Escuela Superior
de Ingeniería y Tecnología
Universidad de La Laguna

Trabajo de Fin de Grado

Grado en Ingeniería Informática

Análisis de los resultados de los sistemas de entrenamiento del Pensamiento Computacional

Analysis of the results of Computational Thinking training systems

Samuel Valcárcel Arce

La Laguna, 16 de junio de 2018

D[a]. **Coromoto Antonia León Hernández**, con N.I.F. 78.605.216-W profesora Titular de Universidad adscrita al Departamento de Lenguajes y Sistemas Informáticos de la Universidad de La Laguna, como tutora

D. **Carlos Segura González**, con N.I.F. 78.704.244-S profesor Titular adscrito al Área de Ciencias de la Computación del Centro de Investigación Matemática en Guanajuato, México, como cotutor

C E R T I F I C A N

Que la presente memoria titulada:

"*Análisis de los resultados de los sistemas de entrenamiento del Pensamiento Computacional*"

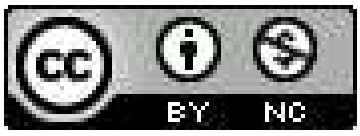
ha sido realizada bajo su dirección por D. **Samuel Valcárcel Arce**, con N.I.F. 54.063.506-M.

Y para que así conste, en cumplimiento de la legislación vigente y a los efectos oportunos firman la presente en La Laguna a 16 de junio de 2018

Agradecimientos

xxx
xxx
xxx
xxx

Licencia



Resumen

El objetivo de este trabajo ha sido .... bla, bla, bla bla, bla, bla bla, bla, bla

La competencia [E6], que figura en la guía docente, indica que en la memoria del trabajo se ha de incluir: antecedentes, problemática o estado del arte, objetivos, fases y desarrollo del proyecto, conclusiones, y líneas futuras.

Se ha incluido el apartado de 'Licencia' con todas las posibles licencias abiertas (Creative Commons). En el caso en que se decida hacer público el contenido de la memoria, habrá que elegir una de ellas (y borrar las demás). La decisión de hacer pública o no la memoria se indica en el momento de subir la memoria a la Sede Electrónica de la ULL, paso necesario en el proceso de presentación del TFG.

El documento de memoria debe tener un máximo de 50 páginas.

No se deben dejar páginas en blanco al comenzar un capítulo, ya que el documento no está pensado para se impreso sino visionado con un lector de PDFs.

También es recomendable márgenes pequeños ya que, al firmar digitalmente por la Sede, se coloca un marco alrededor del texto original.

El tipo de letra base ha de ser de 14ptos.

Palabras clave: Palabra reservada1, Palabra reservada2, ...

Abstract

Here should be the abstract in a foreing language...

Keywords: *Keyword1, Keyword2, Keyword3, ...*

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

En este apartado se detallan los diferentes motivos y objetivos a desarrollar, así como la metodología y el plan de trabajo llevado a cabo para la realización del Trabajo de Fin de Grado.

1.1. Motivación

La tecnología con el paso de los años ha ido avanzando a pasos agigantados, de manera que cada día sea más accesible a todas y cada una de las personas que hacen uso de ella en mayor o menor medida. Esto ayuda a que sectores externos a la informática, como la educación, avancen en materia de nuevos conocimientos, surgiendo así términos como el **Pensamiento Computacional**.

Podemos entender como pensamiento computacional la capacidad del ser humano para solucionar problemas, crear sistemas y entender de qué manera se comporta el ser humano a través del ámbito de la informática. Este tipo de "pensamiento"beneficia a los estudiantes de todas las edades, adquiriendo la capacidad de abstraer determinados problemas, que refuerzan y mejoran las habilidades intelectuales, permitiendo que se puedan aplicar en otros ámbitos.

En Internet existen diversas plataformas dedicadas a transmitir el concepto de pensamiento computacional en el ámbito educativo, como son Code.org[3], Codecademy[2], etc. En este trabajo nos hemos centrado en Code.org, una plataforma diseñada con la intención de permitir que este tipo de pensamiento sea divulgado tanto a alumnos de diferentes escuelas o institutos, en todas las partes del mundo, al igual que de un amplio rango de edades, de forma que desarrollen esta capacidad a través de las actividades que se disponen en la plataforma.

Debido a que en la aplicación mencionada anteriormente no se muestran los resultados de los tests realizados a los alumnos de manera gráfica, se ha

desarrollado una plataforma que simule la representación de dichos datos de los estudiantes. Esto permite que se manifieste visualmente en qué medida los alumnos están adquiriendo los conceptos que se intentan transmitir con los cursos o talleres propuestos en Code.org.

1.2. Objetivos

Puesto que Code.org ha sido la plataforma elegida para llevar a cabo el proyecto, el objetivo principal a conseguir es realizar una contribución a su página a través de Github, plataforma donde tienen alojada Code.org, de forma que los profesores no sólo tengan los resultados de los alumnos en los cursos impartidos, sino también la posibilidad de observarlos de manera gráfica, permitiendo ver el progreso de sus alumnos.

En caso de no conseguir dicho hito, se realizará un servicio externo, con la intención de plasmar los datos de los alumnos en gráficas representativas. Dicho servicio simulará la plataforma de Code.org, donde un profesor registrará a los estudiantes en los diferentes cursos o talleres, y podrá observar gráficos filtrados por edades, sexo, niveles completados, etc, complementando así el servicio que ofrece la plataforma de Code.org.

1.3. Metodología y plan de trabajo

Para desarrollar el proyecto, se ha seguido un plan de trabajo que podría dividirse en diferentes etapas:

- Una primera fase, en la que se ha realizado un estudio de los beneficios que supone el aprendizaje del pensamiento computacional en el sector educativo, a través del movimiento global **Horas de Código**[4], unos talleres impartidos a alumnos en un amplio rango de edad, con la intención de fomentar el aprendizaje del pensamiento computacional.

- A continuación, se procedió a valorar las principales páginas (Code.org, Codecademy, Programamos[5]) dedicadas a plasmar este tipo de pensamiento en sus actividades. En última instancia, se decidió que Code.org era la más adecuada para realizar la contribución a los resultados.

- Debido a que se actualizaba constantemente el repositorio de Code.org, con su consiguiente cambio en la estructura, se procedió a realizar una plataforma alternativa que intentara simular el funcionamiento de Code.org, con la particularidad añadida de la representación gráfica.

El resto de la memoria cuenta con un capítulo que aúna los antecedentes y estado actual del tema con respecto al pensamiento computacional, donde se abordará la explicación de algunas de las plataformas similares a la desarrollada para el proyecto. En el capítulo 3 se explican las diferentes tecnologías utilizadas para realizar el proyecto final, complementando así al capítulo 4, en el que se expone tanto la arquitectura como el funcionamiento en detalle de la aplicación (CodeCharts). A continuación, en el apartado 5, algunas de las pruebas y verificaciones usadas para el testeo de la plataforma

Capítulo 2

Antecedentes y estado actual del tema

Siguiendo el curso del campus de la biblioteca, haciendo uso de .Q, una herramienta de búsqueda de información de la Universidad de La Laguna, se realizó una búsqueda de artículos relacionados con el Pensamiento Computacional, algunos de ellos de la famosa revista ACM.

2.1. Estado del arte

Para explicar los antecedentes y el estado actual del tema en este proyecto se tendrán en cuenta las referencias citadas a continuación:

- En un artículo[8] publicado en 2017, *"Teacher Configurable Coding Challenges for Block Languages"*, se explica como una herramienta llamada **COPPER** (CustOmizable Puzzle Programming EnviRonment), desarrollada para crear puzzles de código en una cuadrícula usando lenguajes de programación basado en bloques, similar a los realizados en la plataforma Code.org "Hour of Code", tiene el potencial de incrementar el interés y el compromiso con el pensamiento computacional.

  De esta manera, se fomenta la forma de pensar computacionalmente a través de tareas que mejoren estas habilidades dentro un ambiente similar a un juego, normalmente fomentando que los estudiantes creen programas que dicten las acciones de un personaje a medida que avanza a través de un nivel de rompecabezas, similares a los de Code.org.

  Este proyecto busca aumentar la utilidad de este tipo de puzzles, permitiendo a los profesores personalizar los elementos visuales y mecánicos de los niveles que necesiten para impartir su contenido. Por ejemplo, un

docente que tenga que dar una lección de historia sobre Cristóbal Colón, podría diseñar un nivel donde el estudiante escribiría el código necesario para controlar un barco, que debe navegar a través del océano y las olas del mar para llegar a tierra. Este nivel se podría complementar con diferentes preguntas, formuladas ocurridos ciertos eventos a lo largo del nivel, para comprobar si el alumno está comprendiendo la lección.

- En 2015 la revista llamada **ACM INROADS**, publicó un artículo[9] mencionando la asociación entre la plataforma Code.org y el NSF (National Science Foundation), una agencia federal independiente creada por el Congreso de los Estados Unidos en 1950 para promover el progreso de la ciencia, la salud nacional y muchos otros aspectos relevantes para el país, con lo que muchos de los alumnos en colegios sin recursos o estudiantes de color tuvieran acceso a una educación, tanto secundaria como primaria, digna en las Ciencias de la Computación.

  El rol que adquiere Code.org en esta agrupación es mejorar las aptitudes adquiridas por los alumnos en el ámbito de las Ciencias de la Computación, mientras que NSF continuará apoyando la investigación de alta calidad en este sector.

- Por último, en un artículo[7] publicado en 2016, en el libro llamado "Proceedings of the 47th ACM Technical Symposium on Computer Science Education"se constata que las enseñanzas sobre la Ciencia de la Computación que se componen de actividades que usen la programación basada en bloques, como pueden ser con Scratch, Alice y las "Hour of Code"de Code.org, incentivan tanto a alumnos como profesores a indagar con más profundidad en el mundo del pensamiento computacional.

2.2. Aplicaciones relacionadas

Existen actualmente diversas plataformas dedicadas a fomentar el aprendizaje del pensamiento computacional, algunas de ellas mencionadas en puntos anteriores.

2.2.1. Code.org

Code.org es una organización no gubernamental creada en 2012 fundada por Hadi y Ali Partovi, que tiene como objetivo principal motivar a la gente, sobre todo a estudiantes de diferentes rangos de edades en colegios o institutos, a aprender sobre las Ciencias Computacionales. Es el organizador principal del

movimiento "Hora del Código", en el que se incentiva a todas las personas que lo realicen a que aprendan sobre el pensamiento computacional

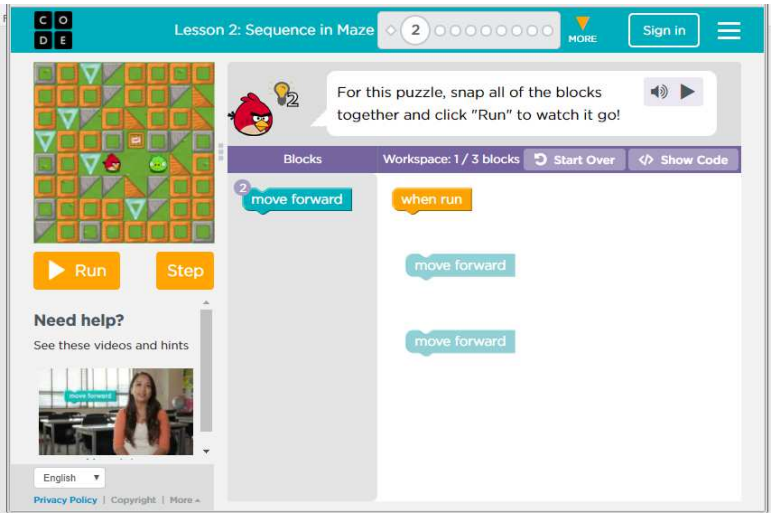


Figura 2.1: Code.org

En su web se encuentran una serie de cursos y actividades destinados a los alumnos que quieran aprender, organizados por temas y edades. Los profesores, a su vez, pueden aprender a crear talleres para enseñar a sus alumnos en la propia plataforma.

Como se explica en puntos anteriores, fue la plataforma elegida para desarrollar el Trabajo de Fin de Grado, debido a su amplia variedad de cursos a la hora de fomentar el aprendizaje del pensamiento computacional. Su página está destinada a que profesores enseñen a sus alumnos a través de actividades, mientras que las otras plataformas están enfocadas a aprender de manera autodidacta.

2.2.2. Codecademy

Codecademy es una empresa enfocada principalmente a la educación y, al igual que Code.org, es una de las plataformas online más relevantes dentro del ámbito de la enseñanza de la programación en la red. Ofrece múltiples cursos gratuitos, de manera que cualquier persona pueda aprender a programar en diversos lenguajes de programación.

Figura 2.2: Codecademy

Los cursos incluidos en la plataforma están estructurados de manera sencilla por temas (sintaxis, funciones, etc.). En caso de que el usuario tenga alguna

duda dentro de cada tema, se disponen numerosas clases para dominar los diferentes apartados.

2.2.3. Programamos

Otra asociación sin ánimo de lucro se llama Programamos, con el objetivo de desarrollar el pensamiento computacional de personas de diferentes edades (desde infantil hasta formación profesional), a través de la programación (en aplicaciones móviles o videojuegos).

Figura 2.3: Programamos

Algunos de los objetivos que persigue conseguir la plataforma son:

- Cambiar el modo de interactuar entre el usuario y la tecnología, para que pase de ser solo consumidor a creador.
- Desarrollar capacidades como la creatividad, imaginación, etc.
- Permitir al profesorado formarse, de manera que posteriormente pueda enseñar a programar a los alumnos.

Capítulo 3

Tecnología utilizada

Al realizar una búsqueda entre las diversas tecnologías que se encuentran en Internet para desarrollo web, en este capítulo se describen con detalle las usadas para realizar el Trabajo de Fin de Grado.

3.1. Ruby on Rails

El "Framework" en un proyecto se puede entender como el entorno de trabajo usado para desarrollar la aplicación web, es decir, una estructura dividido en capas que nos indica qué programas deben o pueden ser construidos y de qué manera se relacionan.

En este proyecto, he usado **Ruby on Rails**[6], un entorno Open Source (de código abierto) para el diseño de toda la plataforma. Rails fue lanzado en 2003 por David Heinemeier Hansson y desde ese momento se ha ido extendiendo entre la comunidad de programadores, consiguiendo así una extensa y gran comunidad.

Figura 3.1: Ruby on Rails

Alguna de las características que definen a este framework son:

- Usa el estilo de arquitectura MVC (Modelo Vista Controlador) que separa los datos, la lógica y la interfaz de la aplicación.

- Evita la repetición de código
- Framework con posibilidad de validación de formularios y manejo de sesiones.

3.2. Active Record

Para gestionar la base de datos de la aplicación, se ha optado por **Active Record**[1], incluido en Ruby on Rails. Active Record se podría definir como la "M" en la estructura "Modelo, Vista, Controlador", ya que es la capa responsable de representar los datos y la lógica de la aplicación.

Rails viene integrado por defecto con SQLite para el manejo de las bases de datos, debido a su sencillez y su uso sin necesidad de un servidor externo. Se utiliza tanto en el entorno de desarrollo (development) como el de pruebas (test) del proyecto. En el caso de que nuestro proyecto se use en el entorno de producción, no se recomienda el uso de SQLite, sino una aplicación más potente para el manejo de los datos.

Figura 3.2: SQLite

3.3. Bootstrap

Para el diseño front-end

Capítulo 4

CodeCharts

Los capítulos intermedios servirán para cubrir los siguientes aspectos: antecedentes, problemática o estado del arte, objetivos, fases y desarrollo del proyecto.

En el capítulo 1 se describió bla, bla, bla.....

Capítulo 5

Conclusiones y líneas futuras

Este capítulo es obligatorio. Toda memoria de Trabajo de Fin de Grado debe incluir unas conclusiones y unas líneas de trabajo futuro

Capítulo 6

Summary and Conclusions

This chapter is compulsory. The memory should include an extended summary and conclusions in english.

6.1. First Section

Capítulo 7

Presupuesto

Este capítulo es obligatorio. Toda memoria de Trabajo de Fin de Grado debe incluir un presupuesto.

7.1. Sección Uno

Tipos	**Descripcion**
AAAA	BBBB
CCCC	DDDD
EEEE	FFFF
GGGG	HHHH

Tabla 7.1: Tabla resumen de los Tipos

Apéndice A

Título del Apéndice 1

A.1. Algoritmo XXX

```
*****************************************************************************
*
* Fichero .h
*
*****************************************************************************
*
* AUTORES
*
*
* FECHA
*
*
* DESCRIPCION
*
*
*****************************************************************************/
```

A.2. Algoritmo YYY

```
/*****************************************************************************
 *
 * Fichero .h
 *
 *****************************************************************************
 *
 * AUTORES
 *
 * FECHA
 *
 * DESCRIPCION
 *
```

```
 *
 ******************************************************************************/
```

Apéndice B

Título del Apéndice 2

B.1. Otro apéndice: Sección 1

Texto

B.2. Otro apéndice: Sección 2

Texto

Bibliografía

[1] Active Record Basics. `http://guides.rubyonrails.org/active_record_basics.html`. Accedido en mayo de 2018.

[2] Codecademy. `https://www.codecademy.com/es`. Accedido en mayo de 2018.

[3] Code.org. `https://code.org/`. Accedido en mayo de 2018.

[4] Hour of Code. `https://hourofcode.com/`. Accedido en mayo de 2018.

[5] Programamos. `https://programamos.es/`. Accedido en mayo de 2018.

[6] Ruby on Rails. `https://rubyonrails.org/`. Accedido en mayo de 2018.

[7] Neil C.C. Brown, Jens Mönig, Anthony Bau, and David Weintrop. Panel: Future directions of block-based programming. In *Proceedings of the 47th ACM Technical Symposium on Computing Science Education*, SIGCSE '16, pages 315–316, New York, NY, USA, 2016. ACM.

[8] Nath Tumlin. Teacher configurable coding challenges for block languages. In *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education*, SIGCSE '17, pages 783–784, New York, NY, USA, 2017. ACM.

[9] Cameron Wilson. Hour of code: Bringing research to scale. *ACM Inroads*, 6(2):18–18, May 2015.