación. Por lo tanto, la información se clasificó en las siguientes categorías de análisis:

- Estrategias y técnicas de enseñanza de la programación.
- Contenidos y habilidades que desarrolla el PC.
- El debate en torno al PC.

Por último, conviene describir concretamente cada una de las categorías de análisis, la primera de ellas, tiene que ver con las estrategias más representativas relacionadas con la programación, con el propósito de identificar

sus semejanzas y diferencias con respecto al PC. Por otro lado, en la segunda categoría se llevó a cabo un análisis sobre los contenidos que se han abordado con el PC, así como las habilidades que desarrolla, de este modo se evidenciarían las diversas aportaciones que tiene en el campo educativo y más aún en la línea de procesos de enseñanza y aprendizaje. En la tercera categoría, se incluye el análisis con *Iramuteq*, en donde se reconocen las tendencias y debates en torno al PC y al PD.

La enseñanza de la programación ¿Sólo se trata de Pensamiento Computacional?

El presente apartado pretende contextualizar al lector, antes dar cuenta de la variedad de estrategias y técnicas de enseñanza de la programación en niños, y que no solamente se trata del PC y las computadoras, como inicialmente podría suponerse, sino que también implica el uso de una amplia gama de dispositivos como las tabletas inteligentes, los teléfonos inteligentes, los mini-robots, y las computadoras de placa reducida como la *Raspberry Pi*.

Como punto de partida, es necesario remontarse a Papert (1980), que llevó a cabo los primeros esfuerzos por estudiar el proceso de aprendizaje de la programación, particularmente con niños. En el marco de la robótica educativa y la creación de un lenguaje llamado *Logo*, que respondía no solamente a la programación secuencial, sino a la teoría pedagógica de Jean Piaget, con base en la cual fue diseñada (Palma & Sarmiento, 2015).

La teoría del aprendizaje por construcción o también llamado construccionismo de Seymour Papert, tiene como característica principal que el estudiante aprende a través de la relación que guarda un nuevo concepto con sus conocimientos previos (Papert, 1993). Cabe señalar que, en el *Massachusetts Institute of Technology* (MIT) se sigue utilizando esta metodología de enseñanza, sin embargo, el lenguaje de programación *Logo* ha sido reemplazado por *Scratch* y el construccionismo por el PC (Lewis, 2010).

Una de las estrategias que aprovecha el carácter multidisciplinario de la robótica educativa es el método Polya[6], que en el estudio de Torrejón y Ventura (2019) ha sido utilizado con base en secuencias de tarjetas de comandos, con el propósito de analizar las estrategias que utilizan los estudiantes para resolver problemas. Las características de este método tienen que ver con una serie de fases, tales como comprender el problema, construir un plan, ponerlo en práctica y por último reflexionar sobre su solución.

Por otra parte, existen otras estrategias relacionadas con la programación que también toman como punto de partida diversas fases para la resolución de problemas, por ejemplo, el aprendizaje por demostración, que ha sido utilizado en trabajos de psicología y neurociencia (Nope, Loaiza & Caicedo, 2011). A diferencia de Torrejón y Ventura (2019), en el estudio de Nope, Loaiza y Caicedo (2011) nombran sub-problemas a aquellas etapas necesarias en la programación, que son la observación, reconocimiento y ejecución de la imitación. Esta estrategia se caracteriza por el uso de algoritmos computacionales en mapas visual-motores, con el propósito de imitar gestos y movimientos. En este sentido, las diferencias clave de ambos estudios tienen que ver con el sujeto de investigación, es decir, que en el primero se hace énfasis en método Polya con niños y en el otro en la programación de un modelo computacional basado en el paradigma del aprendizaje por demostración, no obstante, una de las semejanzas clave es que en ambas la resolución de problemas matemáticos se dividen en sub-problemas o fases, que Papert (1980) llamaba ejercicio de depuración y descomposición en la práctica de enseñanza de la programación.

Siguiendo con el tema de las estrategias de enseñanza, particularmente en la influencia que tiene la robótica educativa y la programación en los estudiantes. Garnica (2014) propone a la robótica ubicua como el elemento clave para la inclusión de los robots en la educación. Su modelo pretende "la integración de la robótica en el entorno de la persona, de manera que los robots no se perciban como objetos diferenciados" (Garnica, 2014, p. 40), Esto es, lograr que los dispositivos computacionales se utilicen en el aula como una necesi-

6. Aprovecha la multidisciplinariedad de la robótica educativa porque en el estudio de Torrejón y Ventura (2019) se utiliza el método Polya para enseñar matemáticas, música, entre otras asignaturas.

dad primaria, de tal forma que, la práctica educativa se centre en las tareas a realizar y no en las herramientas tecnológicas.

Por lo tanto, el objetivo de la robótica educativa para Garnica (2014) tiene que ver con profundizar en contenidos y habilidades a través de los robots, utilizando sistemas mecatrónicos como recursos didácticos. Para lo anterior, la autora recomienda una serie de fases, que son la identificación de la plataforma, programación de la misma y validación del funcionamiento del robot con base en un conjunto de actividades que ayudarían a comprender la resolución de problemas a los estudiantes (Garnica, 2014). Por lo anterior, las estrategias de enseñanza del método Polya, aprendizaje por demostración y la robótica ubicua, se asemejan en la organización metodológica que utilizan, llámense fases o sub-problemas.

Siguiendo con el tema del aprovechamiento de las funciones de los robots para aprender a programar, Ortiz et al. (2019) refiere que en ocasiones las estrategias utilizadas para desarrollar habilidades de programación, se reducen únicamente en la enseñanza de las funciones básicas de los robots. La crítica se sitúa particularmente a las llamadas "estrategias STEM", en donde según su estudio, se quedan en un nivel superficial de la robótica educativa, caracterizado por no resolver problemas, sino en dominar las funciones del robot (en el caso de su investigación, de los kits Lego).

El estudio de Suárez y Soto (2015) se basó en la enseñanza de la programación a través de un paquete de *Software* llamado *Kodu,* esta estrategia consistió en una demostración y explicación de las funciones básicas, para posteriormente mediante trabajo colaborativo, los estudiantes pudieran replicar y explorar lo expuesto inicialmente. Por otro lado, existen investigaciones que toman al paquete de *Software Scratch* como la estrategia didáctica misma, tal es el caso de Serna et al. (2018) que nombró "*estrategia didáctica del Scratch*", aunque lo que se realizó fue utilizar conceptos constructivistas derivados de Vygotsky y el enfoque didáctico de la asignatura de "conocimiento del medio". Por lo tanto, más que una estrategia, *Scratch* tomó el papel de recurso didáctico.

Los estudios de Serna et al., (2018), Correa et al., (2019), así como Suárez y Soto (2015) son ejemplos de cómo se hace énfasis en el robot o paquetes de *Software* de programación, sin embargo, su dominio no garantiza el desarrollo

de habilidades significativas. Con respecto a la presente obra, coincide más con lo expuesto por Bers (2017), en donde la codificación no implica precisamente el desarrollo de un PC, es decir, aunque son conceptos relacionados, es necesario no quedarse simplemente en la programación de los robots, sino profundizar en el tipo de habilidades que desarrolla, así como qué clase de estrategias se utilizaron.

Si bien el PC no es una estrategia de enseñanza de la programación como tal, brinda elementos para pensar en qué tipo de habilidades son indispensables no sólo para programar, sino para resolver problemas que impliquen el pensamiento algorítmico, descomposición, abstracción, depuración, entre otros conceptos, ya sea en problemas matemáticos o no. Por lo tanto, dado su amplio margen de aplicación, en ocasiones del PC se puede ver involucrado con otras estrategias didácticas, incluso confundir y pensar que el robot o *Software* son la estrategia *per se*.

El construccionismo: la base del Pensamiento Computacional

Para autores como Palma y Sarmiento (2015), así como Bocconi (2016), los conceptos básicos de la teoría de Jean Piaget fueron los detonadores del construccionismo de Seymour Papert y a su vez del PC. Estas teorías pedagógicas funcionan bajo la premisa de que el aprendizaje es el resultado de la asimilación de conceptos nuevos, con base en su comparación con los conocimientos previos del estudiante (Papert, 1991). Sin embargo, en el estudio de Guardiola (2018), agrega que los conceptos nuevos deben de poseer un significado particular, en otras palabras, ser del interés del estudiante.

Por otro lado, Jiménez, Ramírez y González (2011) refieren que los nuevos conceptos pueden ser significativos si se relacionan con materiales tangibles, es así como "surge LEGO/LOGO que une el lenguaje de programación *Logo* con el mundo de construcción LEGO" (Jiménez, Ramírez & González, 2011, p. 165). Una de las semejanzas que se identificaron en este apartado, es que los estudios utilizan *Lego* en sus versiones *WeDo* y *Mindstorms*, no obstante, es necesario reconocer cómo fueron utilizados didácticamente, además de cuáles fueron los resultados de su aplicación.

Estrategias y técnicas de enseñanza de la programación

Las estrategias y técnicas utilizadas son diversas, además de que poseen coincidencias y discrepancias. La resolución de problemas es uno de los elementos que resaltan en los estudios analizados, sin embargo, los estudiantes solucionan problemas de distinta manera, Cheng (2018) refiere que la teoría del construccionismo de Seymour Papert, permite a los estudiantes "construir" su propio conocimiento, esto a través de cuatro fases, tales como dividir en sub-problemas, planificación del procedimiento, así como la ejecución y análisis de errores, que además se relacionan con las habilidades digitales del PC. Esta última fase de Cheng (2018) podría relacionarse con la metacognición que propone Di Lieto, et al. (2017), en donde el estudiante debe depurar los códigos de programación que elaboró, además de que enfatiza en el trabajo colaborativo y la socialización durante las prácticas de programación.

Otro de los elementos que destacan en las estrategias de enseñanza analizadas, tienen que ver con el aspecto lúdico del robot y el *software* de programación, tal es el caso de los estudios realizados con niños menores a 7 años (Torrejón & Ventura, 2018; Miranda, 2018; Barrera, 2014; Di Lieto, et al. 2017; Muñoz-Repiso & Caballero-González, 2019). Aunque el PC sigue siendo la propuesta principal, se identificó que existen estrategias lúdicas que involucran a la programación de robots específicos (Barrera, 2015).

En el estudio de Barrera (2015) relacionado con estas estrategias lúdicas robóticas, se evidenció que la depuración de problemas es un punto clave en tareas de programación. Estas tareas simples deben de monitorearse hasta conseguir un producto final, que puede ser la construcción de un robot, hasta la programación del mismo. Por lo que es necesaria la llamada *depuración análoga,* que se concentra en el aprendizaje a partir de los errores cometidos por los estudiantes. En palabras de Papert (1980), refiere que "los errores nos benefician porque nos llevan a estudiar lo que sucedió, a comprender lo que anduvo mal y, a través de comprenderlo, a corregirlo" (p. 135). Por tal motivo, los estudios relacionados con el PC se caracterizan por una depuración de sub-problemas, con el propósito de mejorar los procedimientos para su resolución (Palma & Sarmiento, 2015).

Hasta este punto, los estudios analizados se han enfocado en la resolución de problemas como pieza clave para la enseñanza de la programación, en donde la relación entre el niño y el *software* y/o robot, adquiere características de exploración, que Papert (1980) resume en "los niños como constructores activos de sus propias estructuras intelectuales" (p. 33). Además, existen investigaciones que permiten esta *conversión del niño como epistemólogo* (Papert, 1980, p. 33), en donde éste aprende sin necesidad de que un docente le enseñe, es decir, utilizando el material que se le proporciona. Este objetivo pedagógico puede lograrse con ayuda de los sensores que poseen los robots, al implementarlos como ejes centrales de la enseñanza (Sáez-López, Sevillano-García & Vazquez-Cano, 2019). A diferencia de estudios que pretenden enseñar a programar con base en problemas y su solución general (Scaradozzi, Sorbi, Pedale, Valzano & Vergine, 2015), es así como Sáez-López, Sevillano-García y Vazquez-Cano (2019) utilizan los sensores de proximidad o sigue líneas del *Mbot* para proponer situaciones problemáticas que se resuelven particularmente con el uso de uno o más sensores.

La programación a través de sensores, como ejes centrales para identificar la función de los bloques visuales de los diferentes *software* como *Scratch* o *Mblock*, según Saéz-López, Sevillano-García y Vazquez-Cano (2019), proporciona una visión transversal de los contenidos de aprendizaje, sin embargo, desde la perspectiva de Erol y Kurt (2017) que se concentraron en la resolución de problemas secuenciales sin ningún tipo de sensor, aun así lograron desarrollar habilidades lógico-matemáticas, que en su momento propiciaron el interés por aprender lenguajes de programación más complejos como el *C*++.

Por otra parte, no todos los estudios siguieron la tendencia del PC, desde la perspectiva de Constante, Chimbo, Jiménez y Gordón (2019) es posible enseñar a programar desde una metodología tradicional, que tiene que ver con una exposición por parte del docente, para que posteriormente el estudiante lo imite y logre resolver problemas con base en dicha práctica. En dicho estudio se evidenció el aprendizaje con base en el resultado de evaluaciones estandarizadas, por lo que el análisis cuantitativo de los datos no permitió emitir conclusiones sobre el proceso de resolución de los problemas, es decir, se limitó a probar cualitativamente que la realidad aumentada propiciaba la motivación e interés de los estudiantes (Constante, Chimbo, Jiménez & Gordón, 2019).

Enseñar a programar desde un enfoque tradicional, en donde el alumno juega un rol pasivo y el docente el papel activo, corre el riesgo de relacionarse con lo que Papert (1980) nombró *QWERTY*[7], que se define como la utilización rudimentaria de cualquier sistema. Es decir, utilizar la programación como herramienta de un enfoque de enseñanza tradicional, podría ser el *QWERTY* de nuestro actual sistema de enseñanza. Por tal motivo, la tendencia actual en el presente apartado, es considerar al estudiante como un sujeto activo, que identifica, explora, categoriza, ejecuta y reflexiona sobre sus errores en la programación (Wing, 2017).

A manera de resumen, es posible mejorar la enseñanza de la programación desde diversas perspectivas, sin embargo, en las estrategias de enseñanza analizadas, destacan las siguientes características:

- La resolución de problemas a través de las fases de: Sub-problemas, planificación, ejecución, análisis de resultados y depuración del procedimiento (Papert, 1987).
- Estrategias lúdicas a través de bloques visuales de programación, como andamiaje para abordar tareas de programación (Barrera, 2015).
- Estrategias de enseñanza en donde el estudiante juega el rol activo de la clase (Papert,1993).
- Aprovechamiento de los sensores o herramientas de programación (Saéz-López, Sevillano-García y Vazquez-Cano, 2019).
- El PC es la metodología más utilizada, aunque se han hecho intentos por relacionarla con otras teorías pedagógicas, según el currículum vigente en cada país (Bocconi, et al., 2016).
- Los estudios recomiendan la enseñanza de la programación desde edades tempranas, no obstante, a partir de los 8 años se identifica un mejor aprovechamiento (Palma & Sarmiento, 2015; Guardiola, 2018; Cheng, 2018).

7. Papert (1980) hace referencia a *QWERTY* al orden de las letras en la mayoría de los teclados, ya que anteriormente se colocaron de esa manera para combatir el problema que tenían las máquinas de escribir al atascarse, sin embargo, nunca se modificó dicho orden, lo que es un ejemplo de la justificación histórica que existe sobre cómo se utiliza la tecnología, a veces de manera inconsciente.

- El PC se ha relacionado con diversas asignaturas, tales como historia, música, matemáticas, geografía, entre otras. Se han obtenido resultados eficientes en el aprendizaje de los estudiantes Scaradozzi, et al., (2015).

Las características de los sujetos de investigación

Determinar las características de los estudiantes que aprenden programación es uno de los temas relevantes, ya que permitiría identificar no solamente las edades recomendadas de los alumnos, sino los conocimientos previos necesarios para futuras investigaciones que tomen la presente obra como referencia, así como las dificultades que se presentaron en los estudios analizados con respecto a los sujetos de investigación.

En primer lugar, las edades de los estudiantes influyen en la selección del material a utilizar, por ejemplo, los *Bee-Bot* se relacionan con los estudiantes de preescolar, dado que las características de estos robots benefician las estrategias lúdicas y dominio espacial (Di Lieto, et al., 2017), además de que según Torrejón y Ventura (2019) "son los más adecuados para trabajar el pensamiento computacional en edades tempranas de manera progresiva y adecuada" (p. 24), por otro lado, es posible combinarse con tableros didácticos, que permiten la utilización de todas las instrucciones del robot (Torrejón & Ventura, 2019).

Otro ejemplo sobre la correlación edad-robot, es el uso de los kits *Lego*, que si bien pueden utilizarse desde temprana edad (Scaradozzi, Sorbi, Pedale, Valzano & Vergine, 2015), se identificó que se utilizan en estudiantes de nivel primaria y secundaria. Al igual que los materiales de *Makeblock*, en donde el *Mbot* se ha investigado en contextos de nivel primaria (Sáez-López, Sevillano-García & Vazquez-Cano, 2019; Guardiola, 2018).

Desafortunadamente sólo algunas investigaciones describen las características de los estudiantes a detalle, por ejemplo, en el análisis comparativo que realizó Guardiola (2018), en donde era necesaria su descripción a profundidad, ya que debía analizar los avances del proceso de aprendizaje de la programación en dos grupos, uno en donde se utilizó un método tradicional y otro con el uso de *Scratch*. No obstante, la diferencia clave entre robots como el *Bee-Bot*, *LEGO WeDo*, *LEGO MindStorm* y el *Mbot*, es el nivel de dificultad que

tiene cada uno, por lo tanto, los conocimientos previos que deben de tener los estudiantes varían dependiendo de cada robot e incluso de cada *software* de programación.

A manera de resumen, las características de los sujetos de investigación con base en los estudios analizados son las siguientes:

- Los estudiantes de preescolar se relacionan adecuada y progresivamente con los *Bee-Bot*
- Se recomienda el uso de los sistemas *Lego* y *Makeblock* para estudiantes de nivel primaria y secundaria
- La investigación de Palma y Sarmiento (2015) consideraba la edad de 10 años como la ideal según la revisión de la literatura de ese año, no obstante, actualmente se está enseñando a programar desde edades cada vez más tempranas.
- La revisión de la literatura de Bocconi, et al., (2016) da cuenta, de que el PC cobra mayor sentido cuando se implementa en estudiantes mayores de 9 años de edad.

En conclusión, la presente unidad de análisis da cuenta de la variedad didáctica que existe para la enseñanza de la programación, sin embargo, existe una inclinación por el PC, la resolución de problemas matemáticos, así como el uso de robots físicos y no tanto virtuales[8], además es importante señalar que los estudios analizados tuvieron también una tendencia cualitativa.

Contenidos y habilidades que se han desarrollado con el Pensamiento Computacional

Es necesario complementar el análisis anterior, con los contenidos y habilidades que según los estudios analizados desarrolla el PC. Comparado con la revisión de la literatura elaborada por Palma y Sarmiento (2015), los contenidos

8. En el caso del estado del arte de Palma y Sarmiento (2015), se identificó el uso de diversos paquetes de *Software* de programación en donde no era necesario el uso de un robot físico, sin embargo, actualmente la tendencia marca que se están adquiriendo robots para materializar las prácticas de programación.

y habilidades abordados por estudios entre 2004 y 2005, guardaban mayor relación con las fases de la resolución de problemas, fracciones, números decimales, transición a un lenguaje de programación como Java, interactividad con teclado y mouse, expresiones numéricas, probabilidades de ocurrencias, patrones matemáticos y secuencias (Palma & Sarmiento, 2015).

Por el contrario, los estudios que conforman la presente revisión de la literatura, se relacionan con la resolución de problemas (con sus respectivas fases), procedimientos de solución secuenciales, pensamiento lógico-matemático, inclusión de asignaturas como historia, geografía, ciencias naturales, artísticas, así como trabajo colaborativo y los beneficios que tiene la programación en la creatividad, motivación y autonomía.

El pensamiento lógico-matemático y su relación con el Pensamiento Computacional

Se ha comprobado que los ambientes de aprendizaje en donde se incluye la robótica educativa y el PC, resultan ser una experiencia significativa, Guardiola (2018) refiere que se desarrollan "nuevas habilidades, nuevos conceptos, fortalece el pensamiento sistémico, lógico, estructurado y formal del estudiante" (p. 16). En este sentido, el pensamiento lógico-matemático puede desarrollarse con base en tareas de programación, basadas en ciclos de cuatro etapas, por ejemplo, en el estudio de Guardiola (2018) se elaboró un ciclo de programación, conformado por conectar, construir, contemplar y continuar. Estas fases se repiten las veces que sean necesarias para depurar la resolución de los problemas.

Por otro lado, autores como Muñoz-Repiso y Caballero-González (2019) se relacionan con la propuesta de Guardiola (2018), principalmente en el aspecto de que el pensamiento lógico-matemático debe de categorizarse para una mejor comprensión. Por lo que en su estudio enfatizó en la acción-instrucción, secuenciación de las instrucciones y depuración. Así mismo, Chalmers (2018) también propone que los problemas deban solucionarse con base en las fases

de identificación, descomposición y abstracción[9], que son procesos centrales del PC.

Por lo anterior, el PC es capaz de desarrollar el pensamiento lógico-matemático, así como el desarrollo de nuevas habilidades para la resolución de problemas, específicamente habilidades como la descomposición, abstracción, depuración, automatización, etc. Por lo tanto, cuando se resuelven problemas con base en el PC, es posible identificar ciertas semejanzas, en otras palabras, se trata de evidencia empírica que demuestra que el PC es capaz de organizar de manera sistematizada la resolución de problemas en fases específicas, lo que trae como consecuencia mejores resultados en el aprendizaje, esto se expone en la tabla 1.

Tabla 1. Las fases de resolución de problemas en la programación.

Fases Autores	Fase 1	Fase 2	Fase 3	Elemento independiente
Papert (1980)	Identificación del problema y relación con sus conocimientos previos	División en problemas simples o sub-problemas	Análisis y relación con otros problemas (conocimientos previos)	Depuración Proceso de asimilación
Guardiola (2018)	Conectar	Construir	Contemplar y continuar	Depuración
Muñoz-Repiso y Caballero-González (2019)	Acción-Instrucción	Secuenciación de las instrucciones	Depuración	
Chalmers (2018)	Identificación	Descomposición	Abstracción	Depuración
Cheng (2018)	Identificación de sub–problemas	Planificación y ejecución del procedimiento	Análisis de resultados	

Fuente: Elaboración propia con base en los autores mencionados en la misma.

El desarrollo del PC, entendido éste como "un conjunto de habilidades y destrezas (…) que todos los humanos deberían poseer y utilizar para resolver problemas" (Adell, et al., 2019, p. 173), posee la característica principal de aprender de los errores cometidos a través de la depuración de los mismos (Papert, 1991; Wing, 2017; Chalmers, 2018). Lo que demuestra que los estu-

9. Chalmers (2018) hace énfasis en que en las 3 etapas propuestas es necesario realizar un ejercicio de depuración.

dios que se presentan en la tabla comparativa 1, tienen una estrecha relación con el PC y su vinculación con la resolución de problemas.

El pensamiento lógico-matemático con base en el PC, se desarrolla a través del orden sistémico que se le otorga a la resolución de problemas (Guardiola, 2018), lo que explica el por qué los estudios analizados (véase tabla 1) hacen énfasis en identificar el problema, dividirlo en sub-problemas, elaborar acciones para solucionarlos y por último depurar dichos procedimientos y relacionarlos con sus conocimientos previos, es decir, con problemas que solucionaron en el pasado (Papert, 1991; Guardiola, 2018; Muñoz-Repiso & Caballero-González, 2019; Chalmers, 2018; Cheng, 2018).

El pensamiento algorítmico-secuencial y su relación con el Pensamiento Computacional

El uso de algoritmos en forma secuencial es una de las características principales del PC y la programación (Torrejón & Ventura, 2019), ya que ayuda a los estudiantes a encontrar el camino de vuelta a la hora de depurar los códigos que utilizan, por otro lado, es importante señalar que autores como Torres, González y Carvalho (2018) relacionan la programación secuenciada de órdenes con el desarrollo de la orientación espacial y la estructuración de procesos mentales bajo "una lógica funcional y significativa" (p. 16).

En este sentido, el pensamiento secuencial se relaciona con la algoritmia o pensamiento algorítmico, que Palma y Sarmiento (2015) definen como un conjunto de órdenes secuenciales, que a su vez ejecutan tareas particulares. Es necesario enfatizar en que este tipo de procesos se observan en los paquetes de *software* de programación para niños, ya que poseen bloques de colores y formas distintas, que deben de ordenarse en una secuencia lógica que permita que el robot lleve a cabo las acciones programadas. No obstante, la diferencia que existe entre el PC, el Pensamiento algorítmico y el Pensamiento Secuencial, es que estas dos últimas se desprenden del desarrollo del PC, dado que se trata de un concepto más amplio que incluye habilidades, tipos de pensamiento y destrezas. (Torrejón & Ventura, 2019).

La motivación, creatividad y autonomía

La mejora en la motivación es uno de los temas más repetitivos en los estudios analizados, sin embargo, ¿A qué se debe este fenómeno? Algunos autores refieren que existe una motivación intrínseca y extrínseca en actividades con base en el PC, que se detonan a través del interés por las actividades y el estímulo que brindan los docentes respectivamente (Barrera, 2015). Por otro lado, la resolución de problemas con actividades lúdica-robóticas divierte a los estudiantes, lo que trae como consecuencia la motivación por el aprendizaje (Palma & Sarmiento, 2015; Constante, Chimbo, Jiménez & Gordón, 2019).

Un aspecto relevante tiene que ver con generar motivación en los estudiantes, a través del abordaje de los problemas o temáticas específicas con base en el robot como herramienta didáctica. Ya que, desde la perspectiva de Garnica (2014), la solución de problemas debe de ser la meta principal, de otra manera, enfocarse en aspectos técnicos del robot terminaría agotando la motivación.

La creatividad y autonomía de los estudiantes juegan un papel clave en el desarrollo del PC, ya que desde el enfoque del estudiante como sujeto activo (Muñoz-Repiso & Caballero-González, 2019), es necesario que trabaje colaborativamente (Miranda, 2018) y utilice su imaginación, con el propósito de aprender nuevos conocimientos y desarrollar habilidades, con base en probar sus límites y retroalimentarse (Muñoz-Repiso & Caballero-González, 2019). Cabe señalar que, sin importar la edad del estudiante, la creatividad y autonomía son habilidades que deben estimularse a lo largo de la vida, por lo que la robótica educativa y el PC son herramientas útiles (Jiménez, Ramírez & González, 2010).

Autores como Muñoz-Repiso y Caballero-González (2019) y Chalmers (2018), refieren que los estudiantes pueden recibir retroalimentación (depuración de procedimientos) por parte de sus propios compañeros, es decir, no solamente del docente. No obstante, a pesar de que se mencionan procesos centrales del PC, tales como la depuración o la descomposición, no han sido estudiados a fondo, por lo que es necesario enfocarse en ellos para determinar su influencia en el desarrollo del PC (Bocconi, et al., 2016).

La presente unidad de análisis demuestra que existen tendencias y debates en el estado del conocimiento actual que es necesario señalar:

- El desarrollo del PC a través de la relación entre el pensamiento lógico-matemático y la resolución de problemas secuenciales son uno de los temas más estudiados, dada su importancia en la programación y en la robótica.
- El PC y la programación están siendo aprovechados para enseñar contenidos de historia, geografía, ciencias naturales y artes.
- El estudio de la influencia de los Procesos Centrales del PC es un tema que necesita ser investigado a profundidad (Bocconi, et al., 2016; Adell, et al., 2019).
- Los estudios guardan una estrecha relación con el PC, por lo que su implementación es tendencia en la comunidad científica.

El robot como recurso didáctico

El presente apartado da cuenta de las características de los robots y *software* de programación utilizados. Ya que de esta forma es posible determinar sus semejanzas, diferencias, potencialidades didácticas y posibles usos en diversos contextos. En este sentido, el análisis se dividirá en dos tipos de herramientas, los robots virtuales y los físicos.

Los robots virtuales pueden definirse como aquellos que solamente necesitan de un *software* para programar (Palma & Sarmiento, 2015), por ejemplo, *PiktoMir*, *Lightbot*, *Gidget*, *Squake Etoys* y *Scratch*. No obstante, este último puede utilizarse también para programar robots físicos como *Lego WeDo*, *Mindstorms* y *Mbot*. Para facilitar el análisis, se presenta una tabla comparativa en donde es posible identificar las características de cada una de estas herramientas (véase tabla 2).

Tabla 2. Herramientas para la enseñanza de la programación.

Características Robot	Ofrece al menos estructuras de programación básicas	Código con sintaxis simple y método gráfico	Permite la depuración sin detectar errores en automático	Posee un soporte al usuario (guía de ayuda)	Propicia la colaboración entre usuarios
Bee-Bot	✔	✔	✔	✔	
Mbot	✔	✔	✔	✔	
LEGO WeDo	✔	✔	✔	✔	✔
LEGO Mindstorms	✔	✔	✔	✔	✔
PiktoMir	✔	✔	✔		
LightBot	✔	✔	✔	✔	
Scratch	✔	✔	✔	✔	✔

Fuente: Elaboración propia con base en (Erol & Kurt, 2017; Guardiola, 2018; Muñoz-Repiso & Caballero-González, 2019).

Las estructuras de programación

Los paquetes de *Software* de programación para niños poseen estructuras en forma de bloques, que facilitan su selección, clasificación y orden secuencial de los códigos (Muñoz-Repiso & Caballero-González, 2019), algunos de estos bloques se asignan directamente a sensores específicos del robot, por ejemplo, en el *Mbot* existen estructuras específicas para programar el sensor sigue líneas, motores izquierdo y derecho, así como las luces led (Sáez-López, Sevillano-García & Vazquez-Cano, 2019).

Como puede apreciarse en la tabla comparativa 2, todos los robots analizados ofrecen estructuras de programación básicas, no obstante, unas más sencillas que otras, por ejemplo, en el caso del *Bee-Bot* se programa a través de los botones que posee en la misma herramienta. Esto permite que los estudiantes observen el proceso de programación de sus compañeros y aprenda de los errores de todos (Di Lieto, et al., 2017). Por otro lado, el sistema *Lego* en sus

versiones *WeDo* y *MindStorms*, así como *Scratch*, se caracterizan por sus bloques más complejos que los del *Bee-bot*, *Piktomir* y *LightBot*, ya que estos tres se apoyan en flechas de dirección, por lo que se omite la complejidad de estructuras como ángulos, velocidad, uso de sensores, entre otras (Erol & Kurt, 2017; Guardiola, 2018; Pérez-Marín, Hijón-Neira, Bacelo & Pizarro, 2018).

La sintaxis del código y su método gráfico

El aprendizaje de la programación desde la perspectiva del PC, debe de funcionar con la misma naturalidad con la que se aprende cualquier otro lenguaje, es decir, como se dijo en apartados anteriores, un ambiente en donde el estudiante juegue el rol activo. Similar a la metáfora a la que Papert (1980) definiría como *matemalandia*[10]. Por lo anterior, se sugiere que el *software* se encuentre en el idioma nativo del alumno (Palma & Sarmiento, 2015), además de que se caracterice por un método gráfico que facilite su comprensión (Torrejón & Ventura, 2019; Sáez-López, Sevillano-García, Vazquez-Cano, 2019). En este sentido, el *Mbot* y su *software* de programación de *Mblock* es una variante de *Scratch*, que posee las características recomendadas (Sáez-López, Sevillano-García, Vazquez-Cano, 2019).

Uno de los aspectos relevantes que destacó en su momento Palma y Sarmiento (2015), así como los estudios de Muñoz-Repiso y Caballero-González (2019) y González-Martínez, Estebanell y Peracaula (2018), la depuración[11] es uno de los procesos que necesariamente los realiza el estudiante con el propósito de aprender de sus errores y mejorar los códigos futuros. Sin embargo, es de igual importancia que el *software* de programación no guíe en este tipo de casos, ya que es el alumno el que debe detectar cuál es el bloque (estructu-

10. La metáfora de matemalandia se refiere a que un estudiante puede aprender a programar bajo condiciones adecuadas, es decir, en donde se tenga acceso a computadoras, interactúe con ellas y el sujeto programe a la computadora, no al revés (Papert, 1980). Por otra parte, el autor también recomienda el uso de lenguajes de programación diseñados específicamente para niños, como *Logo* o *Scratch*. Que Palma y Sarmiento (2015) denominarían pseudolenguajes.

11. La depuración definida en el estudio de Muñoz-Repiso y Caballero-González (2019) como la habilidad en donde se identifica y corrigen errores.

ra de programación) que falló en el procedimiento para resolver el problema (Muñoz-Repiso & Caballero-González, 2019).

Soporte de ayuda al usuario y su transición a la colaboración entre programadores

Visualizar a los estudiantes desde una perspectiva en donde ellos son los programadores, permite dejar de pensar en el uso de guías e instructivos tradicionales. Actualmente los sistemas *Lego* y *Scratch* poseen una página web que brinda la oportunidad no solamente de aprender cómo funcionan los robots y su proceso de programación, sino que pueden compartir sus experiencias, visualizar códigos elaborados por otros e incluso modificarlos agregando sus propias ideas. Es necesario señalar que, aunque no se identificó la idea de la transición de los tutoriales comunes a aquellos en donde es posible aprender de los códigos de otros programadores, es posible pensar en que éste puede ser un futuro debate en estudios recientes.

A manera de conclusión, la forma en la que se abordó la presente unidad de análisis no se enfocó en cómo funcionan cada uno de los robots y *software* de programación, ya que esto evidentemente se puede investigar en internet, sino que se priorizó en las características en común, con el objetivo de que futuras investigaciones puedan realizar análisis más profundos, que incluso contradigan lo expuesto en este apartado.

La investigación sobre la enseñanza de la programación: una visión metodológica

La presente sección se caracteriza por realizar una revisión comparativa sobre la metodología de investigación que utilizaron los estudios analizados anteriormente. Como puede observarse en la tabla comparativa 3, la metodología predominante es la cualitativa, sin embargo, el enfoque mixto también posee un número considerable de estudios. Esto se debe al propósito y número de participantes de cada uno de los proyectos, es decir, los estudios enfocados a un número extenso de participantes optaban por un enfoque mixto, por otro lado, aquellos que se enfocaban en un número reducido de sujetos de inves-

tigación utilizaron un método cualitativo. Por último, se identificó que la metodología más utilizada fue el estudio de caso, ya sea único, comparativo o experimental.

Tabla 3. La enseñanza de la programación y metodologías de investigación

Metodología	Cuantitativa	Cualitativa	Mixta
Constante, Chimbo, Jiménez y Gordón (2019)		✔	
Torrejón y Ventura (2019)			✔
Miranda (2018)			✔
Barrera (2015)		✔	
Guardiola (2018)			✔
González-Martínez, Estebanell y Peracaula (2018)			✔
Scaradozzi, Sorbi, Pedale, Valzano y Vergine (2015)		✔	
Chalmers (2018)		✔	
Di Lieto, et al. (2017)	✔		
Tochacek, Lapes y Fuglik (2016)		✔	
Torres, González y Carvalho (2018)		✔	
Muñoz-Repiso y Caballero-González (2019)			✔
Sáez-López, Sevillano-García y Vazquez-Cano (2019)			✔
Cheng (2018)			✔
Jiménez, Ramírez y González (2011)		✔	
Erol y Kurt (2017)	✔		
Garnica (2014)		✔	

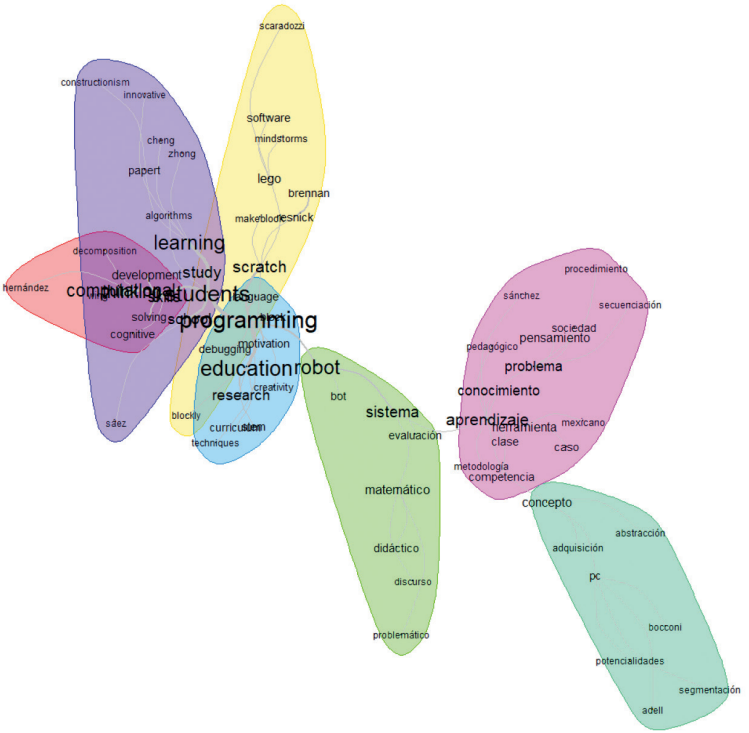
Fuente: Elaboración propia en los autores mencionados en la misma.

El debate en torno al Pensamiento Computacional: partiendo de lo conocido para proponer lo inédito

Las categorías de análisis anteriores dan cuenta de lo que ya se sabe en torno al PC, es decir, las estrategias más utilizadas para la enseñanza de la programación, así como las ventajas que tiene la implementación de la propuesta pedagógica del PC. No obstante, es necesaria una mirada crítica sobre el tema, que visibilice aquellos aspectos que se desconocen, incluso aquellas desventajas que se han identificado en torno al PC. En otras palabras, la presente categoría muestra el otro lado de la moneda, que sirve para complejizar y ubicarse en las tensiones, como parte de un ejercicio de extrañamiento

Para este ejercicio, se llevó a cabo un análisis de los artículos seleccionados. En éste destaca que se incluyeron todos los artículos, además se llevó a cabo una depuración de la información detallada, que requirió de un trabajo minucioso. Cabe señalar que, se incluyen nuevos estudios que presentaron hallazgos distintos con respecto a la implementación del PC en el currículum escolar, algunos con críticas a su definición y refiriéndose a los procesos centrales del PC como la clave para posicionarlo como una propuesta sólida (Adell, et al., 2019; Bocconi, et al., 2016).

El debate en torno al Pensamiento Computacional.



Fuente: Elaboración propia

El PC se encuentra en un debate constante, autores como Bocconi, et al., (2016) evidenciaron los vacíos en el conocimiento con respecto a la didáctica,

formación docente, evaluación, etc. Por lo tanto, este tipo de hallazgos contradicen a aquellos como Brennan y Resnick (2012), Wing (2017) y Csizmadia (2015), que han dedicado sus esfuerzos en impulsar al PC, no obstante, ¿Cómo impulsar conceptos teóricos que todavía necesitan legitimarse? En este sentido, Adell, et al. (2019) se posiciona en un extremo[12], explicando que el PC es una propuesta capitalista en búsqueda de mano de obra barata en la industria de la programación. No obstante, después de llevar a cabo el análisis de literatura especializada, la postura de Adell, et al. (2019) podría ser arriesgada, ya que ni Bocconi et al. (2016), ni Jordi Adell brindan evidencia que sustente ese argumento sobre la llamada *perversión del PC*.

El avance tecnológico en cuestión de las aplicaciones en dispositivos inteligentes, el fenómeno del consumo audio-visual por demanda, el *streaming*, videoconferencias virtuales, el uso de la Inteligencia Artificial (IA), entre otros, han tenido como consecuencia, la necesidad de comprender cómo es que funciona y se programa cada una de estas herramientas. El PC se encuentra justo dentro de este debate, que busca que los estudiantes desarrollen habilidades computacionales específicas que les permitan reconocer las herramientas tecnológicas como una extensión de sí mismos[13], ya sea a través del análisis, automatización y abstracción de procesos.

Por otro lado, existe un punto de acuerdo entre estos dos extremos, ya que tanto Wing (2017), como Bocconi, et al., (2016), Adell, et al., (2019), Brennan y Resnick (2012) y Csizmadia (2015) concuerdan en que es necesaria mayor profundización en el tema, principalmente en la comprensión de cómo se desarrollan ciertos procesos del PC. Ciszmadia (2015) ha dado el siguiente paso, al operacionalizar el PC señalando y exponiendo sus 6 procesos centrales, no obstante, un año después, Bocconi, et al., (2016) demuestra mediante una revisión exhaustiva de la literatura, que es necesario comprender a profundidad

12. Como se puede apreciar en el esquema 1, Adell y Bocconi se posicionan en el extremo derecho. Por lo que sus disertaciones están cargadas de crítica, lo que trae como consecuencia una necesidad por investigadores que partan del debate desde las tensiones, que les brinden una mirada compleja y no reduccionista.

13. La "Extended-Mind" se trata de un concepto atribuido a Clark y Chalmers (1998), sin embargo, con el avance tecnológico cobra mayor sentido al relacionarlo con el PC.

cómo se desarrollan, en qué contextos, qué dificultades existen, entre otros cuestionamientos teórico-pedagógicos.

El Pensamiento Computacional e *Iramuteq* como herramienta de análisis

Iramuteq es una herramienta que permite realizar un conteo léxico, además de identificar comunidades o categorías que se relacionan, esto con base en la frecuencia con la que aparecen en los textos. El análisis que se llevó a cabo con *Iramuteq* y que trajo como resultado el esquema 1, abrió nuevos caminos para la construcción del presente análisis. Por lo tanto, en el presente apartado se expondrán dichas reflexiones, con el propósito de construir relaciones entre la robótica educativa, la programación, el pensamiento computacional y el proceso de descomposición.

Son diversos los estudios que se han dedicado a dar cuenta de la influencia que tiene el PC en la programación de robots educativos, no obstante, aunque las definiciones de Wing (2017) hacen énfasis en que el PC puede desarrollarse en contextos no computables, pareciera todavía indisociable de la programación y por consecuencia de *Scratch*, *Makeblock*, *Lego Mindstorms*, *Blockly*, entre otros. Estas herramientas se relacionan constantemente con autores como Brennan y Resnick (2012), sin embargo, no es una simple casualidad.

Investigadores como Brennan y Resnick (2012) y Wing (2017), provienen de una esfera que no solamente estudia actualmente el PC, sino que lo impulsa en Estados Unidos y que repercute a nivel internacional. Como se observa en el esquema 1, es la programación de donde se desprenden temas como la mejora de la motivación y creatividad a través de esta propuesta pedagógica. Asimismo, uno de los hallazgos más relevantes que se pueden observar en este análisis, es que el PC se encuentra separado de la programación, es decir, cada uno cuenta con una comunidad[14] propia. A pesar de estar conectados, llama la atención que existe un espacio de separación que se llena con el recurso robó-

14. *Iramuteq* nombra como "comunidades" a cada una de las esferas de colores que se encuentran en el esquema 1.

tico, es decir, lenguajes de programación como *Blockly* y *Scratch* que toman el papel de "puente" entre ambas comunidades.

Desde la perspectiva de Adell, et al. (2019), todavía se tiene una pregunta fundamental, "¿Contribuye la programación de robots educativos sencillos al desarrollo del PC en niños y adolescentes?" (p. 177). Esta reflexión posee más sentido cuando se observa a la luz del análisis con *Iramuteq*. Adell, et al. (2019) y Ioannou y Makridou (2018), concluyen que apenas existen nueve publicaciones que relacionan el desarrollo del PC y la programación, en donde se identificaron evidencias no tan claras con respecto a sus procesos centrales, específicamente los de descomposición, abstracción, algoritmos y la depuración.

Lo anterior da cuenta del debate actual en torno al PC, que no solamente se sitúa en llegar al consenso de una definición operacional, sino que también pasa por otros polos que vale la pena mencionar, como la relación PC-programación, que no precisamente se trata de un binomio, sino de una categoría compleja, en donde intervienen los procesos centrales del PC, robots educativos, resolución de problemas, estrategias didácticas y de evaluación.

Con respecto a los procesos centrales del PC, parecieran desorganizados en el esquema 1, ya que por un lado se pueden encontrar la abstracción y secuenciación, mientras que dentro de las comunidades de programación y robótica educativa se encuentra la depuración. De igual forma, el proceso de descomposición es el único que se ubica dentro de la comunidad del PC. Esto puede deberse a diversas razones: Bocconi, et al. (2016) refiere que los seis procesos centrales resuelven la falta de consenso conceptual del PC, no obstante, es necesaria mayor profundización en cada uno de ellos, sobre todo para dar respuesta al cuestionamiento que propuso Adell, et al. (2019) y que en su momento adelantó Wing (2017).

Lo que es una constante, es que la International *Society for Technology in Education* (ISTE) y *Computer Science Teachers Association* (CSTA) en el (2011), además de Brennan y Resnick (2012), Rich y Langton (2016), Corradini et al., (2017), Adell, et al., (2019), así como Bocconi, et al., (2016), concuerdan en que los procesos centrales del PC son clave para su operacionalización y su comprensión teórica.

Conviene recordar que estos procesos son la abstracción, pensamiento algorítmico, descomposición, depuración, automatización y generalización (Bocconi, et al., 2016). A pesar de que ninguno posee mayor peso sobre otro, ya que se relacionan de manera transversal, son éstos los que resignifican la práctica de la programación, dado que la *tecnicidad* al programar un robot se convierte en una tarea más compleja (Martín-Barbero, 2009).

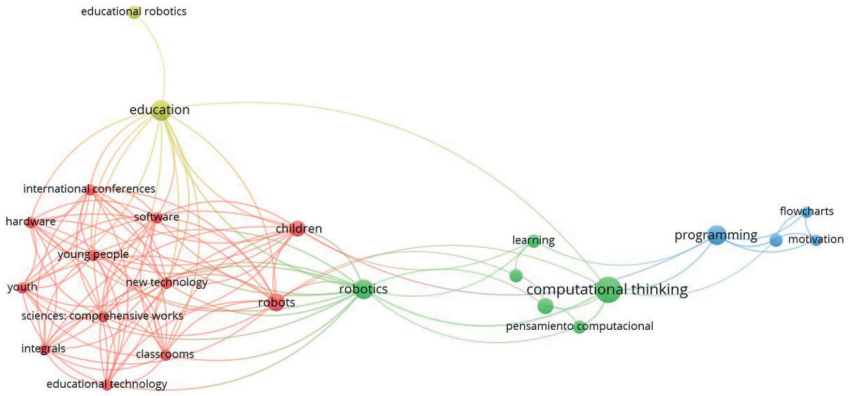
Una vez que se identificó que el proceso de descomposición es el único inmerso en la comunidad del PC, se analizaron las causas de este fenómeno. Llegando a las siguientes conclusiones:

- Entendiendo que la abstracción tiene que ver con un proceso de seleccionar aquello que es relevante y depurar lo irrelevante, es necesario un ejercicio de descomposición para extraer sus elementos más importantes.
- El pensamiento algorítmico se relaciona con llegar a una solución con una serie de pasos estipulados, en donde indudablemente es necesario descomponerlo en estas "fases" de solución.
- La depuración necesita de un análisis y evaluación que sea predictiva y verifique errores en los procedimientos, por lo tanto, nuevamente la descomposición será necesaria para segmentar cada uno de los errores y corregirlos.
- La automatización ejecuta y repite lo que el sujeto codifica, por lo que es necesario descomponer estas ejecuciones de manera lógica, para que tengan sentido para el lenguaje de programación.
- Por último, la generalización se asocia con la identificación de patrones, en donde será necesario descomponer aquellas similitudes y relaciones entre problemas antiguos para que, en su momento, puedan reconstruirse y aplicarse en problemas nuevos.

Estas conclusiones se realizaron con base en la revisión de la literatura elaborada por Bocconi, et al., (2016), que recuperó la esencia de cada uno de los procesos centrales, sin embargo, contrastándolo con el análisis expuesto en el esquema 1, el proceso de descomposición interviene de manera constante en el desarrollo de los otros. Esto explica que se encuentre dentro de la comunidad del PC, y directamente relacionado con la programación, la resolución de problemas y la robótica educativa.

Otro asunto interesante, tiene que ver con la identificación concurrencias en otro *software* de análisis. En éste se observa la decantación clásica del PC por la mejora de la motivación y el uso de los robots educativos. Mientras que, sin importar ir contra en contra de la corriente, existen líneas de relación que saltan el uso de los robots para aterrizar directamente en los estudiantes y los procesos de aprendizaje mismos. ¿Será entonces que esas pequeñas líneas están abriendo el camino del PC desconectado? ¿Será esa la salida para los países que necesitan del PC, pero no cuentan con el poder adquisitivo para la inclusión de robots con tecnología de punta?

Esquema 1. Relaciones entre los conceptos involucrados.



Fuente: Elaboración propia con base en los estudios expuestos en el presente capítulo.

A diferencia que en el análisis con *Iramuteq*, *VosViewer* arrojó resultados que apuntan hacia una separación categorial entre el uso de las tecnologías, es decir, aquellas disciplinas relacionadas al asunto computacional, y por otro lado, el PC, que trae consigo el aprendizaje de la programación. Si se observa el esquema de *VosViewer* verticalmente, es posible ver que, quien mueve los hilos es la educación, y que, detrás de ella se encuentra la educación robótica. A sus pies se encuentra la tecnología, el *hardware*, el *software*, que conectan al pensamiento computacional con la mejora de la motivación.

Aunque pareciera que la robótica es el punto de encuentro entre ambos mundos, existen hilos que saltan a este nodo para ir directo al aprendizaje

(como primera estación), pasar por el aprendizaje (segunda estación) y, por último, desembocar en la educación misma. Este comportamiento inusual habla de los esfuerzos por incluir el PC en contextos no computacionales. ¿Será que la descomposición pueda desarrollarse desde una caja negra alterna? Es decir, aquella que se da fuera de la programación virtual, en contextos desenchufados del robot. En esta obra no se cierra la mirada a uno u otro mundo, si existe algún elemento fuera del escenario de la programación complejizaría todavía más la caja negra, y eso sería, sin lugar a dudas, benéfico para el estado del conocimiento actual.

3. El PC y su relación con el *Mbot*

Para comprender a profundidad los fundamentos del PC y de la propia descomposición, es necesario también dar cuenta de las características del *Mbot*, ya que los supuestos teóricos que se expondrán más adelante, y que sin lugar a dudas son el propósito principal de esta obra, necesitan de una descripción de los elementos que conforman al robot, visto como medio o recurso didáctico que permite a un sujeto desarrollar estrategias de descomposición y programación.

La concepción del PC tiene diversas vertientes que lo han caracterizado. Por un lado, se encuentra la característica principal que lo relaciona con el debate actual sobre su definición operativa, a la que cabe señalar, todavía no se encuentra inacabada e inconclusa. Por otro lado, algunos autores como Papert (1980) lo han vinculado al construccionismo e ideas piagetianas, lo que explica el por qué posee conceptos como la clasificación, pensamiento reversible, permanencia, conocimientos previos, esquemas mentales, entre otros.

El construccionismo de Papert (1993) sentó las bases para la iniciación de los estudiantes en la programación y el desarrollo del PC, no obstante, este proceso teórico-empírico que se expuso en libros como *Mindstorms: Children, computers and powerful ideas*, también fue pieza clave para los inicios del PC con Jeannette Wing (2006). Prueba de lo anterior se refleja en Papert (1991), y su concepto de *procedural thinking*, éste fue tomado nuevamente por Wing (2006), ampliándolo y explicándolo en el PC.

El concepto *Procedural Thinking* fue considerado como el primer acercamiento para pensar en que, los estudiantes sean los que programan a las computadoras, y no como se había planteado en el pasado. No obstante, Wing (2006) construyó un concepto más completo que sería aceptado en la comunidad científica y sería investigado gradualmente con el paso de los años. En este

sentido, esta autora refiere que el PC "implica la resolución de problemas, el diseño de sistemas y la comprensión de la conducta humana, haciendo uso de los conceptos fundamentales de las ciencias computacionales" (p. 33). Asimismo, agrega que implica un conjunto de habilidades y actitudes, que deberían ser incluidas al mismo nivel de la lectura, escritura y aritmética en el currículum de cualquier sistema educativo.

Posteriormente, surgieron otras definiciones en torno al PC, tal es el caso de Brennan y Resnick (2012), que lo definen como un proceso cognitivo relacionado con la resolución de problemas, que pueden resolverse de manera efectiva a través de un agente de procesamiento de información. Por otro lado, Wing (2008) agrega que se trata de un proceso que puede abordarse desde los primeros años en los estudiantes, y que es capaz de acoplarse a diversas disciplinas, no sólo a las matemáticas.

Posteriormente Wing (2017) propuso una nueva definición, que actualmente es una de los más utilizadas en los estudios analizados en la revisión de literatura, en ésta el PC se define como "*the thought processes involved in formulating a problem and expressing its solution(s) in such a way that a computer-human or machine-can effectively carry out*" (p. 8). Cabe señalar que, actualmente el PC se encuentra en un debate constante por la construcción de la definición más acertada.

Para lograr definir al PC integralmente, es necesario identificar los elementos que lo conforman. Por tal motivo, existen autores que se han enfocado no solamente en su definición, sino en rescatar cuáles son sus características, con el propósito de evidenciar qué es y qué no es PC. En este punto, Grover y Pea (2013) refieren que el PC involucra a la abstracción y generalización, procesamiento de información, descomposición estructurada, lógica condicional, noción algorítmica, depuración, sistemas de símbolos, pensamiento iterativo, recursivo y paralelo, así como limitadores de eficiencia y rendimiento (pp. 39-40).

Por otro lado, otros autores han identificado seis conceptos que juegan el papel de habilidades que involucra el PC, tales como el pensamiento algorítmico, descomposición, generalización, abstracción, evaluación y razonamiento lógico. (Csizmadia et al., 2015). En este sentido, se trata de habilidades cen-

trales que permiten comprender el alcance del PC y cómo se relaciona con la programación (Cárdenas, 2017).

Por lo anterior, es necesario describir cada una de las habilidades antes mencionadas. Primeramente, la abstracción ha sido definida por Wing (2017) como la clave del PC, dado que desde su perspectiva se trata del proceso cognitivo de más alto nivel, que es utilizado para definir patrones y generalizaciones. Por otro lado, la abstracción es utilizada para "capturar las propiedades esenciales de una serie de objetivos, mientras se ignorar sus características irrelevantes" (Wing, 2017, p. 8). Por lo tanto, el proceso de abstracción toma una serie de *inputs* ejecutados en una serie de pasos, para producir los *outputs* planificados en objetivos específicos de la resolución de problemas (Wing, 2017).

La abstracción es entonces aquella que nos permite simplificar con base en los datos relevantes, ignorando los irrelevantes. Esta habilidad la utilizamos para gestionar incluso la complejidad que se vive en un contexto escolar. Por ejemplo, al elaborar un horario escolar, en donde se recolecta la información relevante, solamente para dar cuenta de los horarios de clases de cada grupo y asignatura (González, 2016).

Siguiendo con el tema de las habilidades centrales del PC, el razonamiento lógico es aquel que utilizamos para darle sentido a lo que sucede, es decir, clarifica y precisa el por qué los problemas que se nos presentan (Csizmadia et al., 2015), de tal forma que, va estrechamente relacionado con la depuración y evaluación de los algoritmos utilizados al resolver problemas.

Por otra parte, el proceso de evaluación tiene que ver con la identificación de las soluciones propuestas, a través de sus algoritmos, sistemas o procesos, y posteriormente tomar decisiones sobre la existencia de otros procedimientos más efectivos y rápidos (Csizmadia et al., 2015). Otros autores lo llaman depuración, como un proceso que se encarga de identificar errores (Cárdenas, 2017). Se caracteriza principalmente por cuatro etapas, tales como el diagnóstico, detección de errores, eliminación y testeo. (Csizmadia et al., 2015; Cárdenas, 2017).

Otro proceso es la generalización, éste se relaciona con la identificación de patrones, con base en las semejanzas y diferencias con problemas resueltos anteriormente. Es entonces "la manera más rápida de resolver problemas

nuevos con base en la solución de problemas previamente resueltos" (Csizmadia et al., 2015, p. 8). También se trata de una habilidad que permite formular soluciones genéricas para aplicarlas en diversos contextos (Cárdenas, 2017). Su propósito principal es predecir, crear reglas y resolver problemas más amplios (Gonzalez, 2016).

Para la resolución de un problema, es necesario el pensamiento algorítmico, que es también unas de las habilidades/procesos centrales del PC. Consiste principalmente en definir paso a paso la solución al problema, en términos de secuencias y reglas (Csizmadia et al., 2015). En este caso, los algoritmos como la multiplicación o la división son ejemplos claros, Csizmadia et al. (2015) refiere que, en el PC, una vez que se entiende el algoritmo, ya no será necesario empezar desde cero un nuevo problema. Por otro lado, Futschek (2006) relaciona esta habilidad con el análisis de problemas, que a su vez permiten la identificación de acciones para resolverlos.

Por último, pero no menos importante, el proceso de descomposición, que se caracteriza por segmentar el problema en pequeñas partes, de tal forma que puedan ser "entendidas, resueltas, desarrolladas y evaluadas por separado, de modo sistematizado" (Cárdenas, 2017, p. 35). Su propósito principal es convertir los problemas complejos en sub-problemas que sean más fáciles de comprender y resolver (Csizmadia et al., 2015).

En otras palabras, "el proceso de fraccionar un problema en sus partes constitutivas, más pequeñas y manejables, se conoce como descomposición (González, 2016, p. 155). Por lo tanto, se trata de un proceso de vital importancia para el PC, ya que, la abstracción necesita de éste para identificar los datos relevantes a través de la descomposición del problema general.

A manera de cierre, se presenta el esquema 2 que resume las habilidades centrales del PC.

Con respecto al proceso de descomposición, y llevando a cabo un análisis más a profundidad, Rich et al. (2018) lo define como una habilidad fundamental de la resolución de problemas, ya que en su estudio realizó un análisis sobre el estado del conocimiento actual sobre esta habilidad en particular. Con el propósito principal de elaborar una propuesta curricular en donde se defina la trayectoria de aprendizaje por la que debe pasar el estudiante para desarrollar dicha habilidad. No obstante, de entrada, Rich, Binkowski, Strickland y

Esquema 2: Habilidades/procesos centrales del Pensamiento Computacional.

Fuente: Elaboración propia con base en Csizmadia et al. (2015); Wing (2006, 2008 y 2017); Gonzalez (2016); Cárdenas (2017); Brennan y Resnick (2012); Grover y Pea (2013) y Futschek (2006).

Franklin (2018) destaca la falta de información en el tema, que a pesar de que identificó más de 30 artículos que estudiaron la descomposición, solamente tres se enfocaron en este proceso.

Por lo tanto, se ha identificado un concepto llamado EFGP, por sus siglas en inglés (*extremely fine-grained programming),* que da cuenta de la desinformación que se tiene sobre el proceso de descomposición. Ya que Meerbaum-Salant, Armoni y Ben-Ari (2011) determinaron que los estudiantes tenían problemas con este proceso en específico, ya que descomponían el problema en sub-problemas tan pequeños, que era imposible solucionarlos y terminaban obstaculizando el proceso.

Meerbaum-Salant, Armoni y Ben-Ari (2011) también destacan que, en el proceso de descomposición se han identificado dos extremos, por un lado, no descomponer el problema (Papert, 1980)[15], y, por otro lado, el segmentarlo en problemas tan pequeños que no tenga sentido separar uno de otro. Lo anterior demuestra que existe un desconocimiento con respecto al proceso de descomposición (aunque no totalmente). Sin embargo, dado que el PC se está aplicando en diversos sistemas educativos a nivel primaria, es necesario investigarlo a fondo, con el propósito de identificar obstáculos futuros.

Por otro lado, aunque se sabe que con la descomposición se pretende identificar, resolver, desarrollar y evaluar cada sub-problema por separado, las

15. Papert (1980) ya había hecho referencia a este problema que, aunque no lo documentó a detalle, observó su recurrencia en diversos estudios.

investigaciones actuales no definen cómo sucede ese proceso, es decir, ¿Ocurren en ese orden?, ¿Existen estudiantes que no puedan siquiera identificar los sub-problemas?, ¿Qué sucede con estudiantes con niveles de conocimiento distintos en un contenido en particular?, ¿La evaluación de los sub-problemas soluciona el tema del EFGP? Todos estos cuestionamientos podrían tener respuesta al identificar la relación que existe entre dicho proceso central y la programación, de tal forma que sea posible comprender los elementos que intervienen en ambos.

El *Mbot* y *Mblock*: una descripción teórica

En este apartado, se exponen las características del *Mbot*, con el propósito de brindar un marco de referencia útil para la comprensión del robot involucrado en la presente obra. Se trata de un robot que, según su creador (la empresa *Makeblock*), atiende a las tres áreas de la robótica, tales como la mecánica, electrónica y la programación. Este recurso puede ensamblarse manualmente con un kit simple de desarmadores, su sistema electrónico está basado en Arduino, compatible con diversos dispositivos, ya sean computadoras con Windows, Mac, así como aplicaciones de iOS y Android.

Este robot puede programarse a través de *Arduino IDE*[16] y la aplicación *mBlock*, esta última está basada en *Scratch* 2.0. Por otro lado, el dispositivo se conecta de manera alámbrica por puerto USB (por sus siglas en inglés Universal Serial Bus), así como Bluetooth de manera inalámbrica. Asimismo, el cableado que posee es sencillo, ya que se trata de conectores RJ25 con códigos de colores[17].

Ahora bien, con respecto a las partes del robot, conviene señalar que se dividen en dos partes, por un lado, los sensores (que capturan y guardan información) y, por otro lado, los actuadores (que realizan acciones que se le

16. Por sus siglas en inglés IDE (*Integrate Development Enviroment).*

17. Para mayor información es posible ingresar de manera gratuita a los cursos que la Secretaría de Educación Jalisco ofrece, de donde se obtuvo la información sobre las características del Mbot. Dicho curso se encuentra en el siguiente link: http://educacionvirtual.se.jalisco.gob.mx/dipta/course/view.php?id=46

programan). El *Mbot* cuenta con una tarjeta *mCore*, que toma el papel del *cerebro* del robot. Por lo tanto, posee varios dispositivos electrónicos como los LED (por sus siglas en inglés *light-emitting diode*), es decir, diodo emisor de luz (actuador). Este dispositivo es capaz de emitir luces de color rojo, azul y verde[18], por lo que es posible combinar colores.

El *Mbot* también posee un *buzzer,* que se encarga de detonar las vibraciones necesarias para emitir sonidos específicos, por ejemplo, notas musicales. Además, cuenta con un sensor de luz, este calcula la luz que existe, con el propósito de que siga o huya de ella. Asimismo, el robot tiene un receptor infrarrojo, el cual recibe señales infrarrojas por parte de un control, en donde el robot puede intercambiar información para moverse o llevar a cabo acciones particulares.

Otro de los componentes del *Mbot* es el seguidor de líneas, con el que es posible seguir una línea en el suelo, básicamente se dedica a calcular el nivel de luz que existe y de ahí moverse de acuerdo al programa que se le indique. También posee un sensor ultrasónico, que toma el papel de un sonar parecido al que utilizan los murciélagos, en donde mide la distancia entre un objeto y otro, de tal forma que puede evitar el contacto con paredes y otros objetos.

Evidentemente cuenta con un botón de encendido y apagado, además de una placa para conexión *bluetooth*. Por último, los actuadores motores, que transforman la energía eléctrica en energía cinética, que a su vez permite el movimiento del *Mbot*, cabe señalar que los motores se programan con base en la potencia de éstos, lo que determina la velocidad con la que se mueven. Por otra parte, esta velocidad se mide en revoluciones por minuto.

Con respecto a la programación del *Mbot*, como se mencionó anteriormente, esta tarea se lleva a cabo mediante *Mblock*, la cual funciona con los siguientes bloques:

- Bloques de luz y sonido
- Bloques de acción
- Bloques de sensores
- Bloques de eventos
- Bloques de control.

18. De ahí su nombre LED RGB, por sus siglas en inglés (*red*, *green* y *blue*).

Bloques de luz y sonido

Por ejemplo, para los LED RGB, es posible elegir cuáles (del par que posee) debe de encender, además de seleccionar el color y el tiempo de encendido. Otro ejemplo básico de programación de LEDs RGB en *Mblock*, es presionar una tecla específica de la computadora, para que enciendan los LED en color rojo y amarillo durante un segundo de encendido y uno de espera. Por último, este proceso se repita diez veces.

Bloques de acción

Este tipo de bloques se encargan de la programación de los motores, es decir, acciones como avanzar o retroceder en un porcentaje específico de potencia, además de acompañarse por un intervalo temporal. Asimismo, es posible programar giros o editar la potencia específica del motor derecho o izquierdo.

Bloques de sensores

Los sensores ultrasónicos, sensores de luz, seguidor de líneas, temporizador, y botón de encendido son los que se relacionan con estos bloques, en donde es posible adaptarlo según el puerto deseado, además de determinar (en el caso del sensor ultrasónico) la distancia requerida entre el *Mbot* y un objeto.

Bloques de eventos

Este tipo de bloques son determinantes para la utilización de teclas de computadora, además del inicio de algún código insertado. Entre los bloques básicos para su funcionamiento, se encuentran el de bandera verde, presionar tecla, recibir mensaje, emisión, además de emisión y espera.

Bloques de control

Estos bloques se encargan de controlar las veces que se repite un código, el tiempo de espera, elementos condicionales afirmativos o negativos, repetir hasta que un evento específico ocurra, detener, etc.

La interface de *Mblock*

Como se expuso en la revisión de literatura, *Mblock* es intuitivo y sencillo de manipular, y sus resultados en el aprendizaje pueden estar por encima de *Scratch* 2.0 (Guardiola, 2018), en casos específicos, aunque está basado en ese lenguaje de programación. En este sentido, *Mblock*[19] posee dos aproximaciones a la programación del *Mbot*, éstas son: el modo historia y la modalidad de crear.

El primero de ellos tiene que ver con la superación de desafíos guiados, es decir, se trata de tutoriales que enseñan a programar de una manera gradual cada uno de los sensores y actuadores del *Mbot*. Por otro lado, la segunda aproximación es la forma clásica de programación, por ejemplo, para los bloques de movimiento mantiene siete bloques base, además de que, aunque siguen siendo los mismos bloques, en la interface de un dispositivo móvil como la tableta cambian los nombres sin modificar el propósito y contenido de cada uno.

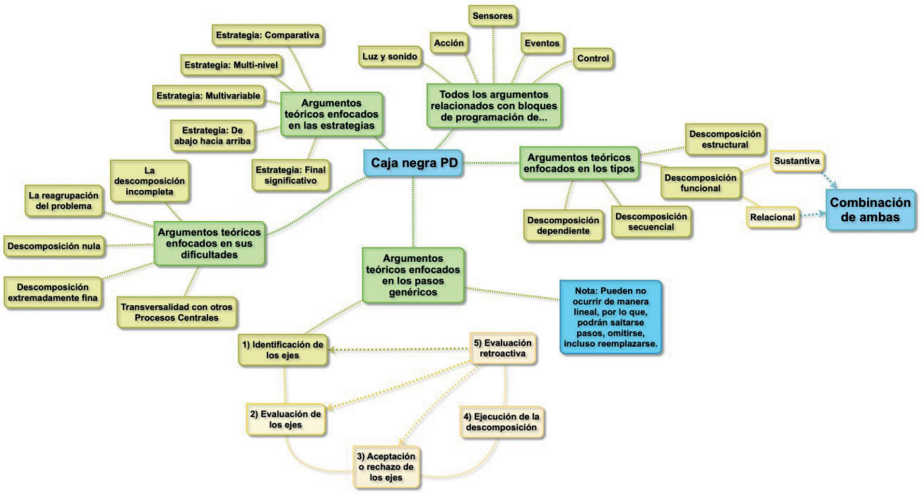
Discusión teórica: develando la caja negra

Una vez analizado el robot *Mbot*, se presenta una reflexión teórica acerca de la relación entre la programación y la descomposición. Por lo tanto, siguiendo con la misma lógica argumentativa, la discusión se organiza en dos apartados, el primero expone las premisas teóricas, tales como, los tipos, estrategias, así como los pasos genéricos de descomposición. El segundo apartado contiene argumentos propios, basados en la construcción de relaciones, identificación de regularidades y diferencias entre los conceptos clave de la descomposición y su relación con la programación. En el Esquema 3 se expone la organización de dichas categorías.

Aunque han sido poco exploradas, no son desconocidas las características básicas de la descomposición, éstas podrían ser vistas como los *inputs* de la llamada caja negra del PD. Asimismo, las dificultades y resultados evidenciados en diversos estudios, organizados de manera sistemática, podrían ser los *outputs*. El propósito del presente apartado, es develar teóricamente, y a

19. Particularmente la aplicación *Mblock-Blockly* para Tablet.

Esquema 3. Los argumentos teóricos.



Fuente: Elaboración propia.

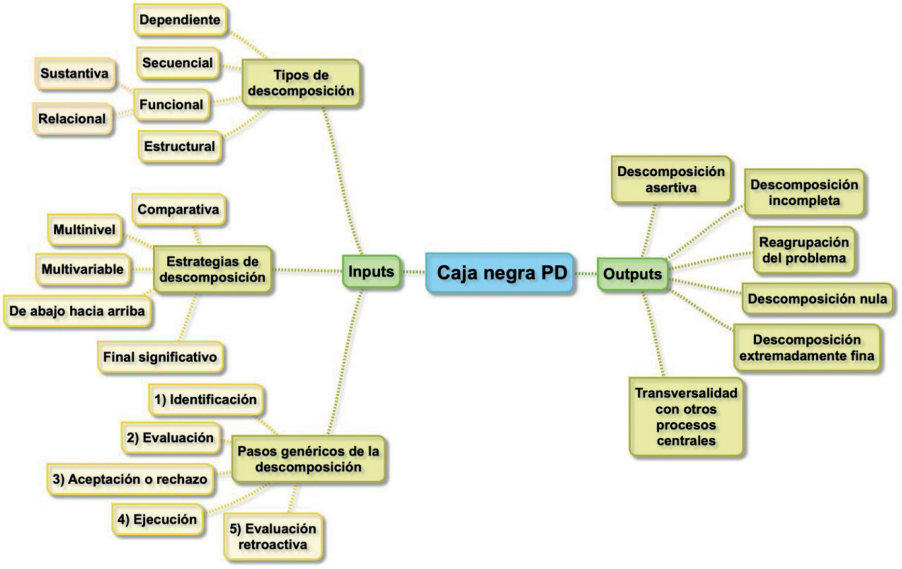
priori, las relaciones que ocurren entre la descomposición y la programación, que traen como consecuencia que ciertos *inputs* vinculados con otros, traigan como resultado *outputs* específicos. Como ejemplo, véase el Esquema 4.

Las estrategias de descomposición

Son escasos los estudios que han profundizado cualitativamente en el PD. Particularmente en el marco del PC, el estudio de Rich, Egan y Ellsworth (2019) abre el camino hacia nuevos análisis teóricos, ya que, hasta el momento, es el único marco teórico publicado en revistas de impacto científico, que ha hecho un esfuerzo por relacionar la descomposición con el PC. No así, con la programación.

En dicho estudio, se exponen diversos argumentos teóricos, tales como las estrategias y tipos de descomposición, así como los pasos genéricos necesarios para descomponer. Con respecto a las estrategias, son seis las que destacan dichos autores:

Esquema 4. La caja negra.



Fuente: Elaboración propia.

- Final significativo: La descomposición comienza por el último componente, es decir, en un pensamiento reversible, la reagrupación del problema se da comenzando por el último elemento descompuesto.
- De abajo hacia arriba: En este caso, la estrategia de descomposición hace énfasis en un sub-componente conocido, por lo tanto, la descomposición en general se basa en las relaciones que guarda este sub-componente con los otros. En otras palabras, se trata de aprovechar el conocimiento o dominio de un elemento del problema, para encontrar sentido a los otros.
- Multi-variable: La descomposición se centra en el uso de una categoría transversal, que contiene a su vez múltiples ejes (conjuntos de sub-componentes), que permiten averiguar cuáles son factibles o no, con base en su relación directa con la categoría base.
- Multi-nivel: En esta estrategia, se sigue la lógica de descomponer en diversos niveles, de tal forma que un componente, tendrá sus propios sub-com-

ponentes y así sucesivamente. En algunos casos, el propósito es llegar al sub-componente más pequeño.
- Comparativa: En este caso, la estrategia hace énfasis en organizar ejes específicos, o incluso, sub-componentes, que permitan reconocer semejanzas y diferencias entre los elementos descompuestos. Es útil para comparar incluso, diversos resultados de descomposición de un mismo problema (Rich, Egan y Ellsworth, 2019).

Cada una de estas estrategias de descomposición, mantienen características específicas. Los autores que se encargaron de identificar y organizar dichas características, refieren que, existe todavía un gran margen para la discusión. En este caso, Rich, Egan y Ellsworth (2019) llevaron a cabo una discusión a priori, a manera de ejemplificación, en donde con base en diversos *softwares* de programación, comenzaron a emitir ciertas relaciones entre las estrategias de descomposición. No obstante, es posible identificar que las relaciones construidas no profundizaron en el proceso de programación, dado que no era el propósito del estudio, solamente se discutió el tema de manera superficial.

Tipos de descomposición

Los tipos de descomposición, es otra categoría que, a pesar de que se presentan por separado, guardan una relación con las estrategias de descomposición. Principalmente porque en algunos casos, el tipo de descomposición marca la pauta para tomar decisiones, acerca de cuál estrategia es la más apropiada. Asimismo, las estrategias pueden llevar consigo diversos tipos de descomposición, no solamente uno, ya que se convierte en una serie de relaciones dependientes e independientes. En este sentido, los tipos de descomposición son los siguientes:

- *Dependiente:* Es aquella descomposición en donde algunas partes dependen de otras, por lo tanto, es clave la identificación de relaciones entre ellas.
- *Funcional:* En este tipo de descomposición, se hace énfasis en el papel o función que poseen cada una de sus partes, de tal forma que, existen elementos que no pueden ser separados y analizados por separado. Cabe señalar que, la función se identifica a partir de los componentes sustantivo y relacional de los componentes del problema.

- *Estructural:* Se caracteriza por la descomposición del problema en sub-partes, que a su vez permitan resolver cada uno individualmente.
- *Secuencial:* Es aquella descomposición en donde se hace énfasis en el orden, tanto de los elementos, como de su reorganización (Rich, Egan y Ellsworth, 2019).

Como se puede observar a simple vista, algunos de estos tipos poseen semejanzas con las estrategias, esto se debe a que, estas últimas han sido diseñadas con base en las características de las primeras. Es decir, inicialmente fueron construidos los tipos de descomposición, para dar lugar a las estrategias. Aún con toda la información que se tiene hasta el momento, se sigue haciendo énfasis en que existe un dispositivo (caja negra), en donde las entradas, salidas e incluso algunas relaciones son conocidas, no obstante, la estructura interna se desconoce. Cabe señalar que, en términos teóricos, existen dos cajas negras por separado, por un lado, la llamada "*black box programming*", y por otro, la "*black box decomposition*", por lo tanto, como se ha mencionado a lo largo de todo el documento, el objetivo es analizarlas como una relación, de tal forma que, recíprocamente, una brinde respuestas para la otra (Rich, Egan y Ellsworth, 2019).

Los pasos genéricos de la descomposición

Existe otro argumento teórico que concibe a la descomposición como un proceso iterativo, que se caracteriza por una serie de pasos que, al menos al momento de descomponer, siguen un ciclo que no siempre es lineal. En este sentido, un aspecto importante tiene que ver con el número de repeticiones necesarias para cumplir con una descomposición eficiente, desde esta perspectiva, se tiene la premisa de que se repetirán cada uno de los pasos las veces que sean necesarias, hasta que el sujeto considere que posee la información completa y necesaria para descomponer y resolver el problema. Los pasos genéricos desde la perspectiva de Rich, Egan y Ellsworth (2019) son los siguientes:

- Identificación de un eje: Se trata de la elección o construcción de uno o más ejes que permitan descomponer el problema sin que sus partes pierdan el

sentido, el eje puede construirse a partir de la función, dependencia, secuencia, estructura, entre otros elementos.
- Evaluación proactiva: En este paso, se evalúa el potencial de cada uno de los ejes o componentes, con el propósito de comprender su viabilidad y significado.
- Aceptación o rechazo: Se trata de un filtro, es decir, depende de este paso si se continúa o no con el proceso, incluso si es necesario agregar un eje para llevarlo al paso anterior y evaluarlo, o en caso contrario, omitir un eje.
- Ejecución de la descomposición: Se ejecuta la descomposición, puede ser con base en un eje o en cada uno de los elementos del problema, depende la profundidad de descomposición a la que se quiera llegar.
- Evaluación retroactiva: Una vez que se tienen resultados totales o parciales, es posible llevar a cabo una nueva evaluación, con el propósito de definir si el problema fue resuelto o no (Rich, Egan y Ellsworth, 2019).

Una de las características principales, es que estos pasos pueden cambiar de orden, repetirse y omitirse. Asimismo, al no ser lineal, es un proceso que puede darse en cualquier momento y en paralelo. Depende, por ejemplo, de la estrategia a utilizar, del tipo de descomposición, incluso el lenguaje de programación que se utilice, dado que la lógica de éste influye en las decisiones que se tomen a la hora de resolver problemas. En el siguiente apartado, se presentan las discusiones en torno a las relaciones construidas entre los pasos, tipos y estrategias de descomposición.

Las relaciones de descomposición y sus implicaciones en el proceso de programación

Para comprender la relación que guarda la descomposición y la programación, es necesario seguir un hilo conductor que guíe el argumento, en este caso, la programación y características específicas de *Mblockly* sirven como escenario contextual, en donde se presentarán cada uno de los elementos clave del PD antes mencionados. Cabe resaltar que, para brindar solidez a cada uno de los apartados siguientes, se siguió con la misma lógica argumentativa de Toulmin, por lo tanto, las premisas son los tipos, estrategias y pasos de la descomposi-

ción, el respaldo los diversos autores que sustentan dichas premisas, el modulador fue concebido como un "supuesto" a *priori*, y por último, las garantías (puentes entre las premisas y la conclusión argumentativa) fueron construidas a partir de cada una de las relaciones que se exponen a continuación.

Las estrategias de descomposición y su implicación en los pasos genéricos

Anteriormente, se adelantó que los pasos genéricos de la descomposición no son lineales, que incluso pueden omitirse y repetirse. Existen dos ejemplos claros, en el primer caso, la estrategia comparativa requiere de mayores repeticiones evaluativas, particularmente en el paso 2 y 5. En un segundo caso, en la estrategia de abajo hacia arriba, la identificación del eje puede omitirse, dado que se hace énfasis en un componente ya conocido, de esta forma, la evaluación y aceptación son pasos que también se dan por hechos. Es en la ejecución de la descomposición y su propia evaluación cuando se identifica si el componente conocido mantenía o no relación con los otros.

En este último ejemplo, la estrategia de abajo hacia arriba desestabiliza la linealidad de los pasos genéricos, no solamente porque omite o salta algunos de ellos, sino porque al momento de descomponer pueden existir dificultades, que se resolverán con base en la toma de decisiones de los pasos a seguir, sin importar si están en orden. En este sentido, sucede de manera similar con las otras estrategias de descomposición, por lo tanto, son éstas las que marcan la pauta para designar el orden o desorden de los pasos genéricos. Su relación es dependiente, ya que, por un lado, se debe a que una no cambia sin la otra, por otro lado, las estrategias de descomposición difícilmente serían efectivas sin una serie de pasos.

Estos pasos genéricos que se han discutido, son solo el inicio de un hilo escondido dentro de la caja negra. Ya que, deben de existir pasos no tan genéricos, que nazcan probablemente de las características particulares de la estrategia de descomposición y del problema específico.

La descomposición en paralelo

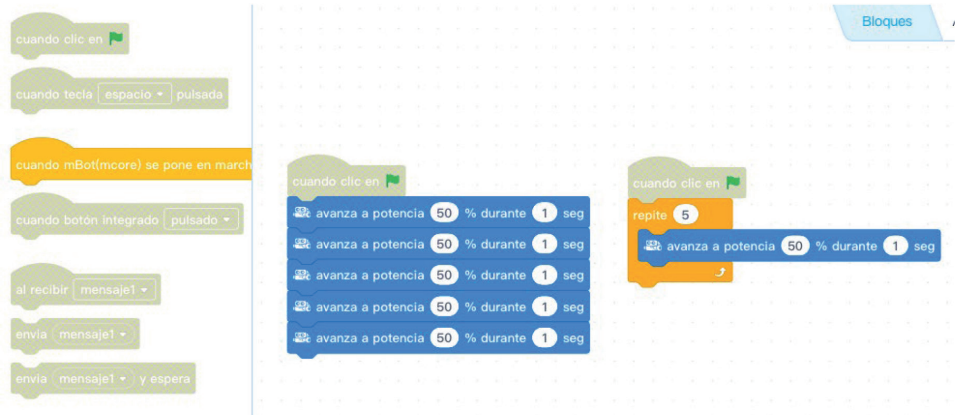
El *software Mblockly* permite llevar a cabo dos o más descomposiciones en una misma interfaz, sin embargo, la descomposición en paralelo hace referencia a los procesos estratégicos que ocurren de manera mezclada, más no en todos los casos simultánea. Tal es el caso de la estrategia multivariable, en donde se construyen los ejes con base en una categoría general, por ejemplo, con base en la descomposición de tipo funcional, es posible seleccionar los ejes con base en la categoría de bloques de control de *Mblockly*, sin embargo, pueden ocurrir dificultades al descomponer ciertos bloques, que dependen unos de otros. Una dificultad específica podría ser cuando el bloque de control de "repetir *n* veces" tiene efecto en varios bloques de acción, de tal forma que, si se descomponen, no solamente la categoría transversal principal carecería de sentido, sino que funcional y dependientemente causaría un caos.

Dicho caos ocurre en la desorganización de la descomposición, por lo tanto, es necesario utilizar otras estrategias para reacomodarlo, en el caso anterior, por ejemplo, puede resolverse con la estrategia comparativa, en donde se llevan a cabo varios prototipos de descomposición y evaluar cuál es el más eficaz. Por lo tanto, lo "paralelo" no tiene que ver con llevar dos procesos al mismo tiempo, sino a la capacidad de cambiar de un proceso a otro, similar a lo que ocurre con la cognición encadenada.

La descomposición funcional y sus implicaciones en la estrategia multivariable

En la descomposición en paralelo se tocó el tema de cómo la funcionalidad de ciertos bloques de programación puede repercutir en la estrategia multivariable, sin embargo, todavía existen asuntos por discutir. Partiendo de que la descomposición funcional depende de dos componentes clave, lo sustantivo y lo relacional, este último juega el papel de puente entre dos o más componentes. Los problemas ocurren cuando dicha relación involucra a varios bloques de programación que a su vez pone en aprietos a la categoría transversal multivariable.

Figura 5. Bloques inhabilitados.



Fuente: Captura de la interfaz de *Mblockly*.

A manera de ejemplo, si se utilizaran los bloques de eventos, particularmente el que indica cuándo iniciar el código ("cuando clic en"), automáticamente se inhabilitan otros bloques. Asimismo, cuando se tienen dos códigos distintos con un mismo propósito, el bloque de evento tiende a relacionarse y funcionar de una misma forma, en la figura 5, se muestra dicho ejemplo.

En este caso, no resulta factible seleccionar como categoría transversal a los bloques de eventos, dado que guardan una relación que no puede descomponerse desde la lógica de *Mblockly*, es decir, la función de inicio impacta en ambos códigos, por lo que una categoría transversal útil, podría ser los bloques de acción. En este sentido, la descomposición de tipo funcional adquiere un peso relevante en contextos de programación, más cuando se trata de estrategias que pretenden abarcar diversos ejes a la vez. Los estudios más enfocados al campo brindarían mayores insumos para determinar si la descomposición ocurre de manera completa o incompleta con este tipo de vicisitudes.

La relación entre la estrategia multinivel y la descomposición extremadamente fina

La descomposición extremadamente fina es una dificultad documentada por Meerbaum-Salant, Armoni y Ben-Ari (2011), tal y como se ha venido argumen-

tando en capítulos anteriores, se trata de un problema común. Sin embargo, el análisis teórico ha brindado elementos para pensar en una posible solución, la estrategia multinivel y su papel en encontrar los sub-componentes más pequeños de un problema, podría detonar tanto la solución como otras dificultades. Es necesario resaltar que, descomponer un problema en sus sub-componentes mínimos no garantiza que la descomposición se dé de manera eficaz. Por lo tanto, es necesario evaluar la pertinencia de cada uno de los componentes.

Para evaluar los componentes del proceso, es necesario hacer énfasis en el segundo paso genérico, sin embargo, particularmente en la estrategia multinivel, es clave que este paso se desprenda de la linealidad, para convertirse en un elemento dinámico, constante y útil, que permita tomar decisiones sobre cuándo se llega al límite de la descomposición. Es preferible entonces, descomponer en componentes significativos, que descomponer en elementos reducidos que carezcan de sentido, función y relación. Al ser un argumento construido a priori, es necesario observar lo que sucede en el campo, es decir, si teóricamente es factible que la estrategia multinivel poniendo en el centro la evaluación proactiva beneficie la descomposición en contextos computacionales.

La estrategia de final significativo y sus repercusiones en la descomposición secuencial

En ocasiones, cuando se tiene un dominio sobre la reorganización de los componentes del problema, es necesario recurrir al último eslabón, y así, sucesivamente, ir recorriendo el camino en un proceso reversible. Dichas características pertenecen a la estrategia de final significativo, sin embargo, la descomposición secuencial es crucial para que dicha estrategia sea eficaz.

En el escenario de *Mblockly*, la secuencialidad se da de manera descendente, por lo que facilita la reversibilidad del proceso, al tener cada bloque un diseño parecido al de un rompecabezas, donde indica si la pieza va arriba o abajo. Por ello, es posible concluir, al menos parcialmente, que el *software* de programación beneficia la descomposición secuencial, y ésta a su vez, a la estrategia de final significativo, habrá que analizar a fondo en el campo, si dicho supuesto mantiene las mismas condiciones.

La estrategia de abajo hacia arriba y sus posibles soluciones ante la descomposición incompleta

La descomposición incompleta ha sido un problema común, una de sus causas probables, puede ser el desconocimiento del problema general y sus componentes. Por lo tanto, una posible solución recae en la estrategia de abajo hacia arriba, es decir, al enfrentarse a un problema totalmente nuevo, es necesario relacionarlo con los conocimientos previos, particularmente, con un componente ya conocido.

Un caso que sirve para ejemplificar dicha estrategia y su relación con la descomposición incompleta, tiene que ver cuando se tiene un dominio de los bloques de eventos, y un desconocimiento de los otros. Es necesario entonces, aprovechar los bloques de eventos y descomponer el problema a partir de las relaciones que guarda con otro tipo de bloques. De esta forma, los componentes del problema cobran sentido, es decir, desde la comprensión de los bloques de eventos, es posible descifrar el funcionamiento de otro tipo de bloques.

Probablemente, no sea la única solución ante la descomposición incompleta, ya que podría ser necesario implementar otro tipo de estrategias, por ejemplo, la de final significativo. Cuando el sujeto mantiene confusiones sobre el orden de reorganización del problema, es necesario comenzar por el final, de tal forma que, pueda seguirse el orden de composición en reversa.

La descomposición nula y su relación con el ensayo y error

Otro de los problemas más comunes tiene que ver con la descomposición nula, es decir, cuando el sujeto no concibe a la descomposición como una opción para facilitar la resolución de un problema. Puede deberse a que nunca antes la haya implementado, o que su implementación parezca a simple vista, demasiado abstracta. En cualquiera de los dos casos, las complicaciones son las mismas, el problema general se vuelve más complicado de resolver, dado que es necesario atenderlo en un solo bloque.

Meerbaum-Salant, Armoni y Ben-Ari (2011) hacen énfasis en que particularmente con esta problemática, los estudiantes van haciendo pruebas de ensayo y error, de tal forma que, en algunos casos logran identificar las bon-

dades de la descomposición y sus ventajas a la hora de resolver problemas, no obstante, existe otro número de estudiantes que sigue omitiendo la descomposición.

Poco se ha hablado de la descomposición estructural, dado que posee una simpleza con respecto a los otros tipos de descomposición, no obstante, para la problemática antes mencionada, descomponer estructuralmente pudiera ser la puerta de salida. En otras palabras, el estudiante principiante difícilmente logrará entender la descomposición dependiente, sustantiva y relacional, si no es a través del ensayo y error de la estructural. Al darse cuenta que algunos componentes carecen de sentido, tendrá que hacer énfasis, en las relaciones entre los componentes, lo que significa un paso más adelante en el PD.

Cuando un estudiante principiante se enfrenta a un problema general, es imposible exigir una descomposición tan compleja como la comparativa o dependiente, por lo tanto, la descomposición más simple hasta el momento, es la estructural, ésta puede llevarlo hacia una descomposición extremadamente fina, pero serán esas complicaciones útiles para comenzar a implementar otro tipo de estrategias.

La automatización de procesos y su repercusión en el PD

Existen particularidades de *Mblockly* que tienen implicaciones en el PD, tal es el caso de la posibilidad de automatizar procesos. Se sabe que, la automatización forma parte de los procesos centrales del PC, sin embargo, hasta el momento no se ha ejemplificado dicha habilidad clave, lo que puede traer ciertas confusiones. Es posible utilizar el mismo ejemplo de la figura 5, en donde se observan 2 códigos visualmente distintos, pero totalmente equivalentes, del lado izquierdo se trata de un código largo, en donde se repite 5 veces la acción de avanzar durante 1 segundo. Por el contrario, en el lado derecha se observa un código corto, en donde se utiliza un bloque de control, para repetir 5 veces la acción de avanzar durante 1 segundo.

En el ejemplo anteriormente descrito, se puede observar una automatización del proceso, es decir, en vez de repetir manualmente la acción de avanzar, es posible utilizar un bloque de control para que lleve a cabo dicha repetición.

Como éste, existe diversos ejemplos y oportunidades para automatizar, el punto clave recae en identificar en cuáles situaciones es posible sacar ventaja.

La automatización evidentemente trae consigo ciertas bondades, por ejemplo, facilitar la codificación y ahorrar tiempo y esfuerzo en su diseño. No obstante, al momento de descomponer, puede ocasiones algunas dificultades, ya que los bloques de repetición se encuentran escondidos en el bloque de control, por lo tanto, un estudiante principiante, podría confundir su repetición con una simple acción. Asimismo, puede ocurrir el caso contrario, es decir, que el hecho de reducir el número de bloques utilizado, facilite la comprensión de los componentes del problema general. Ambos supuestos tendrán que comprobarse en el campo, principalmente para comprender la relación que guarda la automatización, la descomposición y el proceso de programación.

El uso de notas o comentarios para la descomposición

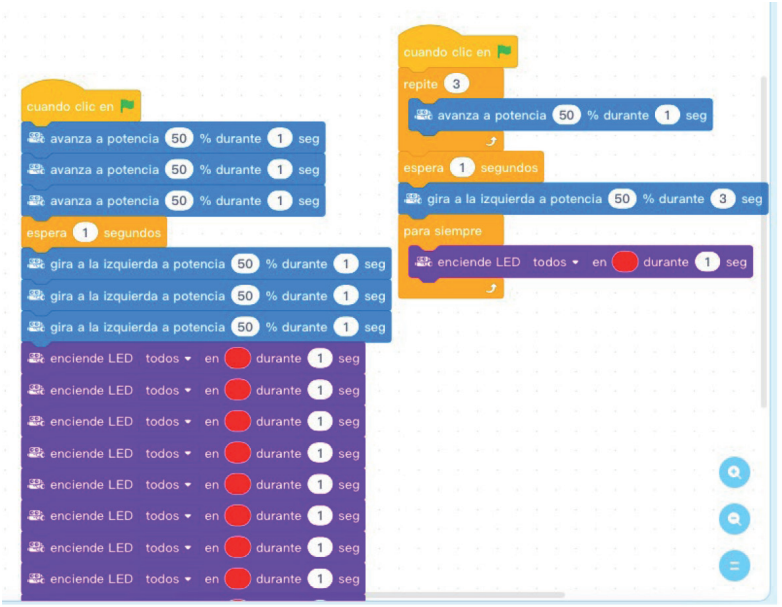
Anteriormente, en problemas que se resolvían con lápiz y papel, era necesario tomar notas sobre elementos clave del problema general. Hoy en día, es posible seguir con notas y comentarios, la diferencia radica en que pueden posicionarse exactamente en el bloque de programación en el que necesitamos poner atención. En este sentido, se trata de un supuesto que los estudiantes tomen nota a la hora de programar, por ejemplo, crear un comentario sobre un conjunto de bloques simplificados con la leyenda “este bloque sirve para repetir la acción de avanzar 5 veces”. De esta forma, podría utilizar la memoria de *Mblockly* para facilitar el proceso y poder concentrar los esfuerzos memorísticos en otros procesos.

Sin duda alguna, puede resultar ventajoso la utilización de comentarios, ya que, al momento de reorganizar los componentes, podrían incluso las notas brindar pistas sobre el orden, estructura, relación y dependencia. Asimismo, será vital asegurarse de que el estudiante conozca la interfaz de *Mblockly*, es decir, que el desconocimiento del uso de esta tecnología no sea una barrera para la implementación de notas y comentarios.

La simplificación de los bloques y su relación con la descomposición

Simplificar implica conocer a profundidad las funciones de los bloques utilizados, tal y como se muestra en la figura 6, en donde se observan dos códigos, en el lado izquierdo se utilizan menos bloques de control, y más de luz y acción. Por el contrario, en el código derecho, se utilizan bloques de control, dado que se hace énfasis en la repetición. Incluso, en el último bloque que implica la repetición infinita de encender la luz color rojo, resulta más sencillo utilizar el bloque de control "para siempre", en vez de copiar y pegar en diversas ocasiones el mismo bloque.

Figura 6. La simplificación del código.



Fuente: Captura de la interfaz de programación en *Mblockly*,

En este sentido, puede resultar más sencillo descomponer el código izquierdo, puesto que el esqueleto estructural del problema es totalmente visible, es decir, cada bloque está explícito en el código, por lo que puede observarse el

orden y relación de cada uno de ellos. Pasa lo contrario con el código derecho, en donde se encuentra implícitas las repeticiones, por lo tanto, al momento de descomponer es necesario mayor dominio del *software* de programación. La incógnita en este argumento es: ¿Qué es más preferible, la simplificación que facilita la programación o la explicitación que facilita la descomposición? Probablemente dependa del nivel de desarrollo que posea cada sujeto.

La incapacidad de descomponer bloques predeterminados

Existe otra problemática que surge solamente en el escenario de *Mblockly*, y tiene que ver con la incapacidad de descomponer bloques predeterminados. Existen algunos bloques que no pueden modificarse, por ejemplo, separar la potencia de los segundos en los bloques de acción, por lo que complica su descomposición y es necesario utilizar bloques de variables y de control. Otra opción sería crear un nuevo bloque, ya que *Mblockly* permite hacerlo, pero no de una manera intuitiva. Esta problemática podría o no presentarse en el campo, y en su momento habrá que analizar sus repercusiones en las estrategias de descomposición.

Recomendaciones metodológicas para el Proceso de Descomposición

En el campo del pensamiento computacional y la descomposición, la elección de una metodología adecuada resulta crucial para abordar las complejidades teóricas que surgen al explorar estos conceptos. Aunque existen diversas aproximaciones para su estudio, la teoría fundamentada emerge como una recomendación pertinente debido a su capacidad para generar teoría a partir de los datos, permitiendo así superar incongruencias clave en las bases teóricas existentes. Esta metodología inductiva ofrece una estructura flexible que facilita la creación de conceptos y categorías que reflejan con mayor fidelidad la naturaleza del fenómeno estudiado.

La teoría fundamentada permite a los investigadores acercarse a fenómenos complejos sin imponer estructuras teóricas preexistentes que podrían limitar el análisis o distorsionar los hallazgos. En el caso del pensamiento

computacional, y particularmente en la descomposición, se encuentran vacíos conceptuales que dificultan la comprensión profunda de sus mecanismos y aplicaciones. Al utilizar un enfoque fundamentado en los datos, es posible identificar patrones y relaciones intrínsecas que, de otra manera, podrían pasar desapercibidos. Este enfoque no solo enriquece la comprensión del pensamiento computacional, sino que también aporta una base sólida para futuras investigaciones.

Otro aspecto clave que refuerza la pertinencia de la teoría fundamentada es su capacidad para abordar la complejidad de la caja negra, una metáfora comúnmente utilizada en el análisis del pensamiento computacional. Las teorías tradicionales a menudo carecen de la flexibilidad necesaria para capturar las dinámicas específicas de este concepto. Por ello, la metodología propuesta no solo facilita la exploración de nuevos ángulos teóricos, sino que también fomenta un enfoque iterativo y reflexivo que asegura la validez y la profundidad del análisis.

Finalmente, la teoría fundamentada promueve una aproximación sistemática que integra herramientas analíticas como la codificación abierta, axial y selectiva, así como la escritura de memos, para construir teorías desde los datos. Estas técnicas son particularmente útiles en un contexto donde el pensamiento computacional y la descomposición requieren un tratamiento detallado y adaptativo. Además, al abordar el fenómeno desde una perspectiva inductiva, se abre la posibilidad de identificar contradicciones teóricas y avanzar hacia la saturación teórica, lo que fortalece la robustez y la aplicabilidad de los resultados obtenidos.

Como se mencionó en el capítulo anterior, al no existir teoría suficiente para explicar el fenómeno de la caja negra, es necesario utilizar una metodología que permita la creación de teoría básica. En este sentido, la teoría fundamentada, que tuvo su origen en Glaser y Strauss (1967), utilizando así, una serie de técnicas de codificación, comparación constante, construcción de muestreos teóricos, utilización de memos, la sensibilidad teórica, entre otras características que se exponen a continuación.

La teoría fundamentada se entiende como un proceso inductivo en donde las conclusiones, teorías sustantivas o supuestos, son fundamentados en los propios datos (Vivar, et al., 2010). Por lo tanto, aunque no se trata de un

proceso lineal, una de las primeras acciones es la conformación del muestreo teórico o selección de los participantes. Al ser inductivo por esencia, dicho muestreo se va construyendo a la par de la primera recolección y análisis de los datos.

Asimismo, el método de comparación de datos es vital para el proceso, ya que permite la identificación de regularidades o diferencias entre los códigos, categorías, eventos y supuestos. Para guiar las comparaciones, utilizar la codificación abierta, axial y selectiva:

- *Codificación abierta:* Permite al investigador nombrar y categorizar los sucesos que ocurren en la práctica.
- *Codificación axial:* Una vez que se llevó a cabo una codificación abierta, se utiliza a la axial para emitir las primeras comparaciones con viejas categorías. En este punto se toman decisiones sobre si los datos son suficientes para la conformación del muestreo teórico, o si es necesario recolectar más.
- *Codificación selectiva:* Tiene como propósito delimitar la nueva teoría, creando una teoría transversal que guíe el análisis de los siguientes datos, por lo tanto, la sensibilidad teórica es vital en este tipo de codificación (Vivar, et al., 2010).

Otro aspecto importante en la teoría fundamentada, es la utilización de memos. Éstos pueden utilizarse desde un *software* de análisis de datos cualitativos, y se caracterizan por guiar en términos metodológicos, al conformar el muestreo teórico; teóricos, al explicar supuestos y relaciones construidas; analíticos, que almacenan las reflexiones del propio investigador; y, por último, descriptivos, que brindan mayor información sobre algunos sucesos (Lora, Cavadias y Miranda, 2017). Por lo anterior, los memos brindan el sello teórico característico del investigador, que le permitirá afinar la sensibilidad teórica que propone Glaser y Strauss (1967).

Lo anteriormente expuesto corresponde a las características básicas de la teoría fundamentada, sin embargo, dicha metodología ha tomado diferentes vertientes, la presente obra se inclinará en la línea adoptada por Strauss y Corbin (2008) y Pandit (1996). Estas vertientes se han inclinado por la especialización de la teoría fundamentada en el uso de entrevistas semi-estructuradas y observación participante, las cuales serán clave para la recolección de datos

sobre el PD. Por otro lado, se apega al uso de *Atlas.ti*, ya que, al llevar a cabo la comparación constante de datos, categorías, entre otros bastos elementos, será necesario apoyarse en un paquete informático que facilite y valide el análisis.

Flick (2015) explica los modelos de Strauss y Corbin (2008) y Pandit (1996) como un proceso circular, en donde es necesario recolectar, interpretar y comparar datos constantemente, con el propósito de llegar a dos sitios, por un lado, una serie de contradicciones teóricas, y, por otro lado, a la saturación teórica. Además, su flexibilidad metodológica posibilita la generación de nuevos muestreos teóricos en función de las necesidades que surjan durante el análisis.

4. Los fundamentos, tipología y estrategias de descomposición: pautas teóricas para su comprensión

La descomposición se concibe como un proceso cognitivo complejo y dinámico que, si bien se sustenta en pilares fundamentales, se ve influido por elementos contextuales y computacionales que evolucionan continuamente. Este proceso implica identificar y analizar el objeto en términos de su orden, estructura, funcionalidades, causas, efectos y otros componentes de entrada y salida. Sin embargo, las relaciones entre estos elementos pueden ser claras o difusas, dependiendo de la interacción de factores internos que generan bifurcaciones en el desarrollo del proceso. La denominada "caja negra" del proceso de descomposición comienza a abrirse a través de ciertos principios teóricos, cuyo eje central es la profundización conceptual de la descomposición[20].

Cabe señalar que, en un primer análisis, la estructura argumentativa parecía dividida entre las lógicas de programación y la caja negra del PD. Sin embargo, un análisis más profundo reveló que las dinámicas particulares de los sujetos que interactúan con el proceso actúan como puente argumentativo entre los elementos clave. De este modo, el PD y las lógicas de programación no deben entenderse como entidades separadas en extremos opuestos, con los actores situados en un punto intermedio. Más bien, la dinámica de la descomposición trasciende esta visión, ya que los sujetos interiorizan ambos

20. A pesar de que el propósito de este libro es identificar los elementos del PD y sus relaciones, de tomó la decisión de implementar cinco categorías que engloban estos elementos y sus relaciones en esferas con lógicas dis tintas, aunque siempre interrelacionadas. Lo anterior con base en el sistema argumentativo de Toulmin, Strauss y Corbin, de tal forma que, la exposición de resultados tuviera mayor coherencia, y apuntara hacia una teoría incipiente con conceptos emergentes.

elementos, los resignifican y, como resultado, emergen características de no linealidad, dinamismo y especificidades únicas.

La tipología de la descomposición

La tipología de la descomposición surge como una herramienta conceptual que permite identificar y clasificar elementos clave del proceso. Esto posibilita la creación de categorías teóricas que agrupan diferentes formas de abordar la descomposición. Entre los tipos más destacados se encuentran la descomposición por funcionalidad, concreta, aritmética, desconectada e invisible. Esta última pone en evidencia que no todos los conceptos son completamente accesibles mediante técnicas analíticas, reconociendo que algunos elementos pueden permanecer difusos o implícitos, lo cual añade una capa de complejidad al proceso.

Los tipos de descomposición constituyen un marco conceptual para comprender no solo qué significa descomponer, sino también cómo se relacionan estos procesos con los escenarios de programación en los que tienen lugar. Este apartado presenta conceptos teóricos incipientes y prioriza explicaciones claras y concretas sobre los tipos de descomposición identificados, evitando descripciones exhaustivas que podrían desorientar al lector. Al final de cada tipo de descomposición, se incluye un organizador gráfico diseñado como un recurso visual para facilitar la comprensión de la esencia y estructura de los conceptos abordados.

Descomposición por funcionalidad

El PD está profundamente vinculado a las lógicas de programación, especialmente aquellas que desencadenan efectos interdependientes, como la relación de un bloque con otro. Para llevar a cabo una descomposición por funcionalidad, es fundamental considerar las características básicas de la estructura de herramientas como *Mblockly*. En este contexto, la funcionalidad se define como la capacidad de los objetos para interactuar en un esquema de causa-efecto; cuando este binomio se separa, pierde sentido y coherencia en relación con ciertos objetivos específicos.

Descomponer por funcionalidad implica, en primer lugar, identificar el objetivo de programación. Esto se traduce en reconocer si el propósito es programar un sensor o un actuador, estableciendo conexiones causa-efecto que, en el análisis teórico, se describen como "códigos puente". Estos códigos representan uniones entre bloques que adquieren significado dentro del contexto del problema. Este significado, sin embargo, es dinámico y depende directamente de las características de la situación problemática.

Un aspecto crucial de este tipo de descomposición es la resignificación de los objetos, un concepto emergente que destaca cómo los sujetos reinterpretan los componentes a lo largo de diversas interacciones. Este fenómeno se observa tanto en problemas computacionales como no computacionales, concretos y abstractos, y refleja una constante: la resignificación de los objetos, que evoluciona hacia la "resignificación de las funciones".

La funcionalidad entre bloques de programación varía según la interpretación del sujeto. Por ejemplo, la relación causa-efecto entre el bloque "avanzar" y un bloque condicional difiere en un contexto de programación para un sensor sigue líneas frente a un actuador de sonidos. Aunque estas relaciones puedan parecer idénticas a simple vista, los elementos de programación que las componen son distintos. Además, surgen conceptos paralelos como el "bloqueo de descomposición", que describe situaciones en las que separar elementos anula la funcionalidad entre ellos, afectando directamente la lógica y los objetivos del programa.

En este sentido, los sujetos que interactúan con procesos de programación no solo asumen un papel de observadores ante las contradicciones entre bloques, sino que toman decisiones estratégicas en torno a ellas. En un análisis inicial, se concluyó que la resignificación de las funciones surge como un concepto inherente al escenario de programación, en el que el entorno establece ciertos límites y reglas. Sin embargo, en fases posteriores de análisis, se identificó a los actores como protagonistas en la construcción y aplicación de este concepto.

La resignificación de las funciones se define como la capacidad de reinterpretar una misma relación causa-efecto entre bloques de programación, permitiendo descomponer sin comprometer la lógica subyacente de una estructura de códigos. Este concepto, no obstante, incluye elementos dinámicos

que dificultan la identificación de constantes claras. Un ejemplo de esto es la categoría denominada "función contradictoria", que se refiere a situaciones en las que una función entra en conflicto con otra, generando errores en la interface o impidiendo que el sistema ejecute una acción específica.

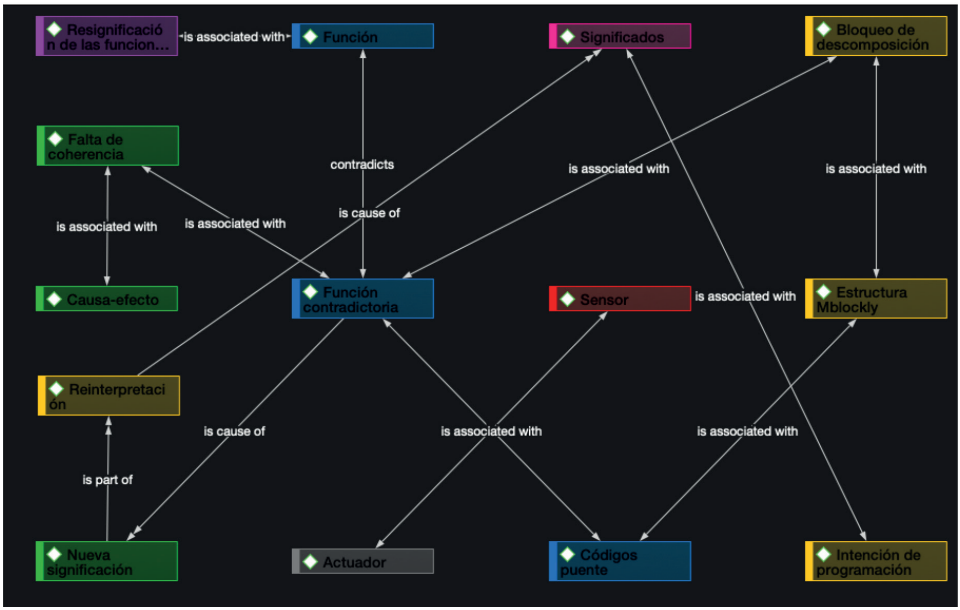
Las funciones contradictorias no son meros accidentes de cálculo, sino que pueden surgir como resultado de decisiones tomadas durante estrategias iniciales de improvisación. Estas estrategias reflejan cómo los agentes, al no reconocer plenamente ciertos elementos basados en sus conocimientos previos, deben improvisar relaciones entre bloques para luego aprender que algunas de estas relaciones resultan conflictivas.

La concentración de este tipo de descomposición en aspectos funcionales puede implicar ciertos riesgos. Existe la posibilidad de que se convierta en una limitante si no se contextualiza adecuadamente. Además, cabe destacar que este fenómeno no constituye una estrategia de descomposición en sí misma, ya que no se identificaron patrones consistentes de planificación relacionados con la resolución de problemas de programación. Más bien, la relación entre códigos y el análisis comparativo de diversas sesiones sugieren que se trata de un componente dentro de una estrategia más amplia. En el esquema "Red semántica", se presenta un organizador gráfico que incluye los códigos más relevantes, ofreciendo una explicación concreta del concepto de descomposición por funcionalidad.

Es importante destacar las diversas interpretaciones de los sujetos respecto a los desafíos encontrados durante la descomposición. Estos aspectos se reflejan en códigos analíticos como frustración, elección de no descomposición y descomposición fallida, que evidencian cómo el reconocimiento de errores dentro de la lógica de programación afecta la dinámica del proceso. Los sujetos tienden a enfocar su atención en las relaciones causa-efecto, ya que estas les permiten profundizar en la funcionalidad de los bloques. Sin embargo, ¿qué ocurre cuando estas relaciones son difusas?

El concepto de relaciones difusas en este análisis se refiere a aquellas conexiones que no presentan una relación evidente con otros elementos, ya sea dentro de la misma tipología o estrategia. En estos casos, los sujetos frecuentemente enfrentan lo que se denomina como "callejón sin salida", un código que se utiliza para describir momentos de incertidumbre. No obstante, estos

Red semántica: La descomposición por funcionalidad.



callejones sin salida se convierten en elementos clave para comprender la complejidad del PD, especialmente en contextos que involucran nuevas funcionalidades o problemas inéditos planteados por herramientas como *Mblockly*.

El término "callejón sin salida" evolucionó posteriormente hacia "relaciones difusas", un concepto que implica que la claridad o ambigüedad de una relación depende del conocimiento previo del sujeto sobre los bloques relacionados. Si uno de los bloques es familiar para el sujeto, la relación puede percibirse como clara. Por el contrario, cuando el sujeto desconoce la mayoría de los elementos involucrados, la relación se vuelve difusa. Este fenómeno destaca la importancia de los conocimientos previos en la construcción de significados dentro del proceso de descomposición.

Las relaciones difusas también pueden adoptar una forma inconsciente, en la que el sujeto percibe una relación clara, aunque los elementos secundarios sigan siendo ambiguos. Este fenómeno resalta el papel activo de los agentes en la definición de relaciones funcionales y difusas, ya que estas interpreta-

ciones varían según el lenguaje de programación y los conocimientos previos combinados.

Descomposición concreta y desconectada

Reducir una estructura en diversos segmentos puede ser un proceso complejo que, en muchas ocasiones, requiere apoyos concretos para facilitar su comprensión. La descomposición concreta se refiere a la separación de los componentes de una estructura de programación mediante el uso de elementos visuales que ayudan a identificar la posición, función y relación de cada bloque. Estos apoyos pueden incluir procedimientos escritos, pruebas realizadas en ventanas paralelas de la interface de *Mblockly*, o incluso objetos físicos y representaciones gráficas.

Este tipo de descomposición tiene su origen en procesos iniciales en los que los sujetos exploran las relaciones entre *software* y *hardware*, las lógicas de programación y los módulos de herramientas como *Mblockly*. La característica principal de la descomposición concreta es que surge al inicio de situaciones nuevas, ofreciendo un apoyo visual y estructural que facilita la comprensión. Sin embargo, con el tiempo, esta dependencia disminuye a medida que la abstracción reemplaza lo concreto, haciendo innecesario el uso de apoyos tangibles para entender los segmentos estructurales.

Uno de los conceptos clave que emergen en este contexto es el "dominio del objeto". Este concepto se define como la capacidad del sujeto para utilizar objetos previamente dominados, cuya experiencia acumulada permite descomponer o explorar nuevas situaciones. Por ejemplo, los agentes pueden recurrir a recursos como cuadernos para representar bloques de programación, calculadoras o incluso objetos físicos para simular el comportamiento del robot en diferentes escenarios. La descomposición concreta, por lo tanto, permite emplear recursos previos para identificar la posición, función y relación de los bloques de programación.

En este sentido, los sujetos aplican sus conocimientos previos para reforzar sus propios procesos cognitivos. Pero, ¿por qué ocurre esto y cuál es el papel del agente en la descomposición concreta? Este fenómeno surge como respuesta a la necesidad de comprender lógicas o problemas abstractos, llevando a los agentes a emplear diversas estrategias. Entre las más comunes se en-

cuentran la improvisación, la planificación y la no descomposición. La improvisación implica utilizar un objeto visual sin haberlo planeado previamente, ya sea físicamente o en un plano imaginativo. La planificación, por su parte, se centra en el uso deliberado de un objeto dominado para abordar un problema específico. Finalmente, la no descomposición se refiere a emplear objetos como una alternativa cuando la descomposición no resulta viable o efectiva.

Cualquiera que sea la vía tomada, el papel del sujeto en la descomposición es central, ya que asume el control y protagonismo de las acciones, independientemente de los errores que puedan ocurrir. Antes de incorporar objetos de dominio, algunos participantes describieron situaciones en las que el robot resultaba incontrolable, agobiante e impredecible. Sin embargo, el uso de estos objetos proporcionó un control hipotético, ya que, al estar fuera del contexto computacional, los sujetos demostraron mayor seguridad en sus estrategias y tipologías de descomposición.

La utilización de objetos dominados previamente, especialmente aquellos que no son computacionales, dio lugar al tipo de descomposición desconectada. Este tipo está estrechamente vinculado con la descomposición concreta, ya que ambas representan diferentes enfoques de un mismo fenómeno. La descomposición desconectada pone énfasis en los recursos no computacionales empleados durante el proceso de descomposición. Para profundizar en este aspecto, se planificaron sesiones específicas en las que no se utilizaban el *Mbot* ni *Mblockly*, reemplazándolos por objetos de dominio de los participantes, así como bloques tangibles.

Durante estas sesiones "desconectadas", los sujetos manifestaron sentirse más cómodos y seguros respecto a sus decisiones y procedimientos. Este fenómeno se identificó a través de códigos analíticos que reflejaban sus interpretaciones sobre el proceso de descomposición. Se observó que, mientras se empleaban los objetos de dominio en contextos no computacionales, la tendencia hacia la no descomposición era más frecuente. Cualquier problema, ya fuera de tipo aritmético, funcional, relacional o condicional, era abordado recurriendo a objetos de dominio mediante un enfoque basado en códigos completos, evitando en muchos casos una descomposición consciente. Este comportamiento resalta el papel del sujeto y sus significados dentro de contextos diversos.

Asimismo, el lenguaje y las reglas propias de *Mblockly* orillaron a los participantes a descomponer y recomponer. Sin embargo, al operar fuera de ese entorno computacional y proporcionarles mayor libertad, la descomposición pasó a un segundo plano. Esto abre un cuestionamiento sobre la utilidad del PD en estos casos, ya que incluso la no descomposición permitió resolver problemas computacionales con eficacia.

La descomposición no está necesariamente vinculada de manera exclusiva al uso del robot. El propósito de este análisis no fue establecer una dicotomía entre el uso del *Mbot* y otros enfoques, ni determinar cuál de ellos podría ser superior. Más bien, el objetivo fue identificar y rescatar las características individuales de cada enfoque, explorando si estos representan tipos distintos de descomposición.

Para apoyar esta exploración, se diseñó un cuadro que permite visualizar las particularidades de cada enfoque, destacando cómo se relacionan con las dinámicas propias del proceso de descomposición.

Elementos de programación	Descomposición concreta	Descomposición desconectada
Lógica	Se caracteriza por tomar objetos de dominio pasado, crear hipótesis sobre las acciones en el *Mbot*, y posteriormente testear los procedimientos. A partir de los propios resultados, se replantean los procedimientos, ya sea en los objetos de dominio o en escenarios computacionales.	Se caracteriza por tomar objetos de dominio pasado, crear hipótesis sobre las acciones en el *Mbot*, recrear el testeo en contextos no computacionales, y si es necesario, ni siquiera ponerlo a prueba en escenarios de *Mblockly*.
Aspectos computacionales	Se testea la hipótesis a prueba y error, sin embargo, el escenario principal son los objetos de dominio.	No se testean las hipótesis, si es necesario, se evita lo computacional. No obstante, la lógica de programación es la misma.
Composición	Una vez se segmenta la estructura de programación, se reincorporan las partes en *Mblockly*, con ayuda de los objetos de dominio.	La reincorporación se lleva a cabo en el mismo objeto de dominio, sin embargo, en ocasiones carece de sentido, debido a que no se realiza un testeo en *Mblockly*.

En ambos tipos de descomposición, el concepto de objeto de dominio resulta fundamental, ya que representa una herramienta mediante la cual los

sujetos pueden aplicar sus conocimientos previos adquiridos en otros contextos a situaciones nuevas. Este traslado de aprendizajes previos facilita la adaptación a diferentes desafíos durante el proceso de descomposición.

En el caso específico de *Mblockly*, la descomposición concreta mostró resultados más consistentes, mientras que la descomposición desconectada presentó inconsistencias en todas sus ejecuciones. Estas inconsistencias llevaron, en algunos casos, a que el proceso evolucionara hacia una forma más abstracta de descomposición, conocida como descomposición "invisible", que será abordada en el siguiente apartado.

La descomposición invisible

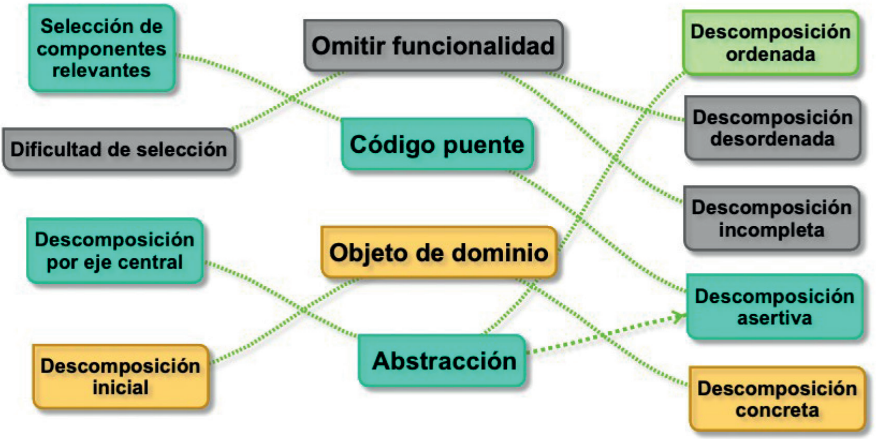
Este tipo de descomposición ocurre en escenarios abstractos, donde los procesos de segmentación y composición de estructuras no son inmediatamente visibles. Su análisis y comprensión dependen de entrevistas y reflexiones que permiten identificar las acciones cognitivas que subyacen al proceso. La denominación "descomposición invisible" no implica una imposibilidad de observarla, sino más bien una desmitificación, ya que estos procesos se hacen evidentes al señalar los elementos de entrada y salida involucrados.

El concepto central de este proceso es la "descomposición abstracta", que engloba todas aquellas acciones de separación, reordenamiento y análisis de segmentos dentro de una estructura de programación, realizadas sin el uso de objetos de dominio. Aunque se percibe como invisible, los elementos de entrada que permiten su desarrollo incluyen códigos como: elementos conocidos, dominio de la lógica de programación y caracterización del problema. Por su parte, los elementos de salida abarcan resultados como: descomposición asertiva, descomposición incompleta o descomposición desordenada.

Se presenta el esquema 5 que ilustra las relaciones entre estos elementos y cómo interactúan en el proceso de descomposición invisible.

La descomposición puede parecer difusa, pero no invisible. Como se observa en el esquema 5, los elementos conocidos, en su relación con la acción de "descartar objetos de dominio" dentro de un contexto nuevo, pueden conducir a una descomposición incompleta. Esto ocurre cuando la situación problemática no reúne las características necesarias para implementar este tipo de des-

Esquema 5. sobre elementos de entrada y salida del PD en su modalidad “invisible”.



composición, lo que sugiere que, en ciertos casos, podría ser más adecuado optar por una descomposición de tipo concreta.

La descomposición difusa fue un concepto recurrente identificado en el corpus analítico, donde muchas relaciones entre elementos de entrada y salida no se materializaron en objetos concretos dentro de los escenarios de programación. Este fenómeno subraya la importancia de analizar las decisiones, interpretaciones y significados asociados a las acciones de los sujetos.

En cuanto a las decisiones, la descomposición difusa no implica necesariamente confusión, desorden o falta de resolución. En ocasiones, los participantes omitieron pasos del proceso cuando ya dominaban ciertas tipologías de descomposición, con el propósito de agilizar el procedimiento. En lugar de analizar cada elemento desde el principio, avanzaban directamente al siguiente paso, estableciendo conexiones entre problemas previos y el actual, un comportamiento reflejado en códigos como “decisión de omisión” o “proceso acortado”.

Esto da pie a un análisis de las interpretaciones y significados relacionados con los elementos de descomposición. Este tipo de decisiones no representan un salto arbitrario de fases, sino una reinterpretación del proceso como idéntico, pero más eficiente. La percepción del problema como más sencillo

refleja una mayor claridad en los elementos de la estructura de programación, así como una intuitiva reorganización de los bloques, respaldada por códigos como "reacomodo intuitivo" o "estructuración simplificada".

¿Por qué ciertos procedimientos son percibidos como más sencillos? Los códigos asociados a procesos metacognitivos evidencian que, al enfrentarse a problemas similares, los sujetos establecen relaciones más directas entre tipos de descomposición y escenarios específicos. Por ejemplo, problemas que implican descomposición por funcionalidad y por oración mostraron características comunes, al igual que aquellos que integran la desconectada con el uso de objetos de dominio. Esta relación directa puede hacer que las resoluciones sean percibidas como difusas por observadores externos, mientras que para los sujetos resultan claras y precisas.

La descomposición difusa representa una etapa intermedia entre procedimientos iniciales y aquellos de mayor dominio. Para los sujetos, este tipo de descomposición implica conexiones implícitas que reflejan un avance en su comprensión y manejo de los procesos. Este fenómeno, inicialmente denominado "invisible" y posteriormente "difusa", podría describirse mejor como "descomposición implícita", destacando su naturaleza como una evolución del aprendizaje. Este proceso, reflejado en códigos como "dominio adquirido" y "fluidez adaptativa", permite aparentar un salto en el procedimiento, ahorrando tiempo y enfocando el esfuerzo en elementos nuevos. Incluso, algunos sujetos decidieron no descomponer estructuras que ya comprendían profundamente, considerando innecesario descomponer algo que ya conocían, según lo reflejado en el código "decisión de no descomposición".

Las estrategias de descomposición

Las estrategias de descomposición incluyen enfoques como el reemplazo, los ordenamientos planificados y las improvisaciones desordenadas, que son implementadas para resolver problemas de programación específicos. Estas estrategias no solo destacan por su diversidad, sino también por su capacidad para adaptarse a contextos y necesidades particulares. Aunque no documentadas exhaustivamente en la literatura, se presentan como ejemplos claros de la naturaleza estratégica, dinámica y compleja del proceso de descomposición.

Esta complejidad se refleja en las decisiones que los sujetos toman en función de sus contextos socio-culturales. Estas decisiones influyen en cómo construyen, modifican, reemplazan y planifican estrategias específicas. Además, la reinterpretación de los resultados de la descomposición permite que los agentes ajusten y reformulen sus propios mecanismos de resolución. Es decir, el sujeto que programa no solo descompone, sino que también decide cuándo y cómo hacerlo, asumiendo un papel activo y reflexivo en el proceso.

En el contexto del PD, una estrategia se refiere a un conjunto de acciones planificadas destinadas a alcanzar un objetivo en el que se incluye algún tipo de descomposición o, al menos, indicios de que la descomposición estuvo presente, ya sea de manera completa o incompleta. Las estrategias identificadas incluyen descomposición por reemplazo, ordenada, desordenada e improvisada. Esta última continúa en debate sobre si se trata de una estrategia en sí misma o una extensión de la tipología.

Descomposición por reemplazo

En las estructuras de programación, existen elementos que pueden ser reemplazados. Por ejemplo, un bloque que ordena avanzar a una velocidad del 30% durante 3 segundos puede ser reemplazado por una combinación de bloques que incluyen avanzar, condicionar la acción por tiempo, y ajustar la potencia a un porcentaje idéntico. Mientras que el primer caso representa un procedimiento simplificado, el segundo es más complejo pero necesario en escenarios donde se requiere ajustar la potencia en un momento específico.

El concepto de reemplazo surge cuando el sujeto identifica que un bloque no puede ser descompuesto de forma directa. En estos casos, se sustituye por un conjunto de bloques que realizan la misma acción, pero permiten descomponerla en segmentos más pequeños. Este fenómeno está estrechamente relacionado con el concepto teórico de "descomposición en segundo nivel", que abarca las acciones realizadas cuando una primera descomposición resulta insuficiente, y se requiere segmentar aún más para resolver un problema.

La descomposición en segundo nivel no siempre garantiza una solución efectiva. En algunos casos, puede añadir complejidad al problema. Esta estrategia suele ser utilizada cuando hay dificultades para comprender completamente el problema. Sin embargo, ¿la descomposición siempre facilita la

comprensión de una estructura? La respuesta es que no en todos los casos ni para todos los problemas. Más adelante, se analizarán los escenarios en los que esta estrategia no fue factible, pero antes es fundamental comprender sus características principales.

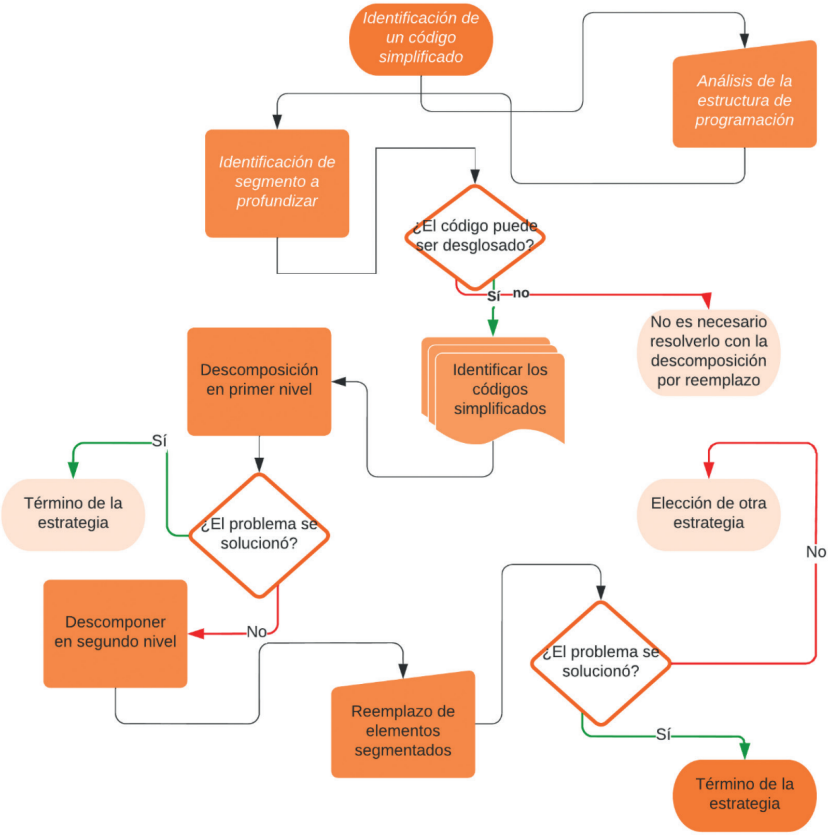
La descomposición en segundo nivel permite identificar la complejidad mínima de una estructura. Este concepto surgió durante escenarios en los que los sujetos enfrentaban problemas que requerían programar acciones precisas, como superar obstáculos milimétricos. La descomposición por reemplazo no pertenece a la tipología, ya que se identificaron patrones específicos que se repetían consistentemente en diversas sesiones de trabajo. Entre las características básicas de esta estrategia se encuentran:

- La descomposición por reemplazo es útil para resolver problemas en espacios reducidos.
- En su fase inicial, implica una descomposición de primer nivel, separando elementos básicos sin alterar sus funciones.
- En una fase intermedia, se identifican segmentos que requieren una visión funcional ampliada.
- En la fase final, se realiza una descomposición en segundo nivel, donde los segmentos seleccionados se reemplazan por otros más detallados. Este proceso incluye la reincorporación meticulosa de elementos, conectándolos uno por uno y respetando la tipología de funcionalidad, sin comprometer la estructura del código.

Sin embargo, para que esta repetición de pasos sea considerada una estrategia, se requiere que cumpla con ciertos criterios: patrones consistentes a través de diversas situaciones, una serie de pasos claramente identificables, y un objetivo intencionado por parte del sujeto. En la imagen del Diagrama de flujo se ilustran los elementos básicos de la estrategia por reemplazo.

Para comprender las relaciones entre los elementos de esta estrategia, es necesario remontarse a su fase inicial, donde el código debe ser interpretado para clasificarlo como simplificado o en extenso. En este sentido, se identificaron las siguientes características clave: en primer lugar, la estructura debía ser clara y contener segmentos identificables; en segundo lugar, al menos uno de los segmentos debía ofrecer la posibilidad de desglosarse; y, por último, la estructura y segmentación debían ser conocidas por el sujeto. Este enfoque

Diagrama de flujo: La descomposición por reemplazo.



demuestra que la estrategia de reemplazo depende directamente de la experiencia previa con problemas similares, ya que su aplicación en estructuras poco exploradas puede romper la lógica de orden, funcionalidad y relación entre bloques.

Se observa que esta estrategia surge tras la repetición continua de enfoques más básicos, como prueba y error, y otras estrategias incipientes, como la descomposición ordenada o desordenada. Durante el proceso, los sujetos no solo observaron el flujo de acciones dentro de los bloques, sino que monitorearon cuidadosamente cada segmento para mantener la relación entre ellos.

Además, asumieron el papel de evaluadores del proceso, tomando decisiones sobre si avanzar hacia una descomposición de segundo nivel o permanecer en el primer nivel, dependiendo de las características del problema. Esto evidencia que cada evaluación fue única según el contexto.

Uno de los aspectos más relevantes es la transición entre la descomposición de primer y segundo nivel, que implica evaluar los resultados iniciales y decidir si es necesario realizar sub-segmentaciones. Este proceso no implica descomponer toda la estructura, sino identificar los grupos de bloques que reflejan errores y segmentarlos específicamente. Sin embargo, esta transición presenta complicaciones que, según los códigos analizados y las co-ocurrencias observadas, requieren una mayor profundización para entender completamente su dinámica.

Los sujetos también desempeñaron un papel de depuración durante este proceso, reacomodando estructuras y asegurando que estas fueran funcionales. En la primera fase de la descomposición, se asumió que este nivel inicial sería suficiente en cualquier caso. No obstante, los códigos relacionados con insuficiencia se fortalecieron a medida que los problemas se volvieron más complejos con el avance en los módulos de *Mblockly*, lo que hizo necesario realizar descomposiciones adicionales. Pero, ¿por qué la descomposición se convirtió en una solución viable? Las razones principales son las siguientes:

- La descomposición en segundo nivel se estableció como una alternativa lógica después de un proceso de evaluación.
- Durante la depuración, se identificaron bloques desconectados o incoherentes que requerían descomposición y reemplazo con bloques adicionales.
- La descomposición ofreció una herramienta eficaz para comprender mejor los problemas y resolverlos de manera más estructurada.

Descomposición ordenada

Existen diversas maneras de implementar una descomposición ordenada, como hacerlo a partir de un eje central, siguiendo una lógica vertical o mediante un ordenamiento jerárquico. El concepto de descomposición ordenada en su modalidad de eje central se refiere a la separación de los elementos de una estructura con base en un propósito inicial (eje). Por ejemplo, si el obje-

tivo es que el *Mbot* encienda luces de un color específico y se toma como eje central a los códigos de control de tiempo, entonces el reordenamiento de los componentes se basa en este conjunto de bloques. A medida que otros bloques se reincorporan a la estructura, se alinean con el eje principal.

La descomposición ordenada se caracteriza como una estrategia en la que los sujetos seleccionan un elemento central, ya sea de la lógica de programación o de la propia estructura de descomposición, y lo utilizan como base para aplicar una serie de pasos. Esto permite mantener la coherencia inicial de la estructura antes de ser segmentada. Sin embargo, las complicaciones surgen cuando el elemento seleccionado no es lo suficientemente robusto para desempeñar el papel de eje central, lo que puede generar errores y complicaciones en el código.

Un elemento que ayuda a mantener la coherencia durante el proceso es la lógica vertical, una variante de la descomposición ordenada que utiliza el ordenamiento de bloques en sentido vertical como eje principal. En este enfoque, los bloques se separan en conjuntos y se organizan de forma vertical, incluso si no están conectados directamente, con el propósito de preservar un orden similar al inicial. Aunque también se observaron variantes horizontales, estas solo se presentaron en casos en los que *Mblockly* las sugirió como opción, y su uso fue limitado.

La descomposición ordenada se identificó principalmente en las últimas sesiones de trabajo, donde existía un mayor dominio de ciertos procedimientos de programación. Sus códigos base incluyen "descomposición lineal", "descomposición repetitiva", "descomposición visible" y "descomposición selectiva". Sin embargo, cuando la planificación no es suficientemente detallada para prever los obstáculos, se corre el riesgo de enfrentar un fenómeno denominado "Colapso del ordenamiento".

El colapso del ordenamiento ocurre cuando, después de realizar una descomposición cuidadosamente ordenada, un elemento externo altera el orden, provocando una pérdida de coherencia que requiere implementar el concepto de "reversibilidad". La reversibilidad puede ser "asertiva" o "fallida" dependiendo de si el sujeto logra reordenar los bloques de manera efectiva o no.

Un ejemplo de este fenómeno surgió durante una sesión en la que un agente externo interrumpió el proceso, desviando la atención del sujeto y haciendo

imposible continuar con la observación del *Mbot*. En este caso, el eje central de la estrategia era observar las acciones del *Mbot* para replantear la composición de los bloques, pero la interrupción impidió aplicar el concepto de reversibilidad, lo que derivó en un colapso del ordenamiento en los módulos siguientes. Este colapso, aunque inevitable, dio lugar al desarrollo de un nuevo concepto que sugiere una estrategia alternativa para casos donde la descomposición ordenada no puede mantenerse.

Descomposición desordenada

Esta estrategia está directamente relacionada con la descomposición ordenada, ya que se recurre a ella cuando la primera falla. Los elementos clave que caracterizan esta estrategia incluyen:

- Observar el código por primera vez, incluso si se encuentra desordenado.
- Identificar elementos similares a situaciones pasadas, con el objetivo de solventar errores utilizando procedimientos previamente aplicados.
- Aplicar el ensayo y error como un procedimiento esencial.
- Implementar la descomposición repetitiva como base para reordenar el código.

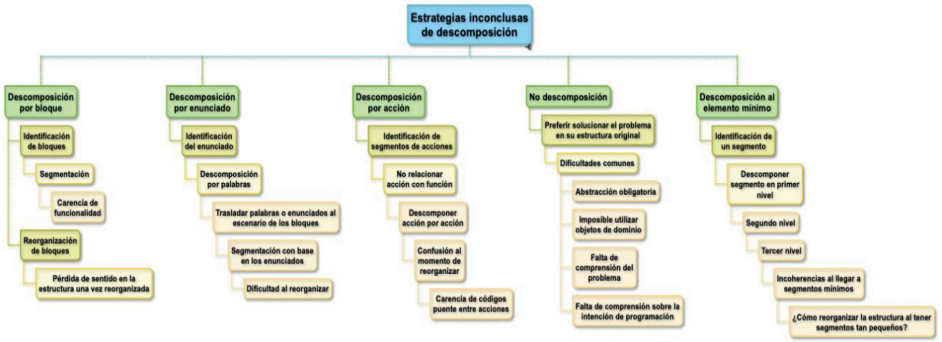
A diferencia de la descomposición ordenada, esta estrategia se caracteriza por su no linealidad; los pasos pueden presentarse en un orden cronológico variable según la situación. Este tipo de descomposición se utiliza como último recurso cuando los problemas son especialmente difíciles de resolver. En algunos casos, si la descomposición desordenada tampoco resulta efectiva, se abandona temporalmente la actividad debido a la frustración o la falta de una solución inmediata.

Además, esta estrategia también se observó en las primeras sesiones de trabajo, junto con la descomposición concreta. Cuando esta última fallaba, los sujetos recurrían al procedimiento de ensayo y error como alternativa. En ciertos momentos, se identificaron códigos como “duda”, “pregunta” y “frustración”, reflejando la necesidad de buscar apoyo externo para continuar con la tarea. Por ejemplo, algunos participantes verbalizaron comentarios como “necesito preguntar porque ya no entiendo esto” (Lozano, entrevista 2, 2022). Sin embargo, se destacó un elemento común que ayudó a resolver estas problemáticas.

El concepto de "repetición de estrategias" fue fundamental para abordar los desafíos asociados con la descomposición desordenada. No obstante, a pesar de haberse identificado en diversas situaciones, no se logró establecer un patrón consistente de comportamiento para esta estrategia. Esto sugiere que podría formar parte de la tipología de la composición en lugar de ser una estrategia autónoma, dado que se caracteriza principalmente por la improvisación, lo cual contrasta con la definición de estrategia planteada al inicio de este apartado.

Las estrategias de descomposición son aspectos especialmente complejos de analizar, ya que tienden a trasladarse con facilidad hacia la tipología. Sin embargo, solo dos estrategias lograron cumplir con las características básicas necesarias para ser identificadas como tales. En el esquema *Estrategias de descomposición*, se presenta un organizador gráfico que incluye otras estrategias tentativas, las cuales, debido a la ausencia de ciertos elementos, fueron clasificadas únicamente como elementos de descomposición.

Estrategias de descomposición.



Elaboración propia.

La evolución de la descomposición: un desarrollo gradual y estratégico

Al considerar que la descomposición es un proceso dinámico y estratégico, emergen elementos que, al trasladarse a escenarios de análisis cronológico, revelan la existencia de un desarrollo gradual. Por lo tanto, la descomposi-

ción se caracteriza por incluir fases iniciales, repetitivas, concretas, visibles, selectivas y abstractas. Estas fases destacan por su no linealidad, la presencia de retrocesos y la incorporación de la reversibilidad intencionada como parte integral del proceso.

La descomposición evolutiva refleja el crecimiento progresivo de los sujetos en su dominio del proceso. Las fases iniciales funcionan como elementos exploratorios que permiten no solo abordar la resolución de problemas, sino también comprender las mecánicas y lógicas subyacentes al contexto. A medida que los agentes repiten estrategias y tipologías, desarrollan una mayor comprensión que les facilita reflexiones metacognitivas. Estas reflexiones consolidan la no linealidad como una elección consciente y la reversibilidad intencionada como un enfoque estratégico clave.

En el análisis realizado, se efectuó un seguimiento cronológico del desarrollo gradual del PD con el objetivo de identificar elementos constantes (fases) y aquellos que son dinámicos y variados. Las fases identificadas incluyen estrategias y tipos de descomposición que no se han presentado en apartados anteriores. Esta decisión busca evitar la repetición de ideas y mantener como elemento sorpresa uno de los hallazgos más significativos de este análisis.

Fase 1. La descomposición inicial

El inicio de la descomposición se caracteriza por la improvisación y la exploración. En esta etapa, al no contar con estrategias dominadas, se recurre a la técnica de ensayo y error. En un primer momento, los códigos son ejecutados sin depuración, lo que genera errores y resultados imprecisos, como que el robot no cumpla con las acciones necesarias para resolver los problemas planteados.

En el contexto del pseudo-lenguaje de programación de *Mblockly*, se identificó un concepto denominado "lógica de formas", que se refiere a la posibilidad de descomponer a partir de las formas físicas de los bloques de programación. Por otro lado, el *software* proporciona un "apoyo de *software*", ofreciendo sugerencias como actualizar el código desde cero o recordatorios sobre el uso de bloques condicionales. Ambos conceptos están relacionados con los objetos de dominio, ya que representan apoyos visuales que permiten a los sujetos implementar una descomposición concreta de manera improvisada.

Un cuestionamiento importante de esta fase es si los participantes eligieron esta ruta cognitiva por iniciativa propia o si fue el *software*, a través de sus apoyos visuales, quien los dirigió hacia este camino. Para investigar esto, se realizó un ejercicio desconectado en una sesión de trabajo, omitiendo los apoyos visuales para observar el comportamiento de los sujetos frente a problemas complejos y nuevos. El resultado mostró que siguieron rutas similares, argumentando en entrevistas que esta decisión se debía a experiencias pasadas. Esto sugiere que podría tratarse de una ruta cognitiva influenciada por aprendizajes previos, lo que amerita un análisis más profundo en contextos desconectados desde el principio.

Otro hallazgo relevante en esta fase es la identificación de conceptos como la "descomposición incompleta", que se caracteriza por reorganizaciones estructurales carentes de elementos necesarios. Esta fase es incipiente y regularmente produce elementos de salida negativos para la resolución de problemas, debido a la ausencia de estrategias iniciales. Sin embargo, estas series de errores son las que dan paso a la segunda fase: la descomposición repetitiva.

Fase 2. La descomposición repetitiva

Después del caos característico de la descomposición inicial, es necesario implementar procedimientos que permitan organizar y dar sentido al proceso. En esta segunda fase, los sujetos recurren a un concepto identificado como "repetición de elementos". Este concepto implica la aplicación de un tipo de descomposición utilizada previamente como base para diseñar estrategias más complejas que las iniciales.

La fase 2 es el momento en que las estrategias comienzan a tomar forma. Sin embargo, su diseño no garantiza necesariamente una correcta aplicación. Un patrón común en esta etapa es la aplicación de elementos de entrada previamente utilizados, relacionándolos de la misma manera con la expectativa de que produzcan resultados idénticos. Sin embargo, al enfrentarse a problemas diferentes o bloques de programación nuevos, las relaciones pueden bifurcarse, generando elementos de salida inesperados que, en algunos casos, no contribuyen a resolver la estructura del problema.

El concepto de descomposición por ensayo y error sigue siendo predominante en esta fase. Los elementos de entrada y sus relaciones se replican sin un análisis detallado de cada situación, lo que en ciertos momentos deriva en una improvisación más que en una planificación consciente. Por esta razón, este fenómeno se interpreta más como una tipología que como una estrategia. Sin embargo, se observa una transición clave cuando los sujetos identifican que la repetición no es suficiente, optando por elementos de entrada relacionados con situaciones similares del pasado. Es en este punto cuando la descomposición comienza a volverse selectiva, pero surge la pregunta: ¿cómo se seleccionan los elementos de entrada?

Fase 3. La descomposición selectiva

Cuando la descomposición repetitiva no funciona en todas las situaciones, surge la necesidad de una selección más deliberada de los elementos de entrada. Estos elementos pueden incluir: situaciones problemáticas, bloques de programación, segmentaciones por función, puentes entre bloques, lógica horizontal y vertical, objetos de dominio, entre otros. Cada uno de estos elementos contiene múltiples códigos que, al relacionarse, aportan solidez teórica al proceso.

En esta fase se identificaron estrategias como la descomposición ordenada, por reemplazo y desconectada, entre otras. Aunque estas estrategias ya han sido explicadas, el patrón común que las une incluye los siguientes pasos:

- Comenzar con la selección de un elemento de entrada que se relacione con al menos un factor puente dentro del escenario de programación.
- El patrón puente representa una conexión entre una experiencia pasada y una situación nueva dentro de la programación. Es el componente clave para diseñar una estrategia.
- Una vez identificados el elemento de entrada y el factor puente, se triangula con el elemento de salida, que puede ser validado mediante ensayo y error o predecido a través del seguimiento lógico del proceso.
- Una de las ventajas del uso de *Mblockly* es la continuidad temática dentro de sus módulos, que agrupan problemas similares. Por ejemplo, el módulo 7 se centra en actuadores de sonido, lo que implica trabajar con bloques específicos relacionados con este tipo de programación. Si un sujeto resolvió

problemas en el módulo 3 y posteriormente enfrentó desafíos similares en el módulo 7 con mayor dificultad, podía utilizar el bloque de sonido como elemento de entrada, las relaciones con otros bloques como factores puente y la descomposición completa o incompleta como elementos de salida. Este proceso ilustra cómo funciona la descomposición selectiva.

- Finalmente, cuando ya no era necesario observar la respuesta del robot para predecir las acciones del programa, surgió lo que se denomina descomposición abstracta, marcando una transición hacia un nivel superior de comprensión y dominio del proceso.

Fase 4. La descomposición abstracta

La abstracción, en el contexto del presente análisis, se define como el punto en el que el PD deja de depender de objetos de dominio, permitiendo una selección deliberada de elementos de entrada, salida y factores puente. Esta fase requiere una capacidad de predicción en la relación causa-efecto entre el sujeto y el *Mbot*, lo que la convierte en el nivel más avanzado de la descomposición. Involucra un ejercicio metacognitivo complejo que solo es posible después de dominar y aplicar estrategias en contextos variados.

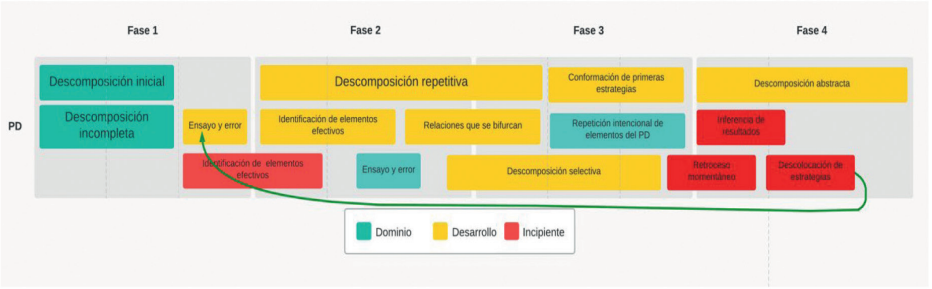
Los elementos dinámicos, identificados como parte de esta fase, incluyen conceptos como la no linealidad, la temporalidad, el retroceso momentáneo y la depuración abstracta. La no linealidad refleja la capacidad de no seguir un orden rígido en las fases del proceso, permitiendo al sujeto combinar estrategias como la descomposición selectiva y la inicial (ensayo y error), dependiendo de las necesidades específicas del problema. Este enfoque fortalece el procedimiento y aumenta las probabilidades de éxito.

La "descolocación de estrategias" es otro concepto fundamental de esta fase. Consiste en la combinación, modificación o reemplazo de estrategias conocidas, una capacidad que surge del dominio de la tipología de la descomposición. Por otro lado, el concepto de "temporalidad" aborda la durabilidad de las estrategias y cómo estas pueden modificarse, reemplazarse o perderse con el tiempo, lo que puede generar retrocesos en el proceso.

La descomposición abstracta se caracteriza por permitir al sujeto imprimir un sello personal en las estrategias que utiliza. Esto implica una flexibilidad para adaptar, combinar o reemplazar estrategias según el contexto, aunque

estas modificaciones no suelen seguir patrones específicos. Por lo tanto, se supone que al alcanzar esta etapa, cada sujeto desarrolla una ruta individual y única que refleja su experiencia y comprensión del proceso.

Esquema 6. La descomposición evolutiva.



Descomposición difusa

Esta categoría teórica agrupa elementos de la descomposición que no presentan relaciones claras o definidas con otras categorías. Por un lado, estos elementos no son lo suficientemente específicos como para integrarse en los hilos conductores previamente definidos, pero, por otro lado, poseen un potencial significativo para abrir nuevos caminos teóricos hacia áreas todavía no exploradas. En este sentido, se postula la posibilidad de una fase intermedia en el desarrollo gradual de la descomposición, que se sitúa en una dicotomía entre la linealidad y la no linealidad. Esta fase contribuye a comprender fenómenos como la predicción de comportamientos en el PD, la improvisación de estrategias intencionadas y las descomposiciones completamente desconectadas de elementos computacionales.

Además, se reconoce que algunos aspectos de la descomposición pueden parecer difusos desde la perspectiva de un observador externo. Sin embargo, los sujetos involucrados en el proceso a menudo identifican patrones o toman decisiones conscientes que les permiten interpretar y manejar dichas relaciones. Esto evidencia cómo la experiencia previa y los conocimientos del agente influyen directamente en su capacidad para percibir la claridad o difusidad de estos elementos.

El modalizador argumentativo: excepciones y características del caso

En la estructura argumentativa propuesta, se identifican elementos que limitan su generalización, ya que los conceptos teóricos presentados no necesariamente aplican de manera uniforme en todos los contextos. Esto se debe a que la teoría, al encontrarse en una etapa incipiente, se caracteriza más por su flexibilidad que por su universalidad. Por lo tanto, esta categoría expone cómo las características de los agentes involucrados, tales como su experiencia previa con herramientas tecnológicas, su interacción con *software* especializado y su relación con procesos cognitivos específicos, pueden influir en la manera en que aplican los principios de descomposición.

El marco de la descomposición

La descomposición es un concepto complejo que, en esta obra, se define como un proceso fundamental en la segmentación de estructuras, ya sean computacionales o no. Este proceso requiere la identificación y separación de los elementos constituyentes, con el propósito de comprenderlos y abordarlos de manera efectiva dentro de la totalidad de un sistema específico. La característica principal de la descomposición radica en su necesidad intrínseca de segmentar, aunque sus aplicaciones y estrategias varían ampliamente según el contexto y las condiciones específicas en que se emplea.

Aunque se reconoce que la descomposición puede darse en contextos no computacionales, en el marco de esta teoría incipiente, el análisis se delimita a escenarios computacionales, particularmente en el pseudo-lenguaje de programación *Mblockly*. La descomposición de estructuras implica desglosar sistemas complejos en unidades más pequeñas, lo que facilita abordar problemas de programación en fases manejables y comprender sistemas densos.

En contextos computacionales como el de *Mblockly*, el proceso de descomposición toma características específicas, entre las que se incluyen:

- Identificar el problema.
- Analizar la estructura y lógica en relación con el lenguaje de programación utilizado.

- Dividir el problema en módulos o segmentos más pequeños.
- Identificar las entradas y salidas del robot o *software* de programación.
- Definir las funciones de cada componente del problema.
- Diseñar, reformular o analizar algoritmos que permitan hacer funcionar cada componente.
- Desarrollar pseudo-códigos o diagramas de flujo que representen el proceso de descomposición.
- Implementar testeos y depuración de cada segmento.
- Reintegrar los componentes garantizando su interacción efectiva.
- Optimizar el proceso mediante la evaluación de interacciones o componentes.
- Manejar errores y excepciones.

Este listado resume las características generales del proceso de descomposición. Es importante señalar que una segmentación aislada dentro de un código no constituye una descomposición completa, sino solo indicios de descomposición en un contexto dado. Además, la composición es parte integral del proceso, ya que la utilidad de la segmentación se pierde si la estructura no puede reorganizarse eficazmente.

El marco teórico presentado aquí está específicamente delimitado al pseudo-lenguaje de programación *Mblockly*, lo que implica que las explicaciones teóricas y los procedimientos ejemplificados están influidos por este entorno. Aunque algunos conceptos pueden ser extrapolados a otros lenguajes de programación como *Java* o *Scratch*, su aplicabilidad podría variar significativamente. Esto subraya la necesidad de futuras investigaciones para profundizar en este proceso cognitivo y poner a prueba los procedimientos descritos en otros contextos.

Finalmente, al no tratarse de una teoría universal, el marco teórico también está influido por las condiciones específicas del escenario estudiado. Este enfoque busca sentar las bases para la comprensión de la descomposición como proceso cognitivo, reconociendo sus limitaciones y abriendo camino para futuros desarrollos teóricos.

Teoría sobre la descomposición: las fases que la conforman

A pesar de que el propósito de la presente obra es dar cuenta de los tipos y estrategias de descomposición. En términos de una nueva teoría, fue necesario pensar en una estructura que brindará una organización distinta, por lo tanto, se diseñaron supuestos explicativos, así como conceptos de primer y segundo orden, que guardan relaciones generales y específicas. No obstante, las fases de la descomposición evolutiva significan el marco esencial para explicar la teoría de salida, además de que permite posicionar cada concepto en un orden jerárquico con base en sus propias relaciones. El Esquema 7, presenta un organizador gráfico que ejemplifica los principales conceptos.

La evolución de la descomposición es un marco conceptual que engloba y organiza conceptos fundamentales, como la desestabilización, los ciclos estratégicos y la descolocación. De esta manera, proporciona una explicación jerárquica y cronológica de los procesos cognitivos y procedimentales que ocurren durante la descomposición. Además, cada uno de estos conceptos principales se relaciona con otros de segundo orden, como la lógica de formas, los objetos de anclaje, la descomposición repetitiva, la descomposición difusa, el patrón puente y la descomposición abstracta. En conjunto, estos conceptos explican lo que aquí llamamos la evolución de los procesos centrales de la descomposición.

El proceso de descomposición presenta cuatro fases distintas, cada una de las cuales cuenta con características fundamentales. Sin embargo, algunas de estas características se ramifican en función del conocimiento previo, que actúa como elementos de entrada que enriquecen los conceptos principales. No obstante, lo que se presenta aquí son los patrones que se observaron, independientemente de la situación o del conocimiento previo específico. También se mencionarán las variantes observadas como elementos a tener en cuenta al aplicar los conceptos de primer orden.

La primera fase se caracteriza por la experimentación y los cambios de procedimiento dentro del ámbito de la programación. Es decir, la fase 1 solo se produce cuando el contexto es nuevo para el individuo, ya que implica una exploración constante de la lógica de programación y las relaciones entre los

diferentes códigos. Por lo tanto, la experimentación con códigos en un contexto específico, en relación con la falta de conocimientos previos en términos informáticos, se conoce como desestabilización.

La desestabilización es un momento clave en el desarrollo del proceso de descomposición en un sujeto determinado, y se enfoca en las descomposiciones iniciales e incompletas que ocurren en un contexto computacional. Se denomina desestabilización debido a la falta de control en las relaciones entre los elementos de entrada y salida. En otras palabras, existen múltiples conexiones que se bifurcan y se modifican como resultado de la observación de diferentes resultados.

Esquema 7. Conceptos de primer y segundo orden.

En términos prácticos, si un sujeto sin experiencia en pseudo-lenguajes de programación se enfrenta a un problema relacionado con los actuadores de un robot en particular, se encontraría en un momento de desestabilización. En este momento, aplicaría códigos sin tener la certeza de que funcionen, y luego recolectaría información y retroalimentaría los mismos códigos en base a los resultados obtenidos.

De ahí que se mencione que esta primera fase mantiene un peso importante, por el ensayo y el error, que caracterizan e incluso forman parte de la esencia de la descomposición inicial e incompleta. Otro de los aspectos a resaltar, tiene que ver con la importancia del error de códigos. Cada uno de ellos, permite al sujeto comprender la lógica de la interface, transiciones, conexiones y funciones de cada bloque de programación. Por lo tanto, la desestabilización se trata de un concepto teórico capaz de develar la propia corriente epistemológica de la presente teoría. Es decir, el sujeto, en interacción con el objeto, y después de una retroalimentación abstracta entre la causa y la consecuencia de un código reflejado en un robot, forma parte de una inclinación por lo constructivista.

La desestabilización no debe de ser entendida como un momento de confusión únicamente, puesto que esto significaría una denominación reduccionista, va más allá de eso, se trata de un momento procesual caracterizado por darle sentido a aquello que todavía es desconocido. Dentro de este desconocimiento, se privilegia la interacción con los elementos intermediarios para otorgarle un significado propio a cada entrada y salida.

Con lo anterior, se afirma que la desestabilización en realidad nunca termina, dado que volverá cada vez que el sujeto se enfrente a una situación desconocida, lo que sucede a menudo dentro de la programación. De igual forma, existen conceptos que explican cuáles con los procedimientos utilizados para estabilizar aquello que carece de sentido propio. Uno de ellos, es la lógica de formas, ésta se define como aquellos mecanismos dentro de escenarios específicos, que proponen reglas y delimitaciones para que un pseudo-lenguaje de programación funcione.

La lógica de formas interviene directamente en el *software*, por lo que, evidentemente será distinta para un pseudo-lenguaje que para otro. En este caso, se privilegiaba el orden tanto vertical más que el horizontal, sin embargo, los

bloques y sus formas reflejaban unidades de sentido esenciales para comprender sus conexiones.

En contra de este concepto, se encuentra el llamado "objeto de anclaje", definido como aquellos elementos, físicos y abstractos, pertenecientes a los conocimientos previos del sujeto y que otorgan sentido a un elemento desconocido. De ahí su nombre de anclaje, como aquel elemento que se adhiere a otro y traslada su significado, provocando así una estabilización momentánea, al menos hasta que la lógica de formas permite su ingreso.

Este binomio conceptual se trata de dos elementos teóricos que se contraponen, pero al mismo tiempo son necesarios para coexistir dentro de una misma desestabilización. En otras palabras, la desestabilización ocurre por la falta de comprensión de la lógica de formas, es decir, no se han comprendido a profundidad las reglas ni delimitaciones del *software* de programación, por lo tanto, el robot resulta incompetente según los códigos diseñados. Por lo tanto, ante este problema, surgen los objetos de anclaje que, como si se tratasen de buscadores de elementos desconocidos, tienen el propósito de identificar aquello que es incomprensible, para relacionarlo con aquello que sí lo es, al menos de dominio. Lo anterior explica el supuesto de que, a mayores conocimientos previos computacionales, mayores las alternativas para resolver problemas en situaciones problemáticas nuevas.

Posteriormente, una vez que la desestabilización se posiciona en segundo plano, surgen los ciclos estratégicos, característicos de la fase 2. Un ciclo estratégico se define como el proceso de reestructuración de estrategias de descomposición, con el propósito de adecuarse a las diversas situaciones en las que se encuentren, de tal forma que, una estrategia puede entrar en un ciclo cuando ésta no logra responder a las necesidades de su entorno, por lo que debe cambiar alguno de sus elementos.

Los ciclos estratégicos se retroalimentan de otros conceptos de segundo orden, como la descomposición repetitiva y la descomposición difusa. Ambos conceptos forman parte de un binomio conceptual. Por un lado, la descomposición repetitiva se enfoca en repetir lo que ha sido útil en situaciones anteriores sin reflexionar sobre la lógica de formas o los objetos de anclaje específicos. Como resultado, la descomposición repetitiva tiende a ser incorrecta en diversos problemas de programación. En este punto, se transforma

en descomposición difusa, cuando la repetición constante no logra resolver los errores cometidos y no se han depurado adecuadamente. La descomposición difusa se refiere a un momento en el proceso de descomposición en el que la lógica de formas y los objetos de anclaje no son suficientes. En este sentido, surge un concepto de segundo orden conocido como "desanclaje conceptual.

El desanclaje hace énfasis a objetos de anclaje que, inmersos en un ciclo estratégico dentro de la lógica de formas, se convierte más que en una solución, en un problema más. Es decir, estos objetos caen en un ciclo sin fin de errores sin depuración, que se utilizan sin ningún tipo de modificación, sin embargo, al tener lógicas de formas distintas a los problemas anteriores, se convierten meramente en objetos inanimados, sin una utilidad y significado para el escenario actual.

En este sentido, el desanclaje conceptual ocurre cuando el sujeto identifica que el ciclo estratégico se ha estancado. Es importante señalar que estos ciclos cuentan con diversas fases, tales como la identificación, diseño, aplicación, evaluación y retroalimentación. En ocasiones la aplicación no se lleva a cabo de manera correcta, por lo tanto, necesita de una evaluación que permita corregir errores cometidos en los códigos de programación, es justamente en esta etapa del ciclo estratégico donde interviene el desanclaje conceptual, que tiene que ver con excluir y en ocasiones ignorar aquellos elementos del pasado, para probar acciones nuevas y arriesgadas que permitan solventar el problema.

El desanclaje conceptual también consiste en bifurcar la función de algún objeto de anclaje. Por ejemplo, supongamos que un objeto específico, como un rompecabezas, se utilizó para resolver anteriormente 8 problemas similares, y siempre tuvo resultados positivos. Sin embargo, al implementarlo en el problema 9, éste no logra resolver el problema, porque las características son distintas o porque la lógica de forma cambió drásticamente. En este caso, el objeto de anclaje debe de reemplazarse o cambiar el sentido del mismo, por ejemplo, en vez de utilizar cada pieza como conteo de bloques, que se utilice para idea un orden distinto, de esa forma cambie el significado de anclaje y se resuelva el problema. Sin lugar a dudas, el desanclaje conceptual enriquece el PD, porque logra que un solo objeto, tenga diversos usos, se profundice en él y se amplíen las posibilidades de solución en el futuro.

Además, las fases 2 y 3 se entrelazan conceptualmente, siendo las principales diferencias la forma en que evolucionan los ciclos estratégicos. En la fase 2, los ciclos estratégicos se diseñan e implementan por primera vez en un contexto de desestabilización. Por otro lado, en la fase 3, los ciclos estratégicos se aplican correctamente y se realizan modificaciones adicionales. Para explicar esto, es necesario destacar el concepto de segundo grado conocido como "ciclos reformulados", que se refiere a la asignación de mayor importancia a una de las etapas que los conforman.

Por ejemplo, para situaciones de actuadores de un robot, los ciclos reformulados se enfocan más en la parte del diseño de la estrategia de descomposición, mientras que, en situaciones de algunos sensores específicos, como los infrarrojos, se enfoca más en la parte de la retroalimentación, ya que se trata de lógica de formas mucho más complejas, que necesitan de mayor cuidado al momento de programar. En otras palabras, la fase 2 representa el inicio de un ciclo estratégico, en donde se encuentran todas las estrategias de descomposición y su tipología, y la fase 3 representa el momento de depuración de los mismos ciclos.

Por último, se encuentra la fase 4, en donde el concepto de primer orden es la descolocación, ésta se define como el proceso clave de diseño, reestructuración y comprensión de los elementos más profundos de la descomposición. Descolocar se trata de un adjetivo conceptual, por así decirlo, de uno de los elementos clave del PD, es decir, aquello que ya existe de una manera determinada, llevado al siguiente nivel de complejidad, dando un salto cualitativo hacia la adaptación a las situaciones específicas que surgen en tiempo real.

Una vez que el sujeto domina la descolocación y la sitúa en el centro del proceso de pensamiento computacional, este se convierte en un proceso dinámico y deja de seguir recetas procedimentales que alguna vez lo obstaculizaron. La descolocación implica la capacidad de modificar estrategias, reestructurarlas y cambiar su enfoque en beneficio del propósito de programación. En este proceso de descolocación, intervienen otros elementos centrales del Pensamiento Computacional, como la automatización, la abstracción, la evaluación, el pensamiento algorítmico y otros elementos clave que enriquecen el proceso.

La descolocación está estrechamente relacionada con dos conceptos de segundo orden: el patrón puente y la descomposición abstracta. El patrón puen-

te se refiere a la capacidad del sujeto para identificar las relaciones clave que permiten que la lógica de la forma funcione de la manera en que lo hace. De esta manera, no importa cuáles sean las nuevas mecánicas de programación, el sujeto es capaz de comprender, a través de la exploración y la desestabilización, los patrones que representan similitudes entre sus conocimientos previos. Podría considerarse como el uso automatizado de objetos de anclaje. Por otro lado, la descomposición abstracta se refiere a la capacidad del sujeto para llevar a cabo el proceso sin necesidad de objetos concretos. Es decir, es un nivel de comprensión tan especializado que permite resolver problemas incluso sin descomponer algunos de sus elementos.

La descolocación se ha convertido en uno de los conceptos teóricos más complejos y difíciles de observar empíricamente, ya que muchos de sus elementos ocurren de manera automatizada y a menudo son ignorados u omitidos por los sujetos durante las entrevistas. Los procedimientos que han sido dominados al punto de realizarse automáticamente forman parte de la descolocación. Metafóricamente, se podría decir que la descolocación es como un banco de elementos de entrada, intermediarios y de salida que solían ser considerados una "caja negra", pero ahora sabemos que forman parte de una serie de fases que en la adaptación y reestructuración pueden parecer difusas.

5. Propuesta pedagógica para el desarrollo de la descomposición y el Pensamiento Computacional

El PC, y en particular la descomposición, se plantean como una vía para fortalecer el pensamiento crítico de los estudiantes en diversos niveles educativos, sin embargo, los supuestos teóricos que se han expuestos tienen su enfoque en nivel primaria. Este capítulo tiene como finalidad proporcionar a los docentes una guía pedagógica para incorporar el PC en el aula, destacando su potencial como herramienta transversal aplicable a diversas disciplinas, contenidos y Procesos de Desarrollo y Aprendizaje (PDA).

Se busca que el docente no sólo sea capaz de desarrollar habilidades técnicas en sus estudiantes, sino que fomente una nueva forma de pensar que transcienda el ámbito digital y tenga aplicaciones en situaciones académicas no computacionales y en la vida cotidiana. Es a través de este capítulo, donde se ofrecerán estrategias y actividades concretas que permitan a los docentes incluir la descomposición dentro del proceso de enseñanza y aprendizaje de manera gradual, además de que se exploran ejemplos de *Scratch* y *Mblockly*, de tal forma de que exista una bifurcación tecnológica en pseudolenguajes de programación de acceso libre, y que solamente necesiten de una computadora o dispositivo móvil para utilizarse.

Se subraya también la importancia de utilizar el PC como un vehículo para el aprendizaje activo, donde los estudiantes asuman un papel protagónico en la construcción de su conocimiento. Lejos de ser un enfoque centrado únicamente en la transmisión de información, el propósito es crear experiencias educativas en donde los estudiantes puedan explorar, experimentar, reflexionar, colaborar y aplicar sus conocimientos. La descomposición se convierte en una herramienta que fomenta el descubrimiento y el deseo de aprender.

Otro de los propósitos del capítulo, por un lado, pretende reforzar la noción de que el PD es exclusivo del ámbito de las ciencias o la tecnología, sino que

se trata de una competencia transversal que fortalece el desarrollo del pensamiento crítico en los estudiantes. Por otro lado, facilitar que los docentes integren el PC en sus planeaciones diarias, no como una carga adicional, sino como un recurso que potencia las actividades ya existentes. Se busca que el docente sea capaz de identificar oportunidades para aplicar la descomposición en proyectos escolares, ejercicios de escritura, resolución de problemas matemáticos o análisis de distintos fenómenos.

La descomposición se presenta como una habilidad que no sólo responde a los desafíos del entorno digital, sino que capacita a los estudiantes para abordar situaciones del mundo real con una perspectiva crítica y resolutiva. A través de ejemplos y recomendaciones que aquí se exponen, los docentes podrán dotar a sus alumnos de herramientas que les permitan desenvolverse con éxito en un entorno complejo y en constante transformación.

La descomposición como eje transversal en el papel del docente

La descomposición se concibe como un pilar fundamental en el desarrollo del PC, destaca no sólo como técnica para resolver problemas complejos, sino como un eje transversal que conecta y enriquece los otros procesos centrales. Al segmentar un objeto, artefacto o tarea compleja en partes más pequeñas y manejables, los estudiantes logran comprender las estructuras subyacentes de un problema, y esto les permite analizar, reorganizar y construir soluciones progresivamente. Por lo tanto, el aprendizaje se convierte en un proceso dinámico, donde el ensayo y error, así como la revisión de procedimientos forman parte esencial del desarrollo de competencias cognitivas superiores.

En educación básica, la descomposición no debe limitarse a contextos tecnológicos o digitales, dado que su aplicabilidad trasciende la programación y se convierte en una estrategia valiosa para asignaturas como matemáticas, ciencias naturales, lenguaje y educación cívica, así como cada uno de los campos formativos que se trabajar hoy en día con la Nueva Escuela Mexicana (NEM). Por ejemplo, descomponer un problema matemático, ayudaría a que los estudiantes puedan identificar operaciones básicas que, al resolverse de manera aislada, contribuyen a la solución integral del ejercicio, de manera

similar ocurre con el lenguaje, al descomponer la estructura de un texto narrativo en introducción, desarrollo y desenlace, permite comprender sus componentes esenciales y facilita la creación de producciones escritas propias.

Además, su carácter interdisciplinario no sólo promueve el desarrollo de habilidades analíticas, sino que también fomenta el pensamiento crítico y la capacidad de transferir conocimientos a diferentes áreas del currículo. La descomposición se presenta como una estrategia transversal, que ayuda a los estudiantes a reconocer patrones, establecer conexiones y aplicar soluciones previas en nuevos contextos. Esta capacidad de transferir aprendizajes a nuevas situaciones es fundamental para construir una educación significativa y duradera, donde los conocimientos no se conciban como elementos estancados, sino como piezas interrelacionadas en un mismo proceso formativo.

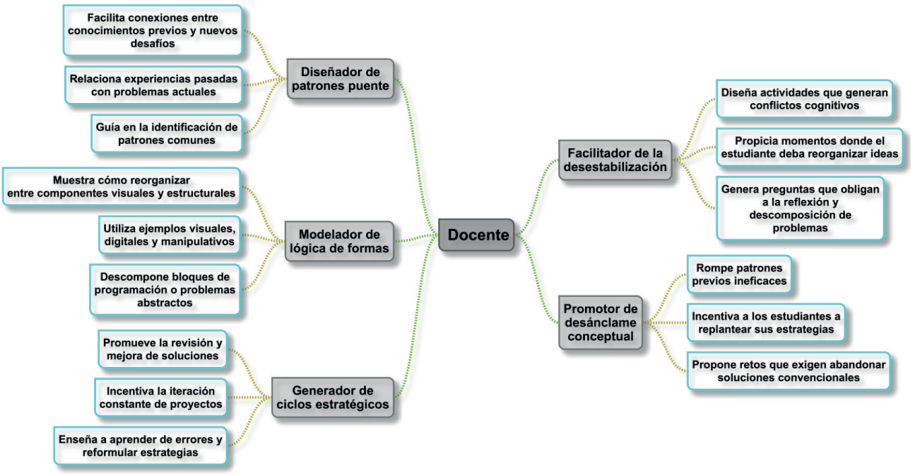
En el aula, la descomposición también se convierte en una herramienta que fomenta el aprendizaje colaborativo, por ejemplo, al dividir un proyecto en tareas específicas de trabajo, en donde cada estudiante asuma un rol dentro del grupo y facilite la distribución equitativa de trabajo con base en las habilidades de cada uno. Esta dinámica de aprendizaje, además de reflejar entornos laborales y académicos reales, promueve en los estudiantes habilidades de comunicación, empatía y cooperación, fundamentales para formar ciudadanos críticos y participativos.

No obstante, ¿Qué papel juega el docente en este proceso? En el Diagrama el papel docente explica cinco características clave para promover la descomposición en el aula.

El docente promueve la desestabilización en el aula, ya que entiende que el aprendizaje profundo surge cuando los estudiantes se enfrentan a situaciones desafiantes, no se trata de generar confusión, sino crear escenarios donde los alumnos cuestionen sus enfoques habituales. Este proceso es clave para que la descomposición se dé, ya que, ante un desafío complejo, los estudiantes deben segmentar el problema en partes más simples para reorganizarlo y comprenderlo de manera integral.

En este sentido, la desestabilización por parte del docente implica proponer problemas que no puedan resolverse de manera inmediata con las estrategias conocidas por los estudiantes, sin embargo, que cuenten los indicios de conocimientos previos básicos para acercarse a una solución. Desestabilizar

Diagrama el papel del docente.



Elaboración propia.

también se manifiesta en cómo el docente formula preguntas, en lugar de ofrecer respuestas directas, guía a los estudiantes mediante interrogantes abiertas para propiciar la reflexión, por ejemplo, ¿Qué sucedería si eliminamos este bloque del código? ¿Cómo podríamos simplificar este proceso?, impulsar así a los alumnos al análisis constante de la estructura de su trabajo y considerar posibilidades inexploradas.

En aritmética, por ejemplo, el docente puede presentar problemas donde la solución no es evidente de inmediato: "Si tienes 124 caramelos y quieres repartirlos en bolsas con cantidades iguales, pero te sobra un número pequeño, ¿cómo podrías resolverlo sin dejar caramelos fuera?" Este tipo de problema obliga a los estudiantes a descomponer el número total en divisiones sucesivas, explorando patrones de reparto y conceptos de divisibilidad. De manera similar, en situaciones cotidianas, el docente puede plantear preguntas como: "Si queremos hacer un mural con recortes de papel y cada fila necesita 6 recortes, ¿cuántas filas completas podemos hacer con 50 recortes?" Aquí, el estudiante debe dividir y razonar sobre los sobrantes, fomentando la descomposición y el pensamiento lógico.

Finalmente, el facilitador de la desestabilización contribuye a formar estudiantes resilientes y autónomos, preparados para enfrentar los desafíos del mundo real. A través de la descomposición, los alumnos adquieren una metodología que pueden aplicar en diversos contextos. En actividades diarias, el docente puede aprovechar situaciones como organizar un paseo escolar: "Tenemos 32 estudiantes y los autobuses tienen 12 asientos cada uno. ¿Cuántos autobuses necesitaremos y cuántos asientos sobrarán?" Este ejercicio impulsa a los alumnos a dividir el total, calcular restos y visualizar soluciones prácticas. Otro ejemplo podría ser el presupuesto de una feria escolar: "Si cada grupo debe recolectar $385 para la decoración y cada alumno aporta $25, ¿cuántos alumnos deben participar para alcanzar la meta?" Estos ejemplos no solo abordan conceptos matemáticos, sino que también integran habilidades de planificación y resolución de problemas que los estudiantes podrán aplicar fuera del aula.

El docente también actúa como promotor del desanclaje conceptual, éste se refiere al proceso que rompe con los patrones previos de pensamiento o estrategias que ya no resultan funcionales, provocando que los estudiantes cuestionen sus procedimientos tradicionales. Uno de los aspectos fundamentales del desanclaje, es fomentar el entorno del error como un elemento valorado y una herramienta de aprendizaje. De tal forma que, el docente refuerce la idea de que el error no se trata de un obstáculo, sino de una oportunidad para analizar y reconstruir los procesos, el propósito principal es distanciarse de las soluciones rígidas y propiciar la creatividad y el pensamiento lateral.

Es comprensible pensar que la desestabilización y el desanclaje conceptual son características del papel del docente similares, sin embargo, mantienen diferencias específicas. Por un lado, la desestabilización ocurre cuando el estudiante enfrenta un problema que desafío sus conocimientos previos, lo que provoca un desequilibrio cognitivo, por otro lado, el desanclaje implica liberar al estudiante de patrones de pensamiento rígidos o soluciones de descomposición repetitivas. En el siguiente cuadro comparativo se observan sus diferencias clave.

Desestabilización y desanclaje conceptual: Similitudes y diferencias.

Elemento clave	Desestabilización	Desanclaje conceptual
Enfoque	Crear desequilibrio o contradicción	Romper patrones repetitivos
Momento de aplicación	Cuando el estudiante se siente cómodo con su conocimiento	Cuando el alumno repite estrategias sin reflexión
Objetivo	Provocar conflicto y reorganización mental	Expandir el repertorio de soluciones
Resultado esperado	Reconstrucción de la solución desde cero	Exploración de nuevos enfoques
Ejemplo de aula	Problemas con restricciones inesperadas	Problemas abiertos con múltiples soluciones

Elaboración propia.

Dentro de estas diferencias existe una relación y complementariedad entre ambas características, es decir, un docente debe de tomar primero el papel de desestabilizador, en donde el estudiante enfrenta un problema que no puede resolver con estrategias habituales, para después tomar el papel promotor de desanclaje, en donde el alumno encuentra una solución y se le alienta a buscar nuevas formas de resolverlo. Ambas características permiten al estudiante ser flexible, creativo y resiliente.

El docente que actúa como diseñador de patrones puente facilita el tránsito de los estudiantes entre conocimientos previos y nuevos aprendizajes, utilizando conexiones que les permitan establecer relaciones significativas entre conceptos. Esta característica es esencial en el desarrollo del PC y la descomposición, ya que los patrones puente funcionan como guías que ayudan a los alumnos a integrar nuevas estrategias a partir de experiencias conocidas. El docente, en este sentido, se convierte en un mediador que habilita la transferencia de habilidades entre distintos contextos, permitiendo que los estudiantes construyan soluciones de forma progresiva y consciente.

La creación de patrones puente implica identificar puntos de anclaje en el conocimiento del estudiante y construir sobre ellos. Por ejemplo, al enseñar fracciones, el docente puede vincular el concepto con la división de objetos cotidianos como pizzas o manzanas. De esta manera, el alumno conecta la operación abstracta con una experiencia concreta, facilitando la comprensión y posterior aplicación en contextos más complejos. En el ámbito del PC, estos

patrones emergen cuando el docente utiliza ejemplos visuales o manipulativos para explicar la descomposición de problemas, permitiendo que los estudiantes asocien estructuras conocidas, como bloques de construcción, con elementos de programación en *Scratch* o *Mblockly*.

Los patrones puente no solo sirven para conectar conceptos, sino que también permiten que los estudiantes generalicen aprendizajes a diferentes áreas. Al enfrentar un problema nuevo, el alumno recurre a estructuras mentales formadas a partir de patrones previos, lo que facilita la descomposición y reorganización del problema. El docente, al diseñar actividades que requieren aplicar conocimientos de diversas disciplinas, fomenta la transversalidad y el pensamiento sistémico. Por ejemplo, un proyecto que combine arte y matemáticas –como diseñar figuras geométricas para construir mosaicos– permite a los estudiantes aplicar conceptos de área y perímetro mientras desarrollan habilidades creativas.

Además, el docente que diseña patrones puente no se limita a ofrecer conexiones directas, sino que también fomenta la identificación de patrones emergentes. A través de preguntas abiertas, el docente guía a los estudiantes hacia la búsqueda de relaciones entre problemas aparentemente disímiles. Preguntas como "¿Dónde más hemos visto un problema similar?" o "¿Cómo resolviste algo parecido antes?" invitan a los alumnos a reflexionar sobre sus experiencias previas y a aplicar estrategias de descomposición en nuevos contextos. Este enfoque no solo fortalece el aprendizaje autónomo, sino que también consolida la capacidad de los estudiantes para transferir habilidades de forma independiente.

En el contexto de la aritmética, el docente puede diseñar patrones puente utilizando ejercicios de suma repetitiva para introducir la multiplicación. Al presentar problemas como "Si cada fila tiene 4 sillas y hay 6 filas, ¿Cómo podríamos calcular el total sin sumar una por una?", el estudiante reconoce la repetición como un patrón que puede simplificarse con una multiplicación. Este proceso permite que el alumno descomponga el problema en pasos más simples y utilice herramientas previas para construir soluciones más avanzadas.

En situaciones cotidianas, los patrones puente pueden surgir de actividades colaborativas, como planificar una feria escolar. El docente puede plan-

tear preguntas como "Si cada stand necesita 3 mesas y hay 8 grupos, ¿cuántas mesas en total necesitaremos?" Este tipo de ejercicio permite al estudiante vincular conceptos de multiplicación con problemas reales, fortaleciendo la comprensión a través de aplicaciones prácticas. Al conectar la resolución de problemas con la vida diaria, el docente no solo facilita el aprendizaje matemático, sino que también refuerza la importancia del PC y la descomposición como herramientas para la toma de decisiones.

El docente que modela la lógica de formas en el aula facilita que los estudiantes comprendan la estructura y funcionamiento de programas visuales a través de la disposición y conexión de bloques de código. En entornos como *Scratch* o *Mblockly*, la manera en que los bloques se organizan refleja directamente la secuencia de acciones, bucles o condiciones. Esta representación gráfica permite a los estudiantes visualizar la lógica de un programa sin necesidad de enfrentarse a líneas de código escritas, lo que convierte la enseñanza del PC en un proceso accesible y atractivo. El docente, en este rol, enseña a interpretar estas estructuras visuales, destacando cómo la disposición de los bloques afecta el flujo de ejecución y la resolución de problemas.

El modelador de la lógica de formas se enfoca en mostrar que los bloques de programación no solo representan acciones individuales, sino que forman parte de un sistema donde la jerarquía y el orden son esenciales. Al observar un programa en *Scratch*, por ejemplo, el estudiante puede identificar que los bloques "si… entonces" condicionan la ejecución de ciertas acciones, mientras que los bloques de repetición crean ciclos que automatizan procesos. El docente, al modelar este tipo de estructuras, guía a los alumnos en la descomposición de programas más grandes, permitiéndoles modificar, reorganizar o añadir elementos para optimizar el resultado final.

Este enfoque también promueve que los estudiantes entiendan que la descomposición visual no es arbitraria; cada bloque tiene un propósito y una ubicación específica dentro de la secuencia. El docente puede demostrar cómo pequeños ajustes en la posición de un bloque cambian por completo el comportamiento del programa. Por ejemplo, mover un bloque de sonido fuera de un bucle hará que se ejecute solo una vez, mientras que dentro del bucle se repetirá indefinidamente. Estos ejercicios permiten que los alumnos experi-

menten con la estructura, fomentando un aprendizaje activo basado en la exploración y la iteración.

Además, el docente que modela la lógica de formas no solo trabaja con programación visual, sino que aplica esta estrategia a la resolución de problemas en otros contextos. Las dinámicas de organización de ideas, como los diagramas de flujo o mapas conceptuales, utilizan principios similares de disposición secuencial y jerárquica. De esta manera, los estudiantes internalizan que la lógica de formas trasciende la programación y puede aplicarse en la planificación de proyectos, la escritura de textos narrativos o la resolución de problemas matemáticos que requieren múltiples pasos.

En situaciones cotidianas, el docente puede aplicar este enfoque a través de actividades como diagramar una receta o planificar un cronograma de clase. Proponer: "Organiza los pasos para armar un papalote. ¿Qué sucede si colocamos el hilo antes de unir las varillas?" ayuda a los estudiantes a identificar la importancia del orden en cualquier proceso. Al visualizar y reorganizar las acciones, los alumnos reconocen la lógica que subyace en tareas diarias, reforzando así su capacidad de descomposición y planificación.

Replantear soluciones es una habilidad fundamental que el docente debe cultivar en los estudiantes. Los ciclos estratégicos no solo se centran en resolver un problema, sino en volver sobre él para perfeccionarlo, descubrir nuevas rutas y optimizar el proceso. La enseñanza del PC se enriquece cuando el aula se convierte en un espacio donde los alumnos comprenden que incluso las respuestas correctas pueden mejorarse. Esta mentalidad fomenta la flexibilidad, el análisis crítico y la disposición a revisar el propio trabajo con una mirada constructiva.

El aprendizaje se vuelve más significativo cuando los estudiantes son guiados a identificar patrones repetitivos y áreas que pueden simplificarse. Por ejemplo, al realizar una suma de fracciones, se puede invitar a los alumnos a encontrar atajos o utilizar descomposición para simplificar denominadores. Estas oportunidades permiten que los estudiantes no solo lleguen al resultado, sino que comprendan cómo y por qué un método puede ser más eficiente que otro. En estos casos, el docente introduce la noción de optimización como parte esencial del proceso, animando a los alumnos a explorar diferentes caminos y tomar decisiones fundamentadas.

Incorporar ciclos estratégicos en la práctica diaria implica normalizar la revisión constante. Al finalizar un proyecto en Scratch o un cálculo matemático, se puede preguntar: "¿Esta solución puede hacerse en menos pasos?" o "¿Qué bloque del código podría simplificarse sin cambiar el resultado final?" Al fomentar este tipo de reflexión, el docente modela la importancia de mirar hacia atrás, no como señal de error, sino como una estrategia consciente de mejora.

Es clave que el estudiante aprenda a identificar por sí mismo qué partes del proceso requieren ajustes, desarrollando autonomía y pensamiento crítico. Actividades como diseñar un recorrido para un robot en *Mblockly* ofrecen un campo fértil para aplicar esta estrategia. Cuando el robot sigue un camino largo o poco preciso, el docente puede guiar a los alumnos a revisar los bloques y reorganizarlos para optimizar el trayecto. Este proceso de reconfiguración no solo fortalece las habilidades técnicas, sino que enseña que los resultados satisfactorios pueden evolucionar hacia soluciones más elegantes y efectivas.

En aritmética, los ciclos estratégicos pueden manifestarse en problemas de reparto. Un ejemplo sería plantear: "Si tienes 56 galletas y quieres repartirlas en partes iguales entre 7 niños, ¿cuántas recibe cada uno?" Una vez resuelto, el docente podría agregar: "¿Cómo podríamos haber descompuesto el 56 para hacerlo más rápido?" Al abordar la solución desde otro ángulo, los estudiantes aprenden que la descomposición puede ser una herramienta de agilidad mental.

Las aplicaciones en la vida diaria refuerzan la utilidad de esta práctica. Cuando los estudiantes organizan un evento escolar, el docente puede invitarlos a reflexionar sobre la distribución de recursos: "¿Cómo podríamos reducir el tiempo de montaje reorganizando las tareas del equipo?" Este ejercicio, lejos de limitarse a la planeación, inculca el hábito de pensar estratégicamente en cada paso, transfiriendo la noción de ciclos estratégicos a cualquier aspecto de su vida académica o personal.

El papel del docente en la enseñanza de la descomposición va más allá de transmitir conocimientos; implica crear un entorno donde los estudiantes desarrollen habilidades cognitivas que les permitan fragmentar problemas, reorganizar ideas y construir soluciones de manera progresiva. A lo largo de las cinco características exploradas (facilitador de la desestabilización, promotor del desanclaje conceptual, diseñador de patrones puente, modelador de la ló-

gica de formas y generador de ciclos estratégicos) se ha delineado un perfil docente que guía, reta y acompaña al alumno en cada fase del aprendizaje. La descomposición no es solo una técnica, sino una forma de pensar que se cultiva desde la práctica diaria en el aula.

Para implementar esta visión, el docente debe iniciar con la desestabilización, rompiendo el equilibrio inicial del estudiante a través de preguntas, actividades o problemas que desafíen sus estrategias previas. Este primer paso busca despertar la curiosidad y generar conflictos cognitivos que conduzcan a la exploración de nuevas rutas. Es en este punto donde el aprendizaje se vuelve significativo, ya que el alumno se enfrenta a la necesidad de reorganizar sus ideas y buscar soluciones fragmentadas que, al integrarse, resuelvan el desafío propuesto.

El desanclaje conceptual surge en el momento en que los estudiantes encuentran una solución, pero el docente les invita a romper con patrones repetitivos y explorar alternativas. Aquí es fundamental cuestionar: "¿Cómo podrías resolverlo de otra manera?" o "¿Qué sucede si cambias esta parte del proceso?" Este enfoque permite ampliar el repertorio de soluciones y evita que los alumnos se conformen con una única respuesta. El docente, entonces, no solo enseña a resolver, sino a reinventar la solución a partir de nuevas perspectivas.

El diseño de patrones puente facilita que los estudiantes conecten aprendizajes previos con problemas actuales, ayudándolos a identificar relaciones y a transferir conocimientos entre disciplinas. El docente actúa como mediador, mostrando similitudes y analogías que permiten descomponer problemas más complejos en términos familiares. Esta práctica se vuelve crucial en la enseñanza interdisciplinaria, donde los estudiantes aplican conceptos de matemáticas en programación o utilizan principios de ciencias naturales para resolver problemas de robótica educativa.

Cuando se modela la lógica de formas, el docente proporciona herramientas visuales y estructurales que ayudan a los estudiantes a interpretar y reorganizar elementos. A través de entornos como *Scratch* o *Mblockly*, se enseña a los alumnos a manipular bloques de código, secuencias y condiciones, comprendiendo que el orden y la disposición afectan directamente el resultado. Esta habilidad se transfiere a otros campos, donde los estudiantes aprenden a

organizar ideas en diagramas, esquemas o proyectos, fortaleciendo su capacidad de descomponer conceptos visualmente.

Finalmente, el docente introduce los ciclos estratégicos, mostrando que todo proceso puede revisarse, ajustarse y optimizarse. Es en este punto donde el aprendizaje se consolida, ya que los estudiantes entienden que siempre existe margen para mejorar. La iteración se convierte en una práctica habitual, permitiendo que los alumnos no solo resuelvan problemas, sino que refinen continuamente sus métodos. El docente, con esta visión, no solo enseña a llegar a una respuesta, sino a construir un camino de aprendizaje donde cada error es una oportunidad y cada intento un paso más hacia el dominio de la descomposición.

La siguiente tabla sintetiza las acciones clave que debe realizar el docente en distintos momentos del proceso de enseñanza para fomentar la descomposición.

El papel del docente.

Papel del docente	¿Qué hacer?	¿Cuándo hacerlo?	¿De qué manera?
Facilitador de la desestabilización	Presentar problemas que rompan con el equilibrio cognitivo	Al inicio de una nueva unidad o proyecto	Plantear preguntas abiertas, problemas con restricciones o variaciones inesperadas
Promotor del desanclaje conceptual	Romper patrones repetitivos y fomentar nuevas estrategias	Cuando los estudiantes encuentran una solución repetitiva	Pedir que resuelva el problema con otro procedimiento o eliminen pasos innecesarios
Diseñador de patrones puente	Conectar conocimientos previos con nuevos problemas	Al introducir conceptos complejos o interdisciplinarios	Usar analogías, ejemplos visuales y experiencias cercanas a los alumnos
Modelador de la lógica de formas	Enseñar a reorganizar elementos visuales y estructurales	Durante la resolución de problemas de programación o proyectos creativos	Guiar en la disposición de bloques de código, diagramas o mapas visuales
Generador de ciclos estratégicos	Impulsar la revisión y mejora continua de soluciones	Después de completar una actividad o proyecto inicial	Pedir que optimicen sus respuestas, agreguen mejoras o simplifiquen procesos.

Elaboración propia.

Propuesta taxonómica de la descomposición

Tal como la taxonomía de Bloom ha sido una guía para docentes de diversos niveles educativos, en este apartado se expone una taxonomía que permite organizar y visualizar el progreso de los estudiantes en la descomposición, a través de cuatro niveles jerárquicos que reflejan el crecimiento progresivo del estudiante.

El primer nivel, *Descomposición Inicial*, representa el punto de partida donde el estudiante enfrenta un problema por primera vez. En esta fase, predomina la exploración y el ensayo-error. Se busca que el alumno reconozca patrones básicos a través de herramientas como la lógica de formas, que permite comprender estructuras visuales en entornos como *Scratch* o *Mblockly*. Los Objetos de Anclaje, aquellos conceptos conocidos que sirven como referencia, también juegan un papel clave en esta etapa. A pesar de los errores frecuentes, esta fase sienta las bases para una descomposición más elaborada en el futuro.

Conforme el estudiante avanza, inicia la etapa de la *descomposición repetitiva*, donde recurre a patrones y soluciones previas, aunque no siempre de forma efectiva. Aquí, el docente permite que el estudiante repita códigos o estrategias conocidas, alentando la exploración de nuevas variaciones. Se introducen conceptos como la repetición de elementos, buscando afianzar habilidades y promover una resolución parcial del problema. Aunque se observan errores reincidentes, esta fase es crucial para consolidar patrones y mejorar la comprensión del proceso.

La *descomposición selectiva* marca un salto cualitativo, donde el estudiante elige de forma deliberada las estrategias que considera más adecuadas, basándose en experiencias previas. Aquí, emergen técnicas como el patrón puente, que conecta situaciones conocidas con problemas actuales, y el desanclaje Conceptual, que permite romper con soluciones ineficaces y explorar alternativas. Este nivel refleja una mayor precisión y eficiencia, con errores cada vez más reducidos.

Finalmente, la *descomposición abstracta* se caracteriza por la capacidad del estudiante para descomponer problemas sin depender de apoyos visuales o tangibles. La descolocación y la automatización permiten que las soluciones

surjan de manera fluida y sin errores evidentes. El estudiante alcanza una comprensión avanzada del problema, lo que le permite anticipar dificultades y adaptar estrategias con rapidez.

Este proceso de descomposición se complementa con una tipología que define las diferentes maneras de segmentar problemas: la descomposición estructural, que divide un problema en partes físicas; la funcional, que se enfoca en las funciones de cada componente; la secuencial, que organiza los elementos en orden de ejecución; y la dependiente, que mantiene la interconexión entre las partes.

Para guiar este proceso, se aplican estrategias clave como la descomposición desde el final significativo, que parte del resultado y retrocede hacia los componentes iniciales; la estrategia de abajo hacia arriba, donde se construye desde elementos conocidos hacia el problema general; y la multivariable, que aborda varios elementos de forma simultánea. La estrategia Comparativa también permite identificar patrones y errores al analizar distintas soluciones.

En la Taxonomía del PD se muestra la propuesta, en donde implícitamente se encuentran inmersa su tipología y estrategias específicas.

Esta taxonomía se integra en el proceso educativo a través de cuatro fases: la exploración inicial, la repetición guiada, la selección de estrategias y la automatización de procedimientos. El rol del docente es fundamental en cada una de estas etapas, facilitando una progresión desde los niveles básicos hasta el dominio avanzado de la descomposición.

La relación entre la taxonomía del proceso de descomposición y la taxonomía de Bloom refleja cómo la enseñanza del PC no sólo desarrolla habilidades técnicas, sino que también impulsa el crecimiento cognitivo progresivo. A medida que los estudiantes avanzan en los niveles de descomposición, desde identificar elementos básicos hasta automatizar soluciones complejas, atraviesan las fases de Bloom, comenzando por recordar y comprender, hasta aplicar, analizar, evaluar y crear. Este paralelismo muestra que la descomposición no es únicamente una herramienta para resolver problemas, sino un proceso que fortalece el pensamiento crítico, la flexibilidad cognitiva y la capacidad creativa. Al guiar a los estudiantes a través de este recorrido, el docente fomenta un aprendizaje profundo y significativo que conecta el análisis computacional con el desarrollo integral de habilidades cognitivas.

Taxonomía del PD.

Elaboración propia.

La taxonomía del proceso de descomposición no solo proporciona un marco para entender el desarrollo del PC, sino que también refleja la progresión natural del aprendizaje, desde los primeros intentos exploratorios hasta la automatización de soluciones complejas. A través de cada nivel desde la descomposición inicial hasta la abstracta, los estudiantes no sólo adquieren habilidades técnicas, sino que fortalecen su capacidad para analizar problemas desde diferentes ángulos, adaptarse a nuevos desafíos y transferir aprendizajes a distintas áreas del conocimiento. Esta evolución está estrechamente ligada al crecimiento cognitivo que plantea la taxonomía de Bloom, donde las habilidades de recordar, comprender y aplicar forman la base para alcanzar niveles superiores de análisis, evaluación y creación.

El papel del docente es fundamental para guiar a los estudiantes en este recorrido. Desde proporcionar estructuras visuales y ejemplos concretos en las primeras fases, hasta promover el desanclaje conceptual y el cuestionamiento de soluciones en niveles más avanzados, el docente facilita que el estudian-

te se apropie del proceso de descomposición de forma autónoma. Al integrar esta taxonomía en la planificación educativa, no solo se enseña a resolver problemas técnicos, sino que se cultivan habilidades que preparan a los alumnos para abordar con creatividad y pensamiento crítico los retos del mundo real.

Cuadro comparativo Bloom y descomposición

Taxonomía de descomposición	Taxonomía de Bloom	Implicaciones cognitivas (Bloom)	Implicaciones de descomposición
Descomposición Inicial	Recordar	Reconocer, listar, identificar y recuperar información.	Separar elementos visibles del problema. Explorar sin tener un plan claro.
Descomposición Repetitiva	Comprender	Explicar, clasificar, interpretar y describir.	Repetir patrones conocidos. Imitar soluciones previas. Ajustar errores sin cambiar enfoques.
Descomposición Selectiva	Aplicar	Usar conceptos en nuevas situaciones. Implementar procesos conocidos.	Elegir estrategias efectivas. Relacionar problemas actuales con soluciones pasadas.
Descomposición Abstracta (Automatización)	Analizar	Desglosar, diferenciar, comparar y organizar.	Dividir problemas en componentes abstractos. Identificar relaciones ocultas entre elementos.
Descomposición Abstracta (Descolocación)	Evaluar	Justificar, argumentar, criticar y defender una postura.	Reformular estrategias. Identificar errores no evidentes. Modificar soluciones previas.
Automatización Completa	Crear	Diseñar, construir, desarrollar y generar nuevos enfoques.	Crear soluciones innovadoras. Descomponer mentalmente sin necesidad de apoyos visuales o físicos.

Elaboración propia con base en taxonomía de Bloom.

Descomponer en educación básica: algunas aplicaciones prácticas

La descomposición es una habilidad esencial que subyace en múltiples procesos de aprendizaje a lo largo de la educación básica. Desde el nivel preescolar hasta secundaria, los estudiantes se enfrentan a situaciones que requieren

dividir problemas, tareas o conceptos complejos en partes más manejables. Este proceso, aunque natural en el desarrollo cognitivo, puede potenciarse de manera consciente a través de proyectos didácticos diseñados específicamente para aplicar la descomposición en diferentes áreas del conocimiento. El objetivo no es solo resolver problemas, sino también construir una mentalidad analítica y flexible que permita abordar cualquier desafío con mayor claridad y precisión.

En el contexto de la Nueva Escuela Mexicana (NEM), el aprendizaje basado en proyectos y la transversalidad de los saberes brindan un escenario propicio para que la descomposición se integre de manera natural en la práctica educativa. A través de proyectos que abarcan desde la exploración de fenómenos naturales hasta la resolución de problemas sociales, los estudiantes tienen la oportunidad de dividir tareas en fases concretas: exploración, indagación, construcción y presentación final. Este enfoque no solo refuerza el contenido académico, sino que también desarrolla habilidades de organización, pensamiento crítico y autonomía.

Aplicar la descomposición en el aula no implica únicamente enseñar a dividir problemas matemáticos o programar secuencias en entornos computacionales. También se extiende a procesos de escritura, análisis de textos, organización de ideas y ejecución de proyectos interdisciplinarios. Cada proyecto, sin importar su naturaleza, puede estructurarse de manera que los estudiantes aprendan a identificar patrones, desglosar conceptos y construir soluciones paso a paso. Esta práctica fortalece no solo el pensamiento lógico, sino también la creatividad y la capacidad de síntesis.

El propósito de este apartado es ofrecer ejemplos concretos de planeaciones didácticas que demuestran cómo la descomposición puede implementarse en diferentes niveles educativos. Desde actividades en preescolar hasta proyectos de mayor complejidad en la media superior, se presentarán casos prácticos que reflejan la aplicación de las cuatro fases del proceso de descomposición. Estos ejemplos buscan inspirar a los docentes a incorporar esta herramienta en su práctica diaria, facilitando el desarrollo integral de sus estudiantes y promoviendo un aprendizaje profundo y significativo.

La descomposición en nivel preescolar

El nivel preescolar representa el primer acercamiento formal de los niños a la resolución de problemas y al desarrollo de habilidades cognitivas fundamentales. Aunque a esta edad los estudiantes no se enfrentan directamente a conceptos complejos de programación o algoritmos, la descomposición se introduce de manera natural a través de juegos, actividades de clasificación y proyectos creativos. La capacidad de dividir tareas grandes en pequeñas acciones manejables es una habilidad que se construye a partir de experiencias prácticas y manipulativas, facilitando el desarrollo del pensamiento lógico y secuencial.

En este contexto, la descomposición se manifiesta en actividades cotidianas como armar rompecabezas, organizar juguetes por categorías o seguir secuencias para realizar tareas. La implementación de proyectos didácticos en preescolar permite que los estudiantes participen en procesos de exploración donde descomponen problemas visuales o físicos, como identificar partes de una planta, construir figuras con bloques o reconocer patrones en una canción. Estas actividades no solo estimulan la creatividad, sino que también fortalecen la capacidad de análisis y resolución de problemas de manera intuitiva y lúdica.

La Nueva Escuela Mexicana (NEM) enfatiza el aprendizaje basado en proyectos y experiencias significativas, lo que permite integrar la descomposición de manera transversal en diversas áreas del desarrollo infantil. A través de proyectos que involucran fases de lanzamiento, indagación, construcción y producto final, se fomenta que los niños participen activamente en su proceso de aprendizaje. A continuación, se presenta un ejemplo de planeación didáctica para un proyecto de cinco días, diseñado para explorar el ciclo de crecimiento de una planta, un contenido presente en el campo formativo de saberes y pensamiento científico. A continuación, se expone una planeación didáctica simplificada[21].

21. Todas las planeaciones didácticas que se expondrán serán simplificadas, es decir, no contendrán una profundización en algunas actividades didácticas, dado que el docente será el encargado de darle ese toque creativo y único a cada secuencia didáctica.

Planeación didáctica preescolar.

Título: “Descubrimos cómo crecen las plantas”
Este proyecto, dirigido a estudiantes de preescolar, tiene como objetivo explorar el crecimiento de las plantas, un contenido alineado con el campo formativo de Saberes y Pensamiento Científico dentro de la Nueva Escuela Mexicana. A lo largo de cinco días, los estudiantes participarán en actividades que les permitirán descomponer el proceso de crecimiento de una planta en fases concretas, favoreciendo la comprensión del entorno natural desde una perspectiva lúdica y sensorial.
Fase: Exploración y desarrollo del pensamiento científico. **Grado:** Preescolar. **Fecha de inicio:** 15 de febrero de 2025. **Fecha de cierre:** 19 de febrero de 2025.
Pregunta generadora: ¿Cómo crecen las plantas y qué necesitan para vivir?
Fase 1: Lanzamiento **Propósito:** Introducir el proyecto, estimular la curiosidad por las plantas y activar conocimientos previos sobre su crecimiento.
Actividades: Observación y exploración de plantas en el entorno escolar (jardín o macetas del aula). Realizar preguntas abiertas: ¿Qué partes tiene una planta? ¿Qué creen que necesita para crecer? Registro de observaciones mediante dibujos o comentarios grupales. **Producto parcial:** Mapa visual con dibujos y palabras que representan las partes de una planta y lo que necesita para crecer.
Fase 2: Indagación **Propósito:** Permitir que los estudiantes experimenten y observen el proceso de crecimiento de una planta desde su germinación.
Actividades: Plantar semillas de frijol en vasos con algodón y agua. Registro diario de los cambios observados. Discusión en grupo: ¿Qué parte creció primero? ¿Cómo sabemos que está creciendo? **Producto parcial:** Diario de crecimiento con dibujos y registros de las semillas germinadas.
Fase 3: Construcción **Propósito:** Crear representaciones físicas que reflejen el conocimiento adquirido sobre las partes de una planta.

Actividades:
Construcción de maquetas de plantas usando materiales reciclados (cartón, papel, plastilina).
Cada estudiante crea una parte de la planta (raíz, tallo, hojas, flores) y luego las ensamblan como grupo.
Reflexión grupal: ¿Qué pasa si falta alguna parte de la planta?
Producto parcial:
Maqueta colectiva de una planta con etiquetas que identifican cada parte.
Fase 4: Presentación Final
Propósito:
Mostrar el trabajo realizado y compartir lo aprendido con la comunidad escolar.
Actividades:
Presentación de las maquetas y explicación del ciclo de vida de una planta a otros grupos de **preescolar.**
Invitación a padres para observar las semillas germinadas y escuchar a los estudiantes explicar su proceso.
Producto Final:
Exhibición de maquetas y germinados en un espacio del aula con descripciones elaboradas por los estudiantes.
Evaluación:
Observación directa: Participación activa en las actividades de siembra, observación y creación de maquetas.
Portafolio de evidencias: Registro de dibujos, diarios de observación y fotografías de las semillas germinadas.
Rúbricas: Evaluación de la maqueta en base a creatividad, participación y explicación de las partes de la planta.

Elaboración propia.

La descomposición en esta planeación didáctica se manifiesta desde la fase de lanzamiento, donde los estudiantes observan las plantas y las dividen visualmente en partes reconocibles como hojas, tallo y raíces. Este primer acercamiento implica una descomposición inicial, guiada por la curiosidad natural de los niños, quienes identifican los componentes más evidentes a partir de la exploración directa. El rol del maestro en esta etapa es esencialmente de facilitador: promueve preguntas abiertas que dirigen la atención de los estudiantes hacia los detalles clave de las plantas, incentivando la observación y fomentando el descubrimiento autónomo.

Durante la fase de indagación, la descomposición se profundiza al dividir el proceso de crecimiento en pasos secuenciales (siembra, riego, germinación). Aquí, los estudiantes aprenden que el desarrollo de una planta no ocurre de forma inmediata, sino a través de etapas concretas que pueden observar y registrar. El docente asume el papel de guía, organizando la actividad de plantación y asegurando que los alumnos sigan cada paso, permitiendo la repetición

de procedimientos y promoviendo la reflexión diaria sobre los cambios observados. Este proceso fortalece la comprensión de que el crecimiento de una planta puede descomponerse y analizarse día a día.

En la fase de construcción, la descomposición toma una forma tangible al invitar a los estudiantes a crear una maqueta de planta. Cada grupo trabaja en una parte específica (raíces, hojas, flores), y el ensamblaje final refleja la integración de estas secciones. Aquí, el docente desempeña el rol de mediador, supervisando que cada estudiante entienda la función de la parte que le corresponde y apoyando el proceso de colaboración. La presentación final del proyecto simboliza la culminación del ciclo de descomposición, donde los estudiantes explican el crecimiento de la planta como una serie de eventos conectados, consolidando el aprendizaje a través de la verbalización y demostración visual.

Durante todo el proyecto, el docente adopta el rol de modelador de la lógica de formas, una de las cinco características clave en el desarrollo de la descomposición. Al guiar a los estudiantes en la construcción de la maqueta, el maestro no solo divide las partes de la planta, sino que también modela cómo cada componente debe ensamblarse en una estructura coherente. A través de ejemplos visuales y manipulativos, muestra cómo el orden y la disposición de cada pieza afectan el resultado final. Esta práctica permite a los niños comprender que, aunque cada parte tiene un propósito individual, su verdadera función se revela cuando todas se integran, reflejando la interdependencia de los elementos de la planta. El docente, al resaltar estas conexiones, enseña a los estudiantes a organizar, comparar y visualizar el problema de forma estructurada, facilitando así una comprensión más profunda del proceso de crecimiento.

La descomposición en nivel primaria

En el nivel de educación primaria, la descomposición evoluciona hacia una práctica más estructurada y reflexiva. A diferencia de preescolar, donde el aprendizaje se centra en la exploración sensorial, los estudiantes de primaria comienzan a analizar problemas de manera más deliberada, separando componentes para comprender cómo se relacionan entre sí. A lo largo de esta etapa, el docente juega un papel crucial como promotor del desanclaje conceptual, guiando a los alumnos para que rompan con soluciones repetitivas y

exploren nuevos enfoques. Esta capacidad de cuestionar y rediseñar estrategias permite que la descomposición se convierta en una herramienta aplicable a diferentes disciplinas, desde las ciencias hasta la escritura y las matemáticas.

El aprendizaje basado en proyectos brinda un escenario ideal para que los estudiantes de primaria experimenten la descomposición en cada fase del desarrollo de una actividad. Al dividir un problema mayor en tareas más pequeñas y manejables, los estudiantes aprenden a organizarse, identificar patrones y colaborar en la búsqueda de soluciones. Durante este proceso, el docente asume el rol de facilitador de la desestabilización, introduciendo desafíos que empujan a los alumnos a reorganizar sus ideas y abordar las tareas de manera crítica. Esta estrategia fomenta el desarrollo de habilidades cognitivas superiores, impulsando a los estudiantes a salir de su zona de confort y buscar respuestas de manera activa.

El proyecto que se presenta a continuación se centra en el análisis y clasificación de los animales según sus características físicas y su hábitat. Este contenido, alineado con el campo formativo de Saberes y Pensamiento Científico de la NEM, permite a los estudiantes descomponer conceptos biológicos de forma gradual, facilitando su comprensión. A través de la observación, investigación y creación de materiales, los alumnos no solo refuerzan sus conocimientos sobre el reino animal, sino que también desarrollan habilidades de organización, clasificación y síntesis, aplicando la descomposición en cada fase del proyecto.

Planeación Didáctica: “Clasificamos y Aprendemos sobre los Animales”
Este proyecto, dirigido a estudiantes de 3º y 4º de primaria, tiene como objetivo que los alumnos identifiquen y clasifiquen animales según sus hábitats y características. El proyecto se desarrolla en cinco días y está alineado con el campo formativo de Saberes y Pensamiento Científico dentro de la Nueva Escuela Mexicana. A través de la observación, indagación y creación de materiales, los estudiantes aplicarán el proceso de descomposición, facilitando la comprensión del mundo natural de manera lúdica y colaborativa.
Fase: Exploración y clasificación de animales. **Grado:** 3º y 4º de primaria. **Fecha de inicio:** 11 de marzo de 2025. **Fecha de cierre:** 15 de marzo de 2025.

Pregunta generadora: ¿Cómo podemos clasificar a los animales y por qué viven en diferentes lugares?
Fase 1: Lanzamiento **Propósito:** Introducir el proyecto y activar conocimientos previos sobre animales y su hábitat.
Actividades: Observación de videos e imágenes de animales de distintos hábitats. Discusión grupal: ¿Qué diferencias observan entre los animales? ¿Dónde creen que viven? Creación de un mapa de ideas con categorías iniciales: animales terrestres, acuáticos y aéreos. **Producto parcial:** Mapa visual con categorías iniciales de animales según su hábitat.
Fase 2: Indagación **Propósito:** Investigar y clasificar animales según sus características físicas y tipo de alimentación.
Actividades: Búsqueda de información en libros y recursos digitales sobre tipos de animales (mamíferos, reptiles, aves, peces). Creación de listas y diagramas de Venn para clasificar animales según su alimentación (herbívoros, carívoros, omnívoros). Registro de características comunes y diferencias en sus cuadernos de trabajo. **Producto parcial:** Diagramas y listas de animales clasificados por hábitat y alimentación.
Fase 3: Construcción **Propósito:** Elaborar un mural que represente hábitats con animales clasificados por los estudiantes.
Actividades: División del grupo en equipos para trabajar en diferentes hábitats (selva, océano, desierto). Creación de dibujos y recortes que representen animales para cada hábitat. Ensamblaje de un mural colectivo con etiquetas que expliquen la clasificación de cada animal. **Producto parcial:** Mural colaborativo con animales clasificados según su hábitat.
Fase 4: Presentación Final **Propósito:** Exponer el mural y explicar el proceso de clasificación a otros compañeros y padres de familia.

Actividades:
Presentación grupal del mural a otros salones de la escuela.
Explicación por parte de los estudiantes sobre por qué cada animal vive en un hábitat específico.
Reflexión final sobre lo aprendido: ¿Cómo afecta el hábitat a la vida de los animales?
Producto Final:
Exposición del mural ante la comunidad escolar.
Evaluación:
Observación directa: Participación en la discusión y creación del mural.
Portafolio de evidencias: Diagramas, listas y fotografías del mural terminado.
Rúbricas: Evaluación del mural con base en creatividad, organización y explicación de los animales.

La descomposición en esta planeación se hace evidente desde la fase de lanzamiento, donde los estudiantes comienzan a clasificar animales en categorías generales (terrestres, acuáticos y aéreos). Esta primera aproximación representa una descomposición inicial, en la que los alumnos, a partir de la observación de videos e imágenes, identifican patrones básicos y dividen los animales en grupos según sus hábitats. El maestro, como facilitador de la desestabilización, introduce preguntas abiertas que empujan a los estudiantes a reflexionar y cuestionar lo que saben, provocando que comiencen a descomponer mentalmente los elementos visuales para crear sus primeras clasificaciones.

Durante la fase de indagación, la descomposición evoluciona hacia un nivel más profundo y repetitivo. A medida que los estudiantes investigan y analizan diferentes tipos de animales, desglosan sus características físicas, hábitos alimenticios y hábitats de manera continua. La repetición de este proceso a través de diagramas y listas permite consolidar categorías, fortaleciendo su comprensión del concepto de clasificación. Aquí, el maestro asume el rol de promotor del desanclaje conceptual, alentando a los alumnos a ir más allá de las primeras categorías, desafiándolos a encontrar nuevas formas de clasificar y explorar más a fondo las diferencias entre especies.

En la fase de construcción, la descomposición se convierte en una práctica selectiva y tangible. Al crear partes del mural que representan distintos hábitats, los estudiantes descomponen el proyecto en secciones específicas (selva, océano, desierto) y luego ensamblan las piezas en un todo unificado. El maestro se transforma en un modelador de la lógica de formas, guiando el proceso

de construcción visual y asegurando que cada elemento del mural esté bien ubicado y cumpla su función dentro del conjunto. Esta etapa no solo refuerza la comprensión del tema, sino que también permite a los estudiantes visualizar cómo las partes individuales contribuyen al producto final, consolidando el aprendizaje a través de la descomposición colaborativa.

La descomposición en nivel secundaria

En secundaria, la descomposición adquiere una dimensión más analítica, donde los estudiantes no solo separan elementos, sino que también deben anticipar cómo cada parte influye en el sistema completo. El enfoque se centra en proyectos que integran conocimientos de diferentes áreas, lo que exige que los alumnos organicen sus ideas, planifiquen por fases y evalúen continuamente sus avances. El docente, en esta etapa, desempeña un papel más reflexivo y crítico, impulsando a los estudiantes a cuestionar sus decisiones y considerar múltiples enfoques. Esta dinámica promueve un aprendizaje más profundo y significativo, ya que cada paso del proceso refuerza la comprensión global del problema.

A medida que avanza el proyecto, la descomposición evoluciona hacia una práctica de prueba y error estructurada. Los estudiantes diseñan, construyen y ajustan componentes, identificando patrones y resolviendo problemas emergentes. El maestro guía este proceso, interviniendo para señalar inconsistencias o plantear desafíos adicionales que obliguen a los alumnos a replantear sus estrategias. Más que ofrecer soluciones directas, el docente impulsa el pensamiento crítico a través de preguntas que estimulan la revisión y mejora del trabajo realizado, consolidando así un ciclo continuo de aprendizaje.

Al concluir el proyecto, la descomposición alcanza un nivel abstracto donde los estudiantes explican y presentan sus resultados de manera integral. En este punto, ya no dependen de diagramas o guías visuales; en cambio, articulan el funcionamiento del sistema de riego a partir de su experiencia práctica. La capacidad de explicar cada componente y su interacción refleja el dominio del proceso de descomposición. El docente evalúa no solo el producto final, sino también la claridad con la que los estudiantes comunican su trabajo, reconociendo la importancia del recorrido y el crecimiento intelectual que este proceso fomenta.

Planeación Didáctica: “Diseñamos un Sistema de Riego Automatizado”
Este proyecto, dirigido a estudiantes de 2º y 3º de secundaria, tiene como objetivo que los alumnos diseñen y construyan un sistema de riego automatizado utilizando sensores de humedad y materiales reciclados. A través de este proyecto, los estudiantes aplicarán la descomposición en cada fase, facilitando la comprensión de los sistemas tecnológicos de manera práctica e interdisciplinaria. El proyecto está alineado con el campo formativo de Saberes y Pensamiento Científico dentro de la Nueva Escuela Mexicana.
Fase: Innovación tecnológica y desarrollo de sistemas. **Grado:** 2º y 3º de secundaria. **Fecha de inicio:** 8 de abril de 2025. **Fecha de cierre:** 12 de abril de 2025.
Pregunta generadora: ¿Cómo podemos automatizar el riego de las plantas para ahorrar agua y tiempo?
Fase 1: Lanzamiento **Propósito:** Introducir el proyecto, activar conocimientos previos sobre sistemas de riego y explorar soluciones tecnológicas.
Actividades: Observación de videos sobre sistemas de riego manuales y automáticos. Discusión grupal: ¿Qué ventajas tiene automatizar un sistema de riego? Lluvia de ideas sobre los componentes necesarios para construir un sistema automatizado. **Producto parcial:** Mapa conceptual con ideas y componentes identificados para el diseño del sistema.
Fase 2: Indagación **Propósito:** Investigar y analizar los elementos necesarios para construir un sistema de riego automatizado.
Actividades: Investigación en recursos digitales y libros sobre sensores de humedad, tuberías y sistemas automatizados. Creación de diagramas de flujo que representen el funcionamiento del sistema. Elaboración de bosquejos individuales sobre posibles diseños de riego. **Producto parcial:** Diagramas y bosquejos con los elementos desglosados del sistema de riego.
Fase 3: Construcción **Propósito:** Construir un sistema de riego automatizado utilizando materiales reciclados y componentes electrónicos.

Actividades:
División del trabajo en equipos: instalación de sensores, montaje de tuberías y ensamblaje del sistema de riego.
Pruebas del sistema para identificar errores y realizar ajustes.
Registro de los resultados y modificación del diseño si es necesario.
Producto parcial:
Sistema de riego automatizado funcional.
Fase 4: Presentación Final
Propósito:
Presentar el sistema de riego y explicar su funcionamiento a la comunidad escolar.
Actividades:
Presentación grupal del sistema ante otros estudiantes y docentes.
Demostración en vivo del sistema de riego en un área verde de la escuela.
Reflexión sobre los aprendizajes adquiridos y propuestas de mejora del sistema.
Producto Final:
Presentación y prueba del sistema de riego automatizado.
Evaluación:
Observación directa: Participación activa durante la investigación y construcción.
Portafolio de evidencias: Diagramas, bosquejos y fotografías del sistema en funcionamiento.
Rúbricas: Evaluación del diseño y la explicación del sistema de riego.

La descomposición en esta planeación se manifiesta en cada fase del proyecto, permitiendo que los estudiantes adquieran una comprensión progresiva del sistema de riego automatizado. Desde el inicio, la identificación de componentes esenciales (sensores, tuberías y válvulas) actúa como punto de partida para fragmentar el problema en elementos más simples. A medida que los alumnos avanzan, el enfoque cambia hacia comprender las interacciones entre las partes, promoviendo una visión integral que conecta la teoría con la práctica. Esta aproximación no solo facilita la construcción del sistema, sino que también desarrolla habilidades analíticas que se extienden a otras áreas del conocimiento.

El papel del docente es clave durante todo el proceso, asegurando que los estudiantes no solo ejecuten tareas mecánicamente, sino que también reflexionen sobre cada decisión tomada. Al intervenir en momentos críticos, el maestro promueve la autoevaluación y la reformulación de ideas, consolidando una metodología de aprendizaje basada en el ciclo de prueba, error y

ajuste. Esta dinámica impulsa a los alumnos a reconocer patrones, identificar errores y optimizar sus soluciones, consolidando así una mentalidad resiliente y proactiva.

Más allá del producto final, el verdadero valor del proyecto radica en la capacidad de los estudiantes para explicar y justificar el proceso de creación del sistema de riego. Al articular cómo y por qué se seleccionaron ciertos componentes y diseños, los alumnos demuestran una comprensión abstracta del problema, reflejando su dominio de la descomposición. Esta habilidad, fomentada por el docente a lo largo de cada fase, prepara a los estudiantes para abordar proyectos futuros con una mentalidad analítica y creativa.

El proceso de descomposición en la educación básica trasciende el desarrollo de habilidades técnicas; representa una vía para moldear el pensamiento crítico, fortalecer la autonomía y preparar a los estudiantes para abordar problemas complejos en cualquier ámbito. A medida que los alumnos avanzan de preescolar a niveles superiores, la descomposición se convierte en una herramienta de análisis que no solo desglosa problemas, sino que también revela patrones, fomenta la creatividad y permite construir soluciones integrales. Las planeaciones didácticas aquí expuestas reflejan cómo esta habilidad se adapta y crece con el estudiante, consolidando una base sólida para el aprendizaje a lo largo de su trayectoria académica.

Más allá de los resultados tangibles, el verdadero impacto de enseñar descomposición radica en la transformación del aula en un espacio de exploración activa. Los estudiantes dejan de ser receptores de información para convertirse en constructores de conocimiento, participando en proyectos donde cada decisión cuenta y donde el error se convierte en una herramienta de aprendizaje. El docente, al facilitar este proceso, actúa como mediador de experiencias, guiando a los alumnos hacia una mayor comprensión sin imponer respuestas, permitiéndoles descubrir sus propias rutas y soluciones.

La importancia de integrar la descomposición en el currículo no solo responde a una necesidad académica, sino a un cambio en la forma en que concebimos el aprendizaje. En un mundo que demanda flexibilidad y resolución de problemas constantes, enseñar a los estudiantes a fragmentar, analizar y reconstruir ideas les proporciona una ventaja crucial. Este enfoque les permite no solo destacar en asignaturas específicas, sino también desarrollar una

mentalidad adaptable, capaz de enfrentar los retos cambiantes de su entorno personal, académico y profesional.

El cierre de este capítulo no marca el final de una propuesta pedagógica, sino el inicio de una práctica continua que evoluciona con cada generación de estudiantes.

La descomposición, aplicada de manera transversal, permite que el aula se convierta en un laboratorio de pensamiento, donde la curiosidad y la lógica se combinan para formar individuos que no temen a los desafíos, sino que los descomponen, los analizan y los resuelven con ingenio y perseverancia. Este es el verdadero valor de integrar la descomposición en la educación: preparar mentes que, desde la infancia hasta la vida adulta, aprendan a ver en cada problema una oportunidad para crecer.

6. Conclusiones

A lo largo de esta obra, se ha puesto de manifiesto que el PD no solamente se trata de un proceso clave en el desarrollo del PC, sino que se han cuestionado sus fundamentos, tipología y estrategias, permitiendo así trasladarlo a contexto no computacionales, en donde toma el papel de herramienta transversal en el diseño de actividades y estrategias didácticas en educación básica.

Como un marco conceptual sólido que aborda una brecha identificada en la comprensión de los procesos cognitivos en contextos computacionales, a través de un meticuloso proceso de análisis teórico y conceptual, se logró la articulación de un conjunto de principios fundamentales que delinean no sólo las etapas, sino también los mecanismos subyacentes en la descomposición. Además, dichos elementos teóricos no sólo tendrán repercusión en el ámbito educativo, sino que también tiene aplicaciones potenciales en una amplia gama de campos, tales como la informática, inteligencia artificial, psicología cognitiva e incluso la neurociencia.

La construcción de los supuestos teóricos propuestos a lo largo de los capítulos anteriores, se trató de un proceso iterativo y colaborativo que ha involucrado la retroalimentación y la contribución de diversos expertos y profesionales en campos relacionados. Esta diversidad de perspectivas ha enriquecido el desarrollo de la teoría y ha garantizado su relevancia y aplicabilidad en una variedad de contextos educativos y profesionales. En este sentido, se ofreció un marco teórico que pretenda abordar los desafíos contemporáneos en la enseñanza y el aprendizaje de habilidades computaciones y de resolución de problemas, posee el potencial para convertirse en un futuro en una teoría integral y coherente que abra nuevas posibilidades para la investigación y la práctica educativa, además de que promueva el desarrollo de individuos

más capacitados y adaptados a los entornos digitales y tecnológicos del siglo XXI.

Justamente en esta idea de los entornos digitales y tecnológicos, el PD emerge como una contribución significativa al campo de la educación y la tecnología al abordar las necesidades contemporáneas de enseñanza y aprendizaje en un mundo digitalmente avanzado. Estas pautas teóricas ofrece una serie de contribuciones que pueden transformar la forma en que se diseñan y se implementan los entornos educativos, tanto en el aula como de manera virtual.

Los supuestos teóricos propuestos representan una valiosa contribución al campo de la educación y la tecnología al ofrecer un marco teórico y práctico para enfrentar los desafíos presentes en la enseñanza y el aprendizaje. Su capacidad para guiar el diseño de estrategias pedagógicas, herramientas educativas y entornos de aprendizaje en línea la convierte en un recurso invaluable para educadores, diseñadores de tecnología educativa y formuladores de políticas educativas en todo el mundo.

En este sentido, el uso de la tecnología en la educación ha tenido esfuerzos considerables, desde la implementación de dispositivos portátiles, tabletas, computadoras, videoproyectores, pizarrones digitales, hasta llegar a lo que hoy conocemos como robots educativos, que son artefactos diseñados específicamente para que los estudiantes puedan manipularlos y aprender a partir de sus componentes y programación. Sin embargo, esta obra demuestra que, no solamente se trata de incorporarlos, por más nuevo que sea, incluso sin importar las características didácticas que pudiese ofrecer.

El *Mbot* que fue descrito y analizado técnica, teórica y didácticamente, se trata de un robot que bien brindó los elementos de entrada en la llamada caja negra, sin embargo, en diversos eventos educativos, congresos sobre robótica educativa, incluso dentro de las mismas aulas de educación básica en México, se trata de un elemento desvinculado con los procesos centrales del PC, en la mayoría de las ocasiones, no se encuentra relacionado a ninguna metodología o didáctica en específico, sino en una exploración pedagógica.

El *Mbot* que fue descrito y analizado técnica, teórica y didácticamente, se trata de un robot que bien brindó los elementos de entrada en la llamada caja negra, sin embargo, en diversos eventos educativos, congresos sobre robótica

educativa, incluso dentro de las mismas aulas de educación básica en México, se trata de un elemento desvinculado con los procesos centrales del PC, en la mayoría de las ocasiones, no se encuentra relacionado a ninguna metodología o didáctica en específico, sino en una exploración pedagógica.

Por ejemplo, en los proyectos de Recrea Jalisco, utilizan los *Mbot* como objetos lúdicos que buscan captar la atención de niños de entre 6 a 11 años de edad, desperdiciándose así las potencialidades de este robot. En la presente obra el robot no se trata del elemento inamovible del objeto de estudio, sino que se defiende la idea que todo elemento tecnológico y robótico debe de ser estudiado a profundidad para después llevarlo a la práctica con mayor seguridad didáctica, de tal forma, se deja de lado la improvisación y se traza una ruta pedagógica que permita a los docentes aplicarlo con mayor sentido.

A la luz de los conceptos teóricos expuestos en este documento, es posible observar a un robot educativo desde una perspectiva única, que ayude a los docentes a desarrollar un PC que pueda trasladarse a diversos contextos, es decir, que no solamente se trata de un robot, sino que dicho recurso tecnológico, sin importar cuál sea, incluso si es de los más básicos, como los *Bee-Bot*, puedan significar una oportunidad para desarrollar pensamiento lógico, algorítmico, automatizado, crítico y computacional.

No se trata de sólo de adquirir robots, ni siquiera de enseñar a programar sólo porque se encuentre dentro de los programas de vanguardia y propuestas pedagógicas contemporáneas que buscan el impulso mercadotécnico de recursos de grandes empresas, disfrazándolo de preocupación por "un futuro cada vez más tecnológico", el sentido de esta obra, es también concientizar acerca de que la enseñanza de la programación tiene como propósito desarrollar habilidades que puedan ser utilizadas en otras situaciones problemáticas, y no sólo matemáticas.

Por lo tanto, el potencial de los supuestos teóricos expuestos es extenso, porque la tipología de descomposición, sus fases, entre otros aspectos que se mencionaron, no son indispensables de un recurso tecnológico en particular, lo que permitirá, al menos en un futuro, seguir con investigaciones aplicadas con otro tipo de robots. Similar a lo que sucedió con Papert y LOGO, en donde su teoría construccionista no se quedó atada a un tipo de tecnología, sino que

aún en la actualidad, pueden aplicarse sus supuestos teóricos a robots que no impliquen la enseñanza de la geometría o de las matemáticas.

Además de esto, el PD permitirá comprender con mayor profundidad los otros procesos centrales propuestos en el PC, dado que seguramente muchos elementos teóricos podrán relacionarse con el pensamiento algorítmico, la evaluación, la automatización, entre otros. Será tarea de investigaciones futuras el hecho de comparar y relacionar cada uno de estos procesos que hasta el momento siguen teniendo vacíos en el conocimiento.

Las implicaciones para la investigación futura derivadas del PD son amplias y prometer abrir nuevas áreas de estudio y desarrollo dentro del campo de la educación y el PC. Una de las áreas clave es la validación empírica, es decir, realizar investigaciones empíricas rigurosas para validar su efectividad en entornos educativos distintos, esto podría implicar estudios longitudinales que examinen el impacto a largo plazo de la aplicación de los supuestos teóricos en cuestión, para el desarrollo de habilidades cognitivas y mejora del rendimiento académico.

Sin lugar a dudas, éste sería uno de los elementos más importantes que se debería concluir en un futuro, es decir, falta un camino largo por recorrer hacia la consolidación como teoría de la propuesta del presente estudio, principalmente porque la crítica principal será la falta de universalización de los resultados, ya que, al tratarse de un estudio cualitativo, con una profundidad específica, tiende a necesitar mayor número de pruebas.

El PD, expuesto en esta obra, se consolida no solo como una herramienta metodológica clave para el desarrollo del PC, sino como un puente hacia propuestas pedagógicas innovadoras que integran la tecnología con una perspectiva crítica y reflexiva. El diseño de una propuesta pedagógica concreta, acompañada de tres planeaciones didácticas, representa un valor añadido significativo. Esta propuesta no sólo aterriza los conceptos teóricos en el aula, sino que proporciona a los docentes una ruta estructurada para aplicar la descomposición en diversos niveles educativos, lo que amplía las posibilidades de formar estudiantes con habilidades analíticas y creativas transferibles a múltiples contextos.

A través de estas planeaciones, se demuestra que el PD no es una abstracción limitada al ámbito de la programación, sino un recurso transversal que,

adecuadamente guiado, facilita la resolución de problemas complejos en distintas disciplinas. Al ofrecer ejemplos prácticos y secuencias didácticas concretas, se facilita que los docentes puedan replicar, adaptar y expandir esta propuesta en función de las necesidades y características de sus estudiantes. Esta dimensión práctica, fundamentada en una sólida base teórica, posiciona a la obra como una contribución valiosa no solo para investigadores y teóricos del PC, sino también para educadores que buscan innovar en sus prácticas pedagógicas y transformar el aprendizaje en un proceso dinámico, significativo y contextualizado.

7. Referencias

Adell, J. S., Llopis, M. A. N., Esteve, M. F. M., y Valdeolivas, N. M. G. (2019). El debate sobre el pensamiento computacional en educación. RIED. *Revista Iberoamericana de Educación a Distancia*, 22(1), 171-186. https://doi.org/10.5944/ried.22.1.22303

Clark, A., Chalmers, D. (1998). The Extended Mind. *Source: Analysis,* 7-19.

Aguilar, M. (2012). Aprendizaje y Tecnología de Información y Comunicación: Hacia nuevos escenarios educativos. *Revista Latinoamericana de Ciencias Sociales, Niñez y juventud*. 10(2), 801-811.

Aguirre Chávez, F. (2016). De la situación problemática al problema científico educacional. *Revista EDUCA UMCH*, (07), 143-151. https://doi.org/10.35756/educaumch.201607.60

Anderson, N. (2016). A Call for Computational Thinking in Undergraduate Psychology. *Psichology Learning and Teaching*, 15(3), (pp. 226-234). DOI: doi:10.1177/1475725716659252

Apple, M. (1986). Ideología y currículo. Akal. Madrid.

Barr, V., y Stephenson, C. (2011). Bringing computational thinking to K-12: What is involved and what is the role of the computer science education community? ACM Inroads, 2(1), 48-54. doi: https://doi.org/10.1145/1929887.1929905

Barrera, L. N. (2015). Uso de la robótica educativa como estrategia didáctica en el aula. *Revista de Investigación y Pedagogía*, 6 (11), (pp. 215-234). http://revistas.uptc.edu.co/index.php/praxis_saber/article/viewFile/3582/3539

Bocconi, S., Chioccariello, A., Dettori, G., Ferrari, A., y Engelhardt, K. (2016). Developing computational thinking in compulsory education implications for policy and practice. Sevilla: *Joint Research Centre*. doi: http://doi. org/10.2791/792158

Brennan, K. & Resnick, M. (2012). New frameworks for studying and assessing the development of computational thinking. Proceedings of the 2012 Annual

Meeting of the American Educational Research Association. Recuperado el 25 de mayo de 2020 en: http://scratched.gse.harvard.edu/ct/files/AERA2012.pdf

Bruner, J. (1985). Vygostky: A historical and conceptual perspective. Culture, Communication, and Cognition: Vygostkian Perspectives, 21-34.

Caballero González, Y., & García-Valcárcel Muñoz-Repiso, A. (2021). Robots en la educación de la primera infancia: aprender a secuenciar acciones usando robots programables. *RIED. Revista Iberoamericana de Educación a Distancia, 24*(1), 77-94. doi:https://doi.org/10.5944/ried.24.1.27508

Cárdenas, P. M. C. (2017). *Pensamiento computacional.* Educación básica. Secretaría de Educación Pública (SEP).

Carretero, S., Vuorikari, R., y Punie, Y. (2017). DigComp 2.1: The Digital Competence Framework for Citizens with eight proficiency levels and examples of use http://doi.org/10.2760/38842

Chalmers, C. (2018). Robotics and computational thinking in primary school*. International Journal of Child-Computer Interaction*, (17), (pp. 93-100). DOI: 10.1016/j.ijcci.2018.06.005

Cheng, G. (2018). Exploring factors influencing the acceptance of visual programming environment among boys and girls in primary schools. *Computers in Human Behavior*, (92), (pp. 361-372). DOI: 10.1016/j.chb.2018.11.043

Cheng, Z. J. (2012). Teaching young children decomposition strategies to solve addition problems: An experimental study. *The Journal of Mathematical Behavior*, *31*(1), 29-47.

Cobo, R. C. y Movareck, J. W. (2011). *Aprendizaje invisible. Hacia una nueva ecología de la educación.* Colección Transmedia XXI. Edicions de la Universitat de Barcelona. Barcelona.

Constante, P. Chimbo, C. Jiménez, V. & Gordón A. (2019). Investigación, diseño e implementación de un sistema de realidad aumentada con asistente robótico para el mejoramiento del aprendizaje, creatividad y entretenimiento para niños de educación primaria. *Revista ibérica de sistemas y tecnologías de la información*, 20 (5), (pp. 566-577). http://repositorio.espe.edu.ec/handle/21000/13448

Corradini, I., Lodi, M., y Nardelli, E. (2017). Conceptions and misconceptions about computational thinking among Italian primary school teachers. En *Proceedings of the 2017 ACM Conference on International Computing Education Research* (pp. 136-144). ACM.

Creswell, J. W. (2014). *Research design: qualitative, quantitative, and mixed methods approaches.* United States of America: SAGE Publications Inc.

Csizmadia, A., Curzon, P., Dorling, M., Humphreys, S., Ng, T., Selby, C. & Woollard, J. (2015). Computational thinking. A guide for teachers. *Computing at School*. Recuperado el 25 de mayo de 2020 en: http://community.Computingatschool.org.uk/resources/2324

Denning, P. J. (2017). Remaining trouble spots with computational thinking. Communications of the ACM, 60(6), 33-39. doi: https://doi.org/10.1145/2998438

Di Bitetti, M. S., & Ferreras, J. A. (2017). Publish (in English) or perish: The effect on citation rate of using languages other than English in scientific publications. *Ambio*, *46*(1), 121-127.

Di Lieto, M. C., Inguaggiato, E., Castro, E., Cecchi, F., Cioni, Giovanni., Dell'Omo, M., Laschi, C., Pecini, C., Santerini, G., Sgandurra, G. & Dario, P. (2017). Educational Robotics Intervention on Executive Functions in preschool children: a pilot study. *Computers in Human Behavior*. (71), (pp. 16-23). DOI: 10.1016/j.chb.2017.01.018

Erol, O. & Kurt, A. A. (2017). The effects of Teaching Programming with Scratch on Pre-Service Information Technology Teachers´ Motivation and Achievement. *Computers in Human Behavior*, 77, (pp. 11-18). DOI: 10.1016/j.chb.2017.08.017

Flick, U. (2015). *El diseño de la investigación cualitativa* (Vol. 1). Ediciones Morata.

Garnica, E. (2014). Robots Herramientas para las Aulas de Clase. *Revista ingeniería, matemáticas y ciencias de la información*, 1 (1), (pp. 33-45). http://www.doc4net.es/cdn/60/57/605701288577/1ZYYNYOM2WwjU0JTOyjTTAUiJz-T2kkE/84bc8f0c.pdf

Glaser, B. y Strauss, A., 1967. *The Discovery of Grounded Theory: Strategies for QualitativeResearch*. Primera Edición. Chicago: Aldine Publishing Company.

González-Martínez, J., Estebanell, M. M. & Peracaula, B. M. (2018). *¿Robots o programación? El concepto de Pensamiento Computacional y los futuros maestros.* Ediciones Universidad de Salamanca EKS, 19(2), (pp. 29-45). DOI: 10.14201/eks20181922945

González, M. R. (2016). *Codigo alfabetización y pensamiento computacional en educación primaria y secundaria: validación de un instrumento y evaluación de programas.* Tesis Doctoral. España: Escuela Internacional de Doctorado.

Grover, S. (2009). Computer science is not just for big kids. *Learning & Leading with technology*, 37(3), (pp. 27-29).

Grover, S., y Pea, R. (2013). Computational Thinking in K-12: A Review of the State of the Field. *Educational Researcher,* 42(1), 38-43. doi: https://doi.org/10.3102/0013189X12463051

Hernández, M. (2020). *Publicar en inglés o morir. La visión del Investigador académico latinoamericano*. México: Universidad de Guadalajara

International Society for Technology in Education (ISTE) y Computer Science Teachers Association (CSTA). (2011). Operational Definition of Computational Thinking for K–12 Education. Recuperado de http://www.iste.org/docs/ct-documents/computationalthinking-operational-definition-flyer.pdf?sfvrsn=2

Ioannou, A., y Makridou, E. (2018). Exploring the potentials of educational robotics in the development of computational thinking: A summary of current research and practical proposal for future work. Education and Information Technologies. doi: https://doi.org/10.1007/s10639-018-9729-z

Jiménez, B. A. y Torres, C. A. (2006). *La práctica investigativa en ciencias sociales*. Colombia: Universidad Pedagógica Nacional.

Jiménez, B. J. A., Ramírez, P. J. F. & González, E. J. J. (2011). Sistema modular de robótica colaborativa aplicado en educación*. Revista Facultad de Ingenieria*, 58(58), (pp. 163-172).

Lanuez, M., Martínez, M. y Pérez, V. (2010). *El maestro y la investigación educativa en el siglo XXI.* La Habana: Pueblo y educación.

Lewis, C. M. (2010, March). How programming environment shapes perception, learning and goals: logo vs. scratch. In *Proceedings of the 41st ACM technical symposium on Computer science education* (pp. 346-350).

Lora, A. A. A., Cavadias, L. M., & Miranda, A. J. M. (2017). La teoría fundamentada en el marco de la investigación educativa. *Saber, ciencia y libertad*, *12*(1), 236-245.

Martín-Barbero, J. (2009). Cuando la tecnología deja de ser una ayuda didáctica para convertirse en mediación cultural. *Revista electrónica Teoría de la Educación*. 10(1), 19-31.

Meerbaum-Salant, O., Armoni, M. & Ben-Ari, M. (2011). Habits of programming in scratch*. Proceedings of the 16th annual joint conference on Innovation and technology in computer science education. ACM*, (pp. 168-172).

Miranda, P. M. S. (2019). Programación y robótica en educación infantil: estudio multi caso en Portugal. *Revista Prisma Social*, (25), (pp. 248-276).

Morozov, E. (2013). *To save everything, click here: The folly of technological solutionism*. Public Affairs.

Mouhaffel, A. G. (2018). Comparación de la enseñanza dos sistemas de programación robótica enfocada a los recursos matemáticos: Arduino+Scratch y Sistema Lego EV3 TT - Assessment Comparative of teaching by two ways for programming robotic based in mathematical resources: Arduino. *International Journal of Innovation and Applied Studies*, *25*(1), (pp. 15-39).

Muñoz-Repiso, A. G. V. & Caballero-González, Y. A. (2019). Robotics to develop computational thinking in early Childhood Education. *Comunicar*, 27(59).

Nishida, T., Kanemune, S., Idosaka, Y., Namiki, M., Bell, T., & Kuno, Y. (2009). A CS unplugged design pattern. *ACM SIGCSE Bulletin*, *41*(1), 231-235.

Nope, S., Loaiza, H. & Caicedo, E. (2011). Programación de un robot bajo el paradigma del aprendizaje por demostración. *Revista Facultad de Ingenieria*, 58(58), (pp. 142-152).

Ortiz, L. C. C., Jiménez, M. M. V., Puerta, J. J. M., & Posada, J. A. T. (2019). Herramienta de robótica educativa basada en Lego Mindstorms y VEX Robotics mediante software 3D y diseño mecatrónico. *RISTI-Revista Ibérica de Sistemas e Tecnologias de Informação*, (34), 1-19.

Palma, S. C. & Sarmiento, P. R. (2015). Estado del arte sobre experiencias de enseñanza de programación a niños y jóvenes para el mejoramiento de las competencias matemáticas en primaria. *Revista Mexicana de Investigación Educativa*, 20(65), (pp. 607-641). http://www.scielo.org.mx/scielo.php?script=sci_arttext&pid=S140566662015000200013&lng=es&nrm=iso&tlng=es

Pandit, N.R., 1996. *The Creation of Theory: A Recent Application of the Grounded. Theory Method.* University of Manchester.

Papert, S. (1980). *Mindstorms: Children, computers and powerful ideas*. Nueva York: Basic Books Inc.

— (1987). Information technology and education: Computer criticism vs. technocentric thinking. *Educational researcher*, *16*(1), 22-30.

— (1993). *The Children´s Machine.* Nueva York: Basic Books Inc.

Papert, S., & Harel, I. (1991). Situating constructionism. *Constructionism*, *36*(2), 1-11.

Pérez-Marín, D., Hijón-Neira, R., Bacelo, A. & Pizarro, C. (2018). Can computational thinking be improved by using a methodology based on metaphors and Scratch to teach computer computer programming to children?. *Computers in Humar Behavior*, (105). DOI: 10.1016/j.chb.2018.12.027

Piaget, J. (1971). Biology and knowledge: An essay on the relations between organic regulations and cognitive processes. Chicago: University od Chicago Press.

— (1977). *The development of thought: Equilibration of cognitive structures*. Viking Press.

Ramírez, G. B. y Anzaldúa A. R. E. (2014). Subjetividad y socialización en la era digital. *Argumentos*, 27(76), 171-189.

Recalde España, E., Serna Agudelo, B. N., Beltrán, G. A., & Cañón Recalde, C. A. (2018). El Scratch como estrategia didáctica para desarrollar la exploración del medio en la educación inicial-Fase I y II.

Rich, K. M., Binkowski, T. A., Strickland, C. & Franklin, D. (2018). Decomposition: A K-8 Computational Thinking Learning Trajectory. *Proceedings of the 2018 ACM Conference on International Computing Education Research*, 18, (pp. 124-132).

Rich, P. J., Egan, G., & Ellsworth, J. (2019). A Framework for Decomposition in Computational Thinking. In *Proceedings of the 2019 ACM Conference on Innovation and Technology in Computer Science Education* (pp. 416-421).

Rich, P. J., y Langton, M. B. (2016). Computational Thinking: Toward a Unifying Definition. En J. M. Spector, D. Ifenthaler, D. G. Sampson, & P. Isaias (Eds.), Competencies in Teaching, Learning and Educational Leadership in the Digital Age (pp. 229-242). Cham: Springer International Publishing. doi: https://doi.org/10.1007/978-3-319-30295-9_14

Rijke, W. J., Bollen, L., Eysink, T. H., & Tolboom, J. L. (2018). Computational Thinking in Primary School: An Examination of Abstraction and Decom-position in Different Age Groups. *Informatics in education*, *17*(1), 77.

Sáez-López. J. M., Sevillano-García, M. L. & Vazquez-Cano, E. (2019). The effect of programming on primary school students´ mathematical and scientific understanding: educational use of mBot. *Association for Educational Communications and Technology*, 67, (pp. 1405-1425). DOI: 10.1007/s11423-019-09648-5

Scaradozzi, D., Sorbi, L., Pedale, A., Valzano, M. & Vergine, C. (2015). Teaching robotics at the primary school: an innovative approach. *Procedia-Social and Behavioral Sciences,* (174), (pp. 3838-3846). DOI: 10.1016/j.sbspro.2015.01.1122

Secretaría de Educación Pública. (2011b). *Desafíos matemáticos para sexto grado*. Guía para el maestro.

— (2011). *Programa de estudios 2011*.

— (2017). *Aprendizajes clave para la educación integral*.

Selby, C., y Woollard, J. (2013). Computational thinking: the developing definition. Presentado en SIGCSE 2014, 5-8 March, Atlanta GA. Recuperado de https://eprints.soton.ac.uk/356481/1/Selby_Woollard_bg_soton_eprints.pdf

Sipitakiat, A. & Nusen, N. (2012). Robo-Blocks: Designing debugging abilities in a tangible programming system for early primary school children*. Proceedings of the 11th International Conference on Interaction Design and Children, Bremen: ACM.* (pp. 98-105). DOI: 10.1145/2307096.2307108

Suarez Castillón, S. A., & Soto Arévalo, F. S. (2015). Qualitative assessment of the kodu visual programming language in primary school children. *Tecnura*, *19*(46), 37-48.

Sullivan, A., Bers, M., & Pugnali, A. (2017). The impact of user interface on young children's computational thinking. *Journal of Information Technology Education: Innovations in Practice*, *16*(1), 171-193.

The Royal Society. (2012). Shutdown or restart? The way forward for computing in UK schools. London. Theory Method. The Qualitative Report, 2 (4), p. 1.

Tochacek, D., Lapes, J. & Fuglik, V. (2016). Developing technological knowledge and programming skills of secondary school students through the educational robotics project. *Procedia-Social and Behavioral Sciences*, (217), (pp. 377-381). DOI: 10.1016/j.sbspro.2016.02.107

Tochacek, D., Lapes, J. & Fuglik, V. (2016). Developing technological knowledge and programming skills of secondary school students through the educational robotics project. *Procedia-Social and Behavioral Sciences*, (217), (pp. 377-381). DOI: 10.1016/j.sbspro.2016.02.107

Torrejón M. M. F. & Ventura-Campos, N. (2019). Enseñanza-aprendizaje músico-matemático utilizando robótica educativa*. 3C TIC: Cuadernos de Desarrollo Aplicados a Las TIC,* 8(3), (pp. 12-37). DOI: 10.17993/3ctic.2019.83.12-37

Torres, N. B., González, R. L. & Carvalho, J. L. (2018). Roamer, un robot en el aula de Educación Infantil para el desarrollo de nociones espaciales básicas. *Revista Ibérica de Sistemas y Tecnologías de Información*, (28), (pp. 14-28). DOI: 10.17013/risti.28.14-28

Vivar, C. G., Arantzamendi, M., López-Dicastillo, O., & Gordo Luis, C. (2010). La teoría fundamentada como metodología de investigación cualitativa en enfermería. *Index de Enfermería*, *19*(4), 283-288.

Voogt, J., Fisser, P., Good, J., Mishra, P., y Yadav, A. (2015). Computational thinking in compulsory education: Towards

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

Wing, J. M. (2006). Computational Thinking. *Communications of the ACM*, 49(3), (pp. 33-35).

— (2008). Computational thinking and thinking about computing*. Philosophical Transactions of the Royal Society*, 366, (pp. 3717-3725). DOI: doi:10.1098/rsta.2008.0118

— (2011). Computational thinking. En 2011 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC 2011) (pp. 3-3). IEEE. doi: https://doi.org/10.1109/VLHCC.2011.6070404

— (2017). Computational Thinking´s influence on research and education for all. *Italian Journal of Educational Technology*, 25(2), 7-14. DOI: 10.17471/2499-4324/922

Zhong, B., & Wang, Y. (2019). Effects of roles assignment and learning styles on pair learning in robotics education. *International Journal of Technology and Design Education*. DOI: https://doi.org/10.1007/s10798-019-09536-2

Pautas teóricas de la descomposición.
Fundamentos, estrategias y aplicaciones en el aula
Se terminó de editar en el mes de noviembre de 2025
en el Centro Universitario de Los Altos,
Av. Rafael Casillas Aceves No. 1200, C.P. 47620,
Tepatitlán de Morelos Jalisco, México.

Tiraje: 1 ejemplar.

El Pensamiento Computacional (PC) es una habilidad crucial del siglo XXI, pero su implementación efectiva en el aula enfrenta un desafío persistente: la comprensión profunda de la descomposición, su proceso cognitivo central. Esta obra va más allá de la mera definición y se adentra en la "caja negra" de la descomposición, develando sus fundamentos teóricos, su tipología y sus estrategias.

A través de un análisis riguroso, el autor establece que la descomposición es un eje transversal que enriquece la práctica pedagógica, al proponer estrategias concretas para docentes de educación básica. El lector encontrará una taxonomía evolutiva del proceso y aplicaciones prácticas en preescolar, primaria y secundaria, demostrando que esta habilidad trasciende la programación y fortalece la creatividad, el razonamiento lógico y la autonomía en contextos computacionales y cotidianos.

Una propuesta teórica no es nada sin una aplicación empírica. Por eso, esta obra contiene la descripción técnica, teórica y didáctica del robot *Mbot* y del *software Mblockly*, que sirvieron de escenario contextual para la construcción de una teoría incipiente sobre el proceso de descomposición, la cual se logró mediante una metodología cualitativa de teoría fundamentada.

CUAltos
Centro Universitario de los Altos

ISBN: 978-607-581-838-2