LAS TECNOLOGÍAS EN (Y PARA) LA EDUCACIÓN

José Miguel García

Sofía García Cabeza

(compilación)

FLACSO
URUGUAY

Compilado por: José Miguel García y Sofía García Cabeza.

Con la autoría de (por orden de aparición): María Teresa Lugo, Florencia Loiácono, Isabel Achard, Manuel Area Moreira, Valeria Odetti, María Barberis, Pablo Bongiovanni, Corina Rogovsky, José Miguel García, Miguel Zapata-Ros.

Coordinación editorial: Sofía García Cabeza
Producción editorial: Susana Aliano Casales
Imprenta: Mastergraf
ISBN: 978-9915-9329-0-3

Las imágenes de esta publicación fueron proporcionadas por los autores de cada capítulo.
La imagen de tapa es de José Miguel García.

Acceso al libro en su versión digital:
http://www.flacso.edu.uy/publicaciones/edutic2020/garcia_garcia_tecnologias_en_y_para_la_educacion.pdf

FLACSO Editorial
Zelmar Michelini 1266, piso 2
11100 - Montevideo, Uruguay
Tel. (+598) 2903 0236
www.flacso.edu.uy

Capítulo 9
El pensamiento computacional, una cuarta competencia clave planteada por la nueva alfabetización

Miguel Zapata-Ros[1]

Introducción

Como hace César Coll (2019) en su prólogo al libro *Pensamiento computacional. Análisis de una competencia clave*, que escribimos Pérez Paredes, de Cambridge University, y quien suscribe, podríamos plantearnos a qué se debe el interés progresivo que se concede al pensamiento computacional en las publicaciones académicas, en los discursos sobre el aprendizaje y en la preocupación educativa contemporáneos. También nos podríamos cuestionar por qué se manifiesta este interés en las reformas educativas acometidas en los últimos años y los intentos con irregular éxito de introducirlo en los currículos de la educación infantil, primaria y secundaria de los países de nuestro entorno; mediante qué metodologías didácticas se hace y qué tipos de actividades se proponen para conseguir los objetivos que se plantean. Igualmente nos podríamos plantear qué organización educativa, incluso ahora en tiempos de pandemia, puede facilitar y promover un desarrollo efectivo del pensamiento computacional en las aulas y en los centros educativos.

Todo ello estará profusamente tratado en multitud de libros, artículos de revistas especializadas, *posts* académicos y foros de opinión. Un capítulo como este no es lugar, pues, para algo que necesitaría un programa amplio intenso y profundo de formación además de una colección de artículos y libros.

1 Este capítulo fue confeccionado por el autor como compilación revisada, actualizada y ampliada de los aportes vertidos en su blog.

En este espacio apenas caben unas pocas reflexiones sobre el sentido de lo que está pasando y algunas consideraciones sobre lo que está por venir.

Sin embargo, ello no obsta, por otro lado, para que nos planteemos, al menos sucintamente, qué es el pensamiento computacional, qué características diferenciales presenta respecto a otras modalidades de pensamiento como el pensamiento lógico, matemático, hipotético-deductivo, inductivo, creativo, etc. Y también para que nos planteemos qué no es el pensamiento computacional, cosa frecuentemente olvidada.

Al comienzo de este trabajo no podemos tampoco eludir formular escuetamente dos consideraciones relativas, respectivamente, a la naturaleza y características del pensamiento computacional y a su traslación a las aulas en forma de contenidos curriculares y de propuestas metodológicas.

Estos dos puntos están subyacentes en el resto del capítulo, particularmente el primero de ellos está en el trasfondo de lo que viene a continuación sobre la definición de pensamiento computacional. La que le otorga sentido.

La primera cuestión, por tanto, que caracteriza al pensamiento computacional tal como lo definimos en los primeros documentos que escribimos al respecto, en el artículo «Pensamiento computacional: Una nueva alfabetización digital» (Zapata-Ros, 2015) y en el libro citado, es, junto con la diversidad y heterogeneidad de los componentes que lo caracterizan, el hecho de que en él se incluyen lo que aparentemente constituye la mayoría, aunque no la totalidad, de los otros tipos o modalidades de pensamiento conocidos. Que son los que de una u otra forma se ponen en acción cuando los informáticos programan o diseñan los algoritmos, diagramas de flujo, análisis de los problemas, etc. Así los identificamos y los describimos cuando en esos primeros trabajos repasamos con ánimo de exhaustividad el quehacer de los programadores.

En esa relación de quince componentes están implícitas casi todas las formas de pensamiento conocidas en el trabajo intelectual. Así, en la relación que se describe en el último trabajo publicado (Zapata-Ros, 2019: 63), encontramos el análisis ascendente, el análisis descendente, la heurística, el pensamiento divergente, la creatividad, la recursividad, la iteración, el ensayo y error, la metacognición, etc.

¿Qué es lo que confiere unidad e identidad a este conjunto de habilidades para que no sea sólo una yuxtaposición?

Como dice Coll (2019):

> La simple convergencia de modalidades y tipos de pensamiento, sin embargo, difícilmente puede dar al conjunto un carácter integrador. Sin un componente vertebrador la convergencia derivaría en realidad en simple amalgama.

Pues bien, lo que confiere unidad e identidad al pensamiento computacional, a este conjunto de habilidades, y permite identificarlo como un tipo de pensamiento diferente a los otros, a los que integra, al pensamiento divergente, al pensamiento abstracto, al pensamiento lógico, al pensamiento elaborativo, etc., es justamente la capacidad que tiene para integrarlos en procesos, en sistemas y en diseños completos orientados a la acción sobre la realidad, a la resolución de problemas reales. Como mencionamos Pérez-Paredes y Zapata-Ros (2018), lo que caracteriza el pensamiento computacional es su orientación a la construcción de «sistemas que interactúan con el mundo real» (p. 60).

A este rasgo que caracteriza el pensamiento computacional queremos añadir otro que es clave para lo que vamos a ver. Se trata de un rasgo que permite comprender y valorar su alcance y el interés creciente que tiene en los ámbitos académico, profesional y educativo: se trata de su estrecha vinculación con la nueva cultura epistemológica, donde el conocimiento es transmitido, procesado, adquirido con códigos y métodos nuevos. El papel que adquiere es el de una nueva alfabetización, a través de la cual los individuos aprenden habilidades para la representación y comunicación del conocimiento en los nuevos códigos, así como el análisis crítico y la intervención en ellos, en programas, en algoritmos y en sus ejecuciones.

De esta forma, el pensamiento computacional está en la base de una cultura «vinculada al desarrollo y la generalización de los medios y de las redes digitales que apareció con la informática personal, la internet, y continuará con la IA y el *blockchain* posiblemente, entre otros medios» (Pérez-Paredes y Zapata-Ros, 2018: viii).

Esto tiene una repercusión decisiva en lo que concierne a la selección y organización de los contenidos curriculares. No solo es indis-

pensable, como hasta ahora se hace, formar a las nuevas generaciones en el conocimiento, dominio y uso de *affordances* de herramientas y dispositivos digitales que median ya en las prácticas sociales y culturales en prácticamente todos los ámbitos de la actividad de las personas y de control de los procesos (robótica), o incluso en la enseñanza de la programación como disciplina o habilidad específica, sino que, además y sobre todo, lo que lo que hace indispensable la incorporación de contenidos específicos de aprendizaje transversal en pensamiento computacional en los currículos escolares es la nueva visión de resolución de problemas que estos medios proporcionan. Y solo se puede hacer conociendo las claves de funcionamiento interno de estos entornos. La lógica de los entornos computacionales tiene mucho que ver con el pensamiento lógico, simbólico, etc. Tradicionales, pero sin el lastre que implican los procesos y medios analógicos. En el libro tenemos multitud de ejemplos sobre aprendizajes concretos: representación de fracciones y decimales, definición de potencias numéricas a los niños, de factoriales con recursividad, etc.

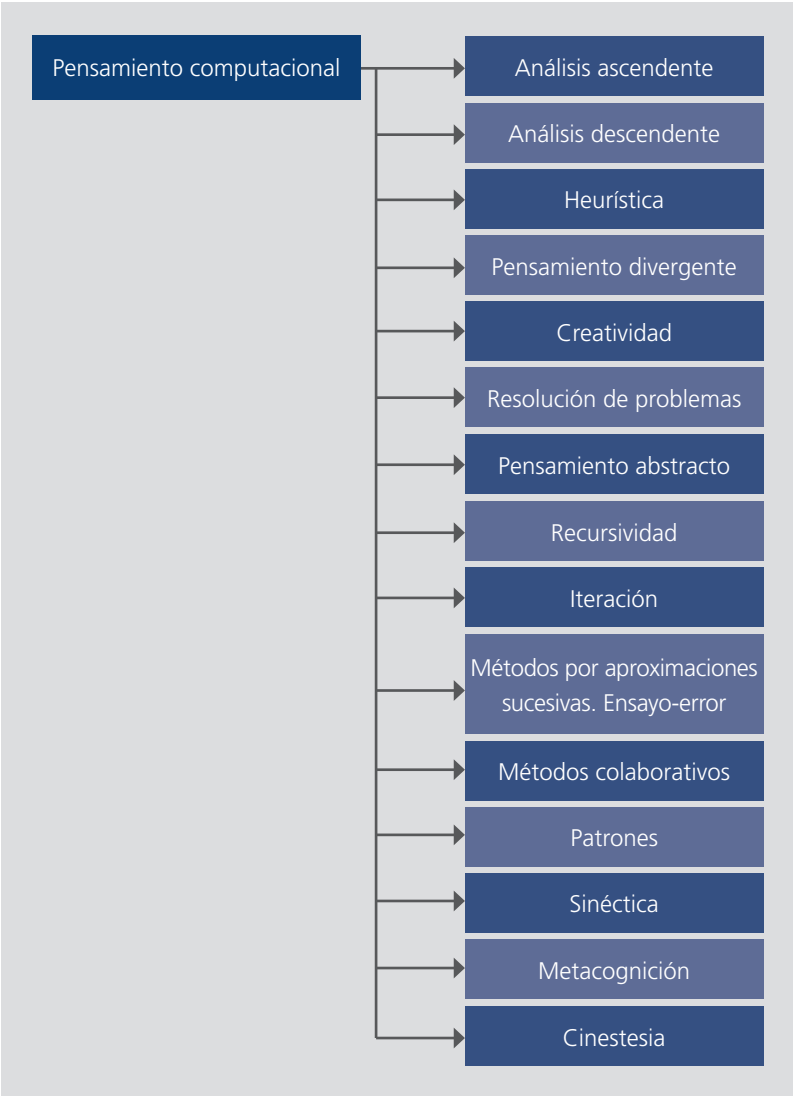
La conclusión, como veremos, es que esta inclusión en el currículo hay que hacerla facilitando el desarrollo del pensamiento computacional de manera transversal en todas las áreas curriculares, con todos los contenidos y como competencia clave desde los primeros ciclos.

Otra cuestión que conviene aclarar desde el principio es la del problema de definir el pensamiento computacional.

En el primer artículo y en el libro en vez de proponernos elaborar una definición acabada y depurada, y por tanto simple, del pensamiento computacional y partir de ella en todo el desarrollo, el procedimiento fue el inverso: lo hicimos constructivamente, nos planteamos en la necesidad de un dominio teórico específico del pensamiento computacional en las teorías del aprendizaje y en la necesidad de un currículo. Y construimos una definición extensiva, basada en los elementos que satisfacían esas necesidades. Pensamos que, por eso, por no adaptarse a las exigencias de la mercadotecnia educativa y de la comunicación académica, nuestra definición no tuvo éxito. Era muy pesada y difícil de digerir frente a la simpleza de definiciones como la de Wing.

Los elementos constitutivos del pensamiento computacional y que lo definen, de esta forma, son los que se presentan a continuación (Zapata-Ros, 2015) (Pérez-Paredes y Zapata-Ros, 2018).

Figura 1. Elementos que están ampliamente desarrollados en los trabajos citados



Fuente: Figura del autor.

En definitiva, esto supone una definición de pensamiento computacional por acumulación de elementos, cohesionados por una función o por un objetivo, que es resolver problemas. En esto consiste la aportación central y clave de nuestro trabajo, iniciada en 2014, en un *post* sobre este tema (Zapata-Ros, 2014b), donde empezábamos la relación de elementos con el pensamiento ascendente, diciendo previamente:

> Conviene decir que estas componentes no están perfectamente delimitadas ni conceptual ni metodológicamente. No son excluyentes, y según en qué contexto se empleen pueden tener significados distintos. De hecho ni tan siquiera se puede decir que constituyan elementos de una taxonomía o que correspondan a un mismo nivel operativo o conceptual. Es perfectamente posible que en métodos o procedimientos que se cataloguen por ejemplo como resolución de problemas haya elementos de análisis ascendente, o descendente, y es difícil que un análisis descendente no tenga elementos de recursividad.

En junio de 2019, un año después de su publicación, llegó a mí este artículo: *The 5th 'C' of 21st Century Skills? Try Computational Thinking* (*Not Coding*), de la profesora Shuchi Grover (2018), de Stanford, en el que por primera vez vi, en alguien distinto y lejano, sin contacto previo de ningún tipo, que se consideraba el pensamiento computacional como una acumulación de habilidades y elementos de conocimiento necesarios para programar y para caminar por la vida profesional y personal en la sociedad digital. Por fin un poco de luz. Hasta ese mismo momento pensaba, ya casi a punto de abandonar al menos en lo fundamental, que esta era una idea estéril y sin futuro.

Antes de entrar en el trabajo de Grover sería bueno que hiciésemos un sucinto recorrido por las definiciones previas. Son definiciones clásicas, que se repiten hasta la saciedad en artículos y conferencias, pero que son, a nuestro juicio, simples o incompletas, porque en el mejor de los casos aluden a unos pocos de los elementos que, según nuestro análisis y según vi después en el de Grove e incluso en el de Wing, pueden encontrarse que lo componen.

Así, la informática Tasneem Raja (2014) en el *post* We Can Code It! , de la revista-blog *Mother Jones* decía:

> El enfoque computacional se basa en ver el mundo como una serie de puzzles, a los que se puede romper en trozos más pequeños y resolver poco a poco a través de la lógica y el razonamiento deductivo.

Pero esta es una forma intuitiva en la que una autora, que proviene del mundo computacional, aborda una serie de métodos ampliamente conocidos en el mundo de la psicología del aprendizaje, además del de la computación. Implícitamente está hablando de análisis descendente y de elaboración: *puzzles* —problemas— que se pueden dividir en *puzzles* —problemas— más pequeños, para ir resolviéndolos. También, en el mismo párrafo, vemos una alusión implícita a la recursividad. Falta la cláusula de parada y la vuelta atrás, pero evidentemente después de armar los *puzzles* pequeños cada uno hay que ensamblarlo en el *puzzle* general. Y también, todo hay que decirlo, habrá que incluir el nivel en el que hay que parar y dar marcha atrás.

La definición es simple y muy útil para el lenguaje periodístico, pero limitada e inexacta. Se limita a señalar un solo aspecto de lo que, en otros ámbitos que veremos, hemos incluido como aspectos particulares del pensamiento computacional, como es el análisis descendente. Y evidentemente reducir el pensamiento computacional a ese tipo de análisis es muy pobre, aunque sea sugestivo y tenga impacto en blogs y en conferencias de divulgación.

La primera definición de Wing (2006), a la que siempre se recurre, es muy general, no hace pensar mucho a la gente y es fácilmente aceptable, pero no nos da pautas concretas para discernir lo que es pensamiento computacional, ahí realmente cabría todo. Wing dice que el pensamiento computacional es una forma de pensar que no es solo para programadores:

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

Y eso está bien, pero no resuelve el problema. En ese mismo artículo continúa diciendo «que esas son habilidades útiles para todo el mundo, no sólo los científicos de la computación».

Pero volvamos al trabajo de Shuchi Grover (2018). En el apartado dedicado a decir qué es el pensamiento computacional, la autora señala que está constituido por:

> (...) los procesos de pensamiento involucrados en entender un problema y expresar sus soluciones de tal manera que una computadora pueda potencialmente llevar a cabo la solución. El pensamiento computacional se basa fundamentalmente en el uso de conceptos y estrategias analíticas y algorítmicas más estrechamente relacionadas con la informática para formular, analizar y resolver problemas.
>
> Al igual que las habilidades de pensamiento general, el pensamiento computacional es un poco como el [concepto de] liderazgo: es difícil de definir, pero lo reconoces cuando lo ves. Si bien muchas personas lo asocian con conceptos como la programación y la automatización, que son todas partes centrales de la informática, los educadores e investigadores han encontrado que es más fácil operacionalizarlo para los propósitos de la enseñanza, el currículo y el diseño de evaluaciones.
>
> Eso significa desglosar las habilidades de pensamiento computacional en sus partes componentes, que incluyen conceptos como lógica, algoritmos, patrones, abstracción, generalización, evaluación y automatización. También significa enfoques como «descomponer» problemas en subproblemas para facilitar la resolución, creando artefactos computacionales (generalmente a través de codificación); reutilizando soluciones, probando y depurando; refinamiento iterativo.
>
> Y sí, ¡también implica colaboración y creatividad! Y, además, no es necesario que involucre una computadora.[2]

2 Traducción del autor.

Veamos, pues. De entrada señala la dificultad de definir pensamiento computacional, entonces, adopta la posición de definir lo que es pensamiento computacional como, o a través de, un conjunto de cosas (eso significa desglosar las habilidades de pensamiento computacional en sus partes componentes). La mayor parte de ellas implican o son habilidades, pero siempre son fáciles de operativizar y, sobre todo, son posibles de incluir en un diseño educativo.

Son habilidades que incluyen facultades para operativizar la lógica (pensamiento lógico), los algoritmos (algoritmia), patrones, abstracción (pensamiento abstracto), generalización (pensamiento ascendente), evaluación y automatización. También significa enfoques como «descomponer» problemas en subproblemas para facilitar la resolución (pensamiento descendente), creando artefactos computacionales (generalmente a través de codificación); reutilizando soluciones, probando y depurando (ensayo y error); refinamiento iterativo (iteración).

Para concluir, dice que «¡también implica colaboración (métodos colaborativos) y creatividad!».

Hasta aquí, si no he contado mal, coincidimos en diez de los quince elementos.

Pero hay más coincidencias, o al menos cierto paralelismo en cuanto a la relevancia del fenómeno en el contexto de la dinámica de las alfabetizaciones y de lo que es la alfabetización digital. En el artículo Groves señala la relevancia del pensamiento computacional en cuanto a que constituye una competencia más a añadir a las ya aceptadas como competencias para la sociedad digital. En cualquier caso, lo que tienen de común ambos desarrollos, el artículo de Graves y los trabajos de quien suscribe, es que el pensamiento computacional supone un punto de inflexión.

En el caso de Shuchi Groves, desde el título [*The 5th 'C' of 21st Century Skills? Try Computational Thinking* (*Not Coding*)] se señala que el pensamiento computacional añade una quinta C a las cuatro de las competencias digitales señaladas y aceptadas por todos. En el título está implícita esta afirmación, porque solo es una pregunta retórica, y en realidad es una afirmación. Y además afirma otra cosa, que la habilidad no consiste en programar, las habilidades son las del pensamiento computacional, una forma de pensamiento que permite programar.

En esta segunda parte de la frase se sintetizan todas las argumentaciones de los trabajos referenciados anteriormente.

En lo siguiente detalla las otras C y argumenta la inclusión de la quinta:

> Desde el comienzo de este siglo, las habilidades de «4C del siglo XXI» (pensamiento crítico, creatividad, colaboración y comunicación) han visto un creciente reconocimiento como ingredientes esenciales de los planes de estudio escolares. Este cambio ha llevado a una adopción de pedagogías y marcos tales como el aprendizaje basado en proyectos, el aprendizaje por indagación y el aprendizaje más profundo en todos los niveles de K-12 que enfatizan el pensamiento de orden superior sobre el aprendizaje de rutina. Sostengo que necesitamos que el pensamiento computacional sea otra habilidad central, o la «5ta C» de las habilidades del siglo XXI, que se enseña a todos los estudiantes.
>
> Existe un creciente reconocimiento en los sistemas educativos de todo el mundo de que la capacidad de resolver problemas de manera computacional, es decir, pensar de manera lógica y algorítmica, y usar herramientas de computación para crear artefactos, incluidos modelos y visualizaciones de datos, se está convirtiendo rápidamente en una competencia indispensable para todos campos.

Para quien suscribe, aceptando el principio general de la quinta C, el cambio es de mucho más impacto y relevancia. Se trata de que estamos en presencia de una nueva alfabetización, que se distingue de las anteriores por el nuevo y potente medio que soporta y transmite el conocimiento: el medio digital, que se incorpora a los ya existentes: los libros, la prensa, los documentos escritos y las imágenes en papel y en los medios audiovisuales. Antes el conocimiento se representaba y se transmitía por la lectura, la escritura y las matemáticas, simbolizadas de forma simplificada por «las tres erres». Ahora, a las tradicionales tres erres: leeR, escribiR y aRitmética, se una cuarta expresión, sin erre, con la cuarta competencia clave para la alfabetización en la sociedad digital, y sin las cuales no se podrían adquirir el resto de conoci-

mientos, representarlos o atribuirles sentido. Esta cuarta competencia clave es el pensamiento computacional. En el libro y en su presentación en University of Cambridge lo decíamos así:

> En la tradición pedagógica anglosajona se les denomina las tres erres: *The three Rs: "Reading, wRiting and aRitmhetic and computational thinking",* que de esta forma se constituyen en una alfabetización de tres erres ampliada (three Rs+): *The three Rs +: «Reading, wRiting, aRitmhetic and computational thinking".*

Hay otro aspecto interesante. El pensamiento computacional no alcanza su verdadero carácter de innovación hasta que no está incluido en los programas oficiales, y esto supone previamente un diseño curricular (o sea: cómo es la nueva programación educativa que queda tras incluirlo) y sobre todo un diseño instruccional (cómo se hace). Es decir, cómo se enlazan los objetivos o resultados que se pretenden con lo que se hace para conseguirlos. La evaluación de aprendizajes y de proceso, los recursos, el propio diseño de objetivos, la metodología docente y sobre todo actividades. Estas últimas serían la pieza clave, la prueba del algodón, de forma que hasta que no lo hagamos no podemos decir que algo constituye una práctica educativa, como bien nos empeñamos en manifestar en el libro, en la parte dedicada a *unplugged*, en *posts* y en los artículos correspondientes (Zapata-Ros, 2019). En esos trabajos incluimos de forma significativa dos actividades: una sobre álgebra en educación infantil y otra sobre puertas lógicas con pegatinas.

En este rápido análisis hay que destacar que si bien Grover no incluye actividades, ni desenchufadas ni de cualquier otro tipo, para el desarrollo de pensamiento computacional (en K12, K9, en Elementary School, o en Key stage-educación infantil) ni tampoco diseño instruccional, sí que se esfuerza en sugerir ejemplos de actividades y de aprendizajes en otras áreas donde pone en evidencia que el pensamiento computacional es útil, muy conveniente o en todo caso que es algo subyacente, y así lo manifiesta para las áreas de lenguaje, matemáticas, ciencias y ciencias sociales.

Por cierto, es curioso que señale el carácter desenchufado («*Some are unplugged*…») de algunas de estas actividades:

> La codificación es un contexto excelente, divertido y útil para desarrollar habilidades de pensamiento computacional. Pero no es la única manera. Aquí hay algunas ideas para fomentar el pensamiento computacional en los sujetos. Algunos están desconectados, mientras que otros se beneficiarían con la codificación. ¡Los maestros pueden reconocer muchas de las actividades no programadas como cosas que ya hacen!

Esperamos y deseamos que, sea como sea, esa convergencia continúe y se plasme como en los demás casos en un nuevo diseño curricular y en programas de capacitación del profesorado, que integre estos elementos desde no solo el K-12, sino, como nos empeñamos en dejar claro, desde las primeras etapas de educación infantil y primaria.

Por lo demás, en este *post* de 2018 se resume lo que más ampliamente exponen Grover y Pea en 2013 y en 2018 (Grover y Pea, 2013 y 2017).

Lo curioso del caso es que en un trabajo posterior de Wing se matiza mucho más en lo que es el pensamiento computacional. Señala la variedad de tipos de pensamiento abstracto que existe, la diferencia con el pensamiento matemático. Así, pues, en ese trabajo podemos rastrear como elementos de pensamiento computacional ciertos rasgos de este que tienen que ver con el lenguaje y el pensamiento de patrones o la secuenciación de acciones.

A veces vemos incluso estos elementos mezclados en formas de pensamiento computacional más complejo (Wing, 2008):

> Mirando hacia el futuro, un pensamiento computacional más profundo, a través de la elección de la abstracción o las abstracciones más sofisticadas, puede permitir a los científicos e ingenieros modelar y analizar sus sistemas en una escala de magnitud mayor de la que pueden manejar hoy en día. Mediante el uso de capas de abstracción, por ejemplo la descomposición jerárquica, esperamos con ansias cuanto antes: modelar sistemas en escalas de tiempo múltiples y en múltiples resoluciones de 3D; modelar

> las interacciones de muchos de estos sistemas complejos, para identificar condiciones que fortalecen los puntos y el comportamiento emergente; aumentar el número de parámetros y conjuntos de condiciones iniciales en estos modelos; jugar estos modelos hacia atrás y hacia adelante en el tiempo; y validar estos modelos confrontando con datos reales.[3]

Lo que parece que sucede es que cuando Wing amplía el diapasón de los niveles, disciplinas y áreas de producción, servicios e investigación, la idea de un pensamiento computacional simple desaparece.

Así lo dice en el epígrafe de su trabajo titulado *Pensamiento computacional en todas partes*:

> El pensamiento computacional está influyendo en la investigación en casi todas las disciplinas, tanto en las ciencias como en las humanidades (Bundy, 2007).

Abunda la evidencia de la influencia del pensamiento computacional en otros campos: el pensamiento computacional está transformando las estadísticas, donde con el aprendizaje automático la automatización de los métodos bayesianos y el uso de modelos gráficos probabilísticos hacen posible identificar patrones y anomalías en voluminosos conjuntos de datos tan diversos como mapas astronómicos, funcionales escaneos de resonancia magnética, compras con tarjeta de crédito y recibos de la tienda de comestibles (por ejemplo, Machine Learning Department 2008).

Hay que pensar, pues, en un constructo conceptual de pensamiento computacional que sea capaz de alojar componentes tan variados y complejos. Una primera aproximación fue la propuesta de Shuchi Grover y modestamente la definición acumulativa de quince componentes de quien suscribe, unificada por su orientación hacia la resolución de problemas y la consecución de objetivos en contextos profesionales, institucionales, investigativos o profesionales.

3 Traducción del autor.

El pensamiento computacional como alfabetización de la cultura digital

La idea más extendida sobre lo que es la alfabetización digital (*Digital Literacy*) es que consiste en una trasposición (UK Government, 2015: 7) de lo que ha sido tradicionalmente una alfabetización a la cultura digital:

> Entonces, alfabetización en su sentido fundamental es compartir el significado a través del lenguaje.

A lo largo de la historia de la humanidad se han ido sucediendo distintas alfabetizaciones y todas han tenido una significación común y un mismo sentido: han supuesto una adaptación de la forma en que los humanos se comunican y representan la realidad a los nuevos medios de comunicación, representación y proceso de la información que han dispuesto.

A grandes rasgos hay tres culturas epistemológicas, según el concepto acuñado por Stehr (2003 citado en Zapata-Ros, 2013), y tres grandes cambios que dan lugar a ellas.

Entre el año 3000 y 1500 a.C. se produce un cambio progresivo en distintas partes del mundo. Aparecen soportes que son capaces de llevar de forma móvil y personal la información y la representación del conocimiento, son las tabletas de arcilla o, de forma predominante para las culturas de la época, el papiro, los rollos de papiro, que son la base administrativa y cultural de los imperios egipcios. Su producción y comercio alcanza su auge en Biblos (Fenicia), nombre de ciudad que ha dado lugar a la palabra griega *Byblos* (*βίβλος*) y de allí la palabra griega también *Biblion* (*βιβλιον*) «libro», que dará origen a los términos Biblia y biblioteca. De hecho, el nombre Biblia con el que se conoce al libro sagrado cristiano es atribuido a esta ciudad, ya que la primera Biblia se realizó en papiro y provenía de esta.

Este medio, que está en la base de una sociedad y de una cultura vastísima, continúa su expansión siendo el soporte para la difusión, la alfabetización y la inculturación de los pueblos y las gentes en la cultura griega y en la cultura romana, a través tanto del imperio de Alejandro Magno como del Imperio Romano.

Aunque material y técnicamente es distinto, al papiro le sucede el pergamino y a los rollos, los códices medievales. Pero es la misma forma de soporte y de medio de difusión: uno a uno, individual y móvil, como base material de información textual, gráfica o icónica. Siempre en entidades únicas e irrepetibles, es la que soporta la cultura en la Edad Media, en los monasterios y cenobios y la traslada hasta los albores del renacimiento y la Edad Moderna, cuando se produce el siguiente cambio que da lugar a la siguiente cultura epistemológica.

La segunda revolución la produce la imprenta y llega hasta nuestros días: la base es el libro impreso y constituye un gran paso, una democratización y una producción industrial de la cultura soportada por este medio. La característica esencial es que el soporte del conocimiento es repetible e ilimitado prácticamente, las tiradas de libros son tan extensas como lectores pueda haber, porque compran libros o porque los leen en bibliotecas u otras instituciones. Igual sucede con el material impreso educativo. Las primeras universidades a distancia lo son por correspondencia con paquetes postales y envíos de libros. Y eso ha sido así hasta ahora en que las crisis disruptivas de la sociedad del conocimiento han tambaleado a instituciones tan sólidas como la UK Open University (Bates, 2018).

En ambos casos todas las manifestaciones sociales o individuales, cómo la gente se expresa, cómo se produce el intercambio de bienes y servicios, cómo queda registrada la propiedad, las ciencias, la medicina, cómo la gente accede al poder y a la riqueza, la promoción interclase queda fuertemente determinada por la forma en que circula el conocimiento y son registrados los datos.

La tercera revolución, con la consiguiente aparición de la cultura epistemológica y con la alfabetización, va vinculada al desarrollo y la generalización de los medios y de las redes digitales, apareció con la informática personal, la internet, y continuará con la inteligencia artificial (IA) y el *blockchain* posiblemente, entre otros medios.

La principal característica de la «sociedad del conocimiento» (Stehr, 2003, citado en Zapata-Ros, 2013) es la transformación radical y progresiva de la estructura económica de la sociedad industrial, con un sistema productivo basado en factores materiales hacia un sistema

en el que los factores simbólicos basados en el conocimiento son dominantes. Factores cognitivos, de creatividad, de conocimiento y de información contribuyen cada vez más a la producción y a la distribución de la riqueza en las sociedades y al bienestar de los individuos. Otra característica clave es la progresiva adquisición de un carácter científico de áreas de actividad de la sociedad (Zapata-Ros, 2013).

Como en los casos anteriores no solo cambia cómo se representa, cómo circula el conocimiento y cómo lo usan los individuos para relacionarse y para otras funciones básicas, sino que afecta a la naturaleza más profunda de las actividades humanas: las económicas, las sociales, las científicas… No hace falta extenderse. Pero en este caso el individuo no solo recibe, incorpora o gestiona el conocimiento, sino que lo crea. Y la realidad se escribe en códigos informáticos, no solo en códigos textuales o literarios. Para acceder a cualquier trabajo es imprescindible una serie de competencias nuevas. Son las competencias computacionales o digitales. Pero no solo de usuario, no basta con usar los medios, también hay que conformarlos, a algún nivel, aunque solo sea para filtrar la información o para interrogar a las bases de datos, o para crear metainformación eficiente, hace falta un conocimiento de codificación, de programación. Esta es la nueva competencia clave en la nueva alfabetización, como antes lo eran la lectura, la escritura o el cálculo.

En todos los casos, en todas las culturas epistemológicas, y la digital es la última, hay una serie de rasgos que son conformados de forma diferente: qué determina las competencias básicas que han de adquirir los individuos, qué sectores sociales las adquieren y para qué, cuáles son esas competencias claves y básicas, y cuál es la tecnología de soporte y de comunicación del conocimiento.

Así, según esta idea, la alfabetización digital consiste en la adaptación y la capacitación de los individuos para esas funciones de comunicación, representación y proceso a las coordenadas de la revolución tecnológica y de la sociedad de la información, consideradas, en sentido no estrictamente tecnológico, como una revolución de medios de comunicación y de difusión de ideas.

Cultura epistemológica	Competencias básicas para	Competencias claves y básicas	Destinatarios	Tecnología
Biblos, Mesopotamia, Egipto, Grecia, Roma, Edad Media. 3000 a.C.	Ser un buen orador. Persuadir. Registrar datos, fechas y acontecimientos.	Leer Escribir Hablar Cálculo Geometría	Una élite de funcionarios, políticos y sacerdotes.	Papiros, códices, tablillas.
Imprenta 1450	Acceder a cualquier sector de la producción, los servicios y a las profesiones. Acceder a cualquier conocimiento.	Lectura Escritura Matemáticas Aprender a aprender (metacognición)*	Toda la sociedad como usufructuaria de algún tipo de conocimiento.	Libro impreso. La fotografía, el cine, la TV, el vídeo, la radiofonía… no digital (aunque para algunos constituye una subcultura epistemológica que justifica una alfabetización propia: La audiovisual e icónica).
Sociedad digital, sociedad del conocimiento 1981-…	Acceder a cualquier sector de la producción, los servicios y a las profesiones. Para acceder a cualquier conocimiento. Son digitales y gobernados por programas o con interfaces de lenguaje estructurado. Para la propia vida.	Lectoescritura Matemáticas Pensamiento computacional	Todos los individuos, como usuarios y como creadores.	Informática personal, internet, tecnología ubicua y s ocial, IA, *blockchain* y otros medios que se desconocen.

* Explicitada recientemente.

En todas las aproximaciones a través de las ideas más importantes o significativas sobre la alfabetización digital se coincide, pues, con el patrón de adaptación de la idea existente sobre alfabetización a una nueva cultura epistemológica.

En el resto de trabajos sobre este tema nos hemos acercado progresivamente a la idea de pensamiento computacional como conjunto de elementos o de formas específicas de pensamiento, que sirven para resolver problemas en diversos ámbitos.

Lo que se propone ahora es la idea de un pensamiento computacional fuertemente relacionado con la alfabetización digital, en el sentido de que está constituido por competencias claves que sirven para aprender y comprender ideas, procesos y fenómenos no solo en el ámbito de la programación de ordenadores o incluso del mundo de la computación, de internet o de la nueva sociedad del conocimiento, sino que es, sobre todo, útil para emprender operaciones cognitivas que, de otra forma, sería más complejo o imposible realizar. O bien, porque sin estos elementos de conocimiento sería más difícil resolver ciertos problemas de cualquier ámbito, no solo de la vida científica o tecnológica, sino de la vida común. Como dijimos, se considera como un conjunto de habilidades esenciales para la vida en la mayoría de los casos y como un talante especial para afrontar problemas científicos y tecnológicos.

En este sentido, a lo dicho en anteriores ocasiones hay que añadir ahora la idea de pensamiento computacional generalizado (*pervasive computational thinking*) como avance y desarrollo de lo tratado en el pensamiento computacional simple, el que Wing (2006) plantea en su primera aproximación.

De esta forma se habla del pensamiento computacional por todas partes (*computational thinking everywhere)* (Wing, 2008). Se habla de ello al considerar que ya no se trata de aspectos puramente asociados a la práctica profesional o vital ordinaria para manejarse por la vida y el mundo del trabajo, sino como una preparación para la investigación básica y para la metodología investigadora en casi todas las disciplinas. El pensamiento computacional está influyendo en la investigación en casi todas las áreas, tanto en las ciencias como en las humanidades (Bundy, 2007). Abundan evidencias sobre la influencia del pensamiento computacional en otros campos: así, el pensamiento computacional está transformando las estadísticas, donde con el aprendizaje automá-

tico, la automatización de los métodos bayesianos y el uso de modelos gráficos probabilísticos es posible identificar patrones y anomalías en voluminosos conjuntos de datos tan diversos como son los corpus lingüísticos (Pérez-Paredes y Zapata-Ros, 2018), los mapas astronómicos, añadir funcionalidades a la práctica de la resonancia magnética o a los hábitos de compra con tarjeta. Esto por señalar solo algunos casos como son los que se asocian con el análisis de grandes datos y la teoría bayesiana. Pero hay muchos más. Recomendamos la lectura del apartado sobre *computational thinking everywhere* en el artículo *Computational thinking and thinking about computing* (Wing, 2008).

Adelantamos que en nuestra próxima definición de pensamiento computacional por elementos añadiremos, en esta dirección de tratar el *computational thinking everywhere*, un nuevo elemento, el de pensamiento bayesiano.

Pensamiento computacional, necesidad de un currículo desde las primeras etapas: el pensamiento computacional desenchufado

La necesidad de un currículo

Según hemos visto en el apartado anterior, hay multitud de áreas del aprendizaje que conviene explorar e investigar en esta nueva frontera. Y la planificación de los currículos tendrá que plantearse esta dicotomía: enseñar a programar con dificultad progresiva (si se quiere incluso de forma lúdica o con juegos) o favorecer este nuevo tipo de pensamiento. Obviamente, no hace falta decir que nuestra propuesta es la segunda, que, además, incluye a la primera. Pero volvamos al tema central, la naturaleza y el tema del pensamiento computacional.

Tropezamos con varios problemas de comienzo: delimitar el contenido y encontrar los términos y conceptos adecuados para definirlo.

En un principio se utilizó la expresión codificación y precodificación. La segunda, extraída de la literatura anglosajona, *coding* o *code*. En este sentido, se utilizó la expresión en los textos que dieron a conocer el año 2014 como el año del código o de la codificación (*Year Of Code*). Es importante acceder al documento de difusión donde además de utilizar el término *code* dan una aproximación bastante general, pero precisa del término ya desde el principio.

Así, se dice: a través de la codificación (*code*) la gente puede descubrir el poder de la informática, cambiando su forma de pensar acerca de su entorno y obtener el máximo provecho del mundo que le rodea.

Más precisa es la definición del informe de 2014 de la Unión Europea *Computing our future Computer programming and coding-Priorities, school curricula and initiatives across Europe*: la codificación (*coding*) es cada vez más una competencia clave que tendrá que ser adquirida por todos los jóvenes estudiantes y cada vez más por los trabajadores en una amplia gama de actividades industriales y profesiones. La codificación es parte del razonamiento lógico y representa una de las habilidades clave que forma parte de lo que ahora se llaman «habilidades del siglo XXI».

Como vemos, es un dominio conceptual muy próximo a lo que hemos visto en el apartado anterior, y veremos en este, que el pensamiento computacional, al menos se expresa con ese sentido que le hemos atribuido.

Por otro lado, de igual forma que se habla de prelectura, preescritura o precálculo para nombrar competencias que allanan el camino a las destrezas claves y a las competencias instrumentales que anuncian, cabe hablar de precodificación o preprogramación para designar las competencias que son previas y necesarias en las fases anteriores del desarrollo para la codificación.

Un planteamiento útil en este sentido es que los niños se familiaricen en las primeras etapas de desarrollo con preconceptos tales como variable, función, valor, parámetro. De forma que, sin necesidad de referencias explícitas, desarrollen habilidades y preconceptos que en el futuro puedan alojar o servir de andamiaje conceptual para operaciones o conceptos más complejos propios de habilidades cognitivas superiores, necesarias para la programación. Así, los equivalentes a variables pueden ser rasgos de objetos como el color, la forma, el tamaño... Y los procedimientos u operaciones con estos rasgos (variables) pueden ser la seriación, el encaje, etc.[4]

4 No hace falta que el lector sea muy avispado para darse cuenta de que, implícitamente, estoy haciendo alusión a los juguetes de seriación, encaje, Torres de Hanoi... que se pueden encontrar para las primeras etapas de desarrollo o para jugar en casa, en los establecimientos de Ikea o que constituyen las bases de las actividades Montessori de rincones.

Evidentemente hay muchas más habilidades y más complejas en su análisis y en el diseño de actividades y entornos para que este aprendizaje se produzca. Así, este ámbito de la instrucción es lo que podría denominarse precodificación o preprogramación, y más recientemente ha sido conocido como *unplugged computational thinking*. Sin embargo, en un principio creímos más adecuado llamarle *precodificación*, pues codificación describe, con más precisión y más ajuste conceptual, la transferencia de acciones e informaciones para que puedan ser interpretadas por los ordenadores y otros dispositivos de proceso, circulación y almacenamiento de la información.

Si solamente hablásemos de algo preparatorio para la programación, podríamos hacerlo así, y sin duda sería correcto. Sin embargo, esto se correspondía con un propósito más amplio que es prepararse para dotarse de claves de comprensión y de representación de los objetos de conocimiento en general. Es por ello que vemos más adecuada la expresión «pensamiento computacional» (*computational thinking*), que a continuación desarrollaremos.

Otra expresión que se propone habitualmente es la de alfabetización digital o «una nueva alfabetización digital». Sin embargo, hay que reconocer que, al menos en español e impropiamente, esta expresión tiene resonancias próximas al término «alfabetización informática», al menos en el sentido que se ha utilizado hasta ahora. Expresión que inevitablemente, por el uso, nos recuerda la informática de usuario, al considerarse esta alfabetización como el conocimiento y la destreza para manejarse en entornos de usuario. Así, es frecuente confundir al buen informático con el que maneja bien, es hábil, con los programas de usuario, las aplicaciones, o al que se maneja con fluidez y rapidez en los ambientes de menús, ventanas y opciones, o simplemente al que tiene habilidad en los pulgares para manejar un *smartphone* o con el índice para moverse por una tableta. En una acepción lamentablemente muy extendida y banal ha dado lugar a que prendan conceptos paracientíficos como son los de nativo y emigrante digital.

En lo que sigue aceptamos la definición de alfabetización digital (*computer literacy*) como el conocimiento y la capacidad de utilizar las computadoras y la tecnología relacionada con ellas de manera eficiente, con una serie de habilidades que cubren los niveles de uso elemental de la programación y la resolución de problemas avanzada (*Wash-*

ington, US Congress of Technology Assessment, OTA CIT-235 April 1984, page 234). Con el reparo, ya citado, de que en ese mismo documento se acepta que la expresión alfabetización digital también se puede utilizar para describir el nivel de acomodo que un individuo tiene con el uso de programas de ordenador y otras aplicaciones que están asociados con las computadoras. La alfabetización digital, por último, se puede referir a la comprensión de cómo funcionan los ordenadores y a la facilidad de operar con ellos.

En lo que sigue hablaremos más de pensamiento computacional y de las iniciativas necesarias para que esta nueva alfabetización digital se produzca: el estudio y la investigación de un nuevo currículo escolar y el análisis de propuestas para la formación de maestros y profesores.

Dilema

Como hemos señalado, una vez aceptada la necesidad de que los niños, desde sus primeras etapas de desarrollo, adquieran las habilidades del pensamiento computacional, constatamos que se han producido dos alternativas que constituyen los términos de un dilema.

Por un lado, la respuesta más frecuente y más simple, a fuerza de ser una respuesta mecánica, ha consistido en favorecer el aprendizaje de las técnicas ya consagradas de programación y de sus lenguajes de forma progresiva o de lenguajes cada vez más complejos: primero juegos con estructuras constructivas de lenguajes —bucles, iteraciones, etc.—, luego lenguajes sencillos utilizados para resolver problemas divertidos, de juegos, etc., y posteriormente ir aumentando la dificultad, sin señalar que en cada uno de estos pasos hemos ido dejando gente por el camino y al final nos quedamos con la élite *friki* de los programadores de siempre. En esencia se trataba de proponer a los niños tareas de programar desde las primeras etapas. De manera que la progresión estuviese en la dificultad de las tareas y en su carácter motivador, desde las más sencillas y más lúdicas a las más complejas y aburridas. Se vincula aprendizaje con la respuesta a un estímulo, no con las características de aprendizaje y cognitivas del niño, en la tradición más clásica del conductismo.

Este es el tipo de planteamiento que está detrás de la idea simple, pero de eficiencia productiva de obtener individuos que hagan muchas líneas de programa y capaces de hacerlas muy rápidamente, sin pensar previamente mucho en el problema a resolver, sin diagramas, sin diseño. En definitiva, es la idea que hay detrás de los concursos de programación, de enseñar a programar a través de juegos, etc.

Como hemos dicho, es un planteamiento competitivo que deja afuera a muchos niños y, como corolario, hace poco deseable para muchos ser programador, o al menos les da una imagen de frikis a los programadores, unos tipos raros con un perfil poco atractivo. Este puede llegar a ser un planteamiento por muchas razones excluyente.

Luego está el otro término del dilema. Lo importante según esta visión no es que los alumnos escriban programas, sino desarrollar en ellos actitudes y capacidades para enfrentar los problemas en las situaciones previas no solo al código, sino incluso al algoritmo. De manera que la organización de la solución, a partir de la visión del problema y de las herramientas cognitivas y metacognitivas, de que dispone para resolverlo, le fluya. Para ello lo importante es que los maestros sepan cómo los alumnos se representan la realidad, su mundo de objetivos y expectativas, pero también cómo funcionan los mecanismos de aprendizaje en estos casos y cuáles son las formas de trabajar exitosas de los que tienen éxito en hacer programas potentes.

Así, pues, lo importante no es el *software* que escriben, sino lo que piensan cuando lo escriben. Y, sobre todo, la forma en que lo piensan.

Conocer este mundo de ideas, de procedimientos y de representaciones, cómo opera, constituye el principio básico del pensamiento computacional. Y cualquier otro conocimiento, como memorizar a la perfección las reglas que constituyen la sintaxis y las primitivas (la gramática) de cualquier lenguaje de programación, no le sirve de nada a los alumnos si no pueden pensar en buenas maneras de aplicarlas.

Este es pues el segundo término del dilema en el que hay que decidir.

Desgraciadamente, como veremos en la segunda parte de este apartado, la modalidad por la que se ha optado de forma más frecuente ha sido la de enseñar a programar directamente. Esa ha sido también la que se ha empezado a utilizar en Uruguay. Así, por ejemplo, se

ha hecho en la Comunidad de Madrid[5] (Valverde y otros, 2015). Simplemente se describen los contenidos y destrezas de programación a conseguir. La novedad consiste en introducir un bloque de contenidos, de forma convencional (no diferente de como se hace con el resto) dentro de las asignaturas de libre configuración autonómica.

Así, en el punto c (1.º y 2.º) del artículo 8, que remiten al Anexo III de la orden, dice:

> 1.º (...) se establecen los contenidos, los criterios de evaluación y los estándares de aprendizaje evaluables de las materias Tecnología, Programación y Robótica, Ampliación de Matemáticas: Resolución de Problemas y Taller de Música.
> 2.º El Departamento de Coordinación Didáctica de Tecnología se responsabilizará de la impartición de la materia Tecnología, Programación y Robótica con carácter prioritario. Secundariamente, podrán impartirla profesores de la especialidad de Informática, siempre que previamente estén cubiertos en su totalidad los horarios de la Familia Profesional de Informática y Comunicaciones.

El patrón es el mismo que cualquier otra disciplina, pero en este caso se hace, además, de forma subsidiaria.

Lo que subyace en la redacción, en este como en otros casos, es un abordamiento convencional: conducir a los alumnos de secundaria por el camino más áspero, el de los contenidos y estándares de aprendizaje, pero en este caso, los de la programación *per se*. En este contexto no se proporcionan, ni se mencionan, otro tipo de ayudas o de claves para conseguir los efectos de que hablamos en el apartado anterior. Está muy lejos, cuando no en otra esfera, de lo que se plantea como pensamiento computacional.

Algunos de los resultados de esta forma de operar pueden ser la exclusión de los que no tienen el don o la habilidad innata para programar directamente. De aquellos alumnos que solo ante la presencia del problema a resolver se les activan mecanismos para con los ele-

5 Decreto 48/2015, de 14 de mayo, del Consejo de Gobierno, por el que se establece para la Comunidad de Madrid el currículo de la Educación Secundaria Obligatoria. Disponible en: <http://www.bocm.es/boletin/CM_Orden_BOCM/2015/05/20/BOCM-20150520-1.PDF>.

mentos de programación (de los que proporciona un lenguaje específico: sintaxis, órdenes, procedimientos, filosofía propia del lenguaje) para elaborar el código. Esta dinámica conduce a la creación de estereotipos y perfiles de alumnos con facilidad para la programación, y del tópico de que la programación es solo cosa de los programadores.

Hay otro efecto derivado. Si se aprende a programar como algo asociado a un lenguaje, es posible que no se produzca la transferencia y que en futuras ocasiones o en distintos contextos no se pueda repetir el proceso. Esto hace que la competencia profesional sea menos y que la inserción no se produzca con toda eficacia que podría ser si se hiciera vinculado a operaciones cognitivas superiores. Las asociaciones profesionales se quejan de que las empresas contraten a informáticos de forma efímera. Sin reparar que es posible que suceda, porque han aprendido de forma efímera, como algo vinculado a lenguajes y a programas pasajeros. De manera que, en un futuro próximo, cuando cambie el programa o la versión actual, no tendrán flexibilidad mental para adaptarse a nuevos entornos, no solo de programación, sino de problemas. Esto no sucede, y las empresas lo saben, si contratan a titulados más familiarizados con elementos de pensamiento heurísticos o de otro tipo de entre los glosados en el repertorio del trabajo de referencia (Zapata-Ros, 2015). Nos referimos a la facilidad con que estas empresas recurren a matemáticos o físicos, que sí tienen esa competencia de abstraer los procedimientos para distinguir aspectos invariantes de la resolución de problemas en entornos cambiantes.

Esto no ocurre así cuando se empieza por desarrollar habilidades generales previas que se puedan activar en situaciones de elaboración de códigos o de resolución de otros problemas. Podemos afirmar que sí existen referencias (Raja, 2014) de investigaciones que ponen de manifiesto que cuando se empieza por enseñar el pensamiento computacional en vez de por la elaboración de códigos, desvinculando la iniciación en el aprendizaje a ser diestros con el ordenador, tal como se entiende habitualmente, se evita el principio de discriminación que hace que algún tipo de niños y de niñas se inhiban.

Una última derivación del tema es que esta forma de organizar el aprendizaje supone un principio de democratización en el acceso a este conocimiento, que de esta forma no queda restringido a las élites de programadores. De manera que incluso los que en un futuro pue-

den ser bibliotecarios, médicos o artistas, pueden ser también buenos programadores. Y, por ende, podría ampliar la base de conocimiento que se vuelca al mundo de la computación, lo que constituye el motor y el combustible de la sociedad del conocimiento.

Con todos estos elementos de juicio y referencias teóricas, elaboramos una definición basada en la necesidad de dominio teórico específico del pensamiento computacional en las teorías del aprendizaje y en la necesidad de un currículo. Ese trabajo lo hicimos entre 2014 y 2018 y constituye el contenido del artículo (Zapata-Ros, 2015) y del libro (Zapata-Ros, 2018) dedicados a este tema y que son referencia continua en este trabajo.

Lo hicimos siguiendo los trabajos teóricos de Eggleston (1980) sobre la idea y la construcción de un currículo.

En el artículo publicado en el número monográfico de RED dedicado a pensamiento computacional (Zapata-Ros, 2015), nos basábamos en la necesidad de contar con un corpus curricular en torno a una serie de habilidades y de elementos más o menos complejos de desarrollo cognitivo. Estos elementos, análisis descendente, ascendente, metacognición, pensamiento lógico, ensayo-error, heurística, pensamientos divergentes, resolución de problemas, etc., al mismo tiempo señalaban lo que debían ser los resultados de aprendizaje, los logros y, por tanto, el resto de dimensiones del diseño instruccional, y lo definían.

Esto puede verse en la Figura 1 de este artículo.

Por último, cabe decir que así lo planteamos, en el contexto de un análisis y de una elaboración interdisciplinar, teniendo presentes las implicaciones de este conjunto de ideas para la redefinición de un dominio teórico específico dentro de las teorías del aprendizaje. Y, desde luego, con la intención de definir descriptivamente, en un primer acercamiento, un currículo adecuado a esos dominios conceptuales para las distintas etapas educativas y para la capacitación de maestros y profesores.

Lo hicimos en una primera aproximación, con las limitaciones de un tratamiento general. Posteriormente lo estamos ampliando y desarrollando como construcción teórica (en el apartado anterior hemos propuesto una nueva componente, la de pensamiento bayesiano, que desarrollaré en futuras elaboraciones). Eso, por un lado. Por otro, intentamos proponer desarrollos prácticos. En una primera parte lo hemos hecho pensando en una modalidad de pensamiento computacio-

nal que merece un particular interés, por no ser el que habitualmente se trabaja y por estar alumnos y docentes bastante alejados de él. Nos referimos al pensamiento computacional desenchufado o *unplugged computational thinking* (Zapata-Ros, 2019) y a los profesores y profesoras de educación infantil y primeros niveles de primaria.

Pensamiento computacional desconectado

La idea de pensamiento computacional desenchufado (*computational thinking unplugged*) hace referencia al conjunto de actividades y su diseño educativo, que se elaboran para fomentar en los niños, en las primeras etapas de desarrollo cognitivo (educación infantil, primer tramo de la educación primaria, juegos en casa con los padres y los amigos), habilidades que luego pueden ser evocadas para favorecer y potenciar un buen aprendizaje del pensamiento computacional en otras etapas o en la formación técnica, profesional o en la universitaria incluso. Actividades que se suelen hacer con fichas, cartulinas, juegos de salón o de patio, juguetes mecánicos, etc.

Para disipar posibles equívocos desde el principio (Zapata-Ros, 2019) utilicé la expresión de *pensamiento computacional desenchufado* (que conceptualmente es algo distinto de desconectado y sin sus connotaciones). En inglés ya existe la expresión *computer science unplugged*. La ha utilizado Tim Bell (Bell y Vahrenhold, 2018) de la Universidad de Canterbury de Nueva Zelanda, con algunas diferencias prácticas y conceptuales que ya veremos. En definitiva, queremos poner de relieve que lo importante es que los niños no jueguen con trastos, no solo digitales, sino ni tan siquiera enchufados, y que a pesar de ello adquieran pensamiento computacional. O quizás por ello.

En este apartado vamos a abordar la necesidad y la conveniencia de trabajar aspectos del aprendizaje previo, convergente con el pensamiento computacional y necesario para él, desde las primeras etapas del desarrollo cognitivo de los individuos. Lo vamos a justificar desde el punto de vista de la teoría del aprendizaje y del de una pedagogía necesaria a ese fin. Vamos a hacerlo desde el punto de vista experiencial.

Para ello, hemos propuesto a título indicativo en un par de actividades y vamos a continuar con investigaciones más consistentes hasta elaborar propuestas más completas que incluyan guías y otros mate-

riales de apoyo para maestros y maestras de estos ciclos. Con ello intentaremos, como no puede ser menos en estos casos, dotarles de valor pedagógico en el sentido de que las propuestas tengan un diseño instruccional, con objetivos del tipo de los tratados en el apartado anterior, pero propuestos para el pensamiento computacional, orientando con ello el diseño hacia la práctica del pensamiento computacional desenchufado.

Hemos sostenido en trabajos anteriores (Zapata-Ros, 2014a)[6] esa necesidad sobre la base de una perspectiva y de una opción, desde el punto de vista de que se trata de una nueva alfabetización y que, como tal, el pensamiento computacional debe constituir globalmente una competencia clave en igualdad a como lo son las otras, las competencias claves de la alfabetización tradicional, la de la época industrial: la lectura, la escritura, el cálculo elemental y la geometría.

Principio de activación

No podemos esperar que las destrezas del pensamiento computacional aparezcan de forma espontánea en el mismo momento en que se necesitan, en los estudios de grado o de secundaria superior.

Las habilidades que son necesarias para la programación de algoritmos complejos, las destrezas del pensamiento computacional en todo su vigor, es decir, las que son necesarias para la programación de ordenadores, para resolver problemas o para organizar el proceso y la circulación de datos, así como para que los ordenadores realicen tareas, las tareas para las que están construidos, estas habilidades, no podemos esperar a que aparezcan o se manifiesten de forma espontánea y que lo hagan en el mismo momento de necesitarlas en los estudios de grado de Computación o de Ingeniería Informática, en la etapa de madurez del alumno que corresponde a esa edad, ni tan siquiera en la etapa de desarrollo del pensamiento abstracto, en la secundaria posobligatoria o incluso en secundaria obligatoria.

En este sentido, estas habilidades no son distintas de otras habilidades complejas que tienen que ver con el desarrollo de los individuos,

6 Disponible en: <https://computational-think.blogspot.com/2014/11/por-que-pensamiento-computacional-i.html>.

que se adquieren de forma progresiva y que solo son utilizables en forma operativa en su última fase.

Esta naturaleza del aprendizaje, el enlace de las situaciones en que se produce con los objetivos finales a través de etapas, niveles y condiciones de aprendizaje, es la que justifica el diseño instruccional y de ello no se libra la adquisición de las habilidades computacionales ni, siendo distinto, el pensamiento computacional. Los aprendizajes complejos se dividen, se fraccionan en aprendizajes más simples, más cercanos a las capacidades de los individuos y más lejanos del momento que adquieren su mayor eficiencia o su mayor operatividad práctica, incluso en el caso de que nunca lo alcancen porque no exista, como sucede frecuentemente con las matemáticas, cuyo sentido lo adquieren en etapas muy diferidas o como habilidades auxiliares de otras. Así sucede con los conocimientos y las competencias claves y con las habilidades básicas.

En este punto es donde obtienen su justificación en las teorías del aprendizaje, en el principio de activación y en la forma en cómo transitar desde que se adquieren las habilidades hasta que son útiles en su destino final. Este tránsito y la forma de organizarlo es lo que constituye la base del diseño instruccional. Por tanto, son dos núcleos claves que están en la justificación en la teoría del aprendizaje y en la base de una pedagogía del pensamiento computacional: el principio de activación y el diseño instruccional.

En esta parte nos vamos a dedicar exclusivamente al principio de activación. Dejaremos para otra ocasión el diseño instruccional. La otra cuestión, la consideración del pensamiento computacional como competencia clave de la nueva alfabetización tampoco la abordaremos en este punto, es una elaboración o una consecuencia elaborada del principio de activación.

Así pues, vamos a justificar con este principio la necesidad y la conveniencia de trabajar aspectos del aprendizaje previos, convergentes con el pensamiento computacional y necesarios para él, desde las primeras etapas del desarrollo cognitivo de los individuos. Es lo que va a justificar después qué actividades y cómo se organizan juegos en la infancia para que habilidades de secuenciación o de encaje, entre objetos computacionales o entre variables y tipos de datos, por ejemplo, se activen y fluyan en la fase de resolver problemas con algoritmos y pro-

gramas en las etapas de enseñanza profesional o universitaria. Esto, obviamente, sería una ejemplificación extrema. En un caso más normal, la adquisición se produciría de una forma más progresiva a través de las distintas etapas educativas, los niveles e incluso dentro de estos y de los módulos y unidades instruccionales que los componen.

En su trabajo, Merrill (2002) desarrolla lo que llama unos principios fundamentales del aprendizaje (*first principles*) y lo hace decantando los principios subyacentes en los que hay consensos, en los que hay un acuerdo esencial, en todas las teorías y que previamente ha identificado. Ese trabajo está expuesto y desarrollado en *First principles of instruction* (Merrill, 2002), en *Educational technology research and development*, incluido como capítulo en el tercer volumen de los libros de Reigeluth *Instructional-design theories and models: Building a common knowledge base* (Merrill, 2009). Y, de forma resumida, en *First principles of instruction: A synthesis* (Merrill, 2007). También son glosados como base del nuevo paradigma instruccional de Reigeluth, cuya versión oficial se puede encontrar en RED número 50, en el artículo «Teoría instruccional y tecnología para el nuevo paradigma de la educación» (Reigeluth, 2016).

En este último trabajo, Reigeluth (2016) distingue entre principios universales y escenarios particulares. Cuando aplicamos con mayor precisión un principio o un método instruccional, por lo general descubrimos que hace falta que este sea diferente para diferentes situaciones y perfiles de aprendizaje, o una mayor precisión para obtener objetivos contextualizados y personalizados. Reigeluth (1999) se refirió a los factores contextuales que influyen en los efectos de los métodos como «escenarios».

Un principio fundamental (Merrill, 2002) o un método básico según Reigeluth (1999), es un aserto que siempre es verdadero bajo las condiciones apropiadas independientemente del programa o de la práctica en que se aplique, que de esta forma dan lugar a un método variable.

Teniendo en cuenta como el mismo Merrill (2002) las define:

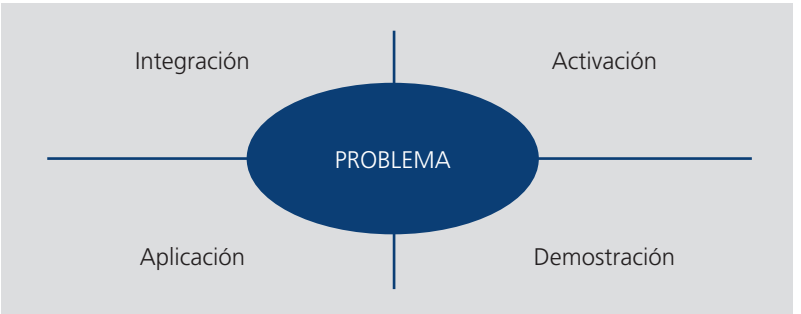
> Una práctica es una actividad instruccional específica. Un programa es un enfoque que consiste en un conjunto de prácticas prescritas. Las prácticas siempre implementan o no implementan los principios subyacentes ya sea que estos principios se especifiquen

o no. Un enfoque de instrucción dado solo puede enfatizar la implementación de uno o más de estos principios de instrucción. Los mismos principios pueden ser implementados por una amplia variedad de programas y prácticas.

De esta forma, Merrill propone un conjunto de cinco principios instruccionales prescriptivos (o principios fundamentales) que mejoran la calidad de la enseñanza en todas las situaciones (Merrill, 2007, 2009) y tienen que ver con la centralidad de la tarea, la activación, la demostración, la aplicación y la integración.

Para ello Merrill (2002) propone un esquema en fases como el más eficiente para el aprendizaje, que se centra en el problema y crea un entorno que implica al alumno para la resolución de cualquier problema En cuatro fases distintas que se presentan en la siguiente figura.

Figura 2. Fases de instrucción



Fuente: Traducido de Merrill, 2002.

Las fases son (a) activación de experiencia previa, (b) demostración de habilidades, (c) aplicación de habilidades y (d) integración de estas habilidades en actividades del mundo real.

La figura anterior proporciona un marco conceptual para establecer y relacionar los principios fundamentales de la instrucción. De ellos, uno tiene que ver con la implicación y la naturaleza real del problema, así percibida por el alumno, y los cuatro restantes para cada una de las fases.

Estos cinco principios enunciados en su forma más concisa (Merrill, 2002) son:

1. El aprendizaje se promueve cuando los estudiantes se comprometen a resolver problemas del mundo real. Es decir, el aprendizaje se promueve cuando es un aprendizaje centrado en la tarea.
2. El aprendizaje se favorece cuando existen conocimientos que se activan como base para el nuevo conocimiento.
3. El aprendizaje se promueve cuando se centra en que el alumno debe demostrar su nuevo conocimiento. Y el alumno es consciente de ello.
4. El aprendizaje se promueve igualmente cuando se centra en que el aprendiz aplique el nuevo conocimiento.
5. El aprendizaje se favorece cuando en el nuevo conocimiento se tiende a que se integre en el mundo del alumno.

De todos estos principios, el que justifica sobremanera la inclusión del pensamiento computacional, como *pensamiento computacional desenchufado*, en las primeras etapas, es el *principio de activación*. En él nos vamos a centrar y no solo en su aplicación para el diseño instruccional en la fase de activación, en la que el conocimiento existente se activa, sino en las fases en las que se crean los conocimientos y habilidades que son activados y en cómo hacerlo para que la activación sea más eficiente.

En su trabajo *Teoría instruccional y tecnología para el nuevo paradigma de la educación*, Reigeluth (2016: 4) caracteriza el principio de activación de manera que:

- El diseño educativo de actividades, organización, recursos, etc., debe ser tendente a activar en los alumnos estructuras cognitivas relevantes, haciéndoles recordar, describir o demostrar conocimientos o experiencias previas que sean relevantes para él.
- La activación puede ser social. La instrucción debe lograr que los estudiantes compartan sus experiencias anteriores entre ellos.
- La instrucción debe hacer que los estudiantes recuerden o adquieran una estructura para organizar los nuevos conocimientos.

Los trabajos de Merrill (2002) y Reigeluth (2016) hacen énfasis en la fase de evocación, pero no en la fase de crear estructuras cognitivas,

experiencias y en general conocimientos y habilidades que puedan ser evocados. Ni tampoco en crear una pedagogía o un diseño educativo que incluya, o tendente a favorecer, elementos cognitivos de enlace que promuevan la evocación. Tampoco a fomentar la investigación sobre estos temas o a investigar qué tipos de enlaces fortalecen más las estructuras cognitivas de enlace y de evocación.

A partir de lo que dicen los ítems anteriores sobre las características del diseño instruccional que implica el principio de activación, podemos concluir que la instrucción, en la fase de crear elementos para ser evocadora, debe:

- Crear estructuras cognitivas que incluyan conocimientos, habilidades, elementos de reconocimiento que permitan distinguir al alumno y otorgar relevancia en su momento de forma fluida a esas habilidades para conseguir su efectividad en ese momento a partir de elementos contextuales, metáforas, etc.
- Otorgar a esas habilidades elementos de reconocimiento que permitan la evocación.
- Asociar esas habilidades a tareas que tengan similitud con las que en su momento serán necesarias para resolver los problemas a los que ayuda la evocación. En nuestro caso, a los problemas computacionales o habilidades propias a los elementos que constituyen el pensamiento computacional.
- Diseñar instruccionalmente las actividades que sean relevantes para evocar los elementos de pensamiento computacional (Pérez-Paredes y Zapata-Ros, 2018).
- Propiciar experiencias de aprendizaje compartido en las primeras etapas y hacer que esos grupos y experiencias sociales sean estables a lo largo del tiempo. Las experiencias compartidas crean elementos de activación a través de grupos o de pares de alumnos. El propiciar grupos y claves de comunicación, de lenguaje, y que esos grupos sean estables a lo largo del tiempo aumenta la potencia de evocación.
- Crear estructuras cognitivas en los alumnos capaces de recomponerse y aumentar en el futuro. Dotar a los conocimientos y habilidades de referencias y de metadatos que permitan ser recuperados mediante evocación.

Debe potenciarse una pedagogía que establezca valores en estas ideas y principios para las primeras etapas.

El principio de activación es clave para cuando se diseña la educación infantil y del primer ciclo de primaria, teniendo en el horizonte los aprendizajes futuros y también el pensamiento computacional.

Merrill ha sido quien más lo ha trabajado, pero no solo.

Como señalamos en otro trabajo (Zapata-Ros, 2018c), Bawden (2008) habla de habilidades de recuperación y remite a lo expuesto en otro trabajo anterior (Bawden, 2001). En las habilidades que señala se constatan ideas como la de construir un «bagaje de información fiable» de diversas fuentes, la importancia de las habilidades de recuperación, utilizando una forma de «pensamiento crítico» para hacer juicios informados sobre la información recuperada y para asegurar la validez e integridad de las fuentes de internet, leer y comprender de forma dinámica y cambiante material no secuencial. Y así una serie de habilidades donde, como novedad, se introducen las *affordances* de conocimiento en entornos sociales y de comunicación en redes, y la idea de relevancia. Solo que en este caso son habilidades sobre el proceso de la información y su posterior recuperación. Obviamente, no son habilidades para desarrollar en esta etapa. Sin embargo, sí sería interesante indagar sobre la recuperación de habilidades que se desarrollan mediante juegos de infancia, como son habilidades cinestésicas.

Pensamiento computacional en la infancia

Desde junio de 2014 hemos argumentado, aportando muy diversas razones, acerca de por qué debían incluirse en el currículo de educación infantil y de primaria actividades de pensamiento computacional. He aquí un resumen.

En el apartado anterior hemos hablado del principio de activación. Basándonos en él hemos sostenido desde hace tiempo la necesidad de favorecer aprendizajes a través de juegos y de otras actividades que estén cognitiva o cinestésicamente conectadas con las habilidades de computación. También hemos sostenido que esto es fácilmente asimilable por el público no especializado (Zapata-Ros, 2014b): al igual que sucede con los deportistas y con los músicos, para que programen bien o simplemente para que no se vean excluidos de esta

nueva alfabetización, que es el pensamiento computacional en la sociedad del conocimiento, debería fomentarse en los niños, desde las primeras etapas, competencias que puedan ser activadas en otras etapas de desarrollo y en otras fases de la instrucción, correspondientes a las etapas del pensamiento abstracto y a las de rendimiento profesional. Y citamos el desarrollo de determinadas habilidades, como son las de seriación, encaje, modularización, organización espacial, etc., que, en estudios posteriores de grado, de bachillerato o de formación profesional, pudiesen ser activadas para elaborar procedimientos y funciones en la creación de códigos o para desarrollar algoritmos propios de esta etapa.

Referencias bibliográficas

Bates, A. (2018): *Online learning and disruptive change at the UK Open University.* Disponible en: <https://www.tonybates.ca/2018/05/02/online-learning-and-disruptive-change-at-the-uk-open-university/>.

Bawden, D. (2001): «Information and digital literacies: a review of concepts». *Journal of Documentation*, 57(2), pp. 218-259.

Bawden, D. (2008): *Origins and concepts of digital literacy. Digital literacies: Concepts, policies and practices*, pp. 17-32. Disponible en: <http://sites.google.com/site/colinlankshear/DigitalLiteracies.pdf#page=19>.

Bell, T.; Vahrenhold, J. (2018): «CS desenchufado: ¿cómo se usa y cómo funciona?». *Aventuras entre límites inferiores y altitudes superiores*, pp. 497-521. Springer, Cham.

Bell, T.; Alexander, J.; Freeman, I.; Grimley, M. (2009): «Computer science unplugged: School students doing real computing without computers». *The New Zealand Journal of Applied Computing and Information Technology,* 13(1), pp. 20-29. Disponible en: <http://www.computingunplugged.org/sites/default/files/papers/Unplugged-JACIT2009submit.pdf>.

Bundy, A. (2007): Computational thinking is pervasive. J. Scient. Pract. Comput.1, pp. 67-69. Google Scholar.

Coll, C. (2019): «Presentación y prólogo del libro: El pensamiento computacional. Análisis de una competencia clave». *Revista de Educación a Distancia*, 19. Disponible en: <https://revistas.um.es/red/article/view/395281>.

Eggleston, J. (1982): *Sociología del currículum*. Troquel, Buenos Aires.

Grover, S. (2018): ***The 5th 'C' of 21st century skills?*** *Try computational thinking (not coding. Retrieved from EdSurge News.* Disponible en: <https://edtechbooks.org/-Pz>.

Grover, S.; Pea, R. (2013): «Computational thinking in K–12: A review of the state of the field». *Educational researcher*, 42(1), pp. 38-43. Disponible en: <https://journals.sagepub.com/doi/pdf/10.3102/0013189X12463051?casa_token=R5cisYlHapoAAAAA:iELmqyonX5dfeBP2ZaTxflrNEjD-Y8N9_kHKQ_5labiLw5As3cKpugJalkod4px4gjbbELG9t2XCWBQ>.

Grover, S.; Pea, R. (2017): Computational Thinking: A competency whose time has come*. Computer science education: Perspectives on teaching and learning in school*, 19. Disponible en: <https://www.researchgate.net/profile/Shuchi_Grover/publication/322104135_Computational_Thinking_A_Competency_Whose_Time_Has_Come/links/5a457813a6fdcce1971a5ce5/Computational-Thinking-A-Competency-Whose-Time-Has-Come.pdf>.

Merrill, D. (2000): *Instructional Strategies and Learning Styles: Which takes Precedence? Trends and Issues in Instructional Technology*, R. Reiser and J. Dempsey (eds.). Prentice Hall.

Merrill, M. D. (2002): «First principles of instruction». *Educational technology research and development*, 50(3), pp. 43-59. Disponible en: <https://link.springer.com/article/10.1007/BF02505024>, <https://mdavidmerrill.files.wordpress.com/2019/04/firstprinciplesbymerrill.pdf>.

Merrill, M. D. (2007): «First principles of instruction: A synthesis». In R. A. Reiser & J. V. Dempsey (eds.), *Trends and issues in instructional design and technology* (2nd ed., pp. 62-71). Upper Saddle River, NJ: Merrill/Prentice-Hall.

Merrill, M. D. (2009): «First principles of instruction». In C. M. Reigeluth & A. A. Carr-Chellman (eds.), *Instructional-design theories and models: Building a common knowledge base* (Vol. III, pp. 41-56). Routledge, New York.

Pérez-Paredes, P.; Zapata-Ros, M. (2018): «El pensamiento computacional, análisis de una competencia clave». Scotts Valley, CA, USA: *Createspace Independent Publishing Platform*, p. 63. Disponible en: <https://www.amazon.es/pensamiento-computacional-analisis-competenciaclave/dp/1718987730/ref=sr_1_1>.

Raja, T. (2014): «We Can Code It! Why computer literacy is key to winning the 21st century». *Mother Jones*. Disponible en: <http://www.motherjones.com/media/2014/06/computer-science-programming-code-diversity-sexism-education>.

Reigeluth, C. M. (1999): «What is instructional-design theory and how is it changing?». In C. M. Reigeluth (ed.), *Instructional-design theories and models: A new paradigm of instructional theory* (Vol. II, pp. 5-29). Mahwah, NJ: Lawrence Erlbaum Associates.

Reigeluth, C. M. (2012): «Instructional theory and technology for the new paradigm of education». RED, *Revista de Educación a Distancia*, 32, pp. 1-18. Disponible en: <http://www.um.es/ead/red/32/reigeluth.pdf>.

Reigeluth, C. M. (2016): «Teoría instruccional y tecnología para el nuevo paradigma de la educación». RED. *Revista de Educación a Distancia*, 50. Disponible en: <http://www.um.es/ead/red/50>.

Reigeluth, C. M.; Carr-Chellman, A. A. (2009): «Situational principles of instruction». In C. M. Reigeluth & A. A. Carr-Chellman (eds.), *Instructional-design theories and models: Building a common knowledge base* (Vol. III, pp. 57-68). Routledge, New York.

Stehr, N. (2003): «Wissenspolitik oder die gesellschaftliche Disziplinierung neuer Erkenntnisse». In Herbert Kubincek, Dieter Klumpp & Alexander Roßnagel (eds.), *Next Generation Information Society? Notweindigkeit einer*

Neuorientiierung. Mössingen-Talheim: Talheimer Verlag, pp. 318-330. Disponible en: <http://www.researchgate.net/publication/266615013>.

UK Government (2015): Green Paper: Digital Literacy. 21st Century Competencies for Our Age: The Digital Age. The Fundamental Building Blocks of Digital Literacy From Enhancement to Transformation.

Valverde-Berrocoso, J.; Fernández-Sánchez, M. R.; Garrido-Arroyo, M.C. (2015): «El pensamiento computacional y las nuevas ecologías del aprendizaje». RED, *Revista de Educación a Distancia*, 46. Disponible en: <http://www.um.es/ead/red/46>.

Wing, J. M. (2006): Computational thinking. it represents a universally applicable attitude and skill set everyone, not just computer scientists, would be eager to learn and use. Commun. ACM 49(3). Disponible en: <https://doi.org/10.1109/vlhcc.2011.6070404>.

Wing, J. M. (2008): Computational thinking and thinking about computing. The Royal Society Publishing. Disponible en: <https://doi.org/10.1098/rsta.2008.0118>.

Zapata-Ros, M. (2013): «La sociedad postindustrial del conocimiento: Un enfoque multidisciplinar desde la perspectiva de los nuevos métodos para organizar el aprendizaje». *CreateSpace Independent Publishing Platform*. Retrieved from Capítulo, 9, 260.

Zapata-Ros, M. (2014a): *Por qué el pensamiento computacional (I).* Blog Pensamiento computacional y alfabetización digital/Computational thinking and computer literacy. Disponible en: <https://computational-think.blogspot.com/2014/11/por-que-pensamiento-computacional-i.html>.

Zapata-Ros, M. (2014b): *Por qué el pensamiento computacional (IV). Un dominio teórico específico en las teorías del aprendizaje y un currículum.* Blog Pensamiento computacional y alfabetización digital/Computational thinking and computer literacy. Disponible en: <https://computational-think.blogspot.com/2014/11/por-que-el-pensamiento-computacional-iv.html>.

Zapata-Ros, M. (2014c): Coding y pre-coding. Blog Microposts, Tumblr Disponible en: <http://miguelzapataros.tumblr.com/post/89143450350/coding-y-pre-coding>.

Zapata-Ros, M. (2014d): *Pensamiento computacional y alfabetización digital (I).* RED. El Aprendizaje en la Sociedad del Conocimiento. Disponible en: <https://red.hypotheses.org/776>.

Zapata-Ros, M. (2015): «Pensamiento computacional: Una nueva alfabetización digital». Blog RED. *Revista de Educación a Distancia,* 46, pp. 1-47. Disponible en: <http://www.um.es/ead/red/46>.

Zapata-Ros, M. (2018a): *Pensamiento computacional. Una tercera competencia clave (IV): Un dominio teórico específico en las teorías del aprendizaje y un currículum.* Blog RED. El Aprendizaje en la Sociedad del Conocimiento. Disponible en: <https://red.hypotheses.org/1079>.

Zapata-Ros, M. (2018b): *Pensamiento computacional. Una tercera competencia clave* (I). Blog RED. El Aprendizaje en la Sociedad del Conocimiento. Disponible en: <https://red.hypotheses.org/1059>.

Zapata-Ros, M. (2018c): *El pensamiento computacional en la transición entre culturas epistemológicas.* Blog RED. El Aprendizaje en la Sociedad del Conocimiento. Disponible en: <https://red.hypotheses.org/1235>.

Zapata-Ros, M. (2019): «Computational Thinking Unplugged». *Education in the Knowledge Society*, 20, pp. 1-29. Disponible en: <https://pdfs.semanticscholar.org/8ea2/7254a97161a9c75acbc26a1350cefdd5637c.pdf>.