Resumen de la tesis

1. Introducción

La inclusión de las habilidades de programación informática y pensamiento computacional (PC) en el curriculum escolar es una de las principales tendencias en el panorama educativo mundial [Info-communications Media Development Authority, 2014, Igbokwe, 2015, Australian Curriculum Assessment and Reporting Authority, 2015, The White House, 2016]. Este movimiento ha provocado un profundo interés entre los académicos y las instituciones de investigación [Furber, 2012, l'Académie des sciences, 2013, Meseguer et al., 2015, Gander et al., 2013], que están analizando y comparando los enfoques y las estrategias de las diferentes iniciativas. Las revisiones sobre el estado del arte que se han realizado en lo relativo al PC en la educación coinciden en tres aspectos principales y fundamentales que requieren atención urgente por parte de la comunidad científica [Bocconi et al., 2016a, Grover and Pea, 2013, Bocconi et al., 2016b]: la evaluación de las habilidades de PC, la transferencia del PC a otros dominios, y los factores que afectan al desarrollo del PC.

En consecuencia, y en concordancia con las lagunas identificadas en la literatura científica, el objetivo principal de esta tesis es proporcionar evidencia que pueda ayudar a los responsables políticos y a los educadores en la introducción del PC en el currículo escolar. Dado que el mejor conocimiento científico sobre el tema hasta la fecha muestra que la forma más efectiva de fomentar el PC desde edades tempranas es mediante actividades de programación [Kalelioglu et al., 2016, Lye and Koh, 2014], esta tesis investiga el desarrollo del PC a través de este tipo de actividades. Además, como los educadores indican que la herramienta que más se utiliza tanto en la educación primaria como en la secundaria es `Scratch` [Sentance, 2015], nuestro trabajo se centra en este lenguaje.

2. Objetivos

Las preguntas de investigación que se abordan en esta tesis son las siguientes:

1. **¿Es posible evaluar automáticamente el desarrollo de habilidades de PC inspeccionando las creaciones `Scratch` de los aprendices?** Con el objetivo de apoyar a los docentes en las tareas de evaluación, desarrollamos una herramienta automática, `Dr.Scratch`, que permite evaluar varios aspectos del PC mediante el análisis estático del código fuente de proyectos `Scratch`. Para validar las evaluaciones que ofrece la herramienta se han realizado diferentes investigaciones trabajando en estrecha colaboración con maestros y estudiantes en colegios e institutos, así como haciendo uso de los proyectos disponibles en el repositorio público de `Scratch`.

Las conclusiones de estas investigaciones proporcionan evidencia que se puede usar para responder a esta pregunta de investigación.

2. **¿El desarrollo de habilidades de PC a través de actividades de programación con `Scratch` mejora el aprendizaje de otras asignaturas?**
   La idea de usar la programación informática para mejorar el aprendizaje de los estudiantes en diferentes dominios lleva discutiéndose en la comunidad científica desde la década de 1960. Sin embargo, todavía hay un déficit importante de estudios que sigan rigurosamente las recomendaciones para realizar investigación en educación. Por lo tanto, hemos desarrollado diferentes intervenciones en escuelas con el objetivo de responder a esta y a otras preguntas consecuentes, tales como si hay diferencias en términos de transferencia de habilidades de PC en función de la edad de los estudiantes o en función de la asignatura objetivo, o si la fomación previa de los docentes en lo relativo a la programación informática tiene un impacto en los resultados de sus estudiantes.

3. **¿Existen factores sociales y no cognitivos que afectan al desarrollo de habilidades de PC cuando se programa con `Scratch`?**
   Las definiciones de PC que se han propuesto recientemente introducen la idea de que los aspectos sociales y no cognitivos son una parte importante de esta habilidad. Sin embargo, hay poca evidencia para respaldar esta idea, y los trabajos que han investigado esta cuestión se basan principalmente en casos de estudio pequeños. Por consiguiente, varias investigaciones que forman parte de esta tesis estudian si los factores sociales y no cognitivos tienen una influencia significativa en el desarrollo de la programación y el PC.

3. Metodología

Desarrollo y validación de Dr. Scratch

Como primer paso en el desarrollo de `Dr.Scratch`, creamos dos *plug-ins* para el entorno `Hairball` [Boe et al., 2013]. Estos *plug-ins* amplían las funcionalidades de `Hairball` permitiendo la detección de algunos malos hábitos de programación que los docentes detectan con frecuencia en los proyectos de sus alumnos, como son el uso de los nombres por defecto que `Scratch` asigna a los nuevos objetos, y la repetición de código. El proceso de desarrollo y la prueba de los plug-ins haciendo uso de proyectos disponibles en el repositorio `Scratch` está documentado en [Moreno and Robles, 2014].

Tras la publicación de la herramienta comenzó su proceso de validación. En primer lugar, uno de los objetivos era que los usuarios pudieran analizar sus proyectos y aprender de forma independiente aprovechando los consejos que brinda la herramienta, por lo que organizamos una serie de talleres con más de 100 alumnos, entre 10 y 14 años, en 8 escuelas diferentes, para verificar en qué medida los estudiantes mejoran sus habilidades de programación mientras usan la herramienta en escenarios de la vida real (validez ecológica). Los detalles de esta intervención, así como el funcionamiento interno de la herramienta, se presentan en [Moreno-León et al., 2015].

Un gran paso en el proceso de validación es la comparación de las evaluaciones proporcionadas por `Dr.Scratch` con otras mediciones de constructos similares (validez convergente). Identificamos dos evaluaciones que podrían usarse en este sentido: métricas de complejidad de ingeniería de software y evaluaciones proporcionadas por humanos expertos.

Por un lado, comparamos las puntuaciones de `Dr.Scratch` con dos métricas clásicas de ingeniería de software reconocidas mundialmente como medidas válidas para la complejidad de un sistema software: la complejidad ciclomática de McCabe [McCabe, 1976] y las métricas de Halstead [Halstead, 1977]. Los resultados después de analizar 95 proyectos `Scratch` que se descargaron aleatoriamente del repositorio se presentan en [Moreno-León et al., 2016b].

Por otro lado, para comparar las puntuaciones automáticas proporcionados por `Dr.Scratch` con evaluaciones de humanos expertos, organizamos un concurso de programación para alumnos de primaria y secundaria. Esto nos permitió recopilar y estudiar más de 450 evaluaciones de proyectos de `Scratch` ofrecidas por 16 expertos en educación informática. Los resultados se presentan en [Moreno-León et al., 2017].

Finalmente, dado que las creaciones `Scratch` se clasifican en diferentes tipos de proyectos, es interesante averiguar si esta topología se replica cuando los proyectos se analizan con `Dr.Scratch` (validez discriminante). En consecuencia, descargamos aleatoriamente 250 proyectos de las categorías principales del repositorio para verificar si `Dr.Scratch` es capaz de detectar diferencias en las dimensiones de PC desarrolladas al programar diferentes tipos de proyectos `Scratch`, tal como se publicó en [Moreno-León et al., 2017].

Pensamiento computacional en el curriculum

Para estudiar si la programación informática, cuando se utiliza como recurso educativo, puede mejorar el aprendizaje de asignaturas no relacionadas con las tecnologías de la información y las comunicaciones, se realizó una revisión sistemática de la literatura para identificar si existen evidencias sobre los beneficios educativos de la programación en primaria y secundaria, para lo que se resumieron los hallazgos empíricos más importantes y se identificaron posibles líneas para futuras investigaciones. El resultado de esta revisión de la literatura se presenta en [Moreno-León and Robles, 2016]. Vale la pena señalar que este artículo recibió el **premio al mejor artículo (Best Paper Award) en la conferencia 2016 IEEE Global Engineering Education Conference** (Abu Dhabi, Emiratos Árabes Unidos).

Las conclusiones de la revisión de la literatura indican que, si bien los hallazgos de los artículos analizados presentan una imagen muy prometedora, es necesario realizar más investigaciones con muestras de alumnos más grandes para justificar el uso de la programación como una herramienta educativa en estos niveles educativos. En consecuencia, buscamos escuelas con las que pudiéramos colaborar para realizar nuevas investigaciones al respecto.

La primera de estas investigaciones se presenta en [Moreno-León and Robles, 2015]. El artículo resume el trabajo realizado en el Colegio San Diego y San Vicente (Madrid, España) durante el año académico 2013/14, donde se desarrolló un estudio con cuatro grupos de estudiantes de 4º y 5º de primaria para medir hasta qué punto el uso de la programación informática en la clase de inglés puede ser una herramienta educativa interesante con un impacto positivo en los

resultados de aprendizaje de los estudiantes. Además, el hecho de que trabajamos con dos profesores que tenían una experiencia de programación previa diferente nos permitió estudiar las diferencias en el aprendizaje de sus alumnos en base a este parámetro.

Siguiendo este enfoque, pero ahora estudiando el desarrollo del pensamiento matemático a través de la programación, participamos en un cuasi-experimento con dos grupos de 6º en la Escuela Hacienda Candelaria (Lorica, Colombia). Específicamente, las preguntas bajo investigación en este artículo son si el uso de la programación en las clases de matemáticas mejora (a) el proceso de modelado y los fenómenos de realidad, (b) el razonamiento, (c) la formulación de problemas y la resolución los mismos, y (d) la comparación y ejecución de procedimientos y algoritmos, como se discute en [Calao et al., 2015].

La última de estas intervenciones empíricas sobre el uso del PC como recurso educativo compara tres diseños de investigación cuasi-experimentales realizados en tres escuelas diferentes [Moreno-León et al., 2016a]. Dado que el curso académico y la asignatura en la que se integró la programación fue diferente en cada una de las escuelas, los resultados se discuten en términos de estos parámetros. Por lo tanto, es una investigación que combina métodos cuasi-experimentales y causales-comparativos, siguiendo el enfoque propuesto por Schenker y Rumrill [Schenker and Rumrill Jr, 2004].

Factores sociales y no cognitivos del pensamiento computacional

Con el fin de proporcionar evidencia sobre los factores sociales y no cognitivos que influyen en el desarrollo de las habilidades de programación y de PC, se han seguido dos enfoques diferentes.

Por un lado, con el objetivo de investigar cómo afecta la socialización al aprendizaje de habilidades de programación, hicimos uso de un conjunto de datos que contiene la actividad pública en la comunidad `Scratch` durante sus primeros cinco años de vida (2007-2012):

> "El conjunto de datos incluye 32 tablas extraídas de la base de datos MySQL utilizada en la plataforma web de `Scratch`. Las tablas capturan muchos tipos de información, incluidos los metadatos en torno a los usuarios y los proyectos, así como el comportamiento del usuario y las comunicaciones que realiza en el sitio. Además de los metadatos, el conjunto de datos también incluye el texto completo de los comentarios públicos, el código fuente completo para cada proyecto y los resúmenes cuantitativos de los contenidos de los proyectos. El conjunto de datos incluye datos sobre más de 1 millón de usuarios, casi 2 millones de proyectos, más de 10 millones de comentarios y metadatos en decenas de millones de eventos como vistas, descargas y expresiones de agradecimiento." [Hill and Monroy-Hernández, 2017, p. 2]

Específicamente, analizamos las interacciones sociales de casi 70.000 usuarios de la plataforma `Scratch` y la sofisticación de más de 1,5 millones de productos de software que crearon, con el objetivo de averiguar si existe una relación entre las conductas sociales de los usuarios y la mejora de sus habilidades de

programación, así como si el tiempo que los usuarios pasan en la comuniad afecta esta relación potencial. Este trabajo se presenta en [Moreno-León et al., 2016c].

Por otra parte, con el objetivo de comprender mejor qué factores no cognitivos subyacen al PC, se hizo que 1.251 estudiantes españoles de 24 escuelas diferentes, matriculados entre 5º de primaria y 4º de Secundaria, realizaran el CTt [Román-González, 2015] y algunos elementos de autoeficacia adicionales, mientras que 99 de esos estudiantes realizaron además el *Big Five Questionnaire-Children version* [Barbaranelli et al., 2003]. El objetivo era estudiar las posibles correlaciones entre el PC, la autoeficacia y las dimensiones del modelo de personalidad de *Big Five*: apertura a la experiencia, conciencia, extroversión, amabilidad y neuroticismo. Los resultados se presentan en [Román-González et al., 2017].

4. Resultados principales

La principal contribución de esta tesis es el desarrollo y validación de `Dr. Scratch`, una herramienta libre/de código abierto que evalúa el grado de desarrollo del PC en proyectos `Scratch`. Esta herramienta está siendo utilizada por cientos de miles de estudiantes, educadores e investigadores de todo el mundo [Lawanto, 2016, Martin, 2016, Garneli and Chorianopoulos, 2017, von Wangenheim et al., 2017, Webber et al., 2014, Aivaloglou and Hermans, 2016, Techapalokul and Tilevich, 2017], y ha recibido el **Premio Google RISE Award**. Se han realizado diferentes acciones para validar las evaluaciones proporcionadas por la herramienta, demostrando su validez ecológica, convergente y discriminante.

Además, las investigaciones empíricas realizadas tanto con respecto a la transferencia de PC en educación primaria y secundaria, como en lo relativo a los factores sociales y cognitivos que afectan al PC, proporcionan ideas y evidencia que pueden ser útiles para docentes, directores y legisladores y que, por lo tanto, podrían tener un impacto en la enseñanza de programación en escuelas, institutos y universidades. De hecho, **las conclusiones de estas investigaciones empíricas se han incluido en varios informes oficiales** que estudian la introducción de la programación y las habilidades de PC en las escuelas para apoyar a los responsables políticos, tanto a nivel nacional [Centro Nacional de Innovación e Investigación Educativa, 2017] como europeo [Balanskat and Engelhardt, 2015, Bocconi et al., 2016a].

Publicaciones asociadas a la tesis

Esta tesis se presentó como un compendio de publicaciones, de acuerdo al Artículo 21 de la normativa de los estudios de doctorado de la Escuela Internacional de Doctorado de la Universidad Rey Juan Carlos.

Las publicaciones incluidas en el compendio son las siguientes:

1. Jesús Moreno-León, Gregorio Robles, and Marcos Román-González. **Towards Data-Driven Learning Paths to Develop Computational Thinking with Scratch**. IEEE Transactions on Emerging Topics in Computing, 2018 (in press).
   DOI: 10.1109/TETC.2017.2734818. JCR Impact factor (2016): 3.826. Q1

2. Marcos Marcos Román-González, Juan-Carlos Pérez-González, Jesús Moreno-León and Gregorio Robles. **Extending the nomological network of computational thinking with non-cognitive factors**. Computers in Human Behavior, 80:441–459, 2018.
DOI:10.1016/j.chb.2017.09.030. JCR Impact factor (2016): 3.435. Q1

3. Jesús Moreno-León, Marcos Román-González, Casper Harteveld, and Gregorio Robles. **On the automatic assessment of computational thinking skills: A comparison with human experts**. In Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems, CHI EA '17, pages 2788–2795, New York, NY, USA, 2017. ACM. DOI: 10.1145/3027063.3053216. GII-GRIN-SCIE (GGS) Conference Rating: Class 1 (CORE:A++, LiveSHINE:A++, MA:A++).

4. Jesús Moreno-León, Gregorio Robles, and Marcos Román-González. **Examining the relationship between socialization and improved software development skills in the Scratch code learning environment**. Jounal of Universal Computer Science, 22(12):1533–1557, 2016. DOI: 10.3217/jucs-022-12-1533. JCR Impact factor (2016): 0.696. Q4

5. Jesús Moreno-León, Gregorio Robles, and Marcos Román-González. **Dr. Scratch: Automatic analysis of scratch projects to assess and foster Computational Thinking**. RED. Revista de Educación a Distancia, 15(46), 2015. DOI:10.6018/red/46/10. Journal in the WoS Emerging Sources Citation Index.

6. Jesús Moreno-León, Gregorio Robles, and Marcos Román-González. **Code to learn: Where does it belong in the K-12 curriculum?** Journal of Information Technology Education: Research, 15:283–303, 2016. Journal in the WoS Emerging Sources Citation Index and Scopus.

7. Jesús Moreno and Gregorio Robles. **Automatic detection of bad programming habits in Scratch: A preliminary study**. In 2014 IEEE Frontiers in Education Conference (FIE) Proceedings, pages 1–4, 2014. DOI: 10.1109/FIE.2014.7044055. GII-GRIN-SCIE (GGS) Conference Rating: Class 3 (CORE:B, LiveSHINE:B).

8. Luis Alberto Calao, Jesús Moreno-León, Heidy Ester Correa, and Gregorio Robles. **Developing mathematical thinking with Scratch: An experiment with 6th grade students**. In Design for Teaching and Learning in a Networked World, pages 17–27. Springer, 2015. DOI: 10.1007/978-3-319-24258-3_2. GII-GRIN-SCIE (GGS) Conference Rating: Class 3 (LiveSHINE:B, MA:B-).

9. Jesús Moreno-León and Gregorio Robles. **Code to learn with Scratch? A systematic literature review**. In Global Engineering Education Conference (EDUCON), 2016 IEEE, pages 150–156. IEEE, 2016. DOI: 10.1109/EDUCON.2016.7474546. GII-GRIN-SCIE (GGS) Conference Rating: Work in Progress (LiveSHINE:B). This work received the **Best Paper Award**.

10. Jesús Moreno-León, Gregorio Robles, and Marcos Román-González. **Comparing computational thinking development assessment scores with software complexity metrics**. In 2016 IEEE Global Engineering

Education Conference (EDUCON), pages 1040–1045, April 2016. DOI: 10.1109/EDUCON.2016.7474681. GII-GRIN-SCIE (GGS) Conference Rating: Work in Progress (LiveSHINE:B).

11. Jesús Moreno-León and Gregorio Robles. **Computer programming as an educational tool in the English classroom: A preliminary study**. In Global Engineering Education Conference (EDUCON), 2015 IEEE, pages 96–966. IEEE, 2015.
DOI: 10.1109/EDUCON.2015.7096089. GII-GRIN-SCIE (GGS) Conference Rating: Work in Progress (LiveSHINE:B).

Además, durante el desarrollo de la tesis el autor estuvo involucrado en las siguientes publicaciones:

1. Christian P. Brackmann, Marcos Román-González, Gregorio Robles, Jesús Moreno-León, Ana Casali and Dante Barone. **Development of Computational Thinking Skills through Unplugged Activities in Primary School**. In Proceedings of the 12th Workshop on Primary and Secondary Computing Education, pages 65–72. ACM, 2017.

2. Jackie Barnes, Amy K. Hoover, Borna Fatehi, Jesús Moreno-León, Gillian Smith, and Casper Harteveld. **Exploring emerging design patterns in student-made climate change games**. Proceedings of the 12th International Conference on the Foundations of Digital Games. ACM, 2017.

3. Efthimia Aivaloglou, Felienne Hermans, Jesús Moreno-León, and Gregorio Robles. **A dataset of scratch programs: scraped, shaped and scored**. In Proceedings of the 14th International Conference on Mining Software Repositories, pages 511–514. IEEE Press, 2017.

4. Gregorio Robles, Jesús Moreno-León, Efthimia Aivaloglou, and Felienne Hermans. **Software clones in scratch projects: On the presence of copy-and-paste in computational thinking learning**. In Software Clones (IWSC), 2017 IEEE 11th International Workshop on, pages 1–7. IEEE, 2017.

5. Marcos Román-González, Jesús Moreno-León and Gregorio Robles. **Complementary tools for computational thinking assessment**. In CTE 2017: International Conference on Computational Thinking Education 2017, pages 154–159, 2017.

6. Jesús Moreno-León, Gregorio Robles and Marcos Román-González. **How social are Scratch learners? A comprehensive analysis of the Scratch platform for social interactions**. In FLOSS Education and Computational Thinking Workshop. 12th International Conference on Open Source Systems, pages 19–26, 2016.

7. Amy K. Hoover, Jackie Barnes, Borna Fatehi, Jesús Moreno-León, Gillian Puttick, Eli Tucker-Raymond, and Casper Harteveld.**Assessing Computational Thinking in Students' Game Designs**. Proceedings of the 2016 Annual Symposium on Computer-Human Interaction in Play Companion Extended Abstracts. ACM, 2016.

8. Jesús Moreno-León and Gregorio Robles. **The Europe code week (CodeEU) initiative: Shaping the skills of future engineers**. In 2015 IEEE Global Engineering Education Conference (EDUCON), pages 561–566. IEEE, 2015.

Referencias

[Aivaloglou and Hermans, 2016] Aivaloglou, E. and Hermans, F. (2016). How kids code and how we know: An exploratory study on the scratch repository. In *Proceedings of the 2016 ACM Conference on International Computing Education Research*, pages 53–61. ACM.

[Australian Curriculum Assessment and Reporting Authority, 2015] Australian Curriculum Assessment and Reporting Authority (2015). Digital technologies, foundation to year 10 curriculum. Available at `http://www.australiancurriculum.edu.au/technologies/digital-technologies/curriculum/f-10`, accessed 2017-01-01.

[Balanskat and Engelhardt, 2015] Balanskat, A. and Engelhardt, K. (2015). Computing our future: Computer programming and coding. Priorities, school curricula and initiatives across Europe. Technical report, European Schoolnet.

[Barbaranelli et al., 2003] Barbaranelli, C., Caprara, G. V., Rabasca, A., and Pastorelli, C. (2003). A questionnaire for measuring the big five in late childhood. *Personality and Individual Differences*, 34(4):645–664.

[Bocconi et al., 2016a] Bocconi, S., Chioccariello, A., Dettori, G., Ferrari, A., Engelhardt, K., Kampylis, P., and Punie, Y. (2016a). Developing computational thinking in compulsory education - Implications for policy and practice. Technical report, Publications Office of the European Union.

[Bocconi et al., 2016b] Bocconi, S., Chioccariello, A., Dettori, G., Ferrari, A., Engelhardt, K., Kampylis, P., and Punie, Y. (2016b). Exploring the field of computational thinking as a 21st century skill. In *EDULEARN16 Proceedings*, 8th International Conference on Education and New Learning Technologies, pages 4725–4733. IATED.

[Boe et al., 2013] Boe, B., Hill, C., Len, M., Dreschler, G., Conrad, P., and Franklin, D. (2013). Hairball: Lint-inspired static analysis of Scratch projects. In *Proceeding of the 44th ACM Technical Symposium on Computer Science Education*, SIGCSE '13, pages 215–220, New York, NY, USA. ACM.

[Calao et al., 2015] Calao, L. A., Moreno-León, J., Correa, H. E., and Robles, G. (2015). Developing mathematical thinking with Scratch: An experiment with 6th grade students. In *Design for Teaching and Learning in a Networked World*, pages 17–27. Springer.

[Centro Nacional de Innovación e Investigación Educativa, 2017] Centro Nacional de Innovación e Investigación Educativa (2017). Informe sobre viabilidad de incorporar el pensamiento computacional como materia propia la secundaria obligatoria y materia transversal en el segundo ciclo de educación infantil y en educación primaria. Technical report, Ministerio de Educación, Cultura y Deporte.

[Furber, 2012] Furber, S. (2012). Shut down or restart? The way forward for computing in UK schools. Technical report, The Royal Society, London.

[Gander et al., 2013] Gander, W., Petit, A., Berry, G., Demo, B., Vahrenhold, J., McGettrick, A., Boyle, R., Mendelson, A., Stephenson, C., Ghezzi, C., et al. (2013). Informatics education: Europe cannot afford to miss the boat. Technical report, ACM.

[Garneli and Chorianopoulos, 2017] Garneli, V. and Chorianopoulos, K. (2017). Programming video games and simulations in science education: exploring computational thinking through code analysis. *Interactive Learning Environments*, pages 1–16.

[Grover and Pea, 2013] Grover, S. and Pea, R. (2013). Computational thinking in k–12. A review of the state of the field. *Educational Researcher*, 42(1):38–43.

[Halstead, 1977] Halstead, M. H. (1977). *Elements of software science*, volume 7. Elsevier New York.

[Hill and Monroy-Hernández, 2017] Hill, B. M. and Monroy-Hernández, A. (2017). A longitudinal dataset of five years of public activity in the scratch online community. *Scientific Data*, 4:170002.

[Igbokwe, 2015] Igbokwe, C. O. (2015). Recent curriculum reforms at the basic education level in Nigeria aimed at catching them young to create change. *American Journal of Educational Research*, 3(1):31–37.

[Info-communications Media Development Authority, 2014] Info-communications Media Development Authority, S. G. (2014). Code for Fun enrichment programme. Available at `https://portal.imda.gov.sg/Sub/Talent/Student-Programmes/Code-for-Fun`, accessed 2017-01-01.

[Kalelioglu et al., 2016] Kalelioglu, F., Gülbahar, Y., and Kukul, V. (2016). A framework for computational thinking based on a systematic research review. *Baltic Journal of Modern Computing*, 4(3):583.

[l'Académie des sciences, 2013] l'Académie des sciences (2013). L'enseignement de l'informatique en France - Il est urgent de ne plus attendre. Technical report, Institute de France.

[Lawanto, 2016] Lawanto, K. N. (2016). *Exploring trends in middle school students' computational thinking in the online Scratch community: A pilot study.* Utah State University.

[Lye and Koh, 2014] Lye, S. Y. and Koh, J. H. L. (2014). Review on teaching and learning of computational thinking through programming: What is next for k-12? *Computers in Human Behavior*, 41:51–61.

[Martin, 2016] Martin, S. A. (2016). *A Descriptive and Explorative Case Study of a Scratch Programming Experience Involving the Creation of a Lunar Simulation/Model with Grade Six Learners.* PhD thesis, University of Calgary (Canada).

[McCabe, 1976] McCabe, T. J. (1976). A complexity measure. *IEEE Transactions on software Engineering*, SE-2(4):308–320.

[Meseguer et al., 2015] Meseguer, P., Moreno, J., Moreno, J. J., Olco, K., Pimentel, E., Toro, M., Velázquez, Á., and Vendrell, E. (2015). Ensenanza de la informática en primaria, secundaria y bachillerato: estado español, 2015. Technical report, Sociedad Científica Informática de España, Conferencia de Directores y Decanos de Ingeniería Informática.

[Moreno and Robles, 2014] Moreno, J. and Robles, G. (2014). Automatic detection of bad programming habits in Scratch: A preliminary study. In *2014 IEEE Frontiers in Education Conference (FIE) Proceedings*, pages 1–4.

[Moreno-León and Robles, 2015] Moreno-León, J. and Robles, G. (2015). Computer programming as an educational tool in the English classroom: A preliminary study. In *Global Engineering Education Conference (EDUCON), 2015 IEEE*, pages 961–966. IEEE.

[Moreno-León and Robles, 2016] Moreno-León, J. and Robles, G. (2016). Code to learn with scratch? a systematic literature review. In *Global Engineering Education Conference (EDUCON), 2016 IEEE*, pages 150–156. IEEE.

[Moreno-León et al., 2015] Moreno-León, J., Robles, G., and Román-González, M. (2015). Dr. Scratch: Automatic analysis of scratch projects to assess and foster Computational Thinking. *RED. Revista de Educación a Distancia*, 15(46).

[Moreno-León et al., 2016a] Moreno-León, J., Robles, G., and Román-González, M. (2016a). Code to learn: Where does it belong in the k-12 curriculum? *Journal of Information Technology Education: Research*, 15:283–303.

[Moreno-León et al., 2016b] Moreno-León, J., Robles, G., and Román-González, M. (2016b). Comparing computational thinking development assessment scores with software complexity metrics. In *2016 IEEE Global Engineering Education Conference (EDUCON)*, pages 1040–1045.

[Moreno-León et al., 2016c] Moreno-León, J., Robles, G., and Román-González, M. (2016c). Examining the relationship between socialization and improved software development skills in the Scratch code learning environment. *J. UCS*, 22(12):1533–1557.

[Moreno-León et al., 2017] Moreno-León, J., Robles, G., and Román-González, M. (2017). Towards data-driven learning paths to develop computational thinking with Scratch. *IEEE Transactions on Emerging Topics in Computing*, PP(99):1–1.

[Moreno-León et al., 2017] Moreno-León, J., Román-González, M., Harteveld, C., and Robles, G. (2017). On the automatic assessment of computational thinking skills: A comparison with human experts. In *Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems*, CHI EA '17, pages 2788–2795, New York, NY, USA. ACM.

[Román-González, 2015] Román-González, M. (2015). Computational thinking test: Design guidelines and content validation. In *Proceedings of the 7th Annual International Conference on Education and New Learning Technologies (EDULEARN 2015)*, pages 2436–2444.

[Román-González et al., 2017] Román-González, M., Pérez-González, J.-C., Moreno-León, J., and Robles, G. (2017). Extending the nomological network of computational thinking with non-cognitive factors. *Computers in Human Behavior*.

[Schenker and Rumrill Jr, 2004] Schenker, J. D. and Rumrill Jr, P. D. (2004). Causal-comparative research designs. *Journal of vocational rehabilitation*, 21(3):117–121.

[Sentance, 2015] Sentance, S. (2015). Annual national survey 2015: Results. Technical report, Computing At School.

[Techapalokul and Tilevich, 2017] Techapalokul, P. and Tilevich, E. (2017). Understanding recurring software quality problems of novice programmers. Technical report, Department of Computer Science, Virginia Polytechnic Institute & State University.

[The White House, 2016] The White House (2016). Computer Science for All. Available at `https://www.whitehouse.gov/blog/2016/01/30/computer-science-all`, accessed 2017-01-01.

[von Wangenheim et al., 2017] von Wangenheim, C. G., Alves, N. C., Rodrigues, P. E., and Hauck, J. C. (2017). Teaching computing in a multidisciplinary way in social studies classes in school–a case study. *International Journal of Computer Science Education in Schools*, 1(2):3–16.

[Webber et al., 2014] Webber, C. G., Spindola, M. M., Otobelli, E. S., Giron, G. R., Dall, G., Poloni, L., Puziski, M., Padilha, R., do Prado, M. d. F. W., et al. (2014). Reflexões sobre o software scratch no ensino de ciências e matemática. *RENOTE*, 14(2).