



**Escuela Superior
de Ingeniería y Tecnología**
Universidad de La Laguna

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

Analysis of the results of Computational Thinking training systems

Autor: Samuel Valcárcel Arce
Tutora: Coromoto León Hernández

Universidad de La Laguna

2 de julio de 2018

1 Motivación y Objetivos

1 Motivación y Objetivos

2 Tecnologías utilizadas

- 1 Motivación y Objetivos
- 2 Tecnologías utilizadas
- 3 Procedimiento experimental
 - Descripción de los experimentos
 - Descripción del material
 - Resultados obtenidos
 - Análisis de los resultados

- 1 Motivación y Objetivos
- 2 Tecnologías utilizadas
- 3 Procedimiento experimental
 - Descripción de los experimentos
 - Descripción del material
 - Resultados obtenidos
 - Análisis de los resultados
- 4 Conclusiones

Definición

*Podemos entender como **Pensamiento Computacional** la capacidad del ser humano para resolver problemas, crear sistemas y entender de qué manera se comporta el ser humano.*

Existen diversas plataformas dedicadas a divulgar de alguna forma este tipo de pensamiento:

- Code.org
- Programamos
- Codecademy

Hitos

- El objetivo principal del proyecto fue integrar en la plataforma una herramienta que facilitara al docente observar de manera gráfica el progreso de sus alumnos en los cursos impartidos.

Problemas

- No se supo como acceder a la **base de datos** de la plataforma de Code.org, junto con la constante actualización de su estructura de su página, hizo imposible su integración

- Para solventar el problema del acceso a la base de datos de la página de Code.org, se optó por diseñar una aplicación que simulara la visualización de los resultados de los alumnos.

Plataforma

Algunas de las características que conforman la aplicación son:

- Posibilidad de crear cursos y secciones de los mismo para la realización de las actividades.
- Registrar a los diferentes alumnos en los talleres.
- Visualización de diferentes gráficas representativas de los resultados.

Ruby on Rails, un entorno de código abierto, para el diseño de toda la plataforma. Se podría definir como:

- Usa el estilo de arquitectura Modelo-Vista-Controlador (MVC), que separa los datos, la lógica y la interfaz de la aplicación.
- Evita la repetición de código.
- Permite el manejo de sesiones y de formularios.



Para gestionar la base de datos de la plataforma, se ha optado por **Active Record**, incluido en Ruby on Rails. Se podría definir como la M en el patrón "Modelo, Vista, Controlador"

- Está preparado para su uso en el entorno de desarrollo (development) y de pruebas (test).
- Para el ámbito de producción, se recomienda un software más potente.



El diseño *front-end* de la aplicación se realizó con **Bootstrap**, un framework de código abierto, que hace uso de HTML, CSS y Javascript, de manera que el usuario pueda usarlo como base para el diseño *responsive* de la plataforma.



Github es una plataforma de desarrollo en la que cualquier desarrollador puede alojar su proyecto de manera cómoda y sencilla.

Se caracteriza por:

- Colaboración entre los desarrolladores de un proyecto dentro de un mismo repositorio.
- Control de versiones, manejo de ramas, etc.
- Permite la posibilidad de usarlo tanto por línea de comandos, como por su interfaz gráfica de usuario.

GitHub

Ha de contar con secciones para la descripción de los experimentos y del material. También deber haber una sección para los resultados obtenidos y una última de análisis de los resultados obtenidos.

Ejemplo

- *Con semilla 1*

Ejemplo

- *Con semilla 1*
- *Con semilla 10*

Ejemplo

- *Con semilla 1*
- *Con semilla 10*
- *Sin semilla*

Ejemplo

- 1 *Descripción del hardware*

Ejemplo

- 1 *Descripción del hardware*
- 2 *Descripción del software*

Medidas de tiempo y Velocidad

| Tiempo (± 0.001 s) | Velocidad (± 0.1 m/s) |
|--|---|
| 1.234 | 67.8 |
| 2.345 | 78.9 |
| 3.456 | 89.1 |
| 4.567 | 91.2 |

Cuadro: Resultados experimentales de tiempo (s) y velocidad (m/s)

Diagrama del tiempo y la velocidad

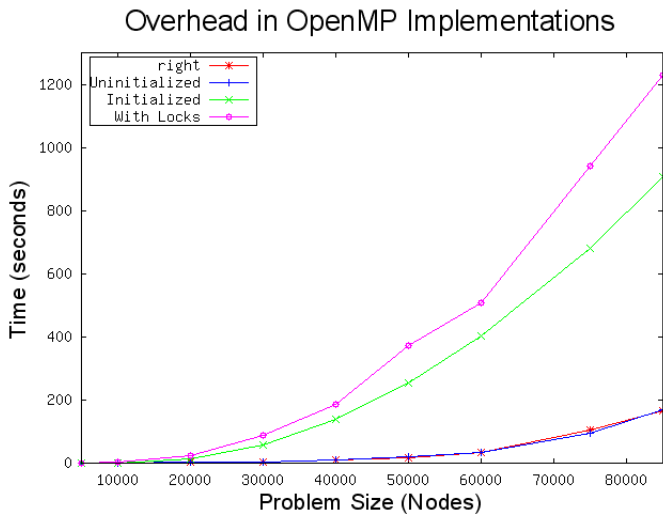


Figura: Ejemplo de figura

Ejemplo

❶ *Conclusión 1*

Ejemplo





- 1 *Conclusión 1*
- 2 *Conclusión 2*

Bibliografía I

-  Code.org. <https://code.org/>
-  Hour of Code. <https://hourofcode.com/>
-  Codecademy. <https://www.codecademy.com/es>
-  Programamos. <https://programamos.es/>
-  Ruby on Rails. <https://rubyonrails.org/>
-  Active Record Basics. http://guides.rubyonrails.org/active_record_basics.html
-  Bootstrap. <https://getbootstrap.com/>
-  Chartkick. <https://www.chartkick.com/>
-  Github. <https://github.com/>
-  RSpec. <http://rspec.info/>

Bibliografía II

-  PuntoQ. http://www.bbtck.ull.es/view/institucional/bbtck/Biblioteca_Digital/es
-  ACM. <https://www.acm.org/publications/magazines>
-  IEEE. <https://www.ieee.org/publications/periodicals.html>
-  SQLite. <https://rubygems.org/gems/sqlite3/versions/1.3.11?locale=es>
-  Devise. <https://rubygems.org/gems/devise>
-  Gem Bootstrap. <https://rubygems.org/gems/bootstrap>
-  Will-paginate. https://rubygems.org/gems/will_paginate
-  Wicked-PDF. https://rubygems.org/gems/wicked_pdf
-  JQuery-Rails. <https://rubygems.org/gems/jquery-rails>

-  Wilson, Cameron. *Hour of Code: Bringing Research to Scale* <http://doi.acm.org.accedys2.bbtck.ull.es/10.1145/2746406>
-  M. Wing. *COMMUNICATIONS OF THE ACM March* <https://www.cs.cmu.edu/~CompThink/papers/Wing06.pdf>
-  Tumlin, Nath. *Teacher Configurable Coding Challenges for Block Languages* <http://doi.acm.org.accedys2.bbtck.ull.es/10.1145/3017680.3022467>
-  Brown, Neil C.C. and Monig, Jens and Bau, Anthony and Weintrop, David. *Panel: Future Directions of Block-based Programming* <http://doi.acm.org.accedys2.bbtck.ull.es/10.1145/2839509.2844661>