Versión Abierta Español – Portugués
de la
Revista Iberoamericana de Tecnologías del/da Aprendizaje/Aprendizagem

Una publicación de la Sociedad de Educación del IEEE (Capítulo Español)
Uma publicação da Sociedade de Educação do IEEE (Capítulo Espanhol)

FEB. 2018 VOL. 6 NÚMERO/NUMERO 1 (ISSN 2255-5706)

Editorial: Afianzando el caracter abierto en español/portugués de IEEE-RITA…….....Martín Llamas Nistal, Senior Member IEEE, Manuel Castro, Fellow IEEE, y María Luisa Carrió Pastor **i**

EDICIÓN ESPECIAL: Pensamiento Computacional
EDITORIAL: Pensamiento Computacional……...…… Francisco J. García-Peñalvo **1**

Pensamiento Computacional entre Filosofía y STEM. Programación de Toma de Decisiones aplicada al Comportamiento de "Máquinas Morales" en Clase de Valores Éticos…… .....,,,

CONSEJO/CONSELHO EDITORIAL

Presidente (Editor Jefe):
Martín Llamas Nistal,
Universidad de Vigo, España

Vicepresidente (Coeditor):
Manuel Castro Gil, UNED, España

Editor Asociado para lengua Portuguesa:
Carlos Vaz do Carvalho,
ISEP, Oporto, Portugal

Secretaría:
Gabriel Díaz Orueta, UNED, España

Miembros:

Manuel Caeiro Rodríguez,
Universidad de Vigo, España

Juan M. Santos Gago,
Universidad de Vigo, España

Francisco J. García Peñalvo,
Universidad de Salamanca, España

Humberto Ramón Álvarez Alvarado,
Universidad Tecnológica de Panamá y LACCEI, Panamá

Jorge Rodríguez,
Western Michigan University,
Kalamazoo, and LACCEI, MI,
EEUU.

José Simó Ten,
Univerisdad Politénica de Valencia, España

Julián Viejo Cortés,
Universidad de Sevilla, España

COMITÉ CIENTÍFICO

Alfonso Lago Ferreiro,
Universidad de Vigo, España

António José Mendes,
Universidade de Coimbra,
Portugal

Antonio Lopez Martin,
Universidad Pública de Navarra, España

António Vieira de Castro,
ISEP, Portugal

Arturo Molina,
Tecnológico de Monterrey,
México

Baltasar Fernández Manjón,
Universidad Complutense de Madrid,
España

Carlos Delgado Kloos,
Universidad Carlos III de Madrid, España

Claudio R. Brito,
International Institute of Education of
COPEC, Brasil

Fernando Pescador,
Universidad Politécnica de Madrid,
España

Gabriel Díaz Orueta, UNED,
España

Francisco J. Azcondo,
Universidad de Cantabria,
España

Francisco Javier Arcega Solsona,
Universidad de Zaragoza,
España

Francisco Jurado,
Universidad de Jaén,
España

Gustavo H. Rossi,
Universidad Nacional de La Plata,
Argentina

Ivan Esparragoza,
Penn State University, EEUU

J. Ángel Velázquez Iturbide,
Universidad Rey Juan Carlos,
España

Jaime Muñoz-Arteaga,
Universidad Autónoma de Aguascalientes, México

Jose Antonio Salvado,
Instituto Politécnico de Castelo Branco,
Portugal

José Bravo Rodríguez,
Universidad de Castilla-La Mancha,
España

José Carlos Lourenço Cuadrado, ISEP,
Portugal

Jose Carlos Meireles Monteiro Metrôlho,
Instituto Politécnico de Castelo Branco,
Portugal

José Carpio,
Universidad Nacional de Educación a Distancia (UNED), España

José Palazzo Moreira de Oliveira,
UFRGS, Brasil

Juan C. Burguillo,
Universidad de Vigo, España

Juan Quemada Vives,
Universidad Politécnica de Madrid,
España

Luis Anido Rifón,
Universidade de Vigo,
España

Luis Jaime Neri Vitela,
Tecnológico de Monterrey,
México

M.Felisa Verdejo,
Universidad Nacional de Educación a Distancia, España

Manuel José Fernández Iglesias,
Universidade de Vigo,
España

Manuel Ortega Cantero,
Universidad de Castilla - La Mancha,
España

Maria José Patricio Marcelino,
Universidade de Coimbra,
Portugal

Oscar Martínez Bonastre,
Universidad Miguel Hernández de Elche, España

Pedro C. C. Pimenta,
Universidade do Minho,
Portugal

Raul V. Ramirez Velarde,
Tecnológico de Monterrey,
México

Samuel Cruz-Lara,
Université de Lorraine,
Francia

Vitor Duarte Teodoro,
Universidade Nova de Lisboa,
Portugal

Xabiel García Pañeda,
Universidad de Oviedo,
España

Yannis Dimitriadis,
Universidad de Valladolid,
España

Afianzando el Caracter Abierto en Español/Portugués de IEEE-RITA

Martín Llamas Nistal, Senior Member IEEE, Manuel Castro, Fellow IEEE, y María Luisa Carrió Pastor

CÓMO REFERENCIAR ESTE ARTÍCULO:

Martín Llamas Nistal, Senior Member IEEE, Manuel Castro, Fellow IEEE, y María Luisa Carrió Pastor. ''…'', 2018
DOI: https://doi.org/...

IEEE-RITA nació en el año 2006 con una clara vocación de facilitar la publicación en español y portugués de artículos de calidad en el tema de tecnologías del aprendizaje, atendiendo principalmente a la comunidad iberoamericana. En el año 2013 se abre una nueva etapa [1] donde IEEE-RITA se convierte en una publicación oficial del IEEE, publicando en inglés a través de IEEE Xplore [2], pero manteniendo su vocación iberoamericana de emplear el español y el portugués como idiomas de trabajo. Así los autores pueden enviar manuscritos en español o portugués para ser revisados técnicamente en estos idiomas. Una vez aceptados son publicados en línea en versión abierta en el idioma recibido (español o portugués) a través de VAEP-RITA (Versión Abierta en Español o Portugués de IEEE-RITA) [3]. Posteriormente son traducidos al inglés por los auotres y revisados en este idioma para tener la mejor versión en inglés del manuscrito original en español o portugués. Una vez aceptados, estos artículos en inglés se publican también en línea en IEEE Xplore y son accesibles sólo a sus suscritores.

De esta manera IEEE-RITA facilita la publicación de artículos de calidad de Tecnologías del Aprendizaje en inglés a aquellos autores cuyo idioma nativo es el español o portugués (fundamentalmente el área iberoamericana), permitiendo que el envío del manuscrito sea en su idioma nativo, así como la posterior revisión hasta la aceptación del mismo. Los autores sólo tienen que pensar en la versión inglesa cuando ya saben que su artículo, desde el punto de vista técnico, es correcto, y la revisión que reciben por parte de IEEE-RITA a través del ETAC (English Technical Advisory Committee) está orientada a ayudar a los autores a conseguir la mejor versión en inglés de su artículo.

Consecuentemente IEEE-RITA consigue que sus artículos sean libremente accesibles a través de VAEP-RITA, sin coste alguno para los lectores, en su versión original en español o portugués, facilitando el acceso a literatura científica a la comunidad iberoamericana en general, al mismo tiempo que consigue la máxima difusión al publicar también en inglés, lingua franca de la ciencia en nuestra época, a través de IEEE Xplore.

El coste actual para los autores de publicar en VAEP-RITA la versión abierta en español o portugués es 150 euros (más impuestos si procede), y gratis para la versión en inglés en IEEE Xplore. Creemos que este coste es pequeño pero desearíamos que este coste de la versión abierta desapareciese, y estamos buscando financiación para conseguirlo. Esperamos poder reducir este coste (inicialmente fue de 175 euros) en años sucesivos y nuestro deseo es poder eliminar para todos los autores el coste de publicar en VAEP-RITA.

Pues bien, IEEE-RITA quiere afianzar y ahondar esta visión de acceso abierto en español o portugués a través de VAEP-RITA, y acceso mundial en inglés a través de IEEE Xplore.

Para ello, por una parte los artículos publicados en inglés a través de IEEE Xplore incluirán el enlace donde se puede encontrar libre y gratuitamente la versión en español o portugués del mismo en VAEP-RITA. Así mismo las versiones en español o portugués indicarán también que son la versión en español o portugués del correspondiente artículo publicado en IEEE-RITA, y así será referenciado.

Y por otra parte se permitirá lo que hasta hora sólo ocurría en casos excepcionales, y es el envío de manuscritos en inglés. En este caso, los manuscritos una vez aceptados deberán ser traducidos al español o portugués para su publicación en abierto en VAEP-RITA.

IEEE-RITA desde sus inicios ha estado involucrada en la publicación de versiones de artículos presentados a conferencias, en la firme creencia de que los congresos son una parte fundamental en el ciclo de publicación de la literatura científica.

Por todo ello queremos a través de este editorial clarificar las distintas formas de envío de manuscritos a IEEE-RITA, incluyendo las versiones extendidas de artículos presentados en congresos:

A. Manuscritos Originales Enviados en Español o Portugués.

El proceso de revisión que se sigue una vez recibido el manuscrito es el siguiente:

1. Se comprueba su originalidad a través de herramientas de plagio y autoplagio. Si se detecta un porcentaje mayor o igual al 25%, serán directamente rechazados, además de las oportunas medidas que se puedan adoptar de acuerdo a las normas al respecto del IEEE [4].
2. Posteriormente serán revisados técnicamente en estos idiomas, por al menos tres revisores bajo la dirección de un editor asociado.

3. Una vez aceptado el manuscrito, el artículo será publicado provisionalmente en VAEP-RITA, y los autores deberán enviar la versión en inglés del mismo, que será comprobada también por las herramientas de plagio y autoplagio. La detección de un porcentaje de al menos un 25% implicará el rechazo no sólo de la versión en inglés, sino también de la versión original que será eliminada de VAEP-RITA, además de las medidas que se puedan adoptar de acuerdo a las normas al respecto del IEEE [4]. La revisión de esta versión en inglés será realizada por dos revisores: uno es el editor asociado que dirigió la revisión de la versión original en español o portugués para verificar que esta versión inglesa se corresponde con la aceptada en su versión original, y el otro es un revisor de la lengua inglesa dirigido por el ETAC cuya misión es revisar el artículo para conseguir la mejor versión posible en inglés. Una vez aprobado por estos dos revisores, el artículo puede ser publicado en inglés.

B. Manuscritos Originales Enviados en Inglés.

El proceso es análogo al de envío de originales en español o portugués, con la salvedad de que la revisión técnica es en inglés, y que una vez aceptado el manuscrito en inglés, los autores deberán traducir su artículo al español o portugués para que sea publicado en abierto en VAEP-RITA, comprobando que esta versión se corresponde con el original y la corrección del idioma. Sin esta versión en VAEP-RITA el artículo no será publicado en IEEE Xplore. Al igual que en el caso A, serán de aplicación todas las medidas de plagio y autoplagio: la detección de un porcentaje de al menos el 25% en la versión inglesa o española/portuguesa supondrá el rechazo directo del artículo tanto de IEEE-RITA como de VAEP-RITA, además de las medidas que se puedan adoptar de acuerdo a las normas al respecto del IEEE [4].

C. Manuscritos Extendidos de un Artículo Presentado en una Conferencia.

En este caso, estos manuscritos han de ser enviados en el mismo idioma en que se publicó ese artículo en la conferencia, y han de cumplir los siguientes requisitos:

- Tengan al menos un 30% de material original y nuevo significativo respecto del artículo publicado y presentado en la conferencia.
- Los autores comenten en la carta de presentación (Cover letter) que se trata de la extensión de un artículo presentado a una conferencia, indicando de qué conferencia se trata y donde fue publicado.
- Los autores deben hacer constar en la sección de Introducción de su artículo que es la extensión de un artículo presentado en la correspondiente conferencia, citando la correspondiente referencia bibliográfica de ese artículo, e indicando claramente en qué aspectos el artículo presentado extiende al original.
- Los autores deberán incluir como fichero adicional en su envío el artículo que fue publicado en la conferencia en cuestión, para que los revisores puedan constatar el material nuevo incluido en el artículo respecto del publicado en la conferencia.

Una vez comprobados estos requisitos, el proceso es similar al seguido en el anterior caso A, salvo la comprobación de plagio y autoplagio, que se permite tener un porcentaje de autoplagio mayor del 25% y siempre compatible con el material nuevo enviado (por ejemplo, si se detecta un 90% de autoplagio, significa que no tiene al menos ese 30% de contenido nuevo). Si esta concordancia no se cumple o se detecta plagio o autoplagio de otros artículos no declarados, el artículo será rechazado.

Además de todos estos aspectos, serán siempre de aplicación las normas del IEEE para los autores de sus publicaciones, que pueden consultarse en [4] y [5].

La publicación en VAEP-RITA supone el pago de la tasa correspondiente por parte de los autores y que podrá consultarse en la página web de VAEP-RITA [3].

Sólo nos queda agradecer a los miembros del consejo editorial, a los editores asociados, miembros del comité científico, a los miembros del ETAC, a nuestro asistente técnico de edición, y a los revisores que hacen posible que IEEE-RITA sea una realidad. También agradecer a la Sociedad de Educción del IEEE por su apoyo, al igual que a la Universidad de Vigo y FEUGA, especialmente en el soporte de VAEP-RITA.

Por último animar a los potenciales autores a enviar su manuscrito a IEEE-RITA. Todas las normas y guías de publicación estarán siempre convenientemente actualizadas en la página web de IEEE-RITA [6].

REFERENCIAS

[1] M. Llamas-Nistal, M. Castro, and M García Lorenzo, " Editorial IEEE-RITA: A New Stage" in IEEE Revista Iberoamericana de Tecnologias del Aprendizaje, vol. 8, no. 1, pp. 1, Feb. 2013. Available at: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6471757 Last accessed Jan 30, 2018

[2] IEEE-RITA: IEEE Revista IberoAmericana de Tecnologías del aprendizaje. Available in IEEE Xplore at: http://ieeexplore.ieee.org/xpl/RecentIssue.jsp?punumber=6245520

[3] Página web de VAEP-RITA. http://rita.det.uvigo.es/VAEPRITA Last accessed Jan 30, 2018

[4] IEEE Publication Services and Products Board Operation Manual. Available at: https://www.ieee.org/documents/opsmanual.pdf Last accessed Jan 30, 2018

[5] Author Information Page on Originality of Content. https://www.ieee.org/publications_standards/publications/rights/author_originality.html Last accessed Jan 30, 2018

[6] Web-site of IEEE-RITA. http://rita.det.uvigo.es Last accessed Jan 30, 2018

Martin Llamas Nistal, Editor in Chief of IEEE-RITA
Manuel Castro, Co-editor of IEEE-RITA
María Luisa Carrió Pastor, Chair of English Technical Advisory Committee (ETAC) of IEEE-RITA

Pensamiento Computacional

Francisco J. García-Peñalvo

CÓMO REFERENCIAR ESTE ARTÍCULO:

Francisco J. García-Peñalvo. "Computational thinking", 2018
DOI: https://doi.org/...

***Title*— Computational thinking**

***Abstract*— Information technologies are the base of the world infrastructure. In this social context, education, like any productive or service sector, is affected by technology. Faced with this reality, educational systems must prepare our young people to live in the digital world, for which they must know a new language without which they will become digital illiterates. Therefore, in school we should not only train in linguistic and numerical literacy, but also in digital literacy. So far, the effort has been oriented mainly to convert our young people into users of computer tools. This has gone from being necessary to being insufficient, because the use of software applications is a digital language that is obsolete in a time that is not proportional, in effort, to the time that was invested in acquiring these skills. Therefore, the challenge is to prepare our young people to face the world in which they will live, giving them the necessary cognitive tools to succeed in the digital world, that is, instead of teaching them only the syntax of a changing language, they should be instructed in the rules that allow to know how the digital language is constructed. Thus, computational thinking emerges as a paradigm of work, and the programming is stablished as the tool to solve problems.**

***Index Terms*— Computational thinking; teaching of computer science; programming; pre-university studies, university studies**

I. Introducción

El contexto social en el que actualmente nos desenvolvemos está completamente ligado a la tecnología. Ante esta realidad, el debate educativo tiene que verse modificado y debe pasar de estar fundamentado en dominar la tecnología como usuarios, para buscar una solución para que nuestros jóvenes dominen los nuevos lenguajes digitales [1] que rigen la sociedad en la que han nacido y van a tener que desarrollarse como profesionales en un futuro próximo.

Esta formación relacionada con la informática debe entenderse en las dos vertientes de la educación: la informática educativa y la educación en informática [2]. Hasta ahora el esfuerzo se ha orientado mayoritariamente a convertir a nuestros jóvenes en usuarios de herramientas informáticas. Esto ha pasado de ser necesario a ser insuficiente, porque el uso de aplicaciones *software* es un lenguaje digital que queda obsoleto en un tiempo que no es proporcional, en esfuerzo, al que se invirtió en adquirir las destrezas.

Esta preocupación por empezar la educación en informática en edades tempranas, incluso en la guardería, aparece a nivel internacional y se ha canalizado mayoritariamente enseñando a programar a los niños.

Este debate propio de la educación en informática en niveles no universitarios está en todos los actores sociales relacionados con la informática y son múltiples las iniciativas al respecto.

Conscientes de la importancia de las habilidades digitales (*eSkills*) relacionadas con las tecnologías de la información y de la necesidad de incluir la informática en los planes de estudio de los niveles no universitarios, la Conferencia de Directores y Decanos de Ingeniería Informática (CODDII - www.coddii.org) y la Asociación de Enseñantes Universitarios en Informática (AENUI – www.aenui.net) llevan años emprendiendo acciones conjuntas al respecto. Una de las primeras fue la redacción en el año 2014 de la declaración "Por la inclusión de asignaturas específicas de ciencia y tecnología informática en los estudios básicos de la enseñanza secundaria y bachillerato".

Por su parte, la SCIE (Sociedad Científica Informática de España – www.scie.es) en 2015 creó un grupo de trabajo (conjunto de SCIE, CODDII y AENUI). Fruto del trabajo de este grupo, se organizó el workshop "Educación en Informática sub 18 (EI<18)" (http://www.congresocedi.es/ei-18), en el marco del V Congreso Español de Informática (CEDI 2016 - http://www.congresocedi.es) celebrado en Salamanca en septiembre de 2016. Al realizarse en la Universidad de Salamanca se incorporó al Comité Organizador el grupo GRIAL (GRupo de InterAcción y eLearning – http://grial.usal.es), de contrastada experiencia en el tema con proyectos internacionales como TACCLE 3 – Coding (http://www.taccle3.eu) [3]. Como continuación de esta labor y para dejar constancia documentada de las buenas prácticas existentes, tanto de las presentadas en el *workshop* como de otras que no se pudieron presentar por limitaciones de tiempo, se planteó la publicación en revistas científicas de monográficos dedicados a la educación en informática en contextos pre universitarios [4, 5].

Existen también otras iniciativas de distintas publicaciones que han dedicado números especiales al tema del pensamiento computacional, como es el caso del número 46, de septiembre de 2015 de la revista RED (Revista de Educación a Distancia - http://www.um.es/ead/red/red.html), que está dedicado íntegramente al tema de "Pensamiento Computacional" [6] o, más recientemente, en el volumen 80 de 2018 de la

Francisco José García Peñalvo, Instituto de Ciencias de la Educación (IUCE), Grupo de Investigación GRIAL, Universidad de Salamanca. Paseo de Canalejas 169, 37008, Salamanca, España (email fgarcia@usal.es).
(https://orcid.org/0000-0001-9987-5584)

revista Computers in Human Behavior se ha publicado la sección especial titulada "*Exploring the computational thinking effects in pre-university education*" [7]. Además, el pensamiento computacional es un tema que empieza a ser un tópico destacado en diferentes congresos internacionales, por ejemplo, en las ediciones de 2016 y 2017 de la Conferencia Internacional TEEM (*Technological Ecosystems for Enhancing Multiculturality* - http://teemconference.eu/) se han organizado sesiones específicas relacionadas con esta temática que han dado lugar a debates extremadamente interesantes y fructíferos, y en el XVIII Simposio Internacional de Informática Educativa (SIIE 2016) hubo suficientes contribuciones como para organizar una sesión temática.

En el trasfondo de todo este debate subyace la idea de cómo afrontar está formación en computación y programación, especialmente en los estudios pre universitarios. Por un lado, están las iniciativas que se centran en la programación, utilizando lenguajes visuales o no visuales como vía para introducir en la computación a los jóvenes de primaria, fundamentalmente, y secundaria; por otro, están las iniciativas basadas en la idea del pensamiento computacional [8, 9], con la idea de contar con una forma de resolver problemas basada en los principios computacionales, pero sin tener que, obligatoriamente, realizar programas de ordenador, es decir, en lugar de enseñarles solo la sintaxis de un lenguaje cambiante, se les debe instruir en las reglas que permiten conocer cómo se construye el lenguaje digital [5].

Las dos ideas que son claramente compatibles y fácilmente integrables en cualquier estrategia, ya que el pensamiento computacional se puede basar en programación [10], robótica [11], control de dispositivos [12], *wareables* [13] o simplemente en conceptos *unplugged* [14], esto es, sin ninguna tecnología y orientados a desarrollar una forma de resolver problemas. En contra del enfoque del pensamiento computacional está la opinión de los que dicen que no hay un consenso claro en su definición [15] y apuestan por la concreción que da el desarrollo de una aplicación utilizando los lenguajes de programación.

La experiencia nos dice que no hay ninguna solución milagrosa para resolver un problema tan complejo como intentar explicar la lógica interna que rige el funcionamiento del mundo digital. Desde este editorial se propone y se defiende que se debe caminar hacia una educación crítica y reflexiva inmersa en un mundo digital, que ayude a los jóvenes a resolver problemas haciendo uso de la tecnología con la que conviven diariamente. Para ello, los docentes disponen de un rico abanico de posibilidades con las que crear escenarios y actividades de aprendizaje que resulten efectivos mediante la combinación de las herramientas y metodologías que tienen a su disposición: pensamiento computacional, programación, robótica, trabajo en equipo, pensamiento crítico, etc. [7].

Además, el pensamiento computacional tiene aplicación en múltiples contextos (pre-universitarios y universitarios) y formatos (compaginando ambientes presenciales, *online* o mixtos).

En definitiva, la enseñanza de la programación y del pensamiento computacional, tanto en niveles pre-universitarios como universitarios, es un tópico de interés para VAEP RITA / IEEE RITA, lo que ha llevado a la organización de esta sección especial que cuenta con 4 artículos:

En el primero de ellos, titulado "Pensamiento Computacional entre Filosofía y STEM. Programación de Toma de Decisiones aplicada al comportamiento de "Máquinas Morales" en Clase de Valores Éticos", se describe una actividad de aprendizaje que emplea el pensamiento computacional en una clase de ética para estudiantes de 14 a 16 años con un enfoque STEM (*Science, Technology, Engineering and Mathematics*). Concretamente, aborda un problema de reflexión sobre las máquinas morales que deben tomar decisiones que pueden afectar a las vidas humanas.

En el artículo "Relación de conocimientos por aprender en metodología de la programación y la evaluación del pensamiento computacional" se presenta la propuesta de evaluar el pensamiento computacional de los estudiantes de nuevo ingreso de la división de Tecnologías de la Información y Comunicación de la Universidad Tecnológica de Puebla para relacionar los conocimientos indicados en la asignatura "Metodología de la programación" y ofrecerles un ambiente inicial de aprendizaje donde acredite, repase o aprenda lo que determine la prueba con el objetivo de motivar al estudiante que ya posee un conocimiento y atender al que requiere enseñanza.

En el artículo "Evolución": Diseño e implementación de material educativo digital para fortalecer habilidades del pensamiento computacional, sus autores describen cómo un material digital permite la mejora de las habilidades de programación de algoritmos relacionados con problemas matemáticos.

El último de los artículos, que lleva por título "Propuesta de una metodología basada en metáforas para enseñar programación a niños", propone una metodología basada en aplicar metáforas, del tipo receta (para programas y secuencias) o cajas (para las variables), a los recursos del profesor orientados a la enseñanza de la programación a niños de entre 9 y 11 años.

AGRADECIMIENTOS

Queremos agradecer en primer lugar a la revista VAEP RITA (http://rita.det.uvigo.es/VAEPRITA/) y a su editor en jefe, el Dr. Martín Llamas Nistal, por el apoyo recibido para la organización de la sección especial sobre pensamiento computacional.

Como no podría ser de otra forma también nuestro agradecimiento a los autores por haber contribuido con sus trabajos a esta sección especial y, por último, a los revisores por ser los garantes de la calidad de los trabajos aceptados.

REFERENCIAS

[1] F. Llorens-Largo, "Dicen por ahí. . . . . que la nueva alfabetización pasa por la programación," *ReVisión*, vol. 8, no. 2, pp. 11-14, 2015.

[2] J. L. Sierra-Rodríguez and F. J. García-Peñalvo, "Informática Educativa y Educación en Informática," *Education in the Knowledge Society (EKS)*, vol. 16, no. 4, pp. 25-31, 2015.

[3] F. J. García-Peñalvo, "A brief introduction to TACCLE 3 – Coding European Project," in *2016 International Symposium on Computers in Education (SIIE 16)*, F. J. García-Peñalvo and J. A. Mendes, Eds. USA: IEEE, 2016.

[4] F. J. García-Peñalvo, F. Llorens Largo, X. Molero Prieto, and E. Vendrell Vidal, "Educación en Informática sub 18 (EI<18)," *ReVisión*, vol. 10, no. 2, pp. 13-18, 2017.

[5] F. Llorens Largo, F. J. García-Peñalvo, X. Molero Prieto, and E. Vendrell Vidal, "La enseñanza de la informática, la programación y el pensamiento computacional en los estudios preuniversitarios," *Education in the Knowledge Society*, vol. 18, no. 2, pp. 7-17, 2017.

[6] W. Bender, C. Urrea, and M. Zapata-Ros, "Introducción al número monográfico sobre Pensamiento Computacional," *RED (Revista de Educación a Distancia)*, vol. XIV, no. 46, 2015.

[7] F. J. García-Peñalvo and J. A. Mendes, "Exploring the computational thinking effects in pre-university education," *Computers in Human Behavior*, vol. 80, pp. 407-411, 2018.

[8] J. M. Wing, "Computational Thinking," *Communications of the ACM*, vol. 49, no. 3, pp. 33-35, 2006.

[9] F. J. García-Peñalvo, "What Computational Thinking Is," *Journal of Information Technology Research*, vol. 9, no. 3, pp. v-viii, 2016.

[10] A. M. Pinto-Llorente, S. Casillas-Martín, M. Cabezas-González, and F. J. García-Peñalvo, "Building, coding and programming 3D models via a visual programming environment," *Quality & Quantity*, vol. In Press, 2018.

Francisco José García Peñalvo realizó sus estudios universitarios en informática en la Universidad de Salamanca y en la Universidad de Valladolid y se doctoró en la Universidad de Salamanca. El doctor García-Peñalvo es el director del grupo de investigación GRIAL (Grupo de investigación en Interacción y eLearning). Sus principales intereses de investigación se centran en el eLearning, Computadores y Educación, Sistemas Adaptativos, Ingeniería Web, Web Semántica y Reutilización de Software. Ha dirigido y participado en más de 50 proyectos de innovación e investigación. Fue Vicerrector de Innovación Tecnológica de la Universidad de Salamanca entre Marzo de 2007 y Diciembre de 2009. Ha publicado más de 200 artículos en revistas y conferencias internacionales. Ha sido editor invitado en varios números especiales de revistas internacionales (*Online InformationReview, Computers in Human Behaviour, InteractiveLearningEnvironments*…). Es el Editor en Jefe de las revistas Education in the Knowledge Society y Journal of Information Technology Research. Coordina el Programa de Doctorado en Formación en la Sociedad del Conocimiento de la Universidad de Salamanca.

[11] C. Fernández-Llamas, M. Á. Conde, F. J. Rodríguez-Lera, F. J. Rodríguez-Sedano, and F. J. García-Peñalvo, "May I teach you? Students' behavior when lectured by robotic vs. human teachers," *Computers in Human Behavior*, vol. 80, pp. 460-469, 2018.

[12] P. Martín-Ramos *et al.*, "First exposure to Arduino through peer-coaching: Impact on students' attitudes towards programming," *Computers in Human Behavior*, vol. 76, no. Supplement C, pp. 51-58, 2017/11/01/ 2017.

[13] D. Reimann and C. Maday, "Enseñanza y aprendizaje del modelado computacional en procesos creativos y contextos estéticos," *Education in the Knowledge Society*, vol. 18, no. 3, pp. 87-97, 2017.

[14] T. Bell, I. H. Witten, and M. Fellows, *CS Unplugged. An enrichment and extension programme for primary-aged students. Version 3.2.2*. New Zealand: University of Canterbury. CS Education Research Group, 2016.

[15] F. J. García-Peñalvo, D. Reimann, M. Tuul, A. Rees, and I. Jormanainen, "An overview of the most relevant literature on coding and computational thinking with emphasis on the relevant issues for teachers," TACCLE3 Consortium, Belgium2016.

Pensamiento Computacional entre Filosofía y STEM. Programación de Toma de Decisiones aplicada al Comportamiento de "Máquinas Morales" en Clase de Valores Éticos

Antonio Miguel Seoane Pardo

CÓMO REFERENCIAR ESTE ARTÍCULO:

Antonio Miguel Seoane Pardo Computational Thinking between Philosophy and STEM. Programming Decision Making applied to the Behaviour of "Moral Machines" in Ethical Values Classroom", en IEEE-RITA, 2018
Doi: https://doi.org/10.1109/RITA.2018.2809940

***Title*— Computational Thinking between Philosophy and STEM. Programming Decision Making applied to the Behaviour of "Moral Machines" in Ethical Values Classroom.**

***Abstract*— This article describes a learning activity on computational thinking in Ethics classroom with compulsory secondary school students (14-16 years old). It is based on the assumption that computational thinking (or better "logical thinking") is applicable not only to STEM subjects but to any other field in education, and it is particularly suited to decision making in moral dilemmas. This will be carried out through the study of so called "moral machines", using a game-based learning approach on self-driving vehicles and the need to program such cars to perform certain behaviours under extreme situations. Students will be asked to logically base their reasoning on different ethical approaches and try to develop a schema of decision making that could serve to program a machine to respond to those situations. Students will also have to deal with the uncertainty of reaching solutions that will be debatable and not universally accepted as part of the difficulty, more ethical than technical, to provide machines with the ability to take decisions where there is no such thing as a "right" *versus* "wrong" answer, and potentially both (or more) of the possible actions will bring unwanted consequences.**

***Index Terms*— Computational thinking, decision making, game-based learning, Ethics, Logic, moral machines, self-driving car.**

I. Introducción

En las últimas décadas, tanto los investigadores como las autoridades educativas han venido discutiendo sobre las competencias y destrezas clave en las que debe incidir el sistema educativo, desde la Educación Infantil hasta la finalización de los estudios universitarios, tanto para la formación de ciudadanos como para la cualificación de trabajadores que encajen en las que (se espera) sean las habilidades clave de los empleos del siglo XXI [1]. Se trata de una discusión que dista mucho de estar cerrada, por más que se tienda a definir sistemas de competencias y destrezas más o menos estandarizados. Por ejemplo, el proyecto DeSeCo, de la OCDE, define un marco de competencias clave estructuradas en tres categorías bastante amplias: *usar herramientas de manera interactiva, interactuar en grupos heterogéneos y actuar de forma autónoma* [2]. Esta taxonomía constituye el fundamento para el desarrollo de las competencias de PISA. Por otra parte, el Marco europeo de competencias clave para el aprendizaje permanente [3] define un total de ocho competencias clave, a saber: *comunicación en la lengua materna, comunicación en lenguas extranjeras, competencia matemática y competencias básicas en ciencia y tecnología, competencia digital, aprender a aprender, competencias sociales y cívicas, sentido de la iniciativa y espíritu de empresa, y conciencia y expresión culturales*. Estas competencias, que adoptaron la forma jurídica de Recomendación del Parlamento y del Consejo Europeo, de 18 de diciembre de 2006, alcanzan el rango de Ley en las sucesivas reformas educativas españolas y, de manera más específica, en la LOMCE, la ley educativa actualmente en vigor [4]. Otros estudios [5] dividen las competencias en aspectos personales y éticos (*vivir en el mundo*), competencias laborales (entre las que se distinguen las *herramientas de trabajo* y las *formas de trabajar*) y de pensamiento (*formas de pensar*). El *pensamiento crítico, la resolución de problemas y la toma de decisiones* es una de las competencias (en este caso multifactorial) que se señalan como esenciales dentro del conjunto de las asociadas con el pensamiento. Se resalta aquí esta competencia en particular porque encaja perfectamente con el enfoque principal de este trabajo, y pone de manifiesto hasta qué punto son una única competencia: el pensamiento crítico es crucial para definir estrategias adecuadas conducentes a la resolución de problemas y debe ser la guía para la toma de decisiones de los individuos.

Además del problema suscitado por la definición de un marco competencial general para la formación académica y el aprendizaje permanente, en los últimos años se ha venido

Manuscrito recibido el 1 de junio de 2017; revisado 1 de julio; aceptado 26 de octubre.

English versión received January, 29th, 2018. Revised February, 6th. Accepted February, 16th.

A. M. Seoane Pardo es Profesor Asociado del Área de Métodos de Investigación y Diagnóstico en Educación en la Universidad de Salamanca y miembro del GRupo de Investigación en InterAcción y eLearning (GRIAL). (e-mail: anton.seoane@usal.es)

(https://orcid.org/0000-0001-8887-3954).

discutiendo el papel que han de desempeñar las competencias STEM (acrónimo formado por las iniciales de las áreas de Ciencias, Tecnología, Ingeniería y Matemáticas, en inglés) en la educación. En este contexto, se asiste a una extraña paradoja: la demanda creciente de puestos de trabajo relacionados con la tecnología y el conocimiento científico (particularmente en el ámbito de la Ingeniería, pero también en otras áreas) no se refleja en el incremento de estudiantes en los grados universitarios que proporcionan dicha cualificación. Antes al contrario, todo parece indicar que algunos países (entre los cuales se encuentra España) podrían empezar a demandar ingenieros extranjeros en la próxima década si esta tendencia no cambia drásticamente [6]. Esto es algo que preocupa tanto en contextos académicos como políticos, puesto que se trata de una situación que puede afectar (y mucho) al desarrollo económico del país, en un contexto, además, en el que hay una elevada tasa de universitarios sin empleo, subempleados o recurriendo a la emigración como única forma de encontrar perspectivas laborales. Si bien la situación de los universitarios españoles no es comparable actualmente a la de otros países del entorno europeo, la falta de estudiantes en las carreras universitarias relacionadas con STEM no es un fenómeno que se produzca únicamente en España, sino que afecta también a otros países europeos [7].

Así pues, cabe plantearse qué ocurre en un sistema educativo que no es capaz de orientar a sus estudiantes hacia estudios que conducen a puestos de trabajo con las mayores y mejores condiciones de empleabilidad, especialmente cuando esta es una tendencia creciente, sostenida y estructural, no una circunstancia coyuntural del sistema productivo. Según algunos estudios realizados en Estados Unidos [8], el problema radica en la forma en que los docentes enseñan Informática actualmente, puesto que los centros educativos han hecho un enorme esfuerzo por incrementar la utilización, integración y enseñanza de las TIC en las aulas, al tiempo que confunden "el uso de la tecnología con la enseñanza de la informática como disciplina académica fundamental en el ámbito de las áreas STEM". Dichos estudios recomiendan, entre otras medidas, "la creación de un conjunto bien definido de estándares para la enseñanza de la informática en educación preuniversitaria, basados en conceptos de pensamiento algorítimico/computacional". Por otra parte, en la Unión Europea, donde se estima un déficit de 800.000 profesionales relacionados con los ámbitos TIC e Informática para 2020, "muchos docentes, padres, economistas y políticos empiezan a pensar que los estudiantes necesitan habilidades de informática y programación", según un reciente estudio de European Schoolnet [9]. En esta misma línea, la Conferencia de Directores y Decanos de Ingeniería Informática reivindicaba en 2014 la inclusión de materias específicas de ciencia y tecnología informática en la Educación Secundaria Obligatoria y el Bachillerato [10]. Además, apuntan, es necesario reforzar las competencias STEM y abordarlas de manera transversal y genérica, sin que queden reducidas solo a las "asignaturas STEM". Para ello, es necesario incrementar y reforzar las acciones formativas destinadas al profesorado, al objeto de garantizar su efectiva preparación.

Por otra parte, el informe *EU Skills Panorama 2014* (publicado en abril de 2015) [11], dedicado a las competencias STEM, establece que, en la mayoría de los países, el número de estudiantes y graduados en titulaciones STEM se ha incrementado. Sin embargo, muchos empleadores manifiestan que los egresados no están preparados para el trabajo, puesto que no poseen las competencias adecuadas, particularmente en lo concerniente a las competencias transversales (*soft skills*), a saber: competencias comunicativas, trabajo en equipo, pensamiento creativo, etc. En consecuencia, todo parece indicar que, además de la falta de competencias STEM, estas tampoco son suficientes, y otras competencias tan necesarias deben acudir "en auxilio" de las anteriores. Ni unas ni otras, además, deben ser solo objeto de estudio de las materias específicas: al igual que la competencia digital no debe circunscribirse a la asignatura de Informática, las competencias comunicativas no son solo atribución de los docentes de Lengua, y así sucesivamente. Se necesita un enfoque mucho más global para abordar las competencias, destrezas y habilidades que se deben impartir a los estudiantes, y no se pueden reducir las competencias clave (ni siquiera el pensamiento algorítmico/computacional) a un modelo de currículum STEM. El Proyecto Europeo TACCLE 3 – Coding [12-15] tiene como finalidad orientar a los docentes y proporcionar la formación necesaria para promover un nuevo *curriculum de informática* que incluya, entre otros aspectos, la programación [9] y el pensamiento computacional [16-20]. La creatividad, el pensamiento crítico, la discusión ética y moral y la conciencia cultural, por mencionar solo algunos ejemplos, son competencias fundamentales para lograr una educación de calidad adaptada a las necesidades del siglo XXI. Por eso este trabajo se centra en el pensamiento computacional "más allá" del modelo STEM, precisamente en un ámbito (la Ética y los Valores Éticos) en el que los problemas a abordar deben ser analizados lógicamente ("computacionalmente"), pero las conclusiones están muy lejos de poder ser consideradas *soluciones* y, antes al contrario, suelen ser el punto de partida para nuevas discusiones y desafíos para el razonamiento y el pensamiento crítico. En pocas palabras, se propone el uso del pensamiento computacional como medio para la discusión ética y moral, en lugar de promover el pensamiento computacional como fin en sí mismo.

En las siguientes páginas se explicará el desarrollo de una actividad de aprendizaje que tiene como finalidad la toma de decisiones en la asignatura de Valores Éticos, con un enfoque basado en el pensamiento computacional. Para ello, en primer lugar se definirán con claridad los conceptos de "Pensamiento Computacional" y "Ética de las máquinas" mediante una sucinta revisión de la literatura científica; a continuación, se presentarán las actividades desarrolladas con estudiantes de Enseñanza Secundaria Obligatoria relacionadas con las máquinas morales; finalmente, se establecerá una comparativa entre esta actividad formativa y otras experiencias, para extraer algunas conclusiones a este respecto.

Este trabajo constituye la versión extendida y traducida de una comunicación presentada en la *Fourth International*

Conference on Technological Ecosystems for Enhancing Multiculturality (TEEM 16) [21]. Respecto a la comunicación original, se ha realizado un análisis más detallado en la mayor parte de las secciones, lo que se explicita a continuación. En primer lugar, se ha hecho un análisis detallado de los diferentes estudios de marcos competenciales que sirven de referencia para los actuales marcos legislativos, en lo que a competencias y destrezas se refiere, y se han aportado más evidencias del problema que se plantea en este estudio, tanto a nivel europeo como global. Además, se ha enfatizado el papel que proyectos como TACCLE 3 – Coding pueden desempeñar en la mejora de las competencias STEM desde edades tempranas. En la Fundamentación Teórica, se ha abordado un análisis más pormenorizado del marco jurídico y legislativo en materia de educación en España, desde los años 70 hasta la actualidad, y se ha explicado en qué medida el fomento de determinadas competencias no está logrando el impacto deseado, en términos de mejora de las habilidades STEM. En la Descripción de la Acción Formativa se han analizado con mayor profusión los diferentes modelos de toma de decisiones que han servido como referencia para el desarrollo del marco propuesto en esta experiencia. Finalmente, se han extendido las secciones de Discusión y Conclusiones y Trabajo Futuro, y se han añadido referencias respecto al trabajo original.

II. Fundamentación teórica: Pensamiento Computacional y Ética de las Máquinas

Jeannette M. Wing afirma con rotundidad que "el pensamiento computacional expresa cómo piensan los humanos, no los ordenadores" [19], porque representa la manera en que los seres humanos resuelven problemas y toman decisiones cuando actúan de manera racional. Obviamente, tomar decisiones y resolver problemas no son algo que afecte únicamente al razonamiento lógico o "STEM". Por el contrario, se trata de estrategias que empleamos en nuestra vida cotidiana, ya sea de forma consciente o inconsciente, y el propósito de estas páginas es precisamente poner de manifiesto cómo podemos hacer uso de tales estrategias de forma estructurada y algorítmica (computacional), a fin de abordar, explicar y tratar de resolver una instancia concreta de los dilemas morales, aquellos que se aplican a un tipo específico de artefactos denominados *máquinas morales*, esto es, máquinas programadas para tomar decisiones con implicaciones de carácter ético. Un ejemplo de dichas máquinas morales, cuyo "comportamiento" es relativamente sencillo de estudiar y analizar en Valores Éticos con estudiantes, es el del coche autónomo.

A. *Pensamiento Computacional y Valores Éticos*

Cuando se pretende promover estrategias, herramientas o contextos de innovación en los procesos formativos, es casi inevitable tener que enfrentarse a la *resistencia al cambio* que ejercen los docentes, por una parte, pero también el sistema educativo en su conjunto. Esta amenaza potencial dificulta cualquier proceso de innovación, ya sea didáctica, metodológica o tecnológica. Por el contrario, en el sistema educativo, como en cualquier colectivo humano, hay profesionales que sienten una poderosa atracción hacia cualquier nueva moda o tendencia, pues son naturalmente innovadores, perciben con entusiasmo cualquier novedad y tratan de experimentarla. Estos, curiosamente, en lugar de ser percibidos como catalizadores de la innovación, lamentablemente (y por supuesto sin pretenderlo), con frecuencia alimentan todavía más, si cabe, la resistencia al cambio de los primeros, que se ven incapaces de "seguir el ritmo" de los más avanzados. De este modo, sienten que la innovación es algo parecido al castigo divino del titán Sísifo, una tarea tan interminable como inútil que se les impone sin que saber muy bien por qué debe llevarse a cabo. He ahí otro de los riesgos de la innovación. Por último, citando a Descartes en el *Discurso del método*, están aquellos que, "sin ser llamados ni por su alcurnia ni por su fortuna al manejo de los negocios públicos, no dejan de hacer siempre, en idea, una nueva reforma" [22]. No es fácil plantear un marco estable y duradero para la innovación donde, desde 1970, se han vivido ocho leyes educativas diferentes en Primaria y Secundaria (LGE, LOECE, LODE, LOGSE, LOPEG, LOCE, LOE y LOMCE), una en Formación Profesional (Ley Orgánica de Cualificaciones) y cuatro en enseñanzas universitarias (LGE, LRU, LOU y LOMLOU). La "innovación legislativa", cuando no produce cambios significativos en los procesos de enseñanza-aprendizaje (tal es el caso de las reformas antedichas en España, que no han definido un marco para la innovación ni la mejora de los procesos docentes) deviene una especie de gatopardismo: "si queremos que todo siga como está, es necesario que todo cambie", decía uno de los personajes de Giuseppe Tomasi di Lampedusa en su novela.

Las reformas educativas no parecen suponer un estímulo para Las últimas reformas educativas han favorecido el estudio de disciplinas relacionadas con la Economía y la necesidad de incluir contenidos económicos y de emprendimiento entre los contenidos básicos de todas las materias, en detrimento de materias como Ética, Educación para la Ciudadanía e Historia de la Filosofía, incluso cuando muchos autores (entre ellos afamados economistas) ponen el acento en la importancia del pensamiento crítico que, si bien no es territorio exclusivo de las materias del área de Filosofía, sí tienen en ellas un terreno fértil para su estímulo. Recuérdese que el pensamiento crítico es clave para la resolución de problemas, según un estudio anteriormente citado [5].

Junto con el incremento de los contenidos de Economía, la presencia creciente de las materias sobre TIC e Informática es otro de los elementos que llama la atención. En este último caso, el refuerzo de estas áreas les confiere la responsabilidad de incrementar la competencia digital y, en cierto modo, de hacer que los estudiantes tomen conciencia de la relevancia de las materias STEM, atrayéndoles hacia ellas, dada la baja demanda de este tipo de estudios y su alto potencial de empleabilidad. Sin embargo, con toda probabilidad habría sido deseable realizar un análisis en profundidad sobre las causas que subyacen a esta falta de "vocación y formación en STEM" para reflexionar sobre cómo el sistema educativo puede contribuir a resolver este problema, porque incrementar "sin más" la presencia de materias relacionadas con las TIC y la Informática es inútil

si no se fomenta la creatividad, la abstracción, el razonamiento lógico, el pensamiento crítico, etc.; y estas competencias pueden y deben ser estimuladas en todas las materias, a condición de que se haga con los enfoques metodológicos y didácticos adecuados.

Entre las competencias que un estudiante debe poseer destaca el pensamiento lógico o algorítmico, últimamente denominado también *pensamiento computacional*. Este concepto surge precisamente en el contexto de la discusión sobre las competencias en alfabetización digital y la relevancia que actualmente se atribuye al desarrollo de código informático como parte de la adquisición de dichas competencias. Pues bien, el desarrollo de código informático está estrechamente relacionado con las tareas de programación, pero el pensamiento computacional comporta un uso más amplio, como indica García Peñalvo: "aunque el desarrollo de código informático es interesante, resulta más útil enfatizar en la idea del pensamiento computacional como aplicación de alto nivel de abstracción y aproximación algorítmica a la resolución de cualquier tipo de problemas" [16]. En consecuencia, es importante distinguir entre pensamiento computacional y desarrollo de código informático o lenguajes de programación. La programación constituye un modo particular de pensamiento computacional [23].

El pensamiento computacional, tal como ha sido definido por J. M. Wing, es un tipo de pensamiento analítico [19] que "describe la actividad mental de formular un problema que admite una solución computacional" [20]. De este modo, su influencia se extiende a áreas de investigación tanto científicas como humanísticas [24], y puede ponerse en práctica en cualquier materia del currículo escolar. Otra definición (más sistemática y comprehensiva) de Wing y otros autores, establece que "el pensamiento computacional es el conjunto de procesos de pensamiento implicados en la formulación de problemas y sus soluciones para que dichas soluciones puedan ser representadas de tal modo que puedan ser ejecutadas por un agente de procesamiento de la información" [25]. Así pues, el pensamiento computacional no puede restringirse a materias relacionadas con la computación o la programación, y la robótica no puede ser el único modo de poner en práctica esta competencia. Las llamadas competencias STEM no son solo responsabilidad de las materias STEM.

El pensamiento computacional es una competencia emergente tanto en la educación primaria como secundaria. Numerosos estudios (como por ejemplo los desarrollados en Reino Unido, Holanda y Estados Unidos) ponen de manifiesto que se están realizando importantes esfuerzos por mejorar esta competencia a través de las materias relacionadas con la Informática [26]. Sin embargo, otros análisis sobre el estado de la cuestión [27] reconocen que, a pesar de las experiencias innovadoras de introducción a la computación (utilizando diferentes entornos y herramientas, tales como programación visual, kits de robótica, videojuegos, etc.), persisten ciertas dificultades para definir con claridad las competencias computacionales e incluso para evaluar su impacto efectivo en términos de mejora del rendimiento académico. Estos estudios revelan la necesidad de seguir investigando en el ámbito de las ciencias de la educación aplicadas a este contexto, incluido el impacto de nuevos enfoques como el aprendizaje situado y distribuido, el uso de agentes inteligentes, el análisis de la interacción, la actividad y el discurso en contextos digitales, aspectos cognitivos implicados en el aprendizaje de conceptos computacionales, etc. Además, y esto es si cabe más relevante, se ha investigado muy poco sobre el impacto de la informática como medio para la enseñanza de otras disciplinas; por ejemplo, en qué medida las estrategias de resolución de problemas, tal como se desarrollan en tareas de programación informática, pueden aplicarse en otras materias.

Probablemente este último aspecto no se esté enfocando de forma adecuada. La cuestión no radica en cómo *exportar* el pensamiento computacional de la informática a otras materias. La mejor manera de promover eficazmente el pensamiento computacional y, en consecuencia, incrementar la capacidad de razonamiento lógico de nuestros estudiantes, consiste precisamente en su introducción como estrategia pedagógica para la formulación y resolución de problemas, tanto en asignaturas científicas como humanísticas, no solo en las materias de informática. Y es precisamente así como este estudio pretende contribuir proponiendo un ejemplo de desarrollo de pensamiento computacional desde el área de Filosofía y, más concretamente, desde la asignatura de Valores Éticos en Enseñanza Secundaria Obligatoria. El desarrollo de un marco para los procesos de toma de decisiones aplicado al análisis de dilemas morales, dadas ciertas variables y condiciones específicas, y conforme a ciertos principios morales, constituye un ejemplo de pensamiento computacional aplicado a esta materia mediante el que los estudiantes pueden poner en práctica el razonamiento abstracto sin recurrir al uso de lenguaje simbólico ni de programación, pero siguiendo los mismos esquemas de abstracción, resolución de problemas y toma de decisiones del pensamiento computacional en el contexto de la inteligencia artificial.

B. Ética de las Máquinas y Máquinas Morales

Anderson y Anderson explican con claridad [28] un concepto relativamente nuevo que consiste en dotar de dimensión ética a las máquinas y que difiere del que comúnmente se conoce como *ética computacional*. Se trata del concepto de *ética de las máquinas.* A diferencia de la ética computacional, que hace referencia al comportamiento humano y a las implicaciones morales del uso de las máquinas por los humanos, la ética de las máquinas alude a la conducta moral de los propios artefactos dotados de inteligencia artificial.

Este principio de ética de las máquinas, según el cual es posible dotar de conducta moral a seres de inteligencia artificial podría parecer, en primera instancia, simple ciencia ficción. De hecho, el origen de la ética de las máquinas se remonta a las Tres Leyes de la Robótica, conocidas también como "Leyes de Asimov", quien definió este conjunto de reglas por vez primera en un relato corto, "Runaround", de 1942. No obstante, con el posterior desarrollo de la robótica, por una parte, y de la inteligencia artificial, por otra, surge el problema de la toma de decisiones por parte de máquinas

cada vez más complejas que, como parte de su proceso de sofisticación, pronto estarán en condiciones de actuar de forma autónoma. En consecuencia, algunas de las decisiones que puedan tomar comportarán consecuencias éticas y morales.

La ética de las máquinas debe tomarse en consideración, al menos, por tres razones: "en primer lugar, se derivan consecuencias éticas de lo que las máquinas son capaces de hacer actualmente y de lo que se espera que hagan en el futuro […]. En segundo lugar, podría argumentarse que el temor de los humanos a la posibilidad de máquinas autónomas inteligentes se debe a la preocupación acerca de si estas máquinas se comportarán éticamente, por lo que el futuro de la inteligencia artificial puede estar en juego […]. Finalmente, creemos que es posible que la investigación en ética de las máquinas permita avanzar en el estudio de la teoría ética [28]. Se trata, pues, de un problema con implicaciones de ingeniería y filosóficas y, para ser honestos, estas últimas son con diferencia mucho más complejas que las primeras, a tenor de las discusiones de la Filosofía Occidental durante más de 2.500 años. Sea como sea, cualquier intento por definir un conjunto de principios o un marco para el comportamiento ético de las máquinas constituye un buen principio para enfrentarse a este desafío, incluso si nos vemos obligados a discutir metafóricamente sobre "virtudes de las máquinas" [29], a definir principios reguladores de carácter moral para la inteligencia artificial, o a crear incluso una *directiva primera* para el código ético de la interacción humano-robot (IHR) [30].

Algunos autores sugieren que, antes de tratar de *enseñar* a las máquinas a tomar decisiones éticas, quizá sería interesante tratar de aplicar el pensamiento computacional a la propia Ética y, de este modo, tratar de automatizar el proceso de detección y clasificación de los valores expresados a través de la comunicación humana [31]. Así pues, suponiendo que fuera técnica y filosóficamente posible convertir máquinas en sujetos capaces de producir un razonamiento moral explícito, el debate radicaría en la selección del enfoque moral más adecuado [32]. Los enfoques deductivos (*top-down approaches*) se basan en la idea según la cual los principios morales o teorías éticas constituyen la base para la creación de reglas que permitan desarrollar acciones éticamente correctas. La Ética deontológica de Kant o el Utilitarismo constituyen teorías éticas susceptibles de encajar en este tipo de enfoque. Por otra parte, los enfoques inductivos (*bottom-up approaches*) no se basan en teorías éticas concretas sino en principios evolutivos y dinámicos, partiendo del principio según el cual la inteligencia artificial "imita" el desarrollo infantil (tomando como base las teorías de Turing y, posteriormente, de Piaget, Kohlberg, etc.), y de este modo la máquina "aprenderá". Finalmente, y puesto que ninguno de los modelos anteriores satisface los criterios para el diseño de una entidad artificial como agente moral, se requiere el desarrollo de enfoques híbridos que combinen un conjunto de reglas de carácter deductivo con la experiencia y el aprendizaje recopilado por la máquina.

Aunque esta discusión resulta extremamente interesante y merecería sin duda un análisis más pormenorizado, excede con mucho el propósito de estas páginas y deberá dejarse para estudios posteriores. En este caso, nos centraremos en diferentes enfoques morales (en su mayoría deductivos, de acuerdo con la clasificación anterior), con el propósito de proponer y analizar escenarios en los que los estudiantes puedan representar, responder y discutir diferentes posibilidades en relación con una serie de dilemas morales que se experimentarán en contextos en los que ciertas máquinas interactúan con otras máquinas y/o seres humanos.

C. ¿Por qué el Vehículo Autónomo?

Los vehículos autónomos serán pronto una realidad cotidiana en nuestras vidas. El primer test de conducción autónoma se llevó a cabo en 2007 y algunos prototipos, como el coche autónomo de Google, han recorrido miles de kilómetros de conducción en carreteras convencionales [33]. Son numerosos los estudios sobre algoritmos de navegación y modelos de toma de decisiones y sistemas de control para la programación de estos vehículos [34]. Pero probablemente algunas de las principales dificultades para el desarrollo de los vehículos autónomos sean más de índole filosófica (concretamente ética) y sociológica que propiamente técnica. En 2015, la Universidad de Stanford realizó un seminario que reunió a filósofos e ingenieros para discutir e implementar principios éticos que pudieran ser formalizados y codificados para su posterior desarrollo y experimentación en simulaciones e incluso en vehículos reales [35]. El debate en la prensa general y las revistas de divulgación científica mostraba titulares como "¿Comprarías un coche que elegiría matarte para salvar otras vidas?" [36] o "Nuestro dilema sin conductor. ¿Cuándo tu coche debe estar dispuesto a matarte?" [37]. Estas cabeceras subrayan el hecho de que no existe un consenso ético (ni sociológico) sobre el enfoque moral del que debe dotarse a estos artefactos. Algunos estudios muestran que existe una cierta tendencia a aceptar, con carácter general, normas y principios de carácter utilitarista (como en el caso del conocido dilema del tranvía [38-39], en el sentido de minimizar el número de víctimas en la carretera y optar por el menor daño posible en términos cuantitativos). Sin embargo, los mismos sujetos que manifiestan estas actitudes morales preferirían viajar en y poseer un vehículo que protegiera sus vidas a toda costa, con independencia de las consecuencias utilitaristas que se pudieran derivar de las acciones de la máquina [33]. Ciertamente, la mayoría de nosotros accionaría la palanca que desviaría al tranvía de su trayectoria para que "solo" se produzca una víctima en lugar de las cinco que fallecerían si el tranvía siguiera el rumbo establecido. Eso, claro está, si la persona que está sola en la segunda vía no es un niño al que conocemos y que, cosas de la vida, es nuestro propio hijo.

Los principios morales de carácter utilitarista serían quizá aceptables para los fabricantes de automóviles, especialmente si por "utilidad" se entiende "menor coste". En este caso, probablemente, las compañías de seguros también se sumarían a este tipo de directrices, solo que lo que unos y otros entenderían por "menor coste" no sería necesariamente lo mismo. Y algo similar ocurriría si sumáramos a esta ecuación a los gobiernos. Por otro lado, muchos de los demás enfoques (deontológico, por ejemplo,

o incluso el comportamiento aleatorio en ciertos casos, por mencionar únicamente algunos enfoques deductivos) serían perfectamente aceptables y éticamente defendibles. Esto pone sobre la mesa le cuestión (no menor) sobre quién debe decidir los principios morales en función de los cuales se programa a la máquina: ¿los estados?; ¿las compañías de seguros?; ¿las marcas?...

En resumen, "el problema, al parecer, es más filosófico que técnico. Antes de que podamos imponer nuestros valores a las máquinas, debemos resolver cómo hacer nuestros valores claros y consistentes" [37]. Así pues, el vehículo autónomo proporciona un campo de experimentación ideal para el pensamiento computacional en relación con la Ética.

La otra razón para centrar el problema en los coches autónomos es que existe un número significativo de estudios y simuladores que nos permite visualizar y formalizar dilemas morales. Los *comportamientos éticos* del vehículo autónomo se representan habitualmente como trayectorias y movimiento, lo que resulta muy interesante por la claridad con la que se pueden analizar visualmente causas y consecuencias, así como tomar decisiones basándose en determinados principios éticos. Algunos de estos escenarios se han concebido específicamente a partir de enfoques de carácter ético [40], y facilitan la posibilidad de analizar y simular diferentes opciones para el mismo escenario adoptando varios principios éticos o comportamientos [41]. Otros, en cambio, han sido definidos con un enfoque sociológico, y permiten decidir cuál de las posibles opciones es la que se considera más adecuada, comparando la respuesta con la que proporcionan otros usuarios [42]. El estudio de estos diferentes enfoques para la toma de decisiones constituye un ejercicio de "calentamiento" muy interesante para el desarrollo de la acción formativa que se describe en el próximo apartado.

III. Descripción de la Acción Formativa

El propósito de este estudio es describir un conjunto de actividades sobre pensamiento computacional aplicadas a la asignatura de Valores Éticos con estudiantes de Educación Secundaria Obligatoria (14-16 años). Los estudiantes objeto de estas actividades podrían haber cursado previamente Valores Éticos en los cursos inferiores, puesto que en el actual currículo educativo español se trata de una asignatura optativa, alternativa a los estudios de religión confesional. Por tanto, los estudiantes de 3º y 4º de E. S. O. deben estar familiarizados con la resolución de dilemas morales y la resolución de problemas en contextos éticos. Por el contrario, se trata de estudiantes que carecen de conocimiento de lógica simbólica (que se estudia en 1º de Bachillerato, el primer curso de educación secundaria no obligatoria, en la asignatura de Filosofía), de modo que esta actividad será también útil como introducción al razonamiento lógico, si bien se desarrollará sin recurrir al uso de estructuras de lenguaje formal.

En las siguientes subsecciones se mostrará la estructura y las líneas maestras de la actividad formativa. Previamente, sin embargo, se destinarán algunas líneas a explicar las dificultades que se derivarían del uso de otros marcos de toma de decisiones éticas existentes, ya sea por su complejidad o por la falta de un enfoque "puramente ético" para la experimentación en el contexto aquí descrito.

A. *Modelo de Toma de Decisiones Éticas*

Aunque no sea el propósito de este trabajo, se ha considerado oportuno realizar una breve revisión de la literatura sobre modelos y marcos de toma de decisiones éticas, con el objeto de analizar su aplicabilidad en este contexto. Básicamente se han estudiado dos tipos de marcos: unos, relacionados con modelos computacionales de toma de decisiones en inteligencia artificial (o aplicados específicamente a máquinas morales), y otros, procedentes directamente de la investigación filosófica. Ambos deberán ser descartados para el propósito de este estudio, por las razones que se indicarán a continuación.

Los modelos más utilizados aplicables a la programación de inteligencia artificial, tales como BDI o LIDA (por mencionar solo un par de ejemplos), son demasiado complicados para satisfacer los requerimientos de una actividad simple para jóvenes estudiantes de secundaria. El modelo BDI (*Belief-Desire-Intention model of human practical reasoning)*, del filósofo Michael Bratman [43], constituye la base para el desarrollo del modelo de *software* BDI para la programación de agentes inteligentes, así que, en principio, parece resultar interesante para trabajar el pensamiento computacional en procesos de toma de decisiones en Ética debido a la potencialidad de añadir principios morales a seres artificiales. Por otra parte, LIDA es una arquitectura cognitiva que incluye procesos de decisión moral, tanto para humanos como para agentes artificiales [44]. El proceso cognitivo comienza a partir de una colección inductiva (*bottom-up*) de datos sensoriales, valores y experiencias, y a continuación "actúa" mediante procesos deductivos (*top-down*) para dar sentido a todo el comportamiento. LIDA, BDI y otras arquitecturas cognitivas (como por ejemplo GWT, Clarion, ACT-R, etc.), incluso si poseen la capacidad de adoptar decisiones éticas "reales", no satisfacen las condiciones requeridas para el desarrollo de procesos de toma de decisiones sencillos como los requeridos en este contexto.

En relación con los modelos para el análisis de dilemas morales, existe una amplia lista de modelos, desde deontológicos a utilitaristas, humanistas y otros muchos enfoques. Tales modelos, que se han aplicado con éxito en Medicina, Enfermería, Marketing, Economía, Psicología, etc., no son apropiados en este caso debido a su falta de "estructura computacional". Permiten el desarrollo de análisis racionales de "pros" y "contras" y el estudio de las implicaciones de diferentes decisiones, pero son todavía demasiado complicados para el propósito de esta actividad formativa.

Así pues, ¿qué tipo de modelo de toma de decisiones éticas debe emplearse? En primer lugar, es necesario centrarse únicamente en enfoques deductivos (*top-down*) para la toma de decisiones, puesto que el propósito de esta actividad es comprender las implicaciones y consecuencias de la toma de decisiones a partir de diferentes principios éticos, y no definir un modelo de "aprendizaje" de un agente de inteligencia artificial, en el que se combinaran valores morales (evolutivos) con experiencias de aprendizaje

"reales", lo cual constituiría un escenario ideal pero excesivamente complejo para este caso. En segundo lugar, es necesario reducir cada uno de los diferentes principios éticos a una regla (o a un conjunto reducido de reglas), de forma que se puedan programar y ejecutar comportamientos específicos en función de determinados principios prestablecidos en una máquina que habrá de analizar una serie de *inputs* y ejecutar un *output* en función de dichos principios previamente definidos y programados. En tercer lugar, el modelo resultante debe ser aplicable a escenarios en los que existan dos o más opciones en situaciones de colisiones inevitables.

Así pues, se ha definido un conjunto de enfoques éticos para guiar los procesos de toma de decisiones (aunque se pueden definir y añadir otros), a saber:

A. Enfoques Consecuencialistas

- A1. Utilitarismo: "la mejor acción es la que proporciona el mayor beneficio o causa el menor daño posible".
- A2. Egoísmo (auto-protección): "la mejor acción es la que me protege a mí y a los que están conmigo".
- A3. Ética basada en el beneficio: "la mejor acción es la que produce el menor coste económico".

B. Enfoques No Consecuencialistas

- B1. Deontología: "la mejor acción es la que protege a quienes actúan conforme a las normas".
- B2. No determinismo: "el comportamiento es el resultado del azar".

B. Acción Formativa: Toma de Decisiones Éticas por parte de Vehículos Autónomos en Colisiones Inevitables

Al objeto de describir las actividades de aprendizaje a desarrollar en la clase de Valores Éticos, se propone la siguiente plantilla. De este modo, la planificación didáctica puede utilizarse separadamente del resto del presente trabajo.

1) Título

Introducción a las máquinas morales y la toma de decisiones en clase de Valores Éticos

2) Presentación

El objetivo de esta actividad consiste en formar a los estudiantes de E. S. O. en pensamiento computacional aplicado al ámbito de la Ética. Para ello, la planificación de la lección se centra en las implicaciones éticas de la programación de vehículos autónomos para que estos ejecuten determinados comportamientos con consecuencias morales en situaciones en las que una colisión es inevitable y se producirán daños personales y materiales con toda seguridad. Los estudiantes deben aprender a analizar, representar, estudiar posibles soluciones (*outcomes*) y reflexionar sobre los comportamientos éticos de las máquinas, así como "programar" dichos comportamientos como respuesta a determinados estímulos (*inputs*), siguiendo determinados principios morales.

Edad: 14-16 años.

Nivel: Medio.

Competencias: pensamiento computacional, toma de decisiones, razonamiento lógico, discusión ética, dilemas morales.

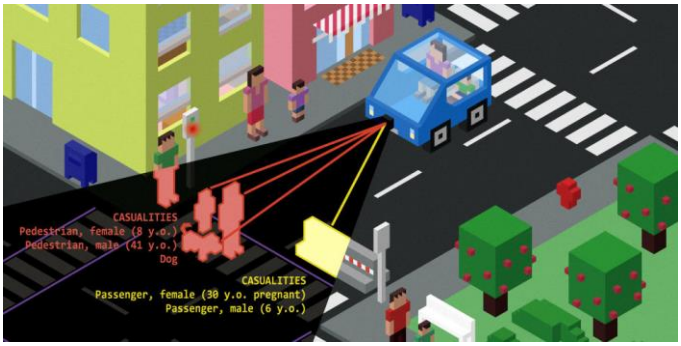


Figura 2. Ejemplo de escenario predefinido para discutir en grupos (Iyad Rahwan. http://www.popularmechanics.com)

3) Finalidad de la Actividad

Esta lección está orientad al desarrollo de competencias relacionadas con el pensamiento computacional, el razonamiento lógico y la toma de decisiones basada en algoritmos. Para ello, se analizarán y estudiarán diferentes enfoques éticos y se formalizarán sus principios morales esenciales al objeto de programar, teóricamente, máquinas inteligentes para que tomen decisiones morales de manera autónoma. Además de estas competencias "computacionales", los estudiantes adquirirán consciencia de la relevancia de los valores y de los enfoques éticos, no solo como principios que sirven como guía de las decisiones cotidianas, sino también para discutir y tratar de alcanzar un consenso en relación con determinados principios morales "socialmente aceptados" para decidir cómo deberían comportarse unas máquinas "inteligentes", tanto en las interacciones con humanos como con otras máquinas y el entorno.

4) Herramientas y Recursos

- ✓ MIT Media Lab: Moral Machine. http://moralmachine.mit.edu.
- ✓ ClaimMS: AccidentSketch.com. http://draw.accidentsketch.com.
- ✓ AV-DMEC *Framework* (véase plan de actividades).
- ✓ Matthieu Cherubini: Ethical autonomous vehicles. https://vimeo.com/85939744.
- ✓ Kahoot: http://kahoot.it.

5) Plan de Actividades

El presente plan de actividades se estructura en cinco sesiones, de la siguiente manera:

Sesión 1. Introducción y Plataforma Moral Machine

La primera sesión se destinará a presentar a los estudiantes los desafíos inherentes a la programación de vehículos autónomos, no solo desde un punto de vista técnico sino también (y quizá principalmente) debido a los problemas de carácter ético.

La primera actividad consistirá en la lectura de un artículo de periódico [2] y la posterior discusión en grupos sobre las diferencias entre los coches autónomos y los actuales vehículos, así como sobre las consecuencias de permitir que sean los vehículos quienes tomen las decisiones. ¿Quién es responsable del daño que provoca el coche? ¿Cómo te sentirías si supieras que, en determinadas circunstancias, el coche estaría dispuesto a matarte en lugar de matar a otros?

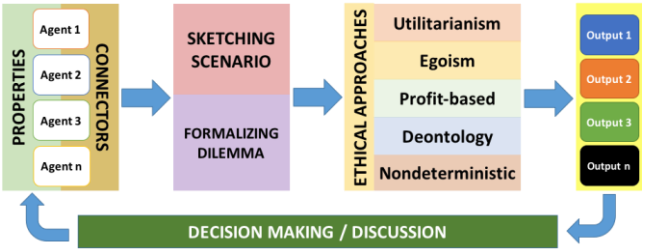


Figura 3. Marco AV-DMEC

En la segunda actividad, los estudiantes habrán de decidir cómo debería comportarse un vehículo en determinadas condiciones. Para ello, se les pedirá que visiten el portal del MIT Lab *Moral Machine*, http://moralmachine.mit.edu (véase figura 1), en el que pueden ver, diseñar y juzgar diferentes escenarios, para posteriormente comparar sus respuestas con los de otros usuarios y sus propios compañeros.

Sesión 2. Análisis de Escenarios Predefinidos

Durante la segunda sesión, el profesor dividirá la clase en grupos y proporcionará a los estudiantes algunos escenarios predefinidos (como se muestra en la Figura 2, por ejemplo). A continuación, los grupos analizarán, discutirán y decidirán cómo debería comportarse en vehículo según determinados enfoques éticos estudiados en temas anteriores, adecuadamente redefinidos para que encajen en las situaciones que se les presenta para su análisis.
A continuación, cada grupo explicará al resto de compañeros el escenario que le ha correspondido para su análisis, así como las posibles opciones de acuerdo con diferentes enfoques éticos. Finalmente, se discutirá cada escenario en clase, tratando de alcanzar un acuerdo sobre el comportamiento y el enfoque ético que implica la "mejor" solución, si tal cosa es posible.

Sesión 3. Escenarios diseñados por los estudiantes (i)

Durante la tercera sesión, se pedirá a los estudiantes que diseñen en grupos sus propios escenarios para la toma de decisiones éticas por parte de máquinas inteligentes, recopilando datos de la matriz presente en la Tabla 1 y que formalicen los procesos de toma de decisiones conforme al marco *Autonomous Vehicle – Decision Making in Ethics Classroom (AV-DMEC)*, tal como se muestra en la Figura 3. Para ello, deberán comenzar por seleccionar el número y naturaleza de *agentes* involucrados en el escenario. A continuación, dotarán a los agentes de determinadas *propiedades* al objeto de completar y clarificar las acciones a analizar. Seguidamente, realizarán un boceto de la escena utilizando algún tipo de herramienta gratuita como AccidentSketck.com, http://draw.accidentsketck.com. Deben describir la escena con proposiciones lógicas utilizando *conectores* y lenguaje natural (por ejemplo: El coche A se cruza en la trayectoria del vehículo autónomo; el vehículo autónomo no puede detenerse a tiempo para impedir la colisión Y atropellará al motociclista O chocará contra la barrera O atropellará a dos peatones en la acera).

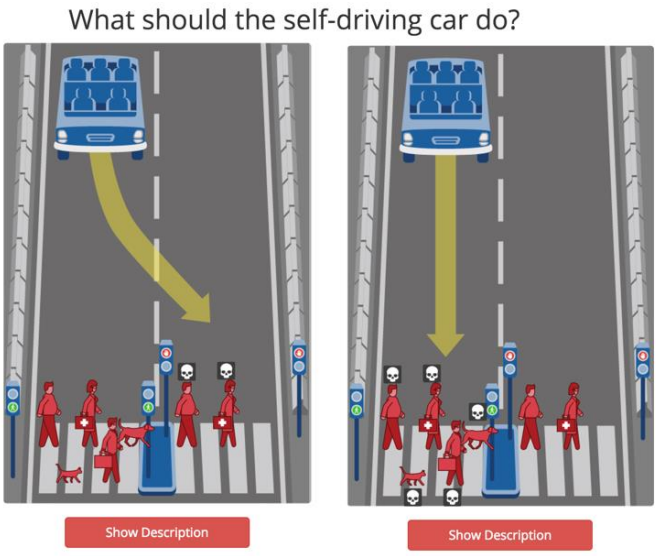


Figura 1. Portal *Moral Machine*. Funcionalidad de juicio

Sesión 4. Escenarios diseñados por los estudiantes (ii)

Tras verificar que el borrador y las proposiciones lógicas encajan y describen con claridad el escenario propuesto por cada grupo, se pedirá a los estudiantes que analicen las posibles respuestas tomando en consideración diferentes *enfoques éticos*. Tras estudiar y discutir tales enfoques, se les pedirá que seleccionen el mejor enfoque ético en función del comportamiento que consideran "más deseable", fundamentando éticamente su decisión.

Sesión 5. Discusión y Feedback

La última sesión se destinará a discutir sobre algunos problemas surgidos de las sesiones 3 y 4. Por ejemplo, ¿existe algún enfoque ético que resulte preferible *por norma general*? ¿Existen escenarios en los que resulta imposible determinar una respuesta "mejor"? ¿Existen escenarios en los que ninguno de los enfoques éticos parece proporcionar una solución razonable?
Con el objetivo de mostrar a los estudiantes la complejidad del pensamiento computacional y los procesos de toma de decisiones, se les invitará a ver el vídeo *Ethical autonomous vehicles*, https://vimeo.com/85939744, en el que Matthieu Cherubini muestra dos casos de estudio (escenarios) bajo tres enfoques éticos diferentes. Se invitará a los estudiantes a analizar los algoritmos éticos presentados y la representación formal, tal como se muestra en http://research.mchrbn.net/eav.
Finalmente, se pedirá a los estudiantes que participen en una actividad de competición de aprendizaje basado en juego (ludificación), al objeto de evaluar lo que han aprendido. Para ello, se realizará un test con Kahoot! (http://kahoot.it), previamente elaborado por el profesor.

TABLA I
MATRIZ CON ELEMENTOS DE EJEMPLO PARA EL DESARROLLO DE ESCENARIOS DE ACCIDENTES CON VEHÍCULOS AUTÓNOMOS

Agentes	**Coche autónomo, Bus, Bus escolar, Moto, Cilcista, Peatón, Obstáculo, Semáforo, […]**
Propiedades	Luz roja/verde, Niño, Bebé, Mujer embarazada, Anciano/a, Sentido contrario, Mismo sentido, Más rápido/Más lento, Con/Sin casco, Cruza correcta/incorrectamente, Impacto, Detención, Atropello […]
Conectoress	Y, O, SI, ENTONCES, O BIEN
Enfoques éticos	Utilitarismo, Egoísmo, Ética basada en el beneficio, Deontología, No determinismo

6) *Evaluación*

El objetivo de evaluación de esta experiencia es doble: por una parte, se pretende evaluar la capacidad de los estudiantes para comprender, definir (tanto visual como lingüísticamente) y tomar decisiones éticas sobre los problemas propuestos. Para ello, el docente tomará nota de las experiencias y resultados, discusiones y argumentaciones de los estudiantes, y les guiará para que alcancen un mejor y más correcto razonamiento lógico y ético, en el caso de que no se estén desarrollando de la manera adecuada. Por otro lado, es necesario evaluar el nivel de satisfacción de los estudiantes con la acción formativa en su conjunto, saber cómo se han sentido y hasta qué punto creen que han mejorado su capacidad para analizar lógicamente dilemas morales y para aplicar diferentes enfoques éticos a diversos escenarios predefinidos o creados por ellos mismos. Esto se llevará a cabo mediante un conjunto de cuestionarios desarrollados con Kahoot! (http://kahoot.it) y un proceso de reflexión final, al que seguirá un breve ensayo individual para su entrega al docente. De este modo, la actividad puede ser evaluada y mejorada para sucesivas iteraciones.

IV. Discusión

Los modelos actualmente disponibles para describir y formalizar procesos de toma de decisiones aplicables a las máquinas morales son todavía demasiado complicados como para que puedan ser utilizados en clases de Ética con estudiantes de enseñanza obligatoria. Dichos modelos se centran en cómo la máquina "actúa" para recopilar información del entorno, cómo su inteligencia artificial procesa la información y cómo es capaz de ejecutar comportamientos programados (incluso con enfoques éticos). En lugar de esto, la actividad formativa aquí descrita está dirigida a reflexionar en la moralidad de la decisión misma más que a tratar de explicar cómo debería comportarse la máquina para actuar éticamente. Interesa más conocer *qué* debería hacer una máquina moral y *por qué* un vehículo autónomo debería actuar de cierta forma adoptando determinadas decisiones, en lugar de explicar *cómo* la máquina sería capaz de llevar a cabo tales tareas, porque nuestros estudiantes desempeñan una función de "filósofos", no de "informáticos". Además, es necesario que los estudiantes sean conscientes de la relevancia de formalizar los dilemas morales a resolver por parte de estas máquinas morales y que sean capaces de analizar las posibles conductas (*outputs*) bajo ciertos presupuestos éticos o enfoques éticos.

Plataformas como *Moral Machine*, del MIT Media Lab, constituyen una herramienta muy interesante para visualizar y juzgar escenarios predefinidos, o incluso para el diseño de nuevas situaciones en contextos similares. Sin embargo, *Moral Machine* se ha diseñado para analizar múltiples actitudes hacia las decisiones morales de vehículos autónomos y, por tanto, las implicaciones éticas se descubren *a posteriori*, en tanto que conjuntos de valores que se conforman en comparación con las respuestas mayoritarias del resto de usuarios. Para el propósito de esta experiencia de aprendizaje, por el contrario, es necesario analizar y tomar decisiones éticas desde enfoques éticos *a priori*, de forma que se puedan comparar diferentes comportamientos que podrían ser preferibles en función del algoritmo moral que fundamente los procesos de toma de decisiones en cada caso. Además, este instrumento del MIT permite diseñar, visualizar y juzgar en escenarios donde solo dos opciones son posibles, lo cual es interesante para el análisis y discusión de un dilema, pero la experiencia resulta más interesante si los escenarios permiten contar con tres o más respuestas o comportamientos posibles, y también, por qué no, si estos comportan un cierto grado de controversia moral.

Esta propuesta no alcanza el nivel de complejidad técnica de un marco para la toma de decisiones aplicable a inteligencia artificial ni la simplicidad (todavía) de una herramienta en la que los usuarios pueden añadir variables, definir *outputs* y evaluar dichas respuestas en función de diferentes enfoques éticos. Sin embargo, posee las virtudes de permitir a los estudiantes definir un conjunto potencialmente infinito de escenarios a partir de un conjunto finito de variables, analizar posibles respuestas mediante el boceto y formalización de los escenarios utilizando el lenguaje natural (pero con estructuras lógicas y conectores) y, finalmente, comparar y discutir las consecuencias de las decisiones adoptadas tomando como referencia los enfoques éticos que los estudiantes conocen perfectamente. Durante la experiencia ponen en práctica competencias de pensamiento computacional, procesos de toma de decisiones, análisis filosófico de comportamientos morales y otras competencias transversales: trabajo en equipo, comunicación oral, debate y argumentación, etc.

Esta planificación didáctica debería favorecer, en consecuencia, que los estudiantes tomaran conciencia de la relevancia de las decisiones éticas y supieran hasta qué punto el desarrollo de la inteligencia artificial presupone la definición de un conjunto de principios éticos (que tanto filósofos como científicos y políticos están muy lejos de consensuar todavía) o un marco ético para la toma de decisiones por parte de las denominadas máquinas morales. El enfoque del pensamiento computacional aplicado a la Ética ayudará a los estudiantes a comprender la necesidad de combinar reflexión ética y computación para formalizar algoritmos capaces de llevar a cabo, cuanto menos, decisiones éticas "débiles", en tanto en cuanto la tecnología, la ciencia y la ética no sean capaces de desarrollar auténticas máquinas morales.

V. Conclusiones y Trabajo Futuro

De todas las ramas de la Filosofía, la Ética es sin duda alguna una de las que se enfrenta a mayores y más importantes desafíos y que comporta implicaciones de mayor relevancia. Aparte de la importancia intrínseca de enseñar a los estudiantes a adquirir competencias para comportarse de manera éticamente responsable, tanto como personas como en su faceta de ciudadanos, es evidente que los enfoques éticos permean y afectan a las ciencias hasta el punto de determinar y condicionar su desarrollo futuro, tal como este trabajo ha demostrado en el caso de las máquinas morales y la inteligencia artificial.

Además, vivir en el siglo XXI comporta la necesidad de adquirir una serie de competencias que todavía se siguen discutiendo, pero entre las cuales se encuentran, sin duda alguna, el pensamiento computacional, la resolución de problemas y las competencias STEM. La educación obligatoria debe garantizar la adquisición efectiva de dichas competencias mediante el desarrollo de planes formativos

que ayuden a nuestros estudiantes a adquirirlas en el seno del currículum escolar. Sin embargo, el pensamiento computacional no es solo cuestión de asignaturas de informática, como tampoco el comportamiento ético y moral es dominio exclusivo de los estudios de carácter filosófico.

El desarrollo de actividades formativas con un enfoque interdisciplinar, que abarcan materias del ámbito de la Informática, Ética, Matemáticas, Artes Plásticas, Lengua, etc., permitirá alcanzar el objetivo de preparar a los estudiantes para converger con su futuro como trabajadores y como ciudadanos.

Esta iniciativa permite a los estudiantes comprender y experimentar las implicaciones de las decisiones morales, al tiempo que ponen en práctica competencias filosóficas como el razonamiento lógico, resolución de dilemas morales, etc. Además, conectar constantemente Ciencias y Humanidades contribuye a comprender el conocimiento humano como un *continuum* en lugar de presentarse como compartimentos estancos, lo cual es un error demasiado frecuente en nuestras sociedades actuales y, por ende, también en nuestros sistemas educativos.

Este conjunto de actividades introduce a los estudiantes en la representación (tanto gráfica como mediante formalización lógicas) de las implicaciones de los comportamientos de las máquinas, en la medida en que se espera que lleguen a convertirse en "máquinas morales" en un futuro próximo. Mientras tanto, los estudiantes pueden analizar las implicaciones de la programación de marcos de toma de decisiones y algoritmos en esas máquinas y discutir sus consecuencias.

Futuros desarrollos de esta investigación avanzarán hacia la implicación de las materias de Informática, de forma que las tareas iniciadas durante la clase de Valores Éticos puedan seguir representándose, mediante el desarrollo de iniciativas de programación o *coding*, en las clases de Informática o TIC. Para ello, y empleando lenguajes de programación como Arduino o Scratch, los escenarios definidos podrían insertarse en robots virtuales o reales, al objeto de mejorar al tiempo las habilidades de pensamiento computacional y comprobar los algoritmos y procesos de toma de decisiones en escenarios "físicos", de forma que se pueda poner en práctica el significado de la *roboética* en auténticos robots.

VI. Agradecimientos

Con el apoyo del Programa de la Unión Europea Erasmus+ Programme. KA2 project "TACCLE 3 – Coding" (2015-1-BE02-KA201-012307).

Este proyecto se ha financiado con el apoyo de la Comisión Europea. Esta comunicación refleja únicamente las opiniones del autor, y la Comisión no puede ser considerada responsable de cualquier uso que se pudiera hacer de la información contenida en este documento.

Referencias

[1] UNESCO. Level-setting and recognition of learning outcomes. The use of level descriptors in the twenty-first century. Paris, United Nations Educational, Scientific and Cultural Organization, 2015.

[2] D. S. Rychen and L. H. Salganik (eds). *Key Competencies for a Successful Life and a Well-Functioning Society*. Göttingen, Hogrefe & Huber, 2003.

[3] European Commission. *Key Competences for Lifelong Learning. European Reference Framework.* Luxembourg, Office for Official Publications of the European Communities, 2007.

[4] Ministerio de Educación Cultura y Deporte. Orden ECD/65/2015, de 21 de enero, por la que se describen las relaciones entre las competencias, los contenidos y los criterios de evaluación de la educación primaria, la educación secundaria obligatoria y el bachillerato. Boletín Oficial del Estado (n. 25, 29th January 2015, pp. 6986-7003). http://www.boe.es/diario_boe/txt.php?id=BOE-A-2015-738.

[5] M. Binkley, O. Erstad, J. Herman, S. Raizen, M. Ripley, M. Miller-Ricci and M. Rumble, "Defining twenty-first century skills" In P. Griffin, B. McGaw, and E. Care (Eds.), *Assessment and teaching of 21st century skills* (pp. 17–66). Netherlands, Springer, 2012.

[6] D. Justo, "España necesitará importar ingenieros si no aumentan los titulados en diez años". *Cadena Ser* (2016-11-10). http://cadenaser.com/ser/2016/10/11/sociedad/1476184570_764354.html.

[7] M. Caprile, R. Palmén, P. Sanz and G. Dente. *Encouraging STEM studies. Labour Market Situation and Comparison of Practices Targeted at Young People in Different Member States. Study*. European Parliament. Directorate General for Internal Policies. Policy Department A: Economic and Scientific Policy, 2015. DOI= 10.2861/519030.

[8] C. Wilson, L. A., Sudol, C. Stephenson and M. Stehlik. *Running on Empty: The Failure to Teach K-12 Computer Science in the Digital Age*. New York, Association for Computing Machinery (ACM), 2010.

[9] A. Balanskat and K. Engelhardt. *Computing our future. Computer programming and coding. Priorities, school curricula and initiatives across Europe*. Brussels, European Schoolnet, 2015.

[10] Conferencia de Directores y Decanos de Ingeniería Informática, Asociación de Enseñantes Universitarios de Informática, *Declaración por la inclusión de asignaturas específicas de ciencia y tecnología informática en los estudios básicos de la enseñanza secundaria y bachillerato.* CODDII-AENUI, 2014. http://coddii.org/wp-content/uploads/2015/06/declaración-CODDII-AENUI.pdf.

[11] CEDEFOP. *EU Skills Panorama (2014) STEM skills Analytical Highlight*. European Commission, 2015. http://skillspanorama.cedefop.europa.eu/sites/default/files/EUSP_AH_STEM_0.pdf.

[12] F. J. García-Peñalvo. "A brief introduction to TACCLE 3 – Coding European Project", in *2016 International Symposium on Computers in Education (SIIE)*, F. J. García-Peñalvo and J. A. Mendes (eds). IEEE, USA, 2016.

[13] F. J. García-Peñalvo, D. Reimann, M. Tuul, A. Rees and I. Jormanainen, "An overview of the most relevant literature on coding and computational thinking with emphasis on the relevant issues for teachers," TACCLE3 Consortium, Belgium, 2016. doi: 10.5281/zenodo.165123

[14] F. J. García-Peñalvo. "Proyecto TACCLE3 – Coding", in *XVIII Simposio Internacional de Informática Educativa, SIIE 2016*, F. J. García-Peñalvo and J. A. Mendes (eds.). Salamanca, Ediciones Universidad de Salamanca, pp. 187-189, 2016.

[15] TACCLE 3 Consortium. TACCLE 3: Coding Erasmus + Project website. City, 2016. http://www.taccle3.eu/en/.

[16] F. J. García-Peñalvo. "What Computational Thinking Is". *Journal of Information Technology Research*, *9*(3), pp. v-viii, 2016.

[17] J. M. Wing. "Research notebook: Computational thinking—What and why?" *The Link Magazine, Spring*. Pittsburgh, Carnegie Mellon University, 2011. Retrieved from https://www.cs.cmu.edu/link/research-notebook-computational-thinking-what-and-why.

[18] J. M. Wing. "Computational Thinking". *Communications of the ACM*, 49(3), pp. 33-35, 2006. doi:10.1145/1118178.1118215.

[19] J. M. Wing. "Computational thinking and thinking about computing". *Philosophical Transactions of the Royal Society, 366*, pp. 3717-3725, 2008. DOI= 10.1098/rsta.2008.0118.

[20] J. M. Wing. "Computational Thinking". In *2011 IEEE Symposium on Visual Languages and Human-Centric Computing*, G. Costagliola, A. Ko, A. Cypher, J. Nichols, C. Scaffidi, C. Kelleher and B. Myers (eds.), 3-3, 2011.

[21] A. M. Seoane Pardo. "Computational thinking beyond STEM: an introduction to "moral machines" and programming decision making in ethics classroom". *Proceedings of the Fourth International Conference on Technological Ecosystems for*

Enhancing Multiculturality, pp. 37-44, 2016. DOI= https://doi.org/10.1145/3012430.3012494.

[22] R. Descartes. *Discurso del método.* Madrid, Alianza Editorial, 2011 (original, 1637).

[23] S. Y. Lye and J. H. L Koh. "Review on teaching and learning of computational thinking through programming: What is next for K-12?" *Computers in Human Behavior, 41,* pp. 51-61, 2014. DOI= 10.1016/j.chb.2014.09.012.

[24] A. Bundy. "Computational thinking is pervasive". *Journal of Scientific and Practical Computing 1,* pp. 67–69, 2007.

[25] J. Cuny, L. Snyder and J. M. Wing. Demystifying Computational Thinking for Non-Computer Scientists. Unpublished work, 2011.

[26] A. Yadav, J. Good, J. Voogt and P. Fisser. "Computational Thinking as an Emerging Competence Domain". In M. Mulder (ed.), *Competence-based Vocational and Professional Education, Technical and Vocational Education and Training: Issues, Concerns and Prospects, 23*, pp. 1051-1067, 2017. DOI= 10.1007/978-3-319-41713-4_49.

[27] S. Grover and R. D. Pea. "Computational Thinking in K-12. A Review of the State of the Field". *Educational Researcher, 42*(1), pp. 38-43, 2013. DOI= 10.3102/0013189X12463051.

[28] M. Anderson and S. Anderson. "Machine Ethics: Creating an Ethical Intelligent Agent". *AI Magazine, 28*(4), pp. 15-26, 2007.

[29] K. G. Coleman. "Android arete. Toward a virtue ethic for computational agents". *Ethics and Information Technology, 3*, pp. 247-265, 2001. DOI=10.1023/A:1013805017161.

[30] L. D. Riek and D. Howard. "A Code of Ethics for the Human-Robot Interaction Profession". *Proceedings of We Robot* (Coral Gables, FL, April 4-5, 2014). Available at SSRN: https://ssrn.com/abstract=2757805.

[31] K. R. Fleischmann, D. W. Oard, A.-S. Cheng, P. Wang and E. Ishita. "Automatic classification of human values: Applying computational thinking to information ethics". *Proceedings of the Association for Information Science and Technology, 46,* pp. 1–4, 2009. DOI= 10.1002/meet.2009.1450460345.

[32] C. Allen, I Smit and W. Wallach. "Artificial morality: Top-down, bottom-up, and hybrid approaches". *Ethics and Information Technology, 7,* pp. 149-155, 2005. DOI=10.1007/s10676-006-0004-4.

[33] J.-F. Bonnefon, A. Shariff and I. Rahwan. "The social dilemma of autonomous vehicles". *Science, 352*(6293), pp. 1573-1576, 2016. DOI= 10.1126/science.aaf2654.

[34] S. M. Veres, L. Molnar, N. K. Lincoln and C. P. Morice. "Autonomous vehicle control systems - A review of decision making". *Proceedings of the Institution of Mechanical Engineers, Part I - Journal of Systems and Control Engineering 225*(3), 2010. DOI= 10.1177/2041304110394727.

[35] W. Knigth. "How to Help Self-Driving Cars Make Ethical Decisions". *MIT Technology Review* (July 29, 2015). Retrieved from https://www.technologyreview.com/s/539731/how-to-help-self-driving-cars-make-ethical-decisions/.

[36] J. Salas. "¿Comprarías un coche que elegiría matarte para salvar otras vidas?". *El País* (June 24, 2016). Retrieved from http://elpais.com/elpais/2016/06/22/ciencia/1466610816_591801.html.

[37] J. D. Greene. "Our driverless dilemma. When should your car be willing to kill you?". *Science 352*(6293), pp. 1514-1515, 2016. DOI= 10.126/science.aaf9534.

[38] P. Foo. *The Problem of Abortion and the Doctrine of the Double Effect in Virtues and Vices.* Oxford, Basil Blackwell, 1978.

[39] J. J. Thomson. "Judith Jarvis Thomson, The Trolley Problem", *Yale Law Journal*(94), pp. 1395-1415, 1975.

[40] N. Goodall. "Ethical Decision Making During Automated Vehicle Crashes". *Transportation Research Record: Journal of the Transportation Research Board, 2424*, pp. 58-65. Washington, D. C. Transportation Research Board of the National Academies, 2014. DOI= 10.3141/2424-07.

[41] M. Cherubini. "Ethical Autonomous Vehicles". Retrieved from http://research.mchrbn.net/eav, 2014.

[42] I. Rahwan, J.-F. Bonnefon and A. Shariff. "Moral Machine". Retrieved from http://moralmachine.mit.edu, 2016.

[43] M. Bratman. *Intention, plans and practical reason.* Chicago, IL,The University of Chicago Press, 1987.

[44] W. Wallach, S. Franklin and C. Allen. "A Conceptual and Computational Model of Moral Decision Making in Human and Artificial Agents". *Topics in Cognitive Science, 2,* pp. 454–485, 2010. DOI= 10.1111/j.1756-8765.2010.01095.x.

Antonio Miguel Seoane Pardo (A Coruña, 1971) es Doctor por la Universidad de Salamanca (2014), Licenciado en Filosofía y Ciencias de la Educación (1994, Universidad de Salamanca) y funcionario en activo del Cuerpo de Profesores de Enseñanza Secundaria de la Junta de Castilla y León (España). Imparte docencia de Filosofía y Geografía e Historia desde 1998, actualmente en el IES Venancio Blanco, de Salamanca. Es Profesor Asociado en el Departamento de Didáctica, Organización y Métodos de Investigación de la Universidad de Salamanca. Ha sido Profesor Colaborador en el Máster Universitario en Informática Pluridisciplinar (Especialidad de Enseñanza y Aprendizaje Electrónico) de la UAH (España) y es docente del Máster en eLearning: Tecnologías y Métodos de Formación en Red, de la USAL (España), y en el Máster TIC en Educación, de esta misma universidad.

Es Coordinador Académico e investigador activo del Grupo de Investigación en InterAcción y eLearning (GRIAL). Es investigador reconocido en metodología de la formación online y diseño estratégico de instituciones virtuales. Fue el primer director de la Universidad Virtual de la USAL hasta diciembre de 2009, a cargo de desarrollar la metodología de formación para los profesores de la Universidad de Salamanca y la estrategia de apoyo a los más de 30.000 usuarios del campus virtual. Colabora habitualmente en iniciativas de formación del profesorado (ha impartido más de 75 cursos, tanto en ámbito nacional como internacional) y es autor de casi un centenar de publicaciones entre libros, capítulos y artículos de impacto sobre eLearning, la metodología de la formación online, la teoría de la comunicación y filosofía y retórica griega antigua.

Ha participado y participa en más de una veintena de proyectos de investigación, tanto nacionales como internacionales, y 16 contratos de transferencia I+D+i. Ha presentado numerosas ponencias en congresos nacionales e internacionales y es autor de un buen número materiales didácticos disponibles en línea sobre herramientas y recursos TIC para la formación en línea.

Escenarios de Aprendizaje para la Asignatura Metodología de la Programación a partir de Evaluar el Pensamiento Computacional de Estudiantes de Nuevo Ingreso

Arturo Rojas-López, y Francisco José García-Peñalvo

CÓMO REFERENCIAR ESTE ARTÍCULO:

Arturo Rojas-López, y Francisco José García-Peñalvo, "Learning scenarios for the subject Methodology of programming from evaluating computational thinking of new students.", en IEEE-RITA, 2018
Doi: https://doi.org/10.1109/RITA.2018.2809941

***Title*—Learning scenarios for the subject Methodology of programming from evaluating computational thinking of new students.**

***Abstract*—Computational thinking is a cognitive process that allows generating solutions to problems through the use of skills such as abstraction, decomposition and algorithmic design. This paper presents a proposal for evaluating the computational thinking of new students, to relate knowledge indicated in the subject Methodology of Programming and to offer an initial environment of learning. The article describes exercises selected to evaluated skills indicated above, and correspondence between contents of the course; using a qualitative methodology, experience of 65 students was evaluated. Results obtained not only favor activity carried out, they also indicated improvements for future generations.**

***Index Terms*—Computational thinking, Computer programming course, Higher education, New students, Programming learning.**

I. Introducción

La aplicación del pensamiento computacional (PC) en educación superior está direccionada hacia las disciplinas de Ciencia, Tecnología, Ingeniería y Matemáticas como lo expone Swaid [1]; para Román-González et al [2] representa el "foco de la innovación educativa por el conjunto de habilidades en la solución de problemas que debe ser adquirido por las nuevas generaciones de estudiantes". El PC "se considera la mejor práctica para la enseñanza de la computación y más ampliamente para resolver problemas y diseñar sistemas" [3], también "ha sido aclamado como valioso para todos en la población, y de ser así, debe ser enseñado a todos" [4], en consecuencia, su promoción es muy alta [5,6].

Las habilidades involucradas en el PC, pueden ser aprendidas por medio de un modelo de juego y a su vez introducir conceptos de programación que eviten la "frustración y la demanda de actividad de los estudiantes" [7,8] de un curso inicial de programación, sin olvidar que el PC no es sinónimo de programación [9]. Un trabajo que relaciona el PC con el estudio inicial de programación de computadoras, es presentado por Lorkins y Harvey [10], donde consideran no solo a estudiantes del área de Ciencia de la Computación (CC) sino aquellos que vienen de una variedad de ciencias e ingenierías para solucionar problemas en sus áreas, inclusive a nivel doctoral [11]; para el caso específico de CC se tiene que ir más allá de "entrenar a los alumnos para solucionar problemas usando un lenguaje de programación en práctica" [12], se tiene que buscar su motivación pues "es un factor importante para mejorar el desempeño del estudiante" [13]. "La construcción de un ambiente de clase basado en la colaboración dirigida por los estudiantes" [14] permite el aprendizaje de habilidades del PC.

En los últimos trece años el modo habitual de impartir la materia inicial a la programación en la Universidad Tecnológica de Puebla ha sido cien por ciento presencial, buscando unificar el aprendizaje de todos los estudiantes basándose completamente en lo indicado en la hoja de asignatura diseñada en su momento a nivel nacional por la Coordinación General de Universidades Tecnológicas. En Puebla factores tales como la deserción, índice alto y variable de reprobación, falta de desempeño académico de los estudiantes fuera de las instalaciones, hacen necesario proponer otras formas que beneficien la calidad educativa. En este panorama, se propuso rediseñar la práctica docente a través del uso del PC que impacte en el aprendizaje de la programación. Nuestro trabajo presenta la novedad de evitar un contenido único para todos los estudiantes en el curso Metodología de la Programación (MP). La organización del

Manuscrito recibido el 8 de mayo, 2017; revisado el 22 de junio; aceptado el 3 de octubre.

English versión received January, 18th, 2018. Revised February, 6th. Accepted February, 19th.

Arturo Rojas López, Universidad Tecnológica de Puebla, Antiguo camino a la resurrección No. 1002-A Zona industrial oriente, 72300, Puebla, México (e-mail: arturo.rojas@utpuebla.edu.mx)

(ORCID no disponible).

Francisco José García Peñalvo, Instituto de Ciencias de la Educación (IUCE), Grupo de Investigación GRIAL, Universidad de Salamanca. Paseo de Canalejas 169, 37008, Salamanca, España (e-mail: fgarcia@usal.es).

(https://orcid.org/0000-0001-9987-5584)

artículo, que es una extensión de los trabajos presentados en el 4th International Conference in Technological Ecosystems for Enhancing Multiculturality (TEEM'16) [15,16], es la siguiente. En la sección Contexto, se presenta el marco de conceptos que ayudan a justificar la propuesta, principalmente la relación de PC y el contenido del curso inicial de programación. En la sección 5 Habilidades – 5 preguntas para evaluar el PC, los reactivos seleccionados para evaluar las habilidades de abstracción, descomposición, generalización, evaluación y diseño algorítmico son comentados. La sección Resultados es importante al proporcionar escenarios de aprendizaje para los 65 estudiantes que realizaron la evaluación del PC, se discuten los comentarios de los estudiantes a la propuesta realizada y además se muestra un experimento de evaluación con estudiantes de tercer cuatrimestre. Finalmente, la sección de Conclusiones expone la importancia del trabajo realizado y el trabajo a futuro considerando la intervención realizada en el cuatrimestre septiembre – diciembre de 2016. Los resultados y conclusiones representan la extensión de los trabajos mencionados anteriormente debido a que contienen información actualizada a partir de la intervención hecha, es decir, su evaluación en el aula, además de actualizar referencias y enriquecer la redacción a lo largo del trabajo.

II. Contexto

El trabajo realizado busca rediseñar la asignatura MP, que está incluida en el primer cuatrimestre de la carrera Técnico Superior Universitario de la división de Tecnologías de la Información y Comunicación (TIC) en la Universidad Tecnológica de Puebla (UTP); cada septiembre inicia una nueva generación y desde el departamento de Servicios Escolares se van formando los grupos para el turno matutino y vespertino que en promedio suman doce de aproximadamente treinta alumnos cada uno. La directora de la división de TIC asigna a los profesores que impartirán la materia MP con base a la disponibilidad reportada por los profesores asignatura o tiempo completo. El objetivo de la asignatura, el objetivo de cada unidad temática, así como la descripción del Saber y Saber Hacer de cada tema está redactado con base a una categoría cognitiva de la Taxonomía de Bloom [17].

En la definición inicial del PC Wing [18] propone que éste involucre la resolución de problemas, el diseño de sistemas, y la comprensión de la conducta humana, basándose en conceptos fundamentales para la CC por lo que resulta útil en un curso inicial de programación. Con el paso del tiempo la solución de problemas continúa siendo parte de la definición [19,20], además de la relación con la CC como en la definición de la Royal Society [21]. El interés por enseñar el PC a una edad temprana es visible en los siguientes escenarios: el documento *Standars in Computer Science* en Norte América [22], el proyecto europeo y sitio Web TACCLE3 – Coding [23 - 26], y el documento Computing our Future [27]; el impacto de aprender PC es importante que James y George lo consideran un elemento clave para el éxito del estudiante en CC [28]. El estudio de Selby [29] explora la relación entre el PC, enseñar programación y la taxonomía de Bloom; desarrolla inicialmente una definición del PC, usada en el presente trabajo, que incluye las habilidades de generalización, descomposición, abstracción, diseño algorítmico y evaluación; posteriormente las mapea al dominio cognitivo de la taxonomía de Bloom para concentrarlas de la siguiente forma: en el nivel de Aplicación la habilidad de generalización, en el nivel de Análisis las habilidades de abstracción y descomposición, en el nivel de Síntesis la habilidad de diseño algorítmico, y el nivel de Evaluación corresponde a la habilidad de evaluación; finalmente relaciona los conocimientos de programación que correspondería enseñar. La definición de la relación que proponemos entre los conocimientos de la asignatura MP y el PC está concentrada en la Tabla I, la descripción y justificación es la siguiente:

- La unidad temática 1 titulada "*conceptos básicos*", tiene seis temas y en la redacción del pilar *Saber* están los verbos Identificar, Reconocer y Describir que corresponden al nivel dos de la taxonomía de Bloom, en el pilar *Saber Hacer* el verbo Determinar, también del nivel dos de Bloom y otro del cuarto, Esquematizar, por lo que se encuentran en el nivel de Comprender y Analizar, en consecuencia se eligió que correspondan a las habilidades de Abstracción y Descomposición.
- La unidad temática 2 nombrada "*expresiones*", usa en los tres temas el verbo Identificar para el pilar *Saber*, que corresponde al nivel dos de Bloom, mientras que en el pilar *Saber Hacer* se usa en tres ocasiones verbos del nivel tres Aplicar por lo que se eligió corresponda a la habilidad de Generalización.
- La unidad temática 3: "*algoritmos y diagramas de flujo*" tiene cuatro temas, en el pilar *Saber* aparecen verbos del nivel cognitivo dos Comprender y en el pilar Saber Hacer un verbo del nivel cinco Evaluar, además de indicar la programación de ejercicios usando el paradigma de programación estructurada por lo que se eligió que correspondan a las habilidades de Evaluación y Diseño algorítmico.

Por lo anterior, aprovechando la relación mencionada y la

Tabla I.
Relación de habilidades del PC y conocimientos de MP

Unidad Temática	Verbos Usados	Nivel taxonomía Bloom	Habilidad PC
1 Conceptos básicos, 6 temas	Saber: Identificar, Reconocer y Describir	2 Comprender	Abstracción
	Saber Hacer: Esquematizar y Determinar	4 Analizar	Descomposición
2 Expresiones, 3 temas	Saber: Identificar	2 Comprender	Generalización
	Saber Hacer: Localizar, Resolver y Convertir	3 Aplicar	
3 Algoritmos y diagramas de flujo, 4 temas	Saber: Reconocer, Identificar y Describir	2 Comprender	Evaluación Diseño algorítmico
	Saber Hacer: Comparar y Resolver	5 Evaluar	

forma en que están determinados los contenidos de la materia se eligieron los reactivos para evaluar el PC de los estudiantes de nuevo ingreso y ofrecer un ambiente inicial de aprendizaje donde se motive a cursar la materia con base a las habilidades detectadas.

III. 5 Habilidades – 5 Preguntas para Evaluar el PC

Los reactivos se eligieron de dos fuentes reconocidas a nivel Internacional: el concurso "sobre fluidez informática y computacional en edades escolares" Bebras [30], y la Olimpiada de Computación de Búsqueda de Talento [31] "que pretende orientar a los estudiantes sudafricanos con más aptitudes en PC hacia carreras técnicas" [2]; debido a que su redacción original es en inglés, se realizó una traducción al español. El motivo por el cual solo se selecciona un ejercicio para cada una de las habilidades del PC, es el tiempo utilizado para responder la evaluación, un promedio de 40 minutos, no es intensión que el estudiante invierta mucho tiempo en responder ejercicios; la validación de cada pregunta está garantizada al ser instrumentos de fuentes internacionales usadas en eventos de competencia, aunque cada ejercicio seleccionado no se limita a evaluar exclusivamente la habilidad para la cual fue seleccionada.

El ejercicio de Bebras denominado Móviles evalúa en gran medida la habilidad de descomposición. La redacción del ejercicio es la siguiente: Un móvil es una pieza de arte que cuelga del techo, generalmente en los dormitorios. Un móvil consiste de palos y figuras. Cada palo tiene unos cuantos puntos donde figuras u otros palos pueden ser atados. Además, cada palo tiene un punto para colgar, donde se cuelga a un palo hacia abajo (o hacia el techo).

La figura 1 es un ejemplo de móvil que puede ser descrito usando números y paréntesis, Pregunta: ¿Cuál de los siguientes móviles puede ser construido usando éstas instrucciones? (-3 (-1 4) (2 (-1 1) (1 1))) (2 (-1 6) (2 3)). El estudiante debe seleccionar la imagen correcta a partir de establecer que la relación de la imagen con la expresión de los números y paréntesis conduce a determinar que el móvil es una composición de pequeños móviles.

El ejercicio nombrado Canguro, también de Bebras, destaca por su evaluación de la abstracción pues el estudiante decide, analiza y determina qué información le va asistiendo para obtener la respuesta correcta del problema planteado, ver Figura 2. La redacción del ejercicio anterior en la evaluación es la siguiente: Hay 10 platos en una fila. Hay una manzana en cada plato. Al canguro Tomás le encanta saltar. Primero, el salta desde el plato más a la izquierda con la letra A. En cada salto después del inicial, salta dos platos hacia adelante, o tres platos hacia atrás. (Un ejemplo de los dos posibles saltos desde un plato es mostrado con flechas en la imagen.) Tomás sólo salta hacia platos con una manzana. Si el salta hacia un plato, recoge la manzana. Pregunta: Si Tomás recoge todas las 10 manzanas, ¿cuál manzana recoge al final? A, B, C, D, E, F, G, H, I, o J. El tercer ejercicio tomado de Bebras con nombre Espías, se enfoca en evaluar la habilidad de generalización, pues una vez que verifica la solución indicada para el problema inicial, aplica a una situación similar la forma de resolver el problema que contiene características parecidas. La redacción del ejercicio es la siguiente: Cada viernes, seis espías intercambian toda la información que han reunido durante la semana. Un espía nunca puede ser visto con más de otro espía al mismo tiempo. Así, tienen que tener varios encuentros de reunión en pares y compartir la información que poseen. El grupo de 6 espías sólo necesitan tres encuentros para distribuir todos sus secretos. Antes del encuentro cada espía mantiene una sola pieza de información (espía 1 conoce 'a', espía 2 conoce 'b', etc.). En el primer encuentro espía 1 y 2 se encuentran y comparten información entonces ahora ambos conocen 'ab', ver Figura 3. Pregunta: Después de un incidente internacional un espía ha dejado de atender los encuentros. ¿Cuál es el número mínimo de reuniones necesitadas por los cinco espías restantes para intercambiar toda la información?

Dos reactivos se tomaron de la Olimpiada de Computación. El primero nombrado Castores en movimiento se eligió para evaluar la habilidad del diseño algorítmico debido a que el alumno que lo resuelva muestra su capacidad de entender la organización de instrucciones para resolver un problema, en este caso las indicaciones para que los castores crucen los hoyos; la redacción del ejercicio en la evaluación es la siguiente: Una colonia de castores está viajando a través de un bosque oscuro. El camino es estrecho, así que tienen que viajar en una fila sin pasar uno del otro. Algunas veces hay un hoyo en el camino. Un hoyo es cruzado de la siguiente manera:

- Primero saltan tantos castores sean necesarios para llenar el hoyo.
- La colonia entera pasará entonces a través del hoyo.
- Los castores que saltaron treparán para salir del hoyo, y unirse al final de la línea

Fig. 1. Descripción del móvil con una expresión: (-3 (-1 1) (1 1)) (2 3)

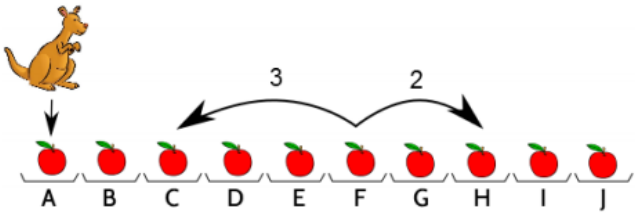


Fig. 2. La tarea de recoger las manzanas

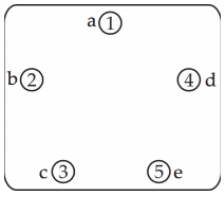


Fig. 3. Cinco espías intercambiando información

Una colonia de 7 castores pasa a través del bosque. Cruzan 3 hoyos. El primer hoyo se ajusta a 4 castores, el segundo se ajusta a 2, y el último hoyo se ajusta a 3 castores, ver Figura 4, ¿En qué orden se encontrarán los castores después de que hayan pasado el tercer hoyo? El segundo reactivo de nombre Salto de charcos, se escogió para determinar la habilidad de evaluación, es decir, realizar paso a paso las instrucciones indicadas para completar el ejercicio y obtener la respuesta solicitada, la redacción del reactivo es la siguiente: Ana (edad 7), Berta (edad 8), Carlos (edad 9), Dora (edad 10) y Luisa (edad 11) están jugando un juego donde saltan de un charco a otro. Ellos han ubicado flechas entre los charcos, y todos inician del lado izquierdo como se indica en la Figura 5. Cuando un niño salta dentro de un charco él o ella espera la llegada de un segundo niño. El niño mayor en el charco entonces saltará de acuerdo a la flecha gruesa, el más joven sigue la flecha delgada. ¿Cuál es el orden (de arriba hacia abajo) en el cual los niños terminarán a la derecha?

IV. Resultados

A. *Escenarios de aprendizaje*

En la plataforma Moodle se crearon tres unidades pero etiquetadas como etapas, con el fin de indicar un aprendizaje progresivo; cada etapa corresponde a las unidades temáticas del curso y contiene el objetivo para que el estudiante conozca cuando ha superado la respectiva etapa. En el encabezado del curso se agregaron dos archivos, el primero contiene el encuadre de la materia y el segundo una definición del concepto Tecnologías de la información, además del objetivo de la asignatura y la encuesta final, que fue habilitada en su respectivo momento al igual que la encuesta a mitad del cuatrimestre. También en cada etapa hay carpetas tituladas como Lecturas, Audios y Evaluaciones donde se encuentran los materiales de estudio en su respectivo formato y archivos de texto con los ejercicios de evaluación; se agregaron actividades tipo Lección para alojar videos y actividades en línea

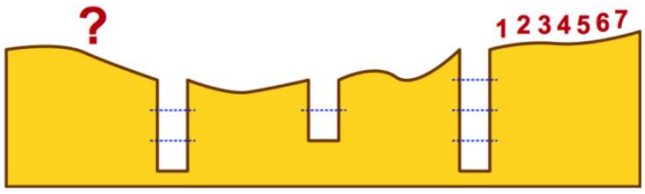


Fig. 4. Siete castores buscando cruzar hoyos

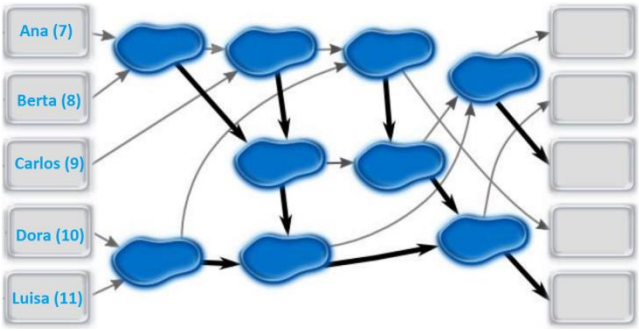


Fig. 5. Salto de Charcos

(crucigramas y sopa de letras) de los temas según la unidad temática; finalmente, se encuentran foros donde los estudiantes responden a preguntas de los temas, suben las evidencias de evaluación y comentan las aportaciones de los compañeros.

Las actividades que se proponen para ofrecer al estudiante un entorno inicial de aprendizaje con base a los reactivos contestados correctamente están determinadas en 10 escenarios:

Escenario 1. 5 respuestas correctas

El estudiante tiene derecho al material del curso en línea y puede realizar quince días después de iniciado el cuatrimestre la evaluación de las unidades temáticas a través de los exámenes elaborados usando la plataforma Moodle y entregando los ejercicios prácticos determinados por la academia. De ser aprobatorios los resultados, el estudiante acredita el curso, en caso contrario se determina una fecha para la revisión presencial docente-estudiante de los ejercicios, se aclaran dudas y si es decisión del estudiante se determinan nuevos ejercicios prácticos a evaluar o tomar el curso en línea.

Escenario 2. Incorrecto el ejercicio de Castores en movimiento

Reforzar el diseño algorítmico con ejercicios a través de la plataforma Moodle, realizar un mes después de iniciado el cuatrimestre las evaluaciones del curso completo y determinar si acredita la materia o elige tomarlo en línea.

Escenario 3. Incorrecto el ejercicio de Salto de charcos

Reforzar las estructuras secuenciales, decisión y repetición codificando ejercicios específicos que resuelven un problema usando las estructuras de control. Calendarizar asesorías presenciales para esos temas y prácticas de laboratorio, realizar un mes después de iniciado el cuatrimestre las evaluaciones del curso completo y determinar si aprueba la materia o elige tomarla en línea.

Escenario 4. Incorrectos los ejercicios de Castores en movimiento y Salto de charcos

El estudiante tiene derecho al material del curso en línea y realizar actividades de práctica para programar los ejercicios de programación estructurada propuestos por la academia. Puede calendarizar tres sesiones de asesoría presencial con el profesor y solicitar al mes de iniciado el cuatrimestre la evaluación de los exámenes en línea y los ejercicios prácticos. Si acredita sus evaluaciones, también el cuatrimestre, en caso contrario puede elegir por tomar el curso en línea o semi-presencial.

Escenario 5. Incorrecto el ejercicio de Espías

Reforzar la evaluación de expresiones aritméticas, lógicas y relacionales considerando la jerarquía de operadores por medio de codificar ejercicios que usen cálculo de operaciones y condicionales; realizar un mes después de iniciado el cuatrimestre las evaluaciones del curso completo y determinar si acredita la materia o elige tomarlo en línea.

Escenario 6. Incorrecto los ejercicios de Castores en movimiento, Salto de charcos y Espías

El estudiante puede trabajar un modelo semi-presencial, donde el material del curso y las actividades deberán ser liberadas con base al tiempo determinado en la hoja de asignatura; el estudiante puede solicitar sesiones de asesoría con el profesor del curso, realizar las prácticas y los

exámenes en los momentos marcados por la academia.

Escenario 7. Correcto el ejercicio de Canguro o Móvil

El estudiante tiene habilidades básicas, pero necesarias para el aprendizaje particularmente de programación, por lo que se puede optar por un entorno semi-presencial, pero con una atención de asesoría presencial regular en las prácticas de laboratorio. Realizar las evaluaciones departamentales indicadas por la academia y ejercicios prácticos evaluativos para desarrollar las habilidades que requieren refuerzo, es decir, Generalización y Diseño algorítmico.

Escenario 8. Incorrecto en todos los reactivos

Existe la posibilidad de que el estudiante no tenga habilidades para el estudio de la Carrera y en la unidad temática tres sea difícil la comprensión de las estructuras de control y diseño algorítmico, por lo que se recomienda en reunión presencial docente-directivo-estudiante valorar el perfil vocacional de éste último. Es muy seguro que el curso presencial sea la mejor opción para el estudiante.

Escenario 9. Incorrectos los ejercicios de Móviles y Canguro

El estudiante trabajará con un modelo semi-presencial, deberá elegir un día a la semana para práctica de laboratorio.

Escenario 10. Correctos los ejercicios de Castores en movimiento y Salto de charcos

El estudiante puede estudiar en línea las primeras dos unidades temáticas, después de cinco semanas trabajar con un modelo semi-presencial.

B. Intervención en el Aula

Durante los primeros días de septiembre de 2016 la evaluación fue aplicada a 65 nuevos estudiantes que correspondieron a dos grupos, 1° C y 1° D, asignados por la dirección de la carrera para que personalmente fuera su docente del curso, así se determinaron los grupos experimentales y de control, este último representado por los grupos 1° A, 1° B, 1° E, 1° F y 1° G que sirvieron para comparar resultados, debido a que los profesores respectivos compartieron las calificaciones de los estudiantes. Los resultados obtenidos de la evaluación del PC por los grupos experimentales son los siguientes:

Para el ejercicio que evalúa la Descomposición 33,8% obtuvo la respuesta correcta. Para el ejercicio que evalúa la Abstracción 52,3% contestó correctamente. Para el ejercicio que evalúa la Generalización 36,9% respondió correctamente. Para el ejercicio que evalúa el Diseño Algorítmico 73,8% respondió apropiadamente. A la pregunta que evalúa la habilidad de Evaluación 67,7% seleccionó la respuesta correcta.

La cantidad de estudiantes que se tuvo para cada escenario de los 2 grupos experimentales está indicada en la Tabla II, a pesar de que existieron mismos escenarios en los dos grupos (excepto el escenario 1 para el 1° D) la variación en las cantidades mostró la individualidad de los estudiantes y la justificación de personalizar su entorno de aprendizaje, es decir, reflejó la diversidad de conocimiento y habilidades con el que ingresaron los estudiantes. Al final del cuatrimestre el resultado de acreditados en los periodos ordinario y extraordinario, así como el número de estudiantes que no acreditaron o se dieron de baja están concentrados en la Tabla III. La calificación aprobatoria corresponde a los valores de Satisfactorio-8, Destacado-9, y Autónomo-10.

El número de estudiantes aprobados es similar a los grupos de control, así como el número de estudiantes que no acreditaron o se dieron de baja, excepto de forma muy drástica con el 1° E y 1° F. El dato de observación está en el número de estudiantes que acreditaron en el periodo extraordinario con el profesor del curso que es mayor en los grupos experimentales, para el caso del 1° G los aprobados en extraordinario fue promovido en un periodo posterior al cuatrimestre por la directora de la carrera.

A mitad del cuatrimestre se solicitó a los estudiantes contestar voluntariamente una encuesta, las preguntas están indicadas en la Tabla IV, así como los valores porcentuales de las respuestas, desafortunadamente solo participaron 24 de los 65 estudiantes. Al final del cuatrimestre otra encuesta

TABLA II.
NÚMERO DE ALUMNOS POR ESCENARIO

Grupo	Escenario	Cantidad
C	5	6
	7	5
	9	8
	10	13
D	1	3
	5	2
	7	10
	9	14
	10	4

TABLA III
RESULTADOS AL FINAL DEL CUATRIMESTRE

Grupo	Ordinario			Extraordinario		Baja
	SA	DE	AU	Aprobados		
A	10	5	6	5	26	8
B	8	4	5	8	25	8
C	2	3	5	18	28	4
D	1	3	2	19	25	8
E	0	1	2	9	12	21
F	3	1	1	8	13	20
G	0	1	6	19	26	6

TABLA IV
ENCUESTA A MITAD DE CUATRIMESTRE

Pregunta	Opción de respuesta
La modalidad de trabajo te parece adecuada con tu expectativa de aprendizaje	Si – 75% No – 25%
¿Conoces los objetivos a los qué tienes que llegar o tienes claro el conocimiento y lo qué debes saber hacer al final del curso?	Si – 66,7% No – 33,3%
¿Te sientes perdido usando la plataforma, no sabes qué hacer y para qué?	Si – 16,7% No – 83,3%
De los recursos contenidos en la plataforma ¿Cuál usaste?	Audio – 45,8% Video – 54,2% Lectura – 70,8% Actividades – 83,3%
¿Qué acción sugieres para mejorar el aprendizaje o estás de acuerdo con tu entorno de aprendizaje?	Respecto a la modalidad de aprendizaje: Estar de acuerdo – 54,2% No está de acuerdo – 16,6% Sin comentario – 29,2% Respecto a sugerencias: Modalidad presencial – 37,5% Más ejercicios – 12,5% Contenido correcto – 29,2% Sin comentario – 20,8%

voluntaria fue realizada y nuevamente la participación fue baja, solo 15 estudiantes respondieron; las preguntas y valores están en la Tabla V. La primera parte preguntó por los conocimientos adquiridos, posteriormente se cuestionó la aprobación de la modalidad de aprendizaje, así como la evaluación inicial del PC, y finalmente se pidieron recomendaciones para estudiantes de futuras generaciones.

La evaluación de las habilidades del PC al inicio del cuatrimestre tuvo una aprobación final del 73%, lo que produjo una aceptación del entorno de aprendizaje del 86%, un resultado muy relevante que justifica la propuesta y su intervención en el aula; aunque la mayoría de los estudiantes acreditaron en el periodo extraordinario, los escenarios de aprendizaje tuvieron aceptación y destaca el uso del contenido de actividades, lectura, y visual en la plataforma Moodle. Los resultados por escenario de aprendizaje fueron los siguientes:

- Escenario 1. Del grupo 1°D los dos estudiantes con calificación Autónomo fueron coherentes al usar un entorno de aprendizaje en línea, recurrieron a las asesorías y presentaron las evidencias de evaluación cuando lo determinaron, obteniendo una calificación aprobatoria. El tercer estudiante obtuvo Destacado debido a que la entrega de evidencias fue del 90% de lo acordado, pero suficiente para acreditar el curso.
- Escenario 5. Los 8 estudiantes de ambos grupos optaron por el modelo semi-presencial al mes de haber iniciado el cuatrimestre. Se determinó un día a la semana para práctica en laboratorio, pero a pesar de estar el material de estudio en línea no lo usaban, pues no se observó avance en laboratorio, se tenía que iniciar con explicación teórica y entonces ejercitar la programación. 3 obtuvieron Destacado, 2 Satisfactorio, y 3 acreditaron en periodo extraordinario.
- Escenario 7. Los 15 estudiantes de los dos grupos eligieron el modelo semi-presencial, aunque se determinó un día de clase teórica presencial y otra de práctica. La evaluación fue con base al tiempo indicado en la hoja de asignatura y los ejercicios determinados por la academia. 1 estudiante obtuvo Satisfactorio, 2 Destacado, 5 Autónomo y 7 acreditaron en extraordinario.
- Escenario 9. Los estudiantes que estudiaban el material en línea fueron los que mejor utilizaron el tiempo de laboratorio, 13 acreditaron el curso en periodo extraordinario y 9 fueron baja.
- Escenario 10. Sin problema trabajaron las dos primeras unidades temáticas en línea, para la última unidad, a pesar de pasar a un modelo semi-presencial no tuvieron el mismo desempeño de autonomía. El diseño de algoritmos y la codificación fue la actividad principal cuando solicitaron horas de laboratorio que no estaban consideradas en el escenario. De los 17 estudiantes, 14 acreditaron en extraordinario y 3 desertaron.

TABLA V
ENCUESTA AL FINAL DEL CUATRIMESTRE

Pregunta	Opción de respuesta
Selecciona los conceptos con los que estás familiarizado	
Tipos de datos	73,3%
Operadores aritméticos	60,0%
Creación de identificadores para variables	66,7%
Operadores lógicos	73,3%
Operadores relacionales	66,7%
Jerarquía de operadores	73,3%
Resolver expresiones aritméticas, lógicas, y relacionales	66,7%
Uso de una variable contador y acumulador	73,3%
Estructura de selección	26,7%
Estructura de repetición	73,3%
Definición y creación de un algoritmo	66,7%
¿La modalidad de aprendizaje fue adecuada para adquirir las competencias del curso?	Si – 86,7% No – 13,3%
¿La evaluación de tus habilidades al inicio del cuatrimestre fue una actividad acertada para determinar el mejor entorno de aprendizaje?	Si – 73,3% No – 26,7%
¿Qué recomendación tienes para las futuras generaciones acerca de la forma de aprender el contenido del curso?	Estudiar – 46,6% Presencial – 20% Ninguna – 20% Lo mismo para todos – 6,7% Me gustó y motivó a investigar y aprender más por mi cuenta – 6,7%

Por lo anterior, la forma de ofrecer los contenidos de la asignatura a partir de evaluar las habilidades del PC representa una novedad para nuestro caso de estudio y puede servir para otros contextos similares a nivel global.

El número de escenarios propuestos al inicio de esta sección bien puede corresponder a los casos detectados en un experimento realizado con alumnos de tercer cuatrimestre y que se muestra a continuación.

Previo al periodo de septiembre – diciembre de 2016, un experimento se realizó con 18 estudiantes voluntarios de tercer cuatrimestre bajo las siguientes condiciones. Se solicitó responder una encuesta en línea acerca de los conocimientos con los que se sentían ya familiarizados de su curso de primer cuatrimestre, la Tabla VI contiene los elementos que marcaba el estudiante, y posteriormente realizaron la evaluación del PC. La intención fue verificar la correspondencia entre sus conocimientos y las habilidades del PC; los resultados determinaron 6 casos.

- Caso 1. Estudiantes que indican saber todos los conocimientos y respuestas correctas de los reactivos del PC por unidad temática, la Tabla VII en la segunda columna representa un resultado ideal de relación.
- Caso 2. Resultado contradictorio de estudiantes que dicen saber todos los conocimientos, pero obtuvieron respuestas incorrectas de los reactivos del PC por unidad temática, la Tabla VII en la tercera columna muestra una cantidad de valores no deseados.
- Caso 3. Diferencia entre estudiantes que dicen saber algunos de los conocimientos y respuestas correctas de los reactivos del PC por unidad temática, la Tabla VII con un valor alto en la unidad temática tres, en la cuarta columna, ilustra que esos estudiantes pueden tener una

falta de confianza de sus conocimientos a pesar de poder resolver los ejercicios respectivos.

- Caso 4. Correspondencia entre algunos conocimientos indicados como no familiarizados por unidad temática y un reactivo erróneo para el caso de dos habilidades del PC, la relación es correcta y justifica también el ofrecer contenido de regularización, ver la Tabla VII en la quinta columna.
- Caso 5. Número de estudiantes que respondieron estar familiarizados con todos los conocimientos de cada unidad temática respectiva y obtuvieron un reactivo erróneo para el caso de dos habilidades del PC, el número 4 en la sexta columna de la Tabla VII, muestra que no todo depende de la evaluación del PC, sí justifica ser una guía para el docente así como la evaluación de los contenidos mediante la generación de las evidencias que determine la académica de la asignatura.
- Caso 6. Resultado de los que aún no están completamente seguros de todos los conocimientos por unidad temática y con reactivos erróneos del PC, la Tabla VII indica con el valor 6 de la unidad temática 2 en la séptima columna, el caso de estudiantes que al no responder bien el reactivo se indica que no tienen aún la habilidad respectiva para los conocimientos de la unidad mencionada.

V. Conclusiones

Se tiene una propuesta justificada para usar la evaluación del PC y determinar la forma de cursar la materia de metodología de la programación para los estudiantes de nuevo ingreso. La relación entre las habilidades evaluadas y los temas de la asignatura usan los resultados reportados por el trabajo de Selby [29] y con base a la experiencia docente se determinan las opciones del entorno de aprendizaje; los escenarios propuestos representan la acción de trabajo con el estudiante ante la respuesta incorrecta de una habilidad específica, por ejemplo, si contestó erróneamente el reactivo que evalúa la habilidad de Descomposición, que corresponde en parte a la unidad temática 1 Conceptos y al escenario 9, el entorno propone al estudiante trabajar en los temas respectivos y favorecer el aprendizaje autónomo. Al final del cuatrimestre septiembre – diciembre de 2016 el número de estudiantes acreditados es poco más del 80% (53 de 65 estudiantes), un 18,5% de deserción (12 estudiantes) y un promedio del 65,45% de nivel académico de egreso, porcentajes similares a grupos que no trabajaron la propuesta y mucho mejor a dos grupos en particular. Los estudiantes no están preparados para un aprendizaje autónomo, pues se observaron tres conductas comunes: requieren de recordarles las actividades de estudio, fecha de entrega de trabajos generalmente determinadas por el profesor y limitar su fuente de conocimiento a lo proporcionado por la Universidad; la falta de una disciplina de estudio hace difícil tomar el control de su proceso educativo, así lo indica el número alto de acreditados durante el periodo extraordinario de los escenarios 9 y 10 en comparación con el número de aprobados en extraordinario en los grupos de control que tenían clase presencial; la mayoría de los estudiantes indicó estar cómodo en un ambiente presencial donde la responsabilidad solo sea del docente y las estrategias que realice frente a grupo. Existe un entusiasmo por el trabajo semi-presencial, pero deberá trabajarse una planeación y seguimiento de actividades más puntual que permita al estudiante sentir el control externo de su proceso educativo. El uso de la evaluación del PC indica dos resultados positivos. En primer lugar, existen estudiantes que sienten motivación por que sean reconocidas sus habilidades; y en segundo lugar, la evaluación permite determinar un entorno de aprendizaje inicial. El trabajo a futuro incluye mejorar la empatía por la propuesta de trabajo para aumentar la participación en las encuestas a mitad y final del cuatrimestre y así tener mejores niveles de evaluación del trabajo educativo propuesto.

Tabla VI
Conocimientos generales por unidad temática

Unidad Temática	Conocimientos
1	Tipos de datos Identificadores de variables
2	Operadores aritméticos Operadores lógicos Operadores relacionales Jerarquía de operadores Resolver expresiones
3	Uso de variable contador y acumulador Estructura de selección (condicional) Estructura de repetición (ciclo) Diagrama de flujo Diseño de algoritmos

Tabla VII
Cantidad de estudiantes por caso

Unidad temática	1	2	3	4	5	6
1	6	2	2	4	4	0
2	4	8	0	-	-	6
3	2	0	13	3	0	0

Agradecimientos

El presente trabajo es realizado dentro del programa de Doctorado en Educación en la Sociedad del Conocimiento de la Universidad de Salamanca.

Referencias

[1] S. I. Swaid, "Bringing Computational Thinking to STEM Education", in 6th International Conference on Applied Human Factors and Ergonomics AHFE and the Affiliated Conferences, AHFE 2015 Procedia Manufacturing, Volume 3, 2015, Pages 3657-3662, ISSN 2351-9789, DOI= http://dx.doi.org/10.1016/j.promfg.2015.07.761.

[2] M. Román-González, J. C. Pérez-González, and C. Jiménez-Fernández, “Test de Pensamiento Computacional: diseño y psicometría general”, in III Congreso Internacional sobre Aprendizaje, Innovación y Competitividad (CINAIC 2015), Octubre 14-16, 2015, Madrid, ESPAÑA

[3] J. A. Rode, A. Weibert, A. Marshall, K. Aal, T. Von Rekowski, H. El Mimoni, J. Booker, “From computational thinking to computational making”, in UbiComp 2015 - Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing, pp. 239-250. DOI:= http://dx.doi.org/10.1145/2750858.2804261.

[4] E. Cole, “On pre-requisite skills for universal computational thinking education”, in ICER 2015 - Proceedings of the 2015 ACM Conference on International Computing Education Research, pp. 253-254. DOI:= http://dx.doi.org/10.1145/2787622.2787737.

[5] Y. Li, Research into the computational thinking for the teaching of computer science", in Proceedings - Frontiers in Education Conference, FIE, 2015-February DOI:= http://dx.doi.org/10.1109/FIE.2014.7044465.

[6] B. C. Czerkawski, E. W. Lyman III, Exploring Issues About Computational Thinking in Higher Education", in TechTrends, 59 (2), pp. 57-65. DOI:= http://dx.doi.org/ 10.1007/s11528-015-0840-3.

[7] M. Román-González, J. C. Pérez-González, and C. Jiménez-Fernández, "Which cognitive abilities underlie computational thinking? Criterion validity of the Computational Thinking Test", Computers in Human Behavior, vol. 72, pp.678-691, 2017

[8] C. Kazimoglu, M. Kiernan, L. Bacon, and L. MacKinnon, "Learning Programming at the Computational Thinking Level via Digital Game-Play", in Proceedings of the International Conference on Computational Science, Procedia Computer Science, Volume 9, 2012, Pages 522-531, ISSN 1877-0509, DOI= http://dx.doi.org/10.1016/j.procs.2012.04.056.

[9] F. J. García-Peñalvo, D. Reimann, M. Tuul, A. Rees, and I. Jormanainen, "An overview of the most relevant literature on coding and computational thinking with emphasis on the relevant issues for teachers," TACCLE3 Consortium, Belgium. 2016. DOI= 10.5281/zenodo.165123.

[10] D. B. Larkins, and W. Harvey, "Introductory computational science using MATLAB and image processing", in Procedia Computer Science, Volume 1, Issue 1, 2010, Pages 913-919, ISSN 1877-0509, DOI= http://dx.doi.org/10.1016/j.procs.2010.04.100.

[11] G. W. Shiflet, and A. B. Shifleta, "Introducing Life Science Doctoral Students in Oz to the Wizardry of Computational Modeling: Introducing Computational Thinking with CellDesigner™", in Proceedings of the International Conference on Computational Science, ICCS 2012. Procedia Computer Science, Volume 9, 2012, Pages 1753-1762, ISSN 1877-0509, DOI= http://dx.doi.org/10.1016/j.procs.2012.04.193.

[12] Z. Lingling, X. Su, and T. Wang, "Bring CS2013 Recommendations into c Programming Course", in International Educational Technology Conference, IETC 2014, 3-5 September 2014, Chicago, IL, USA, Procedia - Social and Behavioral Sciences, Volume 176, 2015, Pages 194-199, ISSN 1877-0428, DOI= http://dx.doi.org/10.1016/j.sbspro.2015.01.461.

[13] A. Ramirez, and D. F. Muñoz, "Increasing Practical Lessons and Inclusion of Applied Examples to Motivate University Students during Programming Courses", in International Educational Technology Conference, IETC 2014, 3-5 September 2014, Chicago, IL, USA, Procedia - Social and Behavioral Sciences, Volume 176, 2015, Pages 552-564, ISSN 1877-0428, DOI= http://dx.doi.org/10.1016/j.sbspro.2015.01.510.

[14] B. Worrell, C. Brand, and A. Repenning, "Collaboration and Computational Thinking: A classroom structure", in Proceedings of IEEE Symposium on Visual Languages and Human-Centric Computing, VL/HCC, 2015-December, art. no. 7357215, pp. 183-187. DOI:= http://dx.doi.org/10.1109/VLHCC.2015.7357215.

[15] A. Rojas López and F. J. García-Peñalvo, "Relationship of knowledge to learn in programming methodology and evaluation of computational thinking", in *Proceedings of the Fourth International Conference on Technological Ecosystems for Enhancing Multiculturality (TEEM'16) (Salamanca, Spain, November 2-4, 2016)*, F. J. García-Peñalvo, Ed. (ICPS: ACM International Conference Proceeding Series, New York, NY, USA: ACM, 2016, pp. 73-77.

[16] A. Rojas López and F. J. García-Peñalvo, "Personalized contents based on cognitive level of student's computational thinking for learning basic competencies of programming using an environment b-learning", in *Proceedings of the Fourth International Conference on Technological Ecosystems for Enhancing Multiculturality (TEEM'16) (Salamanca, Spain, November 2-4, 2016)*, F. J. García-Peñalvo, Ed. (ICPS: ACM International Conference Proceeding Series, New York, NY, USA: ACM, 2016, pp. 1139-1145.

[17] B. S. Bloom, *Taxonomía de los objetivos educacionales, Manual I: El dominio cognitivo*. Nueva York: David McKay Co Inc., 1956.

[18] J. M. Wing, "Computational Thinking", *Communications of the ACM*, vol. 49, no. 3, pp. 33-35, 2006.

[19] V. Barr, and C. Stephenson, "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=http://10.1145/1929887.1929905.

[20] F. J. García-Peñalvo, "What Computational Thinking Is", *Journal of Information Technology Research*, vol. 9, no. 3, pp. v-viii, 2016.

[21] Royal Society, "Shut down or restart: The way forward for computing in UK schools", Retrieved from London, UK: https://royalsociety.org/~/media/education/computing-in-schools/2012-01-12-computing-in-schools.pdf

[22] CSTA. 2011. "K-12 Computer Science Standars", Retrieved from http://csta.acm.org/Curriculum/sub/CurrFiles/CSTA_K-12_CSS.pdf

[23] TACCLE 3 Consortium. 2017. TACCLE 3: Coding Erasmus + Project website. Retrieved from http://www.taccle3.eu/

[24] F. J. García-Peñalvo, "Proyecto TACCLE3 – Coding", in *XVIII Simposio Internacional de Informática Educativa, SIIE 2016*, F. J. García-Peñalvo and J. A. Mendes, Eds. no. 222 Salamanca, España: Ediciones Universidad de Salamanca, 2016, pp. 187-189.

[25] F. J. García-Peñalvo, "A brief introduction to TACCLE 3 – Coding European Project", in *2016 International Symposium on Computers in Education (SIIE 16)*, F. J. García-Peñalvo and J. A. Mendes, Eds. USA: IEEE, 2016.

[26] F. J. García-Peñalvo, "Presentación del Proyecto TACCLE3 Coding," presented at the Workshop EI<18. Educación en Informática sub 18, Salamanca, España, 2016. Available: http://repositorio.grial.eu/handle/grial/653

[27] European Schoolnet. 2015. Computing our future – Computer programming and coding, Priorities, school curricula and initiatives across Europe.

[28] L. James, and L. F. George, "Thinking about computational thinking", in Proceedings of the 40th ACM technical symposium on Computer science education, Chattanooga, TN, USA, pp. 260-264.

[29] C. C. Selby, "Relationships: computational thinking, pedagogy of programming, and Bloom's Taxonomy", in *Proceedings of the Workshop in Primary and Secondary Computing Education* (WiPSCE '15). ACM, New York, NY, USA, 80-87, DOI= http://dx.doi.org/10.1145/2818314.2818315.

[30] UK Bebras Computational Thinking Challenge, answers 2015, University of Oxford, available http://www.bebras.org

[31] Talent Search. 2015, Elite: Grade 12+, Institute of IT Professionals South Africa, available http://www.olympiad.org.za.

Arturo Rojas López realizó sus estudios universitarios y de maestría con mención Cum Laude en Ciencias de la Computación en la Benemérita Universidad Autónoma de Puebla. Trabaja como profesor de tiempo completo en la Universidad Tecnológica de Puebla de Enero de 2003 a la fecha en la División de Tecnologías de la Información y Comunicación, particularmente en el área de Sistemas Informáticos. Ha participado en dos ocasiones (2009 y 2017) en la actualización de los planes de estudio en las reuniones de trabajo del sistema de Universidades Tecnológicas de México. En 2013 coordinó el Segundo Congreso Internacional de TIC de las Universidades Tecnológicas – INTED (Innovación, Tecnología y Educación para el progreso social). Ha ofrecido talleres y ponencias en diferentes Universidades Tecnológicas de México con diferentes temáticas en el desarrollo de software. Actualmente estudia el Programa de Doctorado Formación en la Sociedad del Conocimiento en la Universidad de Salamanca; los temas de investigación que son de su interés son la Educación personalizada, el Pensamiento computacional, Gamificación y eLearning.

Francisco José García Peñalvo realizó sus estudios universitarios en informática en la Universidad de Salamanca y en la Universidad de Valladolid y se doctoró en la Universidad de Salamanca. El doctor García-Peñalvo es el director del grupo de investigación GRIAL (Grupo de investigación en Interacción y eLearning). Sus principales intereses de investigación se centran en el eLearning, Computadores y Educación, Sistemas Adaptativos, Ingeniería Web, Web Semántica y Reutilización de Software. Ha dirigido y participado en más de 50 proyectos de innovación e investigación. Fue Vicerrector de Innovación Tecnológica de la Universidad de Salamanca entre Marzo de 2007 y Diciembre de 2009. Ha publicado más de 200 artículos en revistas y conferencias internacionales. Ha sido editor invitado en varios números especiales de revistas internacionales (*Online Information Review, Computers in Human Behaviour, Interactive Learning Environments …*). Es el editor en jefe de las revistas Education in the Knowledge Society y Journal of Information Technology Research. Coordina el Programa de Doctorado en Formación en la Sociedad del Conocimiento de la Universidad de Salamanca.

"Evolución": Diseño e Implementación de Material Educativo Digital para Fortalecer Habilidades del Pensamiento Computacional

Mauricio Javier Rico Lugo, Xabier Basogain Olabe, Nancy Moreno Niño

CÓMO REFERENCIAR ESTE ARTÍCULO:

Mauricio Javier Rico Lugo, Xabier Basogain Olabe, Nancy Moreno Niño, "'Evolution': Design and implementation of digital educational material to strengthen computational thinking skills.", en IEEE-RITA, 2018
Doi: https://doi.org/10.1109/RITA.2018.2809943

***Title*—'Evolution': Design and implementation of digital educational material to strengthen computational thinking skills.**

***Abstract*—This article describes how an educational digital material supports the improvement of skills in the development of algorithms within Math problems, which belong to a part of an operative computational thinking, in students of the subject Logic Programming. This study reflects how students lack the fundamentals of Math and the algorithmic thinking required to solve the problems established in the educational digital material. The results show that doing a reiterative challenge help students not just to get the fundamentals of Math, but also to solve the algorithms used in the educational digital material, learning from the mistakes made.**

***Index Terms*—Algorithms, computational thinking, critical thinking, educational digital material, Information and communication technologies.**

I. Introducción

La mayoría de centros educativos a nivel mundial donde se imparten carreras relacionadas con la informática, se dicta una asignatura de introducción o fundamentos de programación. El nombre de la asignatura puede variar y su contenido incluye los conceptos básicos de algoritmos [1] definidos como pasos finitos y ordenados para resolver un problema que conlleva la realización de procesos matemáticos básicos [2]. El curso de Lógica de Programación (correspondiente al programa de desarrollo de software) es la primera asignatura relacionada con informática que afrontan los estudiantes de primer semestre del Instituto Colombiano de Aprendizaje (INCAP). Los estudiantes de esta asignatura presentan históricamente un bajo desempeño en matemáticas, que como lo refiere el Ministerio de Educación Nacional de Colombia es el área que junto con comprensión lectora presenta un alto porcentaje (70%) de reprobación en las pruebas de Estado. En el informe del Instituto Colombiano para la Evaluación de la Educación (ICFES) sobre los resultados de las pruebas PISA (Programa de Evolución Internacional de Estudiantes) del 2012, se evidencia que en el área de matemáticas sólo dos de cada diez estudiantes pueden realizar interpretaciones literales de problemas matemáticos, que contienen algoritmos básicos, fórmulas y procedimientos para resolver problemas de números enteros.

Linares [3] explica que esta dificultad de los recién ingresados a la educación superior también afecta a sus profesores ya que estos deben retomar conceptos matemáticos como la regla de los signos, reforzar operaciones como las divisiones, y aquellos módulos que se debieron adquirir en la etapa escolar. Este déficit afecta a la docencia de todas las materias que involucran conceptos matemáticos.

En esta nueva era de las tecnologías y la comunicación se requiere combinar habilidades de diferentes formas de pensamiento crítico, matemático y algorítmico que dan lugar a una nueva forma de razonar denominado el Pensamiento Computacional (PC). El PC es un concepto emergente para poder resolver problemas de este siglo que necesitan soluciones innovadoras, creativas y que generen un reto para los estudiantes; para alcanzar soluciones a estos problemas el estudiante debe contar con múltiples habilidades para analizar y comprender el problema. Esto significa que el estudiante debe tomar posturas a favor o en contra sobre un tema en particular en un modo reflexivo [4].

El PC no es un concepto particular de quienes se especializan en las Ciencias de la Computación. El PC es una forma de razonamiento que se debe fomentar con herramientas que permitan hallar más de una solución a un problema o una situación en particular; evaluar entre múltiples posibilidades la solución más adecuada y poder sustentarla. El presente estudio se plantea cómo fomentar este tipo de pensamiento desde el aula y asociarlo con los conocimientos adquiridos en los años escolares.

Manuscrito recibido Septiembre 29, 2017; revisado Noviembre 30, 2017; aceptado Diciembre 29, 2017.

English versión received January, 13th, 2018. Revised February, 6th. Accepted February, 19th.

Mauricio J Rico Lugo. Magister en Informática Educativa de la Universidad de la Sabana, Chía Colombia (e-mail: mauriciorilu@unisabana.edu.co).

(https://orcid.org/0000-0003-1271-5700).

Xabier Basogain Olabe. Doctor Ingeniero de Telecomunicaciones de la Universidad del País Vasco – Euskal Herriko Unibertsitatea (email: xabier.basogain@ehu.eus).

(https://orcid.org/0000-0002-6672-6897).

Nancy Moreno Niño. Magister en Informática Educativa de la Universidad de la Sabana, Chía Colombia (e-mail: nmorenon@educacionbogota.edu.co)

(https://orcid.org/0000-0002-2345-9928).

Esta investigación tiene como objetivo diseñar un Material Educativo Digital (MED) con diferentes retos que permita fortalecer habilidades propias del Pensamiento Computacional y evaluar su efecto en los estudiantes. Para este estudio se implementó una prueba piloto en un grupo de estudiantes de la materia Lógica de Programación.

El enfoque de la investigación es cualitativo; se realizó en un escenario común tanto para el profesor como para los estudiantes. No se pretendió establecer unos datos estadísticos cuantitativos [5] que se podrían obtener del MED de forma automática. Los instrumentos de recolección de datos fueron diarios de campo, software grabador de pantalla, y grabación de voz para los grupos focales. Con estos instrumentos se ha observado la interacción del estudiante con el MED, en particular qué dificultades encontraba, cómo las solucionaba, qué respondía al azar, qué retos eran fáciles o cuáles más difíciles, qué opinaban del MED, cuál es su aporte a nivel personal, y la relación del MED con el PC. Este pilotaje ha servido para realizar correcciones, medir tiempos, adicionar retos, e implementar estrategias que permitan a futuro tener un estudio completo de la población tanto en la sede principal de INCAP como en las otras sedes a nivel nacional.

El presente artículo presenta en primera lugar una revisión teórica de los conceptos de pensamiento computacional y los materiales educativos digitales que enmarcan el estudio. Se presenta la metodología del diseño de la investigación bajo un enfoque cualitativo utilizando la metodología de caso de estudio, realizado en un salón de la materia de lógica de programación del instituto INCAP. En este apartado metodológico se describen las técnicas de recolección datos, la planificación para la elaboración del MED y el pilotaje del material.

En el apartado de resultados se describe cómo se recolectaron los datos y se analizaron mediante diferentes instrumentos; además se presentan los resultados según los momentos de la investigación, permitiendo realizar comparaciones antes y después de la aplicación del MED con los estudiantes.

Por último el apartado de conclusiones describe las principales conclusiones a partir del análisis de los resultados, y los estudios futuros a realizar.

A. *Pensamiento Computacional*

El término Pensamiento Computacional es relativamente nuevo. Wing [6] indica que el PC es un conjunto de habilidades universales para todo el mundo y no solamente para las personas vinculadas con las ciencias de la computación. Se basa en el poder de las computadoras combinado con el pensamiento humano y sirve para solucionar problemas cotidianos, diseñar sistemas y realizar tareas rutinarias [7]. Esta forma de pensamiento debe ser desarrollado en la escuela de la misma manera que se hace con la escritura y la aritmética. Si se tiene en cuenta lo planteado por Wing [6], estas habilidades del pensamiento computacional tienen sus raíces en la teoría construccionista de Seymourt Papert de los años 80 que vincula el aprendizaje con la computación; en esta teoría el conocimiento no se transmite sino que se construye por cada persona. El aprendizaje requiere la ayuda de una persona o de un medio material (particularmente tecnológico) [8]. El lenguaje de programación LOGO creado por Papert [9] fijó las bases de la teoría del construccionismo favoreciendo la autonomía del estudiante para experimentar, cometer errores y reflexionar sobre ellos; en este proceso se construye el conocimiento.

La Sociedad Internacional para la Tecnología en la Educación (ISTE) y la Asociación de Maestros de Ciencias de la Computación (CSTA), acompañados de líderes de la educación superior, la industria y la educación básica secundaria y media, se reunieron en el año 2011 para plantear una definición operativa del pensamiento computacional. Definen el Pensamiento Computacional como un proceso de solución de problemas que incluye las siguientes características básicas [4]:

1) Formular problemas de manera que permitan usar computadores y otras herramientas para solucionarlos.
2) Organizar datos de manera lógica y analizarlos.
3) Representar datos mediante abstracciones, como modelos y simulaciones.
4) Automatizar soluciones mediante pensamiento algorítmico.
5) Identificar, analizar e implementar posibles soluciones con el objeto de encontrar combinación de pasos y recursos más eficientes.
6) Generalizar y transferir ese proceso de solución de problemas a una gran diversidad de estos.

Al hacer una reflexión sobre las aproximaciones dadas por los diferentes autores sobre pensamiento computacional, se encuentra como constante la necesidad de solucionar problemas; sin importar el escenario donde se encuentre el individuo siempre hay algo que requiere ser solucionado, mejorado o inventado. La solución basada en un algoritmo sirve tanto para problemas matemáticos como para problemas del mundo real a los cuales se puede dar respuesta con el desarrollo de una aplicación de software. Es así como en el pensamiento computacional interviene el pensamiento lógico, sistémico y algorítmico [9], [10].

A partir de la revisión sobre los fundamentos del pensamiento computacional y tal como lo presenta el ISTE [4], se encuentra que este tipo de pensamiento está sustentado por las habilidades propias de otros modos de razonamiento. Entre ellos está el pensamiento algorítmico, el pensamiento crítico y el pensamiento matemático. La figura 1 ilustra los otros tipos de pensamiento que intervienen en el pensamiento computacional.

Fig. 1. Relación del pensamiento algorítmico, crítico y matemático con el pensamiento computacional.

La definición operativa de PC lo describe como un proceso para solucionar problemas que incluye características como la organización de datos de una manera lógica, la realización de abstracciones, la representación de soluciones y su automatización mediante el pensamiento algorítmico. El pensamiento algorítmico es una habilidad fundamental para el desarrollo del PC que incluye una reflexión crítica para analizar las diferentes soluciones encontradas y determinar cuál es la mejor. Las habilidades de pensamiento matemático se hacen necesarias en muchas de las actividades diarias como realizar comparaciones, evaluar un orden o una magnitud; estas habilidades son utilizadas de forma natural. El cálculo mental tiene unas connotaciones tradicionales en la atención y en la memoria, pero también en funciones sociales y educativas [11].

B. *Material Educativo Digital*

Hoy día los estudiantes cuentan con entornos que permiten el fácil contacto con las Tecnologías de la Información y Comunicación (TIC), sin importar la edad y supliendo diferentes necesidades. En el ámbito educativo se encuentra un área dedicada al uso de las TIC, y dentro de las materias enseñadas destaca la utilización de software (procesador de texto, hojas de cálculo, presentaciones, etc.); también se enfoca el uso de las TIC en la búsqueda de información y su procesamiento a través de Internet. Desafortunadamente en la mayoría de los casos la enseñanza solo se centra en el uso de las herramientas tecnológicas pero no en descubrir cómo funcionan o cómo crearlas. En los últimos años se ha comenzado a hacer un cambio en la forma de ver las TIC, señalando al estudiante la importancia de ser un productor de tecnología y no simplemente un consumidor de la misma [12][13]. Además conviene tener en cuenta que la educación no puede ser ajena al avance tecnológico, porque las nuevas tecnologías traen oportunidades para beneficiar y mejorar los métodos de aprendizaje. La incorporación de las TIC no es solo un desafío sino una necesidad en la educación, estas tecnologías han roto las barreras de la distancia, creando nuevos ambientes de aprendizaje continuos e incesantes [14].

La aparición de nuevos roles en los docentes y los estudiantes, y la creación de nuevos materiales de enseñanza-aprendizaje son los cambios que más han influido las TIC en el ámbito educativo [15]. Estos materiales pueden ser físicos o virtuales; lo importante es que despierten interés en los estudiantes y que tengan la información adecuada con situaciones cercanas a la realidad. Así se logrará la motivación y comprensión de los contenidos que se quieren transmitir [16].

Estos materiales reciben diferentes nombres como recursos educativos digitales, objetos virtuales de aprendizaje, recursos educativos digitales abiertos, materiales educativos digitales (MED), y materiales educativos computacionales [17].

Pianucci, Chiarani y Tapia [18] definen los materiales educativos digitales como "recursos facilitadores del proceso de enseñanza-aprendizaje en soporte digital, siguiendo criterios pedagógicos y tecnológicos, que integran diversos medios incorporados en un diseño de instrucción". Todo material educativo digital debe fundamentarse en el proceso de enseñanza y aprendizaje, más que en su disponibilidad o entornos gráficos llamativos, aunque son elementos importantes se debe tener claro el objetivo pedagógico, identificando las necesidades de la población, para así realizar las estrategias que permitan dar soluciones que brinden los mejores resultados [19].

Dependiendo de las necesidades educativas, Galvis [19] propone 5 tipos de materiales para fortalecer el aprendizaje. En este estudio se considera el tipo de material, denominado juego educativo que tiene como fin reforzar temas, conceptos y desarrollar destrezas.

II. Metodología

A. *Diseño*

La presente investigación está alineada con los principios del paradigma constructivista [20]. Este paradigma da un rol activo al sujeto de estudio; en nuestro caso el sujeto de estudio son los estudiantes del curso de la materia Lógica de Programación [21].

El enfoque de la investigación es cualitativo; no pretende basarse en estadísticas para presentar sus resultados sino que a partir de las observaciones se interpretan los datos obtenidos [22].

El diseño de la investigación se enmarca bajo la metodología de estudio de caso basado en métodos flexibles y en la utilización de múltiples herramientas para capturar y analizar los datos, para así poder comprender las peculiaridades de los estudiantes [22]. Siguiendo el planteamiento de Stake [23] se seleccionó un grupo de estudiantes con características similares a conveniencia, en este caso estudiantes de la materia Lógica de Programación.

B. *Población*

El estudio se ha realizado en INCAP que es un instituto de educación para el trabajo y desarrollo humano (este tipo de educación es conocida en otros países como educación profesional). La edad mínima de ingreso para los programas de sistemas es de 14 años. Esta institución tiene más de 15000 estudiantes distribuidos en varias sedes (la sede Chapinero objeto del estudio cuenta con 9850 estudiantes). La carrera Técnico en Sistemas cuenta con 100 estudiantes, en su primer semestre los estudiantes tienen que matricular la materia de Lógica de Programación la cual es la base fundamental en el proceso de pensamiento lógico y crítico de los estudiantes para resolver problemas con algoritmos computacionales, y desarrollar a futuro aplicaciones de software (apps).

C. *Muestra*

Se seleccionó para este pilotaje uno de los 10 grupos de la materia de Lógica de Programación, materia identificada en la institución educativa por el código PA (programación uno). El grupo seleccionado fue PA02 conformado por ocho estudiantes, dos mujeres y seis hombres, de edades comprendidas entre 17 y 24 años. La selección de la muestra se realizó por conveniencia como lo plantea Hernández, Fernández y Baptista [22], según el enfoque cualitativo que fundamenta la investigación.

D. *Técnicas de Recolección de Datos*

Las técnicas que permiten la recolección de la información son la observación y la entrevista. La técnica de entrevista contó con el diseño de cuestionarios de tipo estructurado (para las pruebas de entrada-salida, caracterización, interacción con el MED), semiestructurado, y abierto (grupo focal). Además de los cuestionarios se hizo registro de las observaciones mediante un diario de campo. A través de software de captura de pantalla en video se grabaron las acciones de los estudiantes con el MED, y se realizó el registro en audio de las entrevistas en el grupo focal. Los instrumentos y las herramientas diseñadas para la recolección de información fueron validados en la prueba piloto del MED. El procesamiento de la información obtenida en grupo focal, se ha realizado a través del programa Atlas.ti v. 7.5.4.

E. *Planificación*

Se realizó un estudio que fue desarrollado en cuatro fases:

1) *Identificar y definir el problema objeto de estudio:*

En esta fase se determinó el tema a tratar realizando entrevistas con los profesores del área de sistemas mediante un cuestionario. Se han identificado los problemas de conocimientos que presentan los estudiantes de la materia Lógica de Programación. De esta manera se han determinado los posibles retos que debería tener el MED.

2) *Diseñar y desarrollar el material educativo:*

En esta fase se diseñó el MED "Evolución" a partir de un storyboard (guion gráfico) evaluado por un experto en materiales digitales. Después de las correcciones del experto se procedió a realizar los ajustes necesarios para empezar la fase de desarrollo.

El diseño y desarrollo del MED "Evolución" fue un trabajo conjunto entre los investigadores y el Centro de Tecnologías de la Academia (CTA) de la universidad de la Sabana en Colombia [24]. El diseño del MED se basó en las estrategias que se deben llevar a cabo para el desarrollo de materiales digitales planteadas por varios autores [19], [25], [26]. La programación del material se realizó en HTML 5 para permitir una navegación adecuada en los diferentes navegadores de los sistemas operativos más utilizados. La figura 2 muestra la pantalla de inicio y entrada al MED.

Este material en su primera versión tiene siete retos distribuidos en diferentes tipos de pensamiento (matemático, algorítmico, y crítico) fortaleciendo las habilidades asociadas al pensamiento computacional. La figura 3 especifica el tipo de pensamiento ejercitado según reto y la habilidad requerida para superarlo.

El MED "Evolución" está diseñado para permitir al participante realizar los retos dependiendo de las habilidades que tenga el participante. La tabla I muestra una síntesis de los aspectos de cada reto.

3) *Realizar el pilotaje del MED:*

Después del desarrollo del MED "Evolución" se procedió a realizar una prueba piloto para comprobar la usabilidad y accesibilidad del material. Esta prueba se realizó con un

Fig. 2. Diseño de interfaz bienvenida del MED "Evolución".

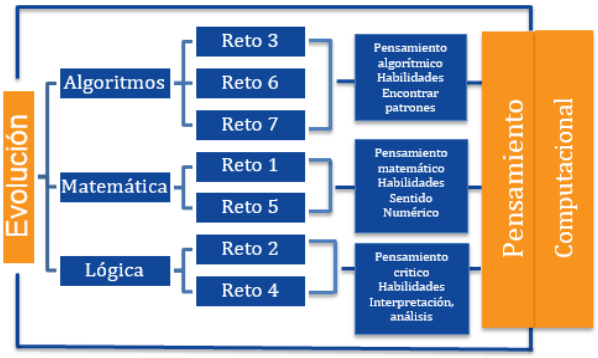


Fig. 3. Esquema general del MED "Evolución".

grupo de estudiantes de INCAP del curso PA04, elegido a conveniencia [22]. De acuerdo con las observaciones del investigador recopiladas en un diario de campo y al grupo focal realizado al final del pilotaje del material, se realizaron cambios en los tiempos de algunos retos y se corrigieron algunos problemas en la programación del MED.

4) *Recolectar los datos y analizarlos:* Cuando el material

fue corregido en su parte gráfica y de programación, se procedió a realizar la implementación para determinar el aporte del MED "Evolución" en las habilidades relacionadas con el pensamiento computacional en los estudiantes de la materia Lógica de Programación. En este pilotaje se recolectaron datos con diferentes herramientas para analizarlos y realizar una interpretación descriptiva de los datos obtenidos.

III. Resultados

Se presenta a continuación el reporte de los datos registrados en los instrumentos de recolección de información según las fases de implementación del MED "Evolución".

A. *Recolección de Datos.*

Se realizó una indagación para reconocer las habilidades del pensamiento computacional con las cuales llegaban los estudiantes antes de la implementación de MED. El instrumento utilizado fue una prueba de entrada con 10

preguntas abiertas que correspondían a habilidades matemáticas, lógicas y algorítmicas, según el esquema del MED (ver figura 3). Esta prueba se realizó en 45 minutos; además se realizaron anotaciones de las observaciones en un diario de campo.

Después de la prueba de entrada se indicó a los estudiantes que podían entrar al link del servidor web donde se aloja el MED "Evolución". La interacción por parte del estudiante con el MED tuvo una duración total de 67 minutos, en esta fase también se registraron observaciones en un diario de campo. Todas las actividades de interacción del estudiante con el material fueron grabadas con un software incluyendo la captura de la pantalla del computador. En total se reunieron aproximadamente 10 horas de video entre todos los participantes de esta implementación.

Terminada la interacción con el MED se entregó la prueba de salida a los estudiantes que contenía 10 preguntas abiertas. Estas preguntas tienen relación con las preguntas de la prueba de entrada, asociadas con los retos del MED "Evolución", y se realizó en 40 minutos.

En este pilotaje el estudiante interactuó una sola vez con el MED para medir si mejora sus habilidades en una primera interacción; además se evitaron posibles contaminantes como realizar operaciones aritméticas con dispositivos electrónicos o pedir ayuda de otro estudiante.

En la última parte de la implementación, se entregó a los estudiantes el cuestionario en formato físico con preguntas abiertas referentes al MED. Estas preguntas se enfocaron en su uso, beneficios, aspectos de agrado y de mejora. Este cuestionario se realizó en 20 minutos. Finalmente se realizó un grupo focal para socializar la actividad con los estudiantes durante 30 minutos. La tabla II muestra un resumen de los momentos de la implementación de la prueba piloto. Análisis de datos

El estudio tiene un enfoque cualitativo basado en varios instrumentos flexibles para recolectar información en diferentes formatos (texto, audio, video) [27]. Esta información fue clasificada y analizada según las siguientes categorías de análisis: a) uso de herramientas en el aprendizaje b) experiencias de aprendizaje, y c) habilidades asociadas al pensamiento computacional. Los grupos focales fueron transcritos, y esta información se clasificó, codificó junto con los cuestionarios en el software Atlas.ti v. 7.5.4.

Con el fin de poder identificar algunas habilidades asociadas a cada tipo de pensamiento, y cómo el MED podría aportar en mejorar en cada una de las habilidades evaluadas, se crearon unos indicadores de desempeño asociados a los retos del MED. La tabla III muestra los indicadores de desempeño utilizados.

TABLA I
DESCRIPCIÓN DE LOS RETOS DEL MED "EVOLUCIÓN"

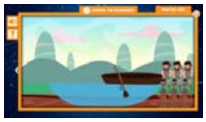
Reto	Nombre	Tema	Objetivo del reto
1	**Nivel célula**	Correlación, operaciones matemáticas básicas	Aumentar la atención y concentración del estudiante, entrenando la memoria a corto plazo
2	**Nivel pluricelular**	Operaciones matemáticas básicas, interpretación de datos, sentido numérico, lógica.	Mejorar la capacidad de interpretación, lógica y de control de datos para resolver un problema, mejorando así sus habilidades matemáticas.
3	**Nivel anfibio**	Algoritmos, patrones, abstracción	Desarrollar el pensamiento algorítmico encontrando patrones para dar solución a un problema.
4	**Nivel marsupial**	Operaciones matemáticas básicas, interpretación de datos, sentido numérico	Mejorar la capacidad de interpretación y de control de datos para resolver un problema, mejorando así sus habilidades matemáticas, sentido numérico, interpretación de lectura.
5	**Nivel orangután**	Operaciones matemáticas básicas	Aumentar capacidad para el manejo de operaciones básicas y la concentración del estudiante, entrenando la memoria corto plazo, matemáticos adquiridos con anterioridad.
6	**Nivel hombre de las cavernas**	Algoritmos, patrones, abstracción.	Desarrollar el pensamiento algorítmico encontrando patrones para dar solución a un problema.
7	**Nivel hombre Recolector** 	Algoritmos, patrones, abstracción	Desarrollar el pensamiento algorítmico encontrando patrones para dar solución a un problema.

TABLA II
MOMENTOS DE LA FASES DEL PILOTAJE

Herramienta o Actividad	Tiempo
Prueba de entrada	45 minutos
Interacción con el MED "Evolución"	67 minutos
Prueba de salida	40 minutos
Cuestionario interacción con el MED "Evolución"	20 minutos
Grupo focal cierre de la actividad	30 minutos
Total tiempo de la actividad	3 horas y 22 minutos

TABLA III
INDICADORES DE DESEMPEÑO SEGÚN HABILIDAD DE PENSAMIENTO OBSERVADA

Componentes del pensamiento computacional	Habilidades	Indicador
1.Pensamiento algorítmico	Encontrar patrones	1.1 Identifica los pasos para resolver los algoritmos 1.2 Realiza abstracciones de una secuencia de caracteres
2.Pensamiento matemático	Sentido numérico	2.1 Interpreta los datos que componen el problema 2.2 Realiza las operaciones matemáticas básicas asociadas al problema
3.Pensamiento crítico	Interpretación Análisis	3.1 Identifica el problema planteado 3.2 Propone una solución acertada al problema 3.2 Explica cómo llegó a resolver un problema

Estos indicadores se verificaron en cada uno de los estudiantes, utilizando una rúbrica según se ejemplifica en figura 4.

Para analizar la interacción de los estudiantes con el MED "Evolución "se creó una rúbrica para describir los principales aspectos a tener en cuenta en el estudiante al utilizar el MED como se muestra en la figura 5.

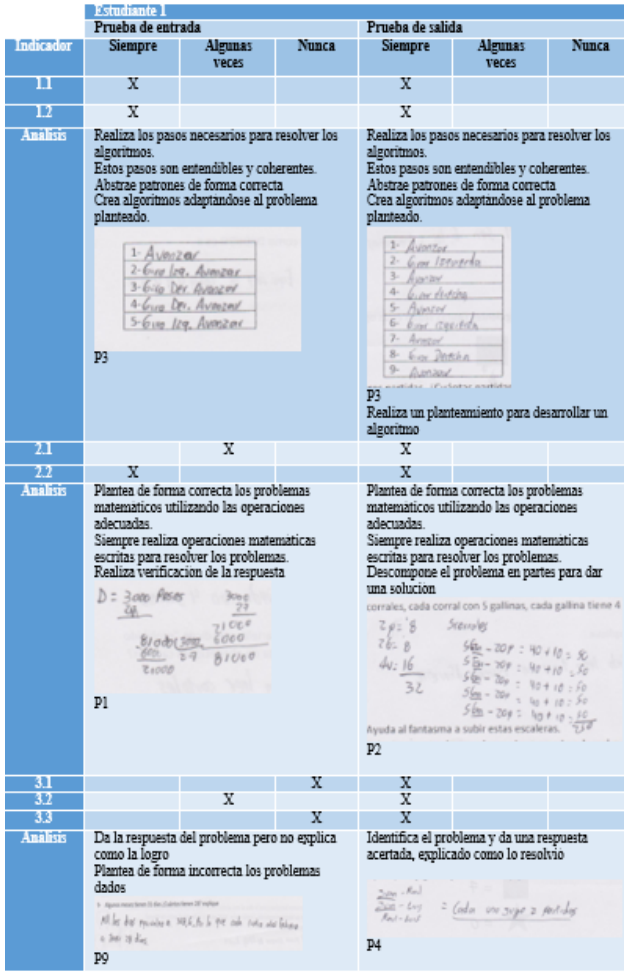


Figura 4. Rúbrica de análisis de los cuestionarios de entrada y salida. Los resultados reportados en esta rúbrica corresponden al estudiante 1 y es para referencia del formato que se siguió en el análisis de todos los casos.

B. *Presentación de Resultados*

Los resultados obtenidos en este estudio del MED "Evolución" se clasifican en tres categorías de acuerdo con los análisis realizados.

1) *Desempeño durante el uso del MED "Evolución":*

El desempeño medido se relaciona con el número de intentos de los estudiantes en superar un reto teniendo en cuenta las habilidades asociadas al PC (ver figura 3).

Los retos 1 y 5 fueron especialmente diseñados para que los estudiantes practicaran las operaciones matemáticas básicas. El número de intentos según los retos 1 y 5 se muestran en la figura 6.

Los retos 3, 6 y 7 fueron diseñados para involucrar habilidades de pensamiento algorítmico. La figura 7

Estudiante 1	Intentos	cambio reto	Operaciones fáciles	Operaciones dificiles	Puntaje	Lee la Ayuda	Dificultad
Reto 1	1	No	multiplicación	Sumas restas	69	Si	fácil
Observaciones **El estudiante encontró la forma de mantener los datos ocultos visibles, no involucra la concentración solo la solución de problemas matemáticos, resolvió primero las multiplicaciones y después las sumas y restas**							
Reto 2	2	si--3			105	Si	dificil
Observaciones **No puede interpretar el problema, Este reto lo deja de ultimo y cuando retoma resuelve sin problema la pregunta**							
Reto 3	17	si--4			64	Si	dificil
Observaciones **Tiende a realizar acciones aun sabiendo que el movimiento no representa un avance en el algoritmo, como abandono el reto y siguió con otros cundo retomo y después de realizar el reto 7 se tomó su tiempo para analizar los pasos sin embargo intenta realizar movimiento aun sabiendo que estos no resolverán el problema, cuando resolvió el reto lo logro en el segundo cero así que el programa lo marco como no resuelto cuando retoma el juego analiza y aunque lo intenta 2 veces más resuelve el patrón.**							
Reto 4	2	No			94	Si	media
Observaciones **Falto análisis en el primer intento**							
Reto 5	1	No	suma resta	división	40	Si	fácil
Observaciones **Un error con una multiplicación y se demora en la división mental**							
Reto 6	1				100	Si	fácil
Observaciones **Entendió el algoritmo y los pasos para solucionarlo**							
Reto 7	13	No			67	si	dificil
Observaciones **Leer de nuevo instrucciones, en 7 ocasiones estuvo a un movimiento de terminar el reto y aun así realizó un movimiento que no se podía, aun conociendo las instrucciones avanzaba y no era la solución correcta**							
Puntaje Total				539		tiempo total	30 minutos

Fig. 5. Rúbrica de análisis de los videos de captura de pantalla. Este es un ejemplo del análisis realizado para el video del estudiante 1 mientras afrontaba cada reto del MED.

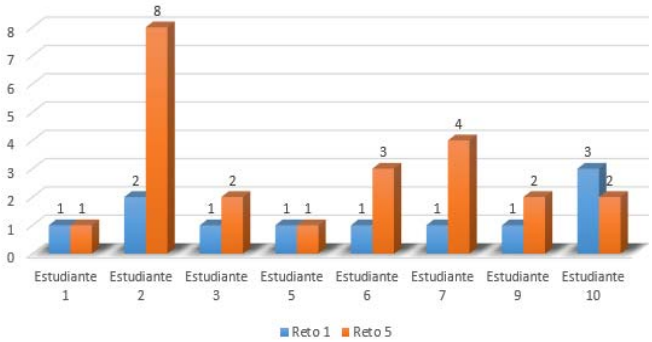


Fig. 6. Número de intentos por estudiante en los retos de habilidades matemáticas.

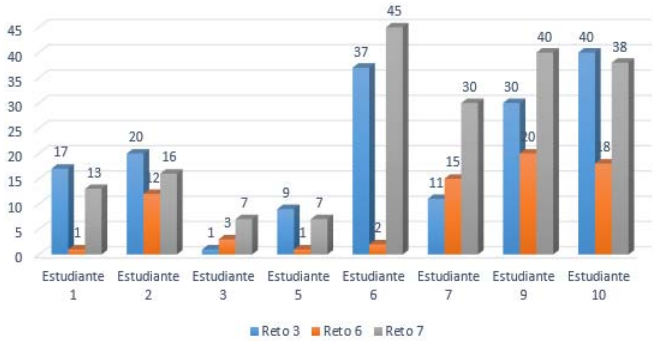


Fig. 7. Número de intentos por estudiante en los retos de habilidades algorítmicas.

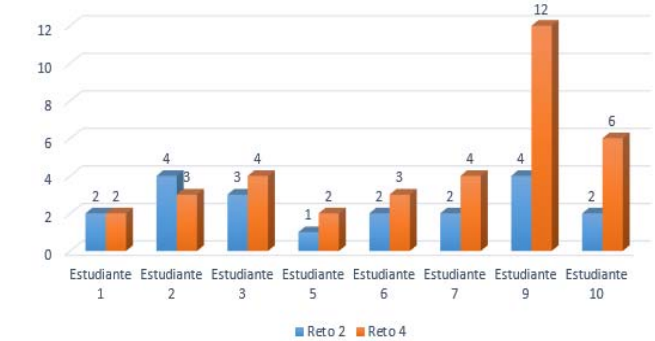


Fig. 8. Número de intentos por estudiante en los retos de habilidades análisis e interpretación.

muestra el número de intentos para estos retos. Se observa en dicha figura que el reto que presentó menos dificultad fue el reto 6.

Por su parte los retos 2 y 4 fueron diseñados para enviar preguntas aleatorias de manera que los estudiantes, además de utilizar operaciones matemáticas como sumas y restas, incorporaran análisis e interpretación de los datos planteados en las preguntas. La figura 8 muestra el desempeño según número de intentos para estos retos.

2) Aporte del MED "Evolución" según las pruebas:

Los casos de mejor desempeño inicial (prueba de entrada) mostraron mejor respuesta tras la implementación del MED, especialmente en aquellas habilidades que inicialmente no alcanzaron un puntaje de mayor eficiencia al compararlos con la prueba de salida.

Las habilidades de mayor dificultad para los casos con bajo desempeño son las asociadas al pensamiento matemático. Después de la intervención con el MED parece potenciarse de manera leve al menos una forma de pensamiento para todos los casos.

En la mitad de los casos se han potenciado habilidades de pensamiento matemático y algorítmico. En dos de los casos con mayor dificultad, se evidencia mejora en habilidades de pensamiento crítico. En uno de estos dos casos se encontró un mejor desempeño según habilidades de pensamiento algorítmico.

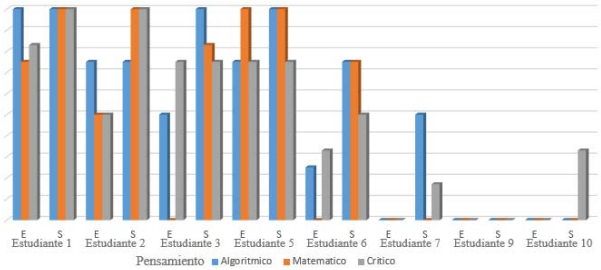


Fig. 9. Comparativa de las habilidades según los tipos de pensamientos en la prueba de entrada (E) y prueba de salida (S).

Uno de los ocho casos analizados de más bajo desempeño no mostró cambio alguno. La figura 9, ilustra el desempeño de los estudiantes antes y después de la intervención con el MED, según el tipo de pensamiento observado.

3) Percepción del estudiante sobre uso del MED:

La mayoría de los estudiantes expresan que MED Evolución es una herramienta que ayuda al desarrollo de habilidades (lógica, interpretación, matemática, resolución de problemas). Además los estudiantes indican que tiene un propósito de aprendizaje que va más allá de superar los retos presentados como se observa en los siguientes testimonios de estudiantes:

"es un juego educativo que nos ayuda a pensar y a ejercitar la lógica de los algoritmos" (Estudiante 2).

"el juego de evolución me parece una muy buena herramienta por que explora todos los niveles, desde el más básico -que uno considera bueno, pues, "bastante elementales-", pero uno a veces a conciencia, con el tiempo olvida varias cosas; entonces digamos que si sirve como para fortalecer las capacidades previas, la capacidad de lógica y organizar los procesos. Me parece que sí es una buena herramienta pedagógica." (Estudiante 5).

Respecto a los aspectos de diseño del MED, (navegabilidad del material, orientación y localización de elementos), los estudiantes consideran que es un recurso con una interfaz agradable a la vista y de fácil navegación. Se considera la ventaja de contar con una ayuda (el manual incorporado en la página web) que facilita el registro y el acceso.

Los retos algorítmicos fueron los de mayor dificultad en opinión de los estudiantes, y entre ellos se destaca el reto de los cavernícolas (este es el último de los retos del MED). Se considera que el MED propone una mayor dificultad a medida que se avanza en los retos:

"naturalmente cada nivel tenía una mayor dificultad que el anterior, pero al usar la lógica y el análisis los pude solucionar sin inconvenientes" (Estudiante 5).

Los retos matemáticos también presentan cierta dificultad, debido, según los estudiantes, a sus escasos conocimientos en matemáticas, la dificultad para desarrollar las operaciones matemáticas (dificultad para concentrarse), y al tiempo límite dado para el logro del reto.

También consideran los estudiantes que el MED aporta en el fortalecimiento de las habilidades matemáticas, particularmente, en la capacidad de memorización, agilidad mental, y en realización de operaciones básicas.

Dentro de los aspectos de mejora del MED Evolución, se menciona aumentar la dificultad de los juegos mediante la disminución del tiempo de los retos y el nivel de complejidad. También se considera importante la incorporación de nuevos retos. Estas propuestas de mejoras se observan en los siguientes testimonios:

"Digamos como juegos de habilidad pero algo más rápido tiempos de respuesta más rápida, algo que lo haga decidir a uno por así decirlo bajo presión" (Estudiante 2)

"Pues en los juegos yo diría que el del mico (orangután) podríamos hacer operaciones matemáticas más grandes y así con el tiempo más restringido" (Estudiante 6)
"quizá el sudoku sería una buena ayuda pero que lo hicieran más interactivo" (Estudiante 3, sobre la incorporación de un nuevo reto).
Respecto a la pertinencia de este tipo de recursos en la enseñanza de la materia de Lógica de Programación, se encuentra consenso en las ventajas que éstos puedan ofrecer en el ámbito educativo y a la falta de implementación de los mismos en el aula como se expresan en los siguientes testimonios:
"Desde mi punto sería una buena herramienta porque lo hace salir de lo fácil de lo que uno está acostumbrado, que la tecnología lo hace todo por uno, y en el juego puede evidenciar que lo hace pensar a uno varias veces para hacer las cosas…" (Estudiante 1).
"Me parece una buena herramienta pedagógica. Digamos si a uno le ayuda a desarrollar esas habilidades de lógica, de razonamiento, yo creo que sí, definitivamente esta es una de esas herramientas que debería ser usada en mayor escala en todas las áreas del conocimiento. Hoy en día que todavía los centros universitarios todavía manejan a la antigua prácticamente, y no s e han dado cuenta que con la globalización, las últimas tecnologías prácticamente son un requisito, también el acceso a estos medios informativos que obviamente lo ponen a uno a la par con personas de otras latitudes" (Estudiante 5).

IV. Conclusiones

En este estudio se trató de abordar una estrategia para incorporar habilidades del Pensamiento Computacional en estudiantes de la materia Lógica de Programación. Esta estrategia se basó en el diseño y posterior pilotaje de un material educativo digital con el nombre "Evolución", que permitiera mediante retos fortalecer habilidades matemáticas, algorítmicas y de análisis.

Las herramientas tecnológicas desarrolladas con una intención pedagógica involucran al estudiante en su aprendizaje y le motiva para que cambie su rol pasivo en las clases por uno de participación en el que puede crear autonomía en su aprendizaje.

En este estudio no se pretende argumentar que con el MED "Evolución" se aprende o se enseña Pensamiento Computacional, sino reforzar conocimientos que traen los estudiantes de su vida escolar y tratar de solucionar algunos déficits en la parte matemática y de análisis relacionadas con habilidades del PC.

En este estudio cualitativo se evidenciaron comportamientos de los estudiantes al interactuar con una herramienta tecnológica, y se indagó por su opinión para describir el grado de participación y compromiso de los estudiantes.

Los estudiantes ven las TIC como herramientas necesarias en los procesos de aprendizaje de cualquier materia en los diferentes niveles educativos. Esta visión de los estudiantes debe crear un compromiso por parte de los profesores para investigar, ensayar y aplicar nuevos métodos de enseñanza en las diferentes asignaturas o materias que dicten actualmente.

La comprensión lectora y las habilidades matemáticas deben ser mejoradas, y para ello se hace necesario crear nuevos materiales y actividades que permitan al estudiante mejorarlas.

Para las materias iniciales de las carreras de sistemas se deben tener en cuenta diferentes materiales interactivos que permita al estudiante ensayar, cometer errores en un entorno controlado, y que sirvan de motivación para avanzar en su proceso de aprendizaje.

Con esta primera intervención del MED "Evolución" se ha evaluado el material tanto a nivel de diseño y desarrollo, y se ha analizado el aporte pedagógico en los temas relacionados con habilidades del PC propias de la materia de Lógica de Programación. Se han corregido los errores de acuerdo con todos los datos analizados. El siguiente paso en esta investigación será desarrollar un estudio general del MED "Evolución" en todos los estudiantes de primer semestre de INCAP que cursan la materia.

Es pertinente mencionar algunas limitantes presentadas en este estudio. Algunos son factores asociados con el MED, y otros con aspectos de la implementación. El tiempo es un factor determinante en esta investigación. Hay que tener en cuenta que el material se implementó una sola vez por cada fase (prueba piloto del material, prueba piloto del estudio) y que se realizó un análisis de la interactividad de cada estudiante en cada una de las fases (observando su comportamiento y resultados); este análisis se realizó en un solo momento para no viciar los datos, ya que el estudiante podría resolver los algoritmos o las preguntas con ayuda de otro compañero o buscando en Internet. En caso de haber dispuesto de más tiempo para el desarrollo de otras implementaciones, se podría haber tenido una mayor certeza de la consolidación de los conocimientos: a) verificando si los ensayos-error fueron significativos, pudiendo encontrar nuevamente el patrón para resolver los algoritmos, b) realizando las operaciones matemáticas mucho más rápido, o más importante, c) no equivocándose al realizarlas.

Se considera que el tiempo de desarrollo de la implementación (más de 3 horas) fue un factor que no permitió la participación de más estudiantes, (algunos estudiantes no disponían de todo este tiempo para realizarla).

Agradecimientos

Los autores agradecen al Centro de Tecnologías para la Academia de la universidad de la Sabana (Chía, Colombia) por ser parte de la elaboración del MED utilizado en este estudio. También a las directivas del Instituto Colombiano de Aprendizaje (Bogotá, Colombia) donde se realizó esta investigación.

Referencias

[1] L. Aguilar, L. Baena, and M. Fernández, *Fundamentos de Programación Libro de Problemas*, 2nd ed. Madrid: McGraw-Hill, 2000.
[2] G. Brassard and P. Bratley, *Fundamentos de algoritmia*. Madrid: Prentice Hall, 1997.
[3] A. Linares, "Por que somos tan malos en Matematicas," *28 Septiembre ,2013*, 2013. [Online]. Available: http://www.eltiempo.com/archivo/documento/CMS-13088961.
[4] Csta.Iste, "Computational Thinking in K–12 Education leadership toolkit," p. 43, 2011.

[5] J. H. Mcmillan and S. Schumacher, *Investigación educativa*, 5th ed. Madrid, Pearson, 2005.
[6] J. M. Wing, "Computational Thinking," *Commun. ACM*, vol. 49, no. 3, pp. 33–35, 2006.
[7] X. Basogain, M. Á. Olabe, and J. C. Olabe, "Pensamiento Computacional a través de la Programación: Paradigma de Aprendizaje," *Rev. Educ. a Distancia*, vol. 46, no. 46, 2015.
[8] P. Blikstein, "Thinking about learning, and learning about thinking," *Diciembre,2013*, 2013. [Online]. Available: https://tltl.stanford.edu/content/seymour-papert-s-legacy-thinking-about-learning-and-learning-about-thinking.
[9] J. M. Wing, "Computational Thinking: What and Why?," *thelink - Magaizne Varnegie Mellon Univ. Sch. Comput. Sci.*, pp. 1–6, 2010.
[10] A. V. Aho, "Computation and computational thinking," *Comput. J.*, vol. 55, no. 7, pp. 833–835, 2012.
[11] S. E. Breen and A. O 'shea, "Mathematical Thinking and Task Design," *Irish Math. Soc. Bull.*, vol. 66, pp. 39–49, 2010.
[12] L. López, "Metodología para el Desarrollo de la Lógica de la Programación Orientada a Objetos," *Rev. Iberoam. Sist. Ciibernética e Informática*, pp. 27–32, 2013.
[13] F. J. García-Peñalvo, "A brief introduction to TACCLE 3 – Coding European Project," pp. 3–6, 2016.
[14] A. Moya, "'Las nuevas tecnologías en la educación,'" *Innovación y Exp. Educ.*, vol. 24, pp. 1–9, 2009.
[15] S. Almeida, J. P. Febles, V. Estrada, and O. Bolaños, "Las Tecnologias de la informacion y las comunicaciones en la universalizacion de la enseñanza medica," pp. 1–18, 2006.
[16] M. Bautista, A. Martínez, and R. Hiracheta, "El uso de material didáctico y las tecnologías de información y comunicación (TIC ' s) para mejorar el alcance académico," pp. 183–194, 2014.
[17] D. Wiley, "Connecting learning objects to instructional design theory: adefinition, a metaphor, and a taxonomy," *Igarss*, vol. 2830, no. 1, pp. 1–35, 2014.
[18] G. Pianucci, M. Chiarani, and M. Tapia, "Elaboracion de Materiales educativos digitales," *1er Congr. Int. Punta del Este*, pp. 1–6, 2010.
[19] A. Galvis, *Ingeniería de Software Educativo*. Bogotá: Universidad de los Andes, 1992.
[20] E. Guba and Y. Lincoln, "Paradigmas en competencia en la investigación educativa," *Por los rincones. Antología de métodos cualitativos en la investigación social.* pp. 113–145, 2002.
[21] C. Coll, "Recopilación De Textos Sobre Las Aplicaciones Pedagógicas De Las Teorías De Piaget," *Psicol. Genética Y Aprendizajes Esc.*, pp. 1–20, 1986.
[22] R. Hernández, C. Fernández, and P. Baptista, *Metodología de la investigación*. Mexico: McGraw-Hill, 2006.
[23] R. E. Stake, *Investigación con estudios de caso*. 1999.
[24] M. J. Rico, "MED Evolución: Un camino al Pensamiento Computacional," *research-technique*, p. 270, 2017.
[25] J. Torrente *et al.*, "Fomentando la Creatividad: Creación de Escenarios de Aprendizaje Basados en Juegos," *Una Guia para Profesores*, pp. 1–46, 2011.
[26] Red.es, "Evaluacion de material educativo digital," *España*, 2005. [Online]. Available: http://www.edubcn.cat/rcs_gene/2_ficha_evaluacion_material.
[27] J. L. Àlvarez and G. Jurgenson, *Cómo hacer investigación cualitativa. Fundamentos y metodología.* Mexico, 2003.

Mauricio Javier Rico Lugo es Ingeniero de sistemas de la Universidad INCCA de Colombia (2005), especialista en gerencia informática de la universidad Remington (2011), Magister en Informática Educativa de la Universidad de la Sabana (2017), ha realizado publicaciones para el sector privado en lógica, fundamentos de programación y administración de redes, profesor desde hace más de 14 años, formador del SENA desde 2014 en programas de ampliación de cobertura en programación de Software y Redes informáticas, en la actualidad es formador de planta en el Instituto Colombiano de Aprendizaje INCAP. Actualmente participa en un convenio investigativo entre la Universidad del País Vasco (España) y la Red Nacional Académica de Tecnología Avanzada de (Colombia) que pretende introducir el pensamiento computacional en los colegios y escuelas de Colombia.

Xabier Basogain Olabe es profesor de la Universidad del País Vasco - Euskal Herriko Unibertsitatea. Ingeniero de Telecomunicaciones por la Universidad Politécnica de Madrid (1987), Doctor Ingeniero de Telecomunicaciones por la Universidad del País Vasco (1991), y miembro del Departamento de Sistemas de Ingeniería y Automática de la Escuela de Ingeniería de Bilbao. Ha impartido cursos sobre sistemas digitales, microprocesadores, control digital, modelado y simulación de eventos discretos, aprendizaje automático y herramientas de colaboración en educación. Sus actividades de investigación incluyen las áreas de: a) soft computing y ciencias cognitivas a STEM; b) tecnologías de aprendizaje y de enseñanza aplicadas a la educación en línea y educación inclusiva; c) realidad aumentada y virtual con tecnologías móviles. Su grupo de investigación (http://ehu.es/gmm) está integrado en los grupos de investigación consolidados del sistema universitario vasco (2010), y tiene un extenso número de publicaciones en foros científicos, académicos y sociales. Es miembro de comités de revistas y comités científicos de congresos, colabora en proyectos europeos, y participa en el desarrollo de proyectos sociales y educativos con instituciones de países de Latinoamérica.

Nancy Moreno Niño es Psicóloga de la Universidad de los Andes (2002), Especialista en Edumática de la Universidad Autónoma de Colombia (2010), Magister en Informática Educativa de la Universidad de La Sabana (2016) y con un entrenamiento para el uso pedagógico de las TIC con la Oficina de Intercambios y Cooperación para la formación de docentes y la Oficina de Educación de Incheon en Corea del Sur (2017). Actualmente se desempeña como profesional docente de apoyo a la inclusión, con población con discapacidad en institución educativa de la Secretaría de Educación de Bogotá, donde ha encaminado sus esfuerzos en investigar sobre el fortalecimiento del pensamiento matemático en población con algún tipo de discapacidad intelectual.

Propuesta de Metodología Basada en Metáforas para la Enseñanza de la Programación a Niños

Diana Pérez-Marín, Raquel Hijón-Neira, Mercedes Martín-Lope

CÓMO REFERENCIAR ESTE ARTÍCULO:

D. Pérez-Marín, R. Hijón-Neira, M. Martín-Lope. ''A Methodology Proposal based on Metaphors to teach Programming to children'', 2018
DOI: https://doi.org/10.1109/RITA.2018.2809944

***Title*— A Methodology Proposal based on Metaphors to teach Programming to children**

***Abstract*— Interest in studying Computer Science has been extended worldwide to children. However, it is unclear which educational method should be used. Teachers need some guides to approaching this task. Therefore, this paper proposes using metaphors such as recipe/program (and sequence), pantry/ memory, and boxes /variables. It also illustrates the possibility of applying these metaphors to any resource available to the teacher. Four step-by-step scripts of how to use the metaphors in class are provided, with the opinions of 62 children (enrolled in 4th, 5th and 6th Primary courses, 9 to 11 years in age) and their teacher's opinion.**

***Index Terms*—Computer Science Education, metaphor, Computational thinking.**

I. Introducción

SABER programar es una de las competencias clave que sería deseable que pudiesen adquirir todos los estudiantes en el siglo XXI [1-3]. Sin embargo, cuando se revisa la literatura de las experiencias realizadas, para investigar si efectivamente los estudiantes adquieren esta competencia, muchos estudiantes universitarios no lo han conseguido. Por el contrario, les cuesta descomponer problemas, desarrollar algoritmos, e implementar estos algoritmos usando algún lenguaje de programación [4].

En los últimos años, el interés en estudiar Programación se ha extendido a niveles inferiores de enseñanza. Esto puede deberse a que algunos estudios sugieren que, si se enseñase programación en edades más tempranas, se podría desarrollar el pensamiento computacional necesario para poder posteriormente crear programas, resolviendo el problema detectado anteriormente, e incluso poder resolver en general problemas que de otra forma no podrían [5].

Muchos estudios se centran en la enseñanza de la programación para niños mediante el uso de Scratch [5-8]. Sin embargo, no hay todavía artículos que proporcionen una metodología de enseñanza de la Programación que permita desarrollar el pensamiento computacional deseado en estos niveles, ni que apoye en este objetivo a los docentes sin conocimientos previos en Informática.

Por el contrario, se encuentran dificultades enseñando conceptos básicos como programa [9], bucles [10], estructuras de control y algoritmos [11]. Las dificultades en muchos casos se pueden deber a la falta de una metodología de enseñanza de estos conceptos en Educación Primaria [9-10]. Se hace patente la necesidad de que los profesores sin conocimientos previos de Informática dispongan de alguna guía para llevar a cabo esta nueva tarea de enseñanza de la programación y el fomento del pensamiento computacional en edades tempranas [5,8,14].

En este artículo, se propone por primera vez el uso de metáforas para introducir a los niños en los conceptos básicos de programación. En particular, se proponen 4 grupos de metáforas: M1, que engloba las metáforas que relacionan programar con cocinar, programa-secuencia con receta, memoria con despensa y variables con cajas; M2, que engloba las metáforas de entrada-salida que relacionan la entrada con el teclado y la salida con la pantalla; M3, que engloba las metáforas de condicionales que relacionan la posibilidad de los niños de tomar decisiones como los ordenadores también pueden tomar decisiones; y M4, que engloba las metáforas de bucles que relacionan la posibilidad de poner la mesa para X personas repitiendo las acciones como los ordenadores pueden repetir las instrucciones varias veces. Se proporcionan también cuatro guías detalladas de cómo poner en la práctica estas metáforas en clase, independientemente de los recursos de los que disponga el profesor.

La razón por la que se propone el uso de metáforas es su utilidad probada como herramienta educativa cuando el dominio de enseñanza es abstracto. Las metáforas se centran en los conceptos, y proporcionan a los estudiantes una mejor organización de las ideas, y una forma de pensar más directa.

Además, usar metáforas no necesita un software ni un hardware especial, tan solo el lenguaje que permita a los profesores convertir conceptos difíciles y abstractos en ideas más simples y fáciles de adquirir. Esto es, justo lo que se quiere conseguir con el pensamiento computacional, una forma de pensar clara y directa, y que permita a los estudiantes desarrollar con éxito la competencia de programar y resolver problemas [15].

Sin embargo, las metáforas también deben tratarse con cuidado. En usos previos de enseñanza basada en metáforas

Manuscrito recibido el 19 de mayo de 2017. Revisado 13 de junio. Aceptado 30 de noviembre.
English versión received January, 12th, 2018. Revised February, 6th. Accepted February, 16th.
D. Pérez-Marín y R. Hijón-Neira trabajan en el campus de Móstoles de la Universidad Rey Juan Carlos, Madrid, España (diana.perez@urjc.es, raquel.hijon@urjc.es).
(https://orcid.org/0000-0003-3390-0251)
(https://orcid.org/0000-0003-3833-4228)
M. Martín-Lope trabaja en el campus de Madrid de la Universidad Rey Juan Carlos, Madrid, España (mercedes.martin@urjc.es).
(https://orcid.org/0000-0002-4442-8215)

en Informática en niveles universitarios se ha advertido que no se debe llegar a "romper" la metáfora [16]. Esto es, hay que usar cada metáfora en su ámbito de acción y cuidar el lenguaje, por este motivo se proporcionan las guías detalladas a los profesores, y es objeto de investigación en este artículo cómo se puede cuidar el lenguaje para que los estudiantes puedan comprender y considerar de utilidad estas metáforas, y los profesores las vean también de utilidad y las quieran aplicar en sus clases. Para validar esta propuesta se proporcionan los resultados de un estudio piloto realizado en un colegio de la Comunidad de Madrid en España.

El objetivo principal de este primer estudio piloto es validar las guías desarrolladas con las metáforas propuestas. 62 niños de 4º, 5º y 6º de Educación Primaria (de 9 a 11 años de edad) y su profesor evaluaron las guías y nos expresaron su opinión en un cuestionario que se les proporcionó.

Tanto los estudiantes como el profesor validaron la utilidad de las metáforas y mostraron su actitud favorable a usarlas en clase. Los indicadores utilizados fueron el porcentaje de estudiantes que encontraron útiles las metáforas, el porcentaje de estudiantes que encontraron difíciles las metáforas, y el porcentaje de estudiantes que solicitaron una forma alternativa de enseñanza de los conceptos informáticos.

El artículo se organiza de la siguiente forma: la Sección 2 revisa el estado del arte de los métodos para enseñar conceptos básicos de programación para desarrollar el pensamiento computacional en los niños; la Sección 3 proporciona las guías que usan por primera vez las metáforas para la enseñanza en Educación Primaria de programación; la Sección 4 proporciona los resultados del estudio piloto realizado; y finalmente, la Sección 5 termina el artículo con la conclusión.

II. Revisión del Estado del Arte

A. *Interés mundial en enseñar programación a los niños*

Heintz et al. presentaron una revisión de cómo Australia, Reino Unido, Estonia, Finlandia, Nueva Zelanda, Noruega, Suecia, Corea del Sur, Polonia y Estados Unidos enseñan programación en Informática en su Educación Primaria y Secundaria [15]. El estudio revela que normalmente la programación impartida en la asignatura de Informática es obligatoria en Educación Primaria y optativa en Educación Secundaria. La Tabla I recoge la situación particular de cada país, con celdas en blanco indicando la falta de información en esos casos. En todo caso, la información recogida pone de relieve el interés mundial en enseñar programación desde edades tempranas.

La tendencia general es ir fomentando desde edades tempranas el pensamiento computacional y la competencia digital. En Educación Secundaria se ofrecen cursos más diversos de Informática y su impacto en la sociedad [15].

En España, la asignatura donde se incluye la enseñanza de la Programación es de Libre Configuración Autonómica. Esto significa que las distintas Comunidades Autónomas pueden decidir si es obligatoria en cada caso.

Por ejemplo, en la Comunidad de Madrid y para la etapa de Educación Primaria, la asignatura se establece como de oferta obligada y se denomina "Tecnología y Recursos Digitales para la mejora del aprendizaje".

Tabla I
Interés mundial en la enseñanza de informática en educación primaria y secundaria [12]

País	Ed. Primaria	Ed. Secundaria
Australia	Obligatoria	Obligatoria
Inglaterra	Obligatoria	
Estonia	Obligatoria	Obligatoria
Finlandia	Obligatoria	
N. Zelanda		Optativa
Noruega		Optativa
Suecia	Obligatoria	Optativa
Korea Sur	Obligatoria	Optativa
EE.UU.		Optativa
Macedonia	Obligatoria	

Esta asignatura tiene un contenido marcado por la ley para toda Educación Primaria, aunque no se establece qué se debe enseñar en cada curso. Se suele impartir a través de las distintas áreas que conforman el currículo (cada colegio puede decidir cómo organizarlo).

En el currículo oficial también se regula el desarrollo la competencia digital como una de las ocho competencias clave que todos los estudiantes deben adquirir al finalizar la etapa de Educación Secundaria obligatoria [17].

B. *Enfoques para Enseñar Programación a los Niños*

Una de las herramientas más utilizadas a nivel mundial para enseñar programación a niños es el entorno de programación visual **Scratch** [18]. El objetivo es que, mientras interactúan con Scratch, los estudiantes puedan aprender conceptos básicos como secuencia, bucle, paralelismo, eventos, condicionales, operadores y datos [7,19]. Así, por ejemplo, el concepto de secuencia se puede programar moviendo una distancia corta un gato y un perro como se muestra en la Figura 1.

El concepto de bucle, como mecanismo para ejecutar la misma secuencia varias veces, se podría enseñar con el comando *repite* seguido del número de iteraciones que se desean. El concepto de condicional, comprendido como la habilidad para tomar decisiones según ciertas condiciones, se podría enseñar con el comando *si* que determina la visibilidad de los objetos. El concepto de dato, para guardar, recuperar y actualizar valores, se podría enseñar con variables que pueden guardar números o cadenas, y listas que pueden guardar una colección de números o cadenas.

Otros enfoques para la enseñanza de Programación y el desarrollo de su pensamiento computacional incluyen crear sus propios programas [6], usar Lego WeDo o robots Mindstorms EV3 [20], y enfoques sin tecnología, para

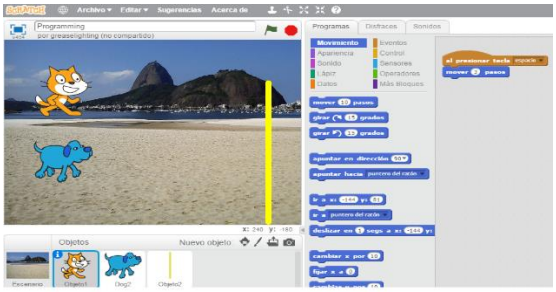


Fig. 1. Concepto de secuencia en Scratch

países en los que aunque consideran que la enseñanza de la Programación es importante, no tienen suficientes ordenadores, o profesores con conocimientos en la materia [5].

En estos enfoques, se enseña Informática mediante el uso de cuentos. Sin embargo, estos enfoques aún no han sido suficientemente evaluados para poder tener resultados significativos de su impacto [21].

C. Uso de Metáforas

El lenguaje metafórico se usa todos los días, y se considera una competencia fundamental de nuestra forma de pensar [22]. Las metáforas conceptuales, esto es, mecanismos que proyectan de un dominio fuente a un dominio destino para facilitar la enseñanza de algún concepto abstracto, pueden ser valiosas herramientas educativas [16].

Las metáforas se han usado para enseñar Biología [23], Química [24], y Matemáticas [25]. El uso de metáforas para enseñar conceptos informáticos a nivel universitario también ha sido objetivo de interés [16,26]. Hay estudios que proponen metáforas concretas para enseñar conceptos abstractos como memoria dinámica [27], o matrices para el manejo de eventos en JAVA [28]. Sin embargo, no se encuentra en la literatura ejemplos del uso de metáforas para enseñar conceptos básicos de programación en Educación Primaria.

III. PROPUESTA: USO DE METÁFORAS PARA ENSEÑAR PROGRAMACIÓN

A partir de la revisión del estado del arte se puede afirmar que todavía no se tiene claro cómo enseñar conceptos básicos de programación a los niños. Los profesores están desorientados, carecen en muchos casos de conocimientos de Informática, y adolecen de la formación adecuada para enseñar programación a los niños [14]. En otros casos, los profesores no tienen demasiados recursos. Sólo tienen la pizarra y quizás un ordenador compartido por varios estudiantes [5].

Como se ha visto en la Sección II.C, las metáforas pueden ser una herramienta educativa poderosa si se usa con cuidado. Se conocen varios casos de éxito a nivel universitario, pero no hay estudios sobre su uso con niños en la literatura. Por otro lado, se pide más investigación sobre este tema [16]. Además, se ha enfatizado la importancia de guiar a los niños paso a paso en su aprendizaje [29].

TABLA II
RESUMEN DE LAS METÁFORAS USADAS EN LOS GUIONES

ID	Guión	Concepto	Metafora
M1	Programa, programación, secuencia, memoria	Programa-secuencia	receta
		Memoria, variables	Despensa, caja
M2	Instrucciones de entrada / salida	Salida	Pantalla Ordenador
		Entrada	Teclado, reflejo en la pantalla PC
M3	Condicionales	Niño tomando decisiones	Ordenador tomando decisiones
M4	Bucles	Tu poniendo la mesa para X personas	Ordenador repitiendo instrucciones X veces

Por todo esto, proponemos en este artículo, por primera vez, cuatro guiones basados en el uso de cuatro grupos de metáforas como una metodología para enseñar conceptos básicos de programación a niños y desarrollar su pensamiento computacional. Estos cuatro guiones engloban seis metáforas que están resumidas en la Tabla II. Además, se ofrece en la Tabla III una temporalización basada en nuestra experiencia utilizando los guiones. La información está clasificada por el curso en el que el estudiante está matriculado. Por ejemplo, 4º, 5º o 6º, entre 9 y 11 años, ya que a partir de esa edad se espera que los niños tengan la adecuada capacidad cognitiva para aprender metáforas [30].

Dependiendo de si los profesores disponen de equipos de visualización, con una aplicación que proyecte las metáforas, o una presentación de Powerpoint que las incluya. En ese caso, el tiempo requerido se reduce porque las metáforas, dibujos y pseudocódigos ya están escritos. De igual forma, si los profesores tienen Scratch podrían usar las metáforas al explicar cada uno de los conceptos de programación y utilizar los bloques visuales del programa para hacer ejercicios prácticos en cada caso. En cualquier caso, los profesores que no tengan la posibilidad de tener pantallas digitales pueden usar una pizarra o similar.

El proceso de explicación en todos los casos se divide en cuatro pasos sucesivos que reflejan el curso normal de introducción a la programación: (1) concepto de programa, programar, secuencia, memoria y variable; (2) instrucciones de entrada y salida (3), condicionales, y finalmente (4) bucles.

Para resaltar cuando se utiliza una metáfora en un guion aparecerá subrayada. Cuando se introduce un nuevo concepto aparecerá en negrita. Las variables aparecerán en cursiva. En algunos casos, se ofrecen diálogos de ejemplo entre profesores (P) y estudiantes (S).

A. Guion para introducir a los niños a los primeros conceptos de programación

(P): “¿Has utilizado alguna vez un juego de ordenador o programa? ¿Conoces Youtube? Es un programa para ver videos. ¿y Paint? Es un programa para dibujar; ¿y Word? Es un programa que usas para escribir cartas, fichas, lo que

TABLA III
TIEMPO SUGERIDO PARA ENSEÑAR CONCEPTOS BÁSICOS DE PROGRAMACIÓN

Nº	Conceptos	Curso	Con Programa	Con Pizarra
1	Programa, programación, secuencia, memoria	6	1 h	2 h
		5	1 h 12’	2 h 24’
		4	1 h 24’	2 h 48’
2	Instrucciones entrada / salida	6	1 h	2 h
		5	1 h 12’	2 h 24’
		4	1 h 24’	2 h 48’
3	Condicionales	6	1 h	2 h
		5	1 h 12’	2 h 24’
		4	1 h 24’	2 h 48’
4	Bucles	6	1 h	2 h
		5	1 h 12’	2 h 24’
		4	1 h 24’	2 h 48’

quieras; vamos a ver sobre Google, ¿sabes lo que es Google? Es un programa para buscar en Internet.
Los alumnos pueden contestar que conocen alguno de estos programas, y en algunos casos todos ellos, pero lo principal es enseñarles que son programas de ordenador, y que pueden estar instalados en sus ordenadores.

(P): "¿Sabéis cómo funcionan los programas? Como una receta, como las que veis en Master Chef Junior por ejemplo"

"Si queréis cocinar una tortilla necesitáis herramientas y alimentos para hacerlo, así: una sartén, aceite de oliva, dos huevos y una pizca de sal"

"Después especificáis los pasos que tenéis que seguir, uno después del otro (para introducir el concepto de secuencia)… como cuando estáis en fila:

1. Pon la sartén en el fuego, 2. Pon aceite, 3. Pon dos huevos en una taza, 4. Bátelos, 5. Échalos en la sartén, 6. Bátelos hasta que estén batidos

"Un programa funciona así, como una secuencia de pasos, y en cada paso se ejecuta una instrucción"

"Vamos a ver un ejemplo: quiero escribir en la pantalla del ordenador un mensaje al usuario (tú eres el usuario, la persona que usa el ordenador)".

La Figura 2 representa la pizarra en ese momento; en el programa el profesor escribirá la instrucción en el lado izquierdo de la Figura 2, y en el lado derecho el recurso disponible; si el recurso es una pizarra el profesor dibujará la ejecución paso a paso; si es una pantalla, se mostrará la ejecución dinámicamente en el programa elegido.

(P): "En mi programa escribiré la instrucción: Escribe_en_pantalla (texto que quiero que salga), y aparecerá en la pantalla" como en la Figura 3.

Después para introducir el concepto de variable y memoria la metáfora de que programar es como cocinar continua:

(P): "pero el ordenador tiene un almacén de comida o despensa donde almacena todo lo que necesita, y lo va modificando para adaptarlo a sus necesidades. Esa despensa es la memoria del ordenador que permite almacenar variables como cajitas. Las variables son cajitas que puedes rellenar con lo que necesites. Por ejemplo (Figura 3): en la despensa tengo una taza donde pongo huevos, o harina, o azúcar,…, todo lo que se necesita para seguir mi receta; lo mismo en el ordenador donde guardo los números, o nombres, o mensajes que necesito mostrar al usuario".

Programa (instrucción)	Pantalla (salida)
escribe_en_pantalla ("Hello")	Hello
Escribe_en_pantalla ("How are you?")	How are you?
escribe_en_pantalla ("Hello") escribe_en_pantalla ("How are you?")	Hello How are you?

Fig. 2. Ejemplos de instrucciones de salida escritas en un programa; izquierda: instrucción de programa, derecha: salida en la pantalla

Despensa		Memoria	
Dibujo una taza		Dibujo una caja (variable)	
La lleno de huevos		Escribo "Hello" dentro de ella	Hello
La vacío y la relleno de harina		La borro y escribo "Bye" en ella	Bye
No la vacío y le añado chocolate. Ahora tengo una taza con harina y chocolate		No la borro y le añado "have a nice day". Ahora tengo en la cajitaVariable "bye, have a nice day"	Bye, Have a nice day

Fig. 3. Ejemplos para ilustrar la metáfora de memoria-variables como despensa-caja (grupo de metáforas M1)

B. Guion para introducir a los niños a las instrucciones de entrada/Salida

Vamos a combinar los conceptos: programa, memoria y pantalla y explicar qué pasa en el ordenador representando en la pizarra o la pantalla del ordenador y la memoria (ver Figura 4), de izquierda a derecha: las instrucciones del programa, la pantalla del PC y la memoria en cada paso secuencial del programa.

Para aplicar los guiones, dependiendo de los recursos disponibles, el profesor puede dibujar la tabla en la pizarra paso a paso y explicar cada fila como un paso secuencial en la pizarra; o también podría utilizar un programa como PrimaryCode [31], que ha sido desarrollado específicamente para mostrarlo interactivamente (en ambos casos se introducen los conceptos de entrada/salida, ver Figura 5 para un ejemplo de pantalla de PrimaryCode).

Por ejemplo, como puede verse en la Figura 4, la instrucción 1 (I1) produce la creación de una "caja" en la memoria llamada NombreVariable vacía. La instrucción 2 (I2) envía un mensaje al usuario pidiendo una entrada, el estado de la cajita variable permanece igual. La instrucción

Programa	Pantalla	Memoria
I1: crea nombreVariable		**nombreVariable**
I2: escribe_en_pantalla ("cuál es tu nombre?")	What's your name?	**nombreVariable**
I3: guarda (nombreVariable)	What's your name? Mary	**nombreVariable** **Mary**
I4: escribe_en_pantalla ("Hello" nombreVariable)	What's your name? Mary Hello Mary!	**VariableName** **Mary**

Fig. 4. Ejecución secuencial de las instrucciones (izda.), salida en la pantalla (centro) y estado de la cajita en memoria almacenando el dato (dcha.)

Fig. 5. Ejemplo de pantalla de PrimaryCode [31], aplicación para enseñar Programación desarrollada basándose en las metáforas propuestas

3 (I3) almacena lo que el usuario escribe usando el teclado que se refleja en la pantalla, es este caso su nombre "Mary" se almacena en la variable. Finalmente, la instrucción 4 (I4) produce una frase escrita en la pantalla con un mensaje escrito por el programador "Hello" y el contenido de la variable en memoria (Mary), que produce el mensaje "Hello Mary" en la pantalla.

A. Guion para introducir a los niños a la programación de instrucciones condicionales.

(P): "Ahora vamos a aprender como el ordenador toma decisiones: vamos a imaginar que tenemos dos números, ¿os acordáis donde los almacena el ordenador?"

(S): "¡Sí! En variables, en cajitas en la memoria del ordenador."

(P): "¿Pensáis que el ordenador puede saber que cajita tiene el valor mayor?

(S): Contestarán "sí" o "no", o intentarán adivinarlo.

(P): "Vamos a ver si el ordenador es capaz de identificar el valor mayor. Si te preguntaran (Figura 6 arriba) cuál de las dos variables (izquierda) tiene el valor mayor, sabrías que contestar (derecha). Igual lo hace el ordenador, pero debes preguntárselo de una forma un poquito diferente: debes crear una instrucción condicional para que el ordenador decida si el contenido de una cajita es mayor que el de la otra y escribes una instrucción para cuando la condición se cumpla, y otra para cuando no se cumpla (esta rama es opcional).

(P) Continua: "Vamos a ver otro ejemplo de como el ordenador resuelve condicionales. Si te dan las notas, si la nota es mayor que 5 has aprobado y si es menor de 5 has suspendido; el ordenador entiende ese tipo de instrucciones y puede tomar decisiones también. Vamos a ver los ejemplos de la Figura 7. La primera tiene la cajita *NotaV* con el valor 6, en la ejecución (en gris) la parte del programa que se está ejecutando, puedes ver que el PC comprueba si *Nota V* es mayor o igual a 5, en ese caso ejecuta la rama "then" y la salida en la pantalla (dcha.) "Passed"; el resto del código correspondiente a la rama "else" no se ejecuta, porque el ordenador ha tomado ya la decisión de que rama ejecutar. En el siguiente ejemplo, *Nota V* tiene el valor 3, el código que el PC va a ejecutar está también en gris, el PC hace la comprobación con el valor guardado en cajita de memoria *Nota V*, en este caso decide que el valor es menor por lo que ejecuta la rama del "else", la salida por pantalla es "Failed".

Memoria	Tu (arriba) ordenador (abajo)	Respuestas
a 7 b 5	¿cuál es mayor?	a
a 7 b 5	if a es mayor que b entonces escribe_en_pantalla (a) sino escribe_en_pantalla (b)	a

Fig. 6. Condicionales, como las resuelves (1ª fila) y el odenador (2ª fila)

A continuación, se pueden hacer más ejemplos de condicionales en los que explicar que el ordenador también puede tomar decisiones más complejas o anidadas. El siguiente ejemplo es sobre esto.

(P): "En Navidad le pedimos a los Reyes Magos o Santa Claus muchas cosas, pero sabemos que recibimos más o menos regalos dependiendo de varios factores. Por ejemplo, vamos a ver…"

(S): "Si me porto bien,..., me lo como todo,…, soy amable"

(P): "de acuerdo, me decís que si os portáis bien, os lo coméis todo y sois amables con los demás recibís muchos regalos, si alguna de esas tareas no se cumplen, recibís menos. Esa clase de decisiones complejas o anidadas también las puede tomar el PC, mirad el ejemplo (Figura 8)".

Memoria	Instrucc. Programa	Salida PC
NotaV 6	if NotaV >= 5 then escribe_en_pantalla ("Passed") else escribe_en_pantalla ("Failed")	Passed
NotaV 3	if NotaV >= 5 then escribe_en_pantalla ("Passed") else escribe_en_pantalla ("Failed")	Failed

Fig. 7. PC ejecutando condicionales. El estado de las variables guardado en memoria (izda.). Las instrucciones que se ejecutan del programa (centro en gris). Salida por pantalla (derecha).

Memoria	Instrucciones Programa	Salida PC

portoBien 6
comoTodo 7
soyMajo 8
	if portoBien >= 5 then
	if comoTodo >= 5 then
	if soyMajo >= 5 then
	escribe_pantalla(`"Santa brings a lot!")
	else
	escribe_pantalla ("Santa brings few")
		Santa brings a lot!

portoBien 3
comoTodo 7
soyMajo 5
	if portobien >= 5 then
	if comoTodo >= 5 then
	if soyMajo >= 5 then
	escribe_por_pantalla ("Santa brings a lot!")
	else
	escribe_por_pantalla ("Santa brings few")
		Santa brings few

Fig. 8. PC ejecutando condicionales complejas. Estado de las variables (izda.), instrucciones que se ejecutan (centro) y salida por pantalla (dcha.)

El profesor puede dibujar en la pantalla las 3 partes representadas en Figura 8. En la memoria escribirá las cajitas que representan los valores de las variables (dependiendo de las respuestas de los niños) las instrucciones de programación incluirán estas variables como se muestran en el centro de la figura, y se irán marcando con color las instrucciones que se van ejecutando dependiendo del valor de las variables. En la parte derecha, se muestra la salida por pantalla.

B. Guion para introducir a los niños a la programación de Bucles.

Para introducir el **concepto de bucle** vamos a usar la metáfora de poner la mesa para el número de personas de la familia. El profesor dará varios ejemplos:

(P): "¿Cuántos sois en tu familia?"

(S): "¡cuatro!...¡tres!"

(P): "OK! numPersonas = 4, o numPersonas = 3, entonces después de poner el mantel tendré que repetir las instrucciones 4 ó 3 veces, dependiendo del número de personas en mi familia"

La Figura 9 muestra los dibujos que el profesor tendrá que hacer en la pizarra para introducir el concepto de bucle, (izquierda arriba) las instrucciones para poner la mesa para una persona (izquierda abajo) variable contenedora del número de personas de la familia; (derecha) pictogramas de las veces que repites las instrucciones:

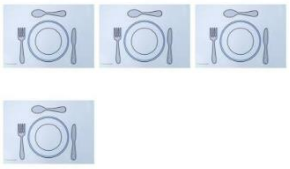
Instrucciones	**Veces que repites las instrucciones**
pon el plato, tenedor, cuchillo y cuchara 4 veces (nPersonas)	
Pon el plato, tenedor, cuchillo y cuchara 3 veces (nPersonas)	

Fig. 9. Representa como el niño repite un conjunto de instrucciones varias veces para poner la mesa para los miembros de su familia

Memoria		**Instrucciones de Programa**	**Salida Pantalla**
Antes	**Después**		
ejecución			
variable 0	**variable** 1	**Mientras (variable <= 3) hacer Escribe_pantalla ("Hello") Suma 1 a variable finMientras**	
variable 1	**variable** 2	**Mientras (variable <= 3) hacer Escribe_pantalla ("Hello") Suma 1 a variable finMientras**	
variable 2	**variable** 3	**Mientras (variable <= 3) do Escribe_pantalla ("Hello") Suma 1 a variable finMientras**	

Fig. 10. Metáforas para el concepto de bucle (grupo de metáforas M4)

Después de comprender el concepto de bucle con la metáfora anterior, el profesor deberá mostrar ejemplos de cómo el PC repite un conjunto de instrucciones en la pizarra o la pantalla, como anteriormente (Figura 10), el estado de las variables en memoria (izquierda) ahora introducimos el concepto de **antes de la ejecución y después de la ejecución**; las instrucciones del programa que se están ejecutando (centro) y la salida en la pantalla (derecha).

IV. CASO DE ESTUDIO PILOTO: EVALUACIÓN

62 estudiantes (41,9% niños y 58,1% niñas) agrupados en 4º (23 niños, 37%), 5º (17 niños, 27,4%) y 6º (22 niños, 35,6%) curso de Educación Primaria (de 9 a 11 años), como se muestra en la Figura 11, participaron en una encuesta para analizar su opinión respecto al uso de metáforas para enseñarles conceptos básicos de programación.

Cuando a los niños se les preguntó si querían crear un programa siguiendo los pasos, como los niños que participan en Master Chef y siguen los pasos de las recetas para cocinar, 55 de los 62 niños (88.7%) dijeron que sí (100% de los estudiantes de cuarto, 88.2% de los estudiantes de quinto, y 90.9% de los estudiantes de sexto).

Cuando a los niños se les preguntó por la **primera metáfora del grupo M1**, esto es, su opinión sobre comparar programar como seguir una receta, **63.3% respondió que la metáfora le facilitaba comprender el concepto**. 1 estudiante de los 23 matriculados en 4º, 4 de los 17 matriculados en 5º, y 1 de los 22 matriculados en 6º manifestaron que la metáfora les resultaba compleja. 2 niños de 6º resaltaron que les sorprendía esta metáfora, la encontraban divertida y nunca se les hubiese ocurrido la relación entre programar y cocinar. 2 niños de los 23 matriculados en 4º, 0 de los 17 matriculados en 5º y 2 de los 22 matriculados en 6º dijeron que ellos hubiesen preferido otra explicación.

Cuando a los niños se les preguntó por la **segunda metáfora del grupo M1**, esto es, su opinión sobre comparar la memoria del ordenador con una despensa, el **65.8% respondió que la metáfora le facilitaba comprender el concepto**. 1 de los 23 estudiantes matriculados en 4º, 2 de los 17 matriculados en 5º, y 1 de los 22 matriculados en 6º dijeron que les parecía difícil entender la metáfora. 2 niños en 4º, 1 en 5º, y 3 en 6º hubiesen preferido otra explicación.

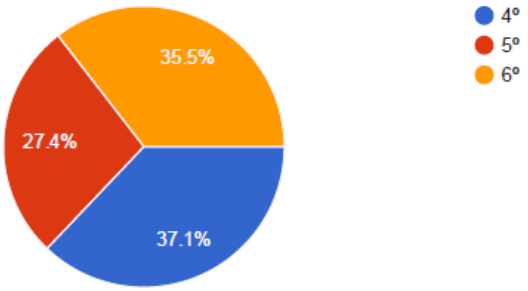


Fig. 11. Distribución de los 62 estudiantes en cursos

Cuando a los niños se les preguntó por la **tercera metáfora del grupo M1**, esto es, su opinión sobre comparar cajitas con variables, **el 73.7% de los estudiantes respondió que la metáfora le facilitaba comprender el concepto**. 6 estudiantes de los 23 matriculados en 4º, 0 de los 17 matriculados en 5º, y 0 de los 22 matriculados en 6º manifestaron que la metáfora les resultaba compleja. 0 niños de 4º, 2 niños de 5º y 2 niños de 6º dijeron que ellos hubiesen preferido otra explicación.

Cuando a los niños se les preguntó por la **cuarta metáfora (grupo M2)**, esto es, su opinión sobre comparar entrada/salida con teclado/pantalla, el **89.9% respondió que la metáfora le facilitaba comprender el concepto**. 2 de los 23 estudiantes matriculados en 4º, 0 de los 22 matriculados en 5º, y 3 de los 22 matriculados en 6º dijeron que les parecía difícil entender la metáfora. 0 niños en 4º, 0 en 5º, y 2 en 6º hubiesen preferido otra explicación.

Cuando a los niños se les preguntó por la **quinta metáfora (grupo M3)**, esto es, su opinión sobre comparar condicionales con tomar decisiones, el **89.6% respondió que la metáfora le facilitaba comprender el concepto**. 1 de los 23 estudiantes matriculados en 4º, 3 de los 17 matriculados en 5º, y 2 de los 22 matriculados en 6º dijeron que les parecía difícil entender la metáfora. Ningún niño indicó que hubiese preferido otra explicación.

Cuando a los niños se les preguntó por la **sexta metáfora (grupo M4)**, esto es, su opinión sobre comparar bucles con repeticiones, el **89.6% respondió que la metáfora le facilitaba comprender el concepto**. 1 de los 23 estudiantes matriculados en 4º, 3 de los 17 matriculados en 5º, y 2 de los 22 matriculados en 6º dijeron que les parecía difícil entender la metáfora. Ningún niño indicó que hubiese preferido otra explicación.

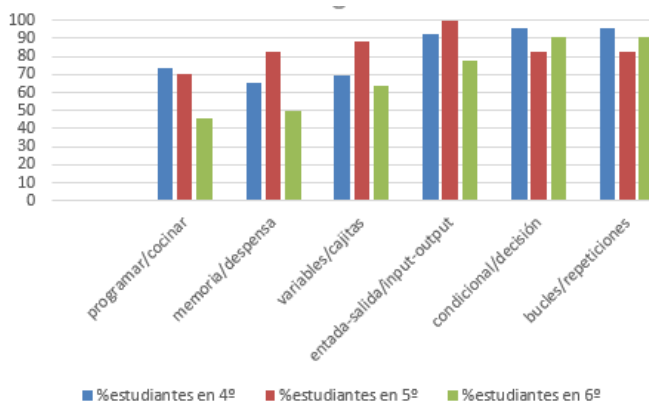


Figura 12. Porcentaje de estudiantes que encontraron útiles las metáforas

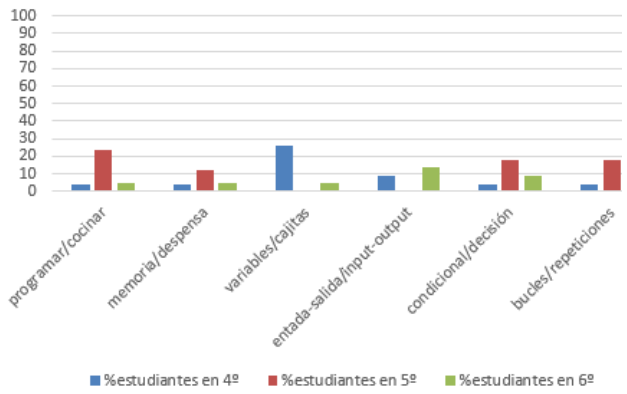


Figura 13. Porcentaje de estudiantes que encontraron difíciles las metáforas

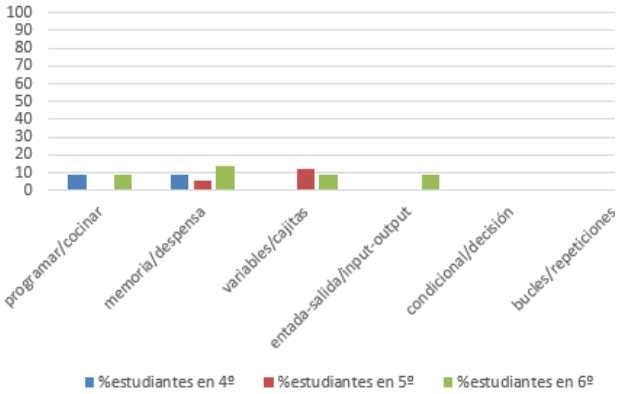


Figura 14. Porcentaje de estudiantes que solicitaron una explicación diferente

explicación.

La Figura 12 muestra un gráfico de barras con la opinión de la utilidad de los estudiantes matriculados en cada curso. Como se puede apreciar, en general, los estudiantes encontraron útiles las metáforas.

La Figura 13 muestra el porcentaje de estudiantes que encontraron difícil comprender las metáforas. Como se puede apreciar, menos del 30% de los estudiantes encontraron difícil estas metáforas, incluso aquellos que estaban matriculados en 4º. Por último, la Figura 14 muestra el porcentaje de estudiantes que solicitaron una forma alternativa de explicación. Como se puede apreciar, menos del 10% de los estudiantes necesitaron otra explicación

V. CONCLUSIONES

Hay un interés mundial en investigar cómo enseñar conceptos básicos de programación a los niños, y desarrollar su pensamiento computacional. Sin embargo, no se ha encontrado todavía ninguna metodología que pueda considerarse cómo la más adecuada para guiar a los profesores en esta tarea. Según la revisión realizada, normalmente los profesores usan Scratch o ejercicios de Code.org, pero no tienen una guía de como transmitir los conceptos a más alto nivel.

En este artículo, el objetivo es proporcionar esta guía a los profesores y a los investigadores basada en el uso de metáforas. Según el RD 126/2014, los estudiantes a partir de 4º de Primaria entienden el concepto de metáfora, y según la revisión de literatura realizada, las metáforas han sido útiles en la enseñanza de programación en niveles universitarios.

La propuesta ha sido validada por 62 niños de 4º, 5º y 6º de Educación Primaria entre 9 y 11 años de edad, que las encontraron útiles (en más de un 65% de los casos), fueron capaces de comprender las metáforas (menos de un 30% de los estudiantes las encontraron difíciles), y sólo en menos del 10% de los casos pidieron una explicación alternativa.

Además, se prueba la validez a nivel de eficacia educativa de la metáfora puesto que los estudiantes aprendieron, como se ha podido cuantificar en los resultados de un pre-post test que se les pasó antes y después de seguir la metodología propuesta tanto sin pizarra como con PrimaryCode [31] ($t=-3.668$, $p=0.001$ y $t=-3.205$, $p=0.003$ respectivamente).

El profesor también validó la metodología seguida. En una entrevista que se le realizó, a la pregunta de que le parecía la enseñanza de la programación basada en metáforas, nos respondió textualmente: “Respecto a lo de la

explicación yo creo que sí está bien porque además se trabajan los textos instructivos y en concreto el de las recetas". Se pretende seguir aplicando la metodología en más colegios y trabajando conceptos informáticos desde edades tempranas para poder ver su impacto en el desarrollo del pensamiento computacional.

AGRADECIMIENTOS

Queremos expresar nuestra gratitud al colegio y a los proyectos TIN2015-66731-C2-1-R y S2013/ICE-2715.

REFERENCIAS

[1] F. J. García-Peñalvo, "Proyecto TACCLE3 – Coding," in XVIII Simposio Internacional de Informática Educativa, SIIE 2016, F. J. García-Peñalvo and J. A. Mendes, Eds. no. 222) Salamanca, España: Ediciones Universidad de Salamanca, 2016, pp. 187-189.

[2] F. J. García-Peñalvo, "A brief introduction to TACCLE 3 – Coding European Project," in 2016 International Symposium on Computers in Education (SIIE 16), F. J. García-Peñalvo and J. A. Mendes, Eds. USA: IEEE, 2016.

[3] F. J. García-Peñalvo, D. Reimann, M. Tuul, A. Rees, and I. Jormanainen, "An overview of the most relevant literature on coding and computational thinking with emphasis on the relevant issues for teachers," TACCLE3 Consortium, Belgium. 2016. doi:10.5281/zenodo.165123

[4] P.Y. Chao, Exploring students' computational practice, design and performance of problem-solving through a visual programming environment. Computers & Education, 95, 2016, pp. 202-215.

[5] C. Brackmann, D. Barone, A. Casali, R. Boucinha, S. Muñoz-Hernandez, "Computational thinking: Panorama of the Americas", In International Symposium on Computers in Education (SIIE), 2016, pp. 1-6.

[6] S. Campe, J. Denner, Programming Games for Learning: A Research Synthesis. Paper presented at the annual meeting of the American Educational Research Association, Chicago, 2015.

[7] I. Ouahbi, F. Kaddari, H. Darhmaoui, A. Elachqar, S. Lahmine, "Learning basic programming concepts by creating games with scratch programming environment", Procedia-Social and Behavioral Sciences, 191, 2015, pp. 1479-1482.

[8] M. Jovanov, E. Stankov, M. Mihova, S. Ristov, M. Gusev, "Computing as a new compulsory subject in the Macedonian primary schools curriculum", In Global Engineering Education Conference (EDUCON), 2016, IEEE, pp. 680-685.

[9] E. Lahtinen, K. Ala-Mutka, H.M. Järvinen, "A study of the difficulties of novice programmers", in ACM SIGCSE Bulletin, vol. 37, no. 3, 2005, pp. 14-18.

[10] D. Ginat, "On novice loop boundaries and range conceptions", Computer Science Education, 14(3), 2004, pp. 165-181.

[11] O. Seppälä, L. Malmi, A. Korhonen, "Observations on student misconceptions—A case study of the Build–Heap Algorithm", Computer Science Education, 16(3), 2006, pp. 241-255.

[12] L.J. Barker, C. McDowell, K. Kalahar, "Exploring factors that influence computer science introductory course students to persist in the major", In ACM SIGCSE Bulletin, Vol. 41, No. 1, 2009, pp. 153-157.

[13] N.J. Coull, I.M. Duncan, "Emergent requirements for supporting introductory programming", Innovation in Teaching and Learning in Information and Computer Sciences, 10(1), 2011, pp. 78-85.

[14] A. Yadav, S. Gretter, S. Hambrusch, P. Sands, "Expanding computer science education in schools: understanding teacher experiences and challenges", Computer Science Education, 2016, pp. 1-20.

[15] F. Heintz, L. Mannila, T. Färnqvist, "A review of models for introducing computational thinking, computer science and computing in K-12 education", IEEE Frontiers in Education Conference, 2016, pp. 1-9.

[16] J.P. Sanford, A. Tietz, S. Farooq, S. Guyer, R.B. Shapiro, "Metaphors we teach by", In Proceedings of the 45th ACM technical symposium on Computer science education, 2014, pp. 585-590.

[17] INTEF, Marco Común de Competencia Digital Docente. Spanish Ministry of Education, Culture and Sports. Available on-line at: https://goo.gl/7pvLve (última visita: 30 de marzo, 2017).

[18] M. Resnick, J. Maloney, A. Monroy-Hernandez, N. Rusk, E. Eastmond, K. Brennan, et al., Scratch: Programming for all. Communications of the ACM, 52(11), 2009, pp. 60-67.

[19] K. Brennan, M. Resnick, "New frameworks for studying and assessing the development of computational thinking". In Proceedings of the 2012 annual meeting of the American Educational Research Association, Vancouver, Canada, 2012, pp. 1-25.

[20] A. Sović, T. Jagušt, D. Seršić, "How to teach basic university-level programming concepts to first graders?", In Integrated STEM Education Conference (ISEC), 2014, IEEE, pp. 1-6.

[21] F. Kalelioğlu, "A new way of teaching programming skills to children: Code. Org", Computers in Human Behavior, 52, 2015, pp. 200-210.

[22] G. Lakoff, M. Johnson, "Metaphors we live by". University of Chicago press, 2008.

[23] N.A. Paris, S.M. Glynn, "Elaborate analogies in science text: Tools for enhancing preservice teachers' knowledge and attitudes", Contemporary Educational Psychology, 29(3), 2004, pp. 230-247.

[24] G.P. Thomas, C.J. McRobbie, "Using a metaphor for learning to improve students' metacognition in the chemistry classroom", Journal of Research in Science Teaching, 38(2), 2001, pp. 222-259.

[25] P. Boero, L. Bazzini, R. Garuti, "Metaphors in teaching and learning mathematics: a case study concerning inequalities". In Pme Conference, Vol. 2, 2001, pp. 2-185.

[26] R.T. Putnam, D. Sleeman, J.A. Baxter, L.K. Kuspa, "A summary of misconceptions of high school Basic programmers", Journal of Educational Computing Research, 2(4), 1986, pp. 459-472.

[27] R. Jiménez-Peris, C. Pareja-Flores, M. Patiño-Martínez, J.A. Velázquez-Iturbide, "The locker metaphor to teach dynamic memory", In ACM SIGCSE Bulletin, Vol. 29, No. 1, 1997, pp. 169-173.

[28] W.W. Milner, "A broken metaphor in Java", ACM SIGCSE Bulletin, 41(4), 2010, pp. 76-77.

[29] D.H. Clements, B.K. Nastasi, S. Swaminathan, "Young children and computers: crossroads and directions from research". Young children, 48(2), 1993, pp. 56–64.

[30] RD 126/2014. Basic Curriculum Primary Education in Spain. Available on-line at: https://goo.gl/nHRs8n (última visita: 6 de marzo, 2018).

[31] PrimaryCode. http://www.lite.etsii.urjc.es/tools/primarycode/ (última visita: 6 de marzo, 2018).

Diana Pérez Marín es Doctora en Ingeniería Informática y Telecomunicación por la Universidad Autónoma de Madrid desde el año 2007. Actualmente, es Profesor Contratado Doctor del Departamento de la Escuela Técnica Superior de Ingeniería Informática de la Universidad Rey Juan Carlos. Su principal línea de interés en investigación es el uso de los ordenadores para la educación con 24 artículos indexados (Thomson Reuters Researcher ID L-4100-2014), H-index 8, y 141 citas (Scopus: 14042379400).

Raquel Hijón Neira tiene el Doctorado Europeo en Informática por la Universidad Rey Juan Carlos, Madrid, España. Ha trabajado como Ingeniera Informática durante cinco años, y enseñando en la Universidad durante 17 años. Actualmente, es Profesor Contratado Doctor en la Universidad Rey Juan Carlos, y miembro del Laboratorio para la Investigación en Tecnologías Educativas (LITE). Sus áreas de investigación son la innovación en la enseñanza de la programación, Interacción Persona-Ordenador y juegos serios. Dr. Hijón-Neira obtuvo el galardón a la mejor tesis otorgado por la IEEE.

Mercedes Martín-Lope es Profesor de Metodología de la Investigación Educativa, Didáctica de las Matemáticas y Diseño Curricular. Es Doctora en Métodos de Investigación y Diagnóstico Educativo por la Universidad Rey Juan Carlos (2015). Martín-Lope tiene una larga experiencia en enseñanza pre-universitaria (1998-2005). A partir del año 2003 comenzó a dar clases en la Universidad. Actualmente, es Coordinadora del Grado de Infantil de la Universidad Rey Juan Carlos. Dr. Martín-Lope ha participado en varios congresos internacionales y coordinado proyectos de Innovación Educativa.

VAEP-RITA (http://webs.uvigo.es/cesei/VAEP-RITA)

Revisores

A

Carlos Alario Hoyos
Ainhoa Álvarez
Humberto R. Álvarez A.
Francisco Javier Álvarez Rodríguez
Rita Arauz de Takakuwa
Juan Ignacio Asensio Pérez
Alejandro Ayala
Francisco Javier Azcondo

B

Francisco Javier Basogain Olabe
Andrea Bellucci
Manuel Benito
Aranzazu Berbey Alvarez
Oriol Borrás
Felix Buendia
Juan Carlos Burguillo Rial

C

Pedro Cacique
Manuel Caeiro Rodríguez
Germán Carro
Jose Carlos Castillo Montoya
Álvaro Castro González
Cristian Cechinel
Alamgir Choudhury
César Collazos
Miguel Ángel Conde
Juan Cruz-Benito

D

Claudia Deco
Juan Manuel Dodero
Elena Domínguez Romero

F

Ramón Fabregat
Patricia Margarida Farias Coelho
Valeria Farinazzo Martins
Camino Fernández Llamas
Miguel Ferrando
Debora Ferreira
Begoña Ferrero Martin

G

Xabiel García Pañeda
Javier García Zubía
Alicia García-Holgado
Miriam Gil Garzón
Carina Soledad González González
Daniel Gutierrez Reina

L

Manuel Lama Penín
Beatriz Legerén Lago
Faraón Llorens
Maria del Carmen Llorente Cejudo

M

Ángeles Manjarrés Riesco
Sergio Martín
Pablo Martín-Ramos
Oscar Martínez Bonastre
Fernando A. Mikic-Fonte
Norma Miller
Xavier Molero
Ana Isabel Molina Díaz
Pablo Molins Ruaro
Javier Montefort
Alvaro Montero
Asuncion Morales Santana
Veronica Moreno
Iveth del Rosario Moreno González
Jesús Moreno León
Lourdes Moreno López
Jose Miguel Mota
Roberto Muñoz
Pedro José Muñoz Merino
Mario Muñoz Organero
Jaime Muñoz-Arteaga

N

Leonardo Ramon Nunes de Sousa
António José Nunes Mendes

O

Francisco Javier Oliver
Teresa Onorati
Pablo Orduña

P

Beatriz Pacheco
Manuel Palomo Duarte
Rafael Pastor Vargas
Rita Pereira
Diana Rosario Pérez Marín
Ana Pinto

R

Gustavo Ribeiro da Costa Alves
Humberto Rodriguez
Jorge Rodríguez
Miguel Rodríguez Artacho
Luis G. Rodriguez V
Arturo Rojas
Marcos Román González
Marco Romano
Susana Romero
Pablo Eduardo Romero-Carrillo
Salvador Ros
Catalina Rus-Casas

S

Hamadou Saliah-Hassane
Elio San Cristóbal Ruíz
Mónica Sánchez
Juan Manuel Santos Gago
María Luisa Sein-Echaluce Lacleta
José Luis Sierra Rodríguez
Ismar Frango Silveira
Hanna Skorczynska
Eduardo Antonio Soto Rodríguez

T

Ana Tajadura Jiménez
Romero Tori

V

Lino Valdenebro
Carlos Vaz de Carvalho
Jesús Ángel Velázquez Iturbide
Julián Viejo Cortés
Klinge Villalba

Z

Telmo Zarraonandia

Asistente Técnico de Edición: Diego Estévez González,
Universidad de Vigo, España

VAEP-RITA es una publicación lanzada por el Capítulo Español de la Sociedad de Educación del IEEE (CESEI). Nuestro agradecimiento a los apoyos recibidos desde el año 2006 por el Ministerio Español de Educación y Ciencia a través de la acción complementaria TSI2005-24068-E y del Ministerio Español de Ciencia e Innovación a través de las TSI2007-30679-E y TIN2009-07333-E/TSI. Gracias también a la Universidade de Vigo y a FEUGA por el apoyo en esta nueva etapa.

VAEP-RITA es una publicación de la Sociedad de Educación del IEEE, gestionada por su Capítulo Español y apoyada por la Universidade de Vigo, España.

VAEP-RITA é uma publicação da Sociedade de Educação do IEEE, gerida pelo Capitulo Espanhol e apoiada pela Universidade de Vigo, España.

VAEP-RITA is a publication of the IEEE Education Society, managed by its Spanish Chapter, and supported by the Universidade de Vigo, Spain.