



**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  
Co-tutor: Carlos Segura González

Universidad de La Laguna

11 de julio de 2018

## 1 Motivación

- Code.org
- Estructura de repositorio de Code.org

## 1 Motivación

- Code.org
- Estructura de repositorio de Code.org

## 2 Objetivos

- 1 Motivación
  - Code.org
  - Estructura de repositorio de Code.org
- 2 Objetivos
- 3 Tecnologías utilizadas

- 1 Motivación
  - Code.org
  - Estructura de repositorio de Code.org
- 2 Objetivos
- 3 Tecnologías utilizadas
- 4 Plataforma CodeCharts
  - Arquitectura de la aplicación
  - Diseño de la aplicación

- 1 Motivación
  - Code.org
  - Estructura de repositorio de Code.org
- 2 Objetivos
- 3 Tecnologías utilizadas
- 4 Plataforma CodeCharts
  - Arquitectura de la aplicación
  - Diseño de la aplicación
- 5 Pruebas y verificación

- 1 Motivación
  - Code.org
  - Estructura de repositorio de Code.org
- 2 Objetivos
- 3 Tecnologías utilizadas
- 4 Plataforma CodeCharts
  - Arquitectura de la aplicación
  - Diseño de la aplicación
- 5 Pruebas y verificación
- 6 Summary and conclusions

## 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 a través de los conceptos fundamentales de la informática.*

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

- Code.org
- Programamos
- Codecademy



## Code.org

Code.org es una empresa, que posee una plataforma diseñada para fomentar el aprendizaje del Pensamiento Computacional entre alumnos y profesores.

Algunas de las **características** que conforman esta aplicación son:

- Posibilidad de crear actividades a medida para un alto rango de edades de alumnos.
- Permite al profesor formarse, a través de cursos, para posteriormente transmitir ese conocimiento a los estudiantes.

# Estructura de repositorio de Code.org



Code.org

- **Dashboard:** dedicado a alojar su plataforma de aprendizaje de Code studio, una aplicación desarrollada en Ruby on Rails, en la que se encuentran sus cursos, tutoriales, cuentas de usuario, etc.
- **Pegasus:** en la que podemos ubicar el servidor en sí de Code.org, una aplicación desarrollada en Sinatra responsable de aspectos como code.org, hourofcode.com, el dashboard del profesor, etc
- **Apps:** donde se encuentra toda el motor usado para los puzzles, tutoriales y herramientas online disponibles para los alumnos.

## Hitos

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

## Problemas

- Al no saber como acceder a la **base de datos** de la plataforma de Code.org, junto con la constante actualización de la estructura de su página en Github, hizo imposible su integración.

## Solución

- 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 la conforman 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:

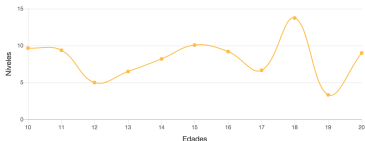
- Colaboración entre los desarrolladores de un proyecto dentro de un mismo repositorio.
- Control de versiones, manejo de ramas, etc.
- Permite la posibilidad de usarlo tanto por línea de comandos, como por su interfaz gráfica de usuario.

# GitHub



# Tecnologías utilizadas

Para la visualización de los resultados de los cursos de manera gráfica, se usó la herramienta **Chartkick**, por su compatibilidad con Ruby (y otros lenguajes como Python, Javascript, etc) y su variedad de representaciones.



La plataforma **CodeCharts** se ha diseñado con la intención de simular el funcionamiento de Code.org, de manera que se aporte una representación gráfica a los resultados de los cursos.

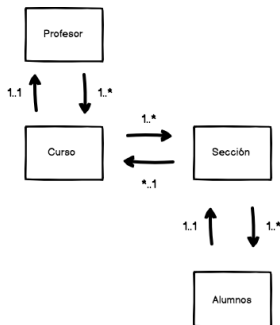
## Características

- Creación de cursos y sus consiguientes secciones (lugares donde se celebra dicho curso).
- Gestión de los alumnos dentro de cada sección.
- Representación gráfica de los resultados de los talleres/cursos.
- Descarga de informe con las representaciones correspondientes a ese curso o sección.

# Arquitectura de la aplicación

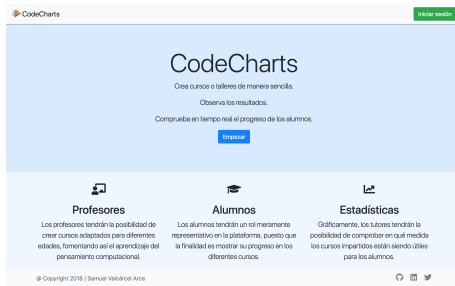
CodeCharts fue desarrollado en Ruby on Rails, como se menciona en diapositivas anteriores, siguiendo la estructura MVC (Modelo-Vista-Controlador).

Desde un principio se usó una base de datos relacional, que nos proporcionaba Rails con ActiveRecord y SQLite.



# Diseño de la aplicación

La primera vez que se accede a la aplicación, el profesor, que será el que se registre en la aplicación, verá una descripción de en qué consiste la plataforma y lo que ofrece al profesorado en los cursos impartidos.



# Diseño de la aplicación

El usuario tendrá tanto la posibilidad de crearse una cuenta y, en caso de estar registrado previamente, podrá iniciar sesión con la misma.

### Crear cuenta

Nombre

Apellidos

Correo electrónico

Contraseña

La contraseña debe tener al menos 6 caracteres.

Confirmar contraseña

[Regístrame](#)

Ya tengo cuenta, [entrar](#)

### Iniciar sesión

Correo electrónico

Contraseña

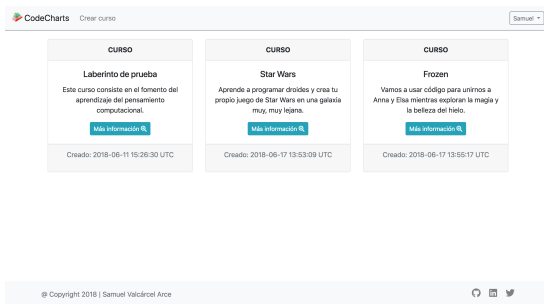
☐ Recuérdame

[Iniciar sesión](#)

Aún no estoy registrado, [crear cuenta](#)

# Diseño de la aplicación

Una vez que el profesor haya accedido a la plataforma, tendrá a su disposición los cursos que ha creado previamente en forma de cuadrícula, con su información asociada.



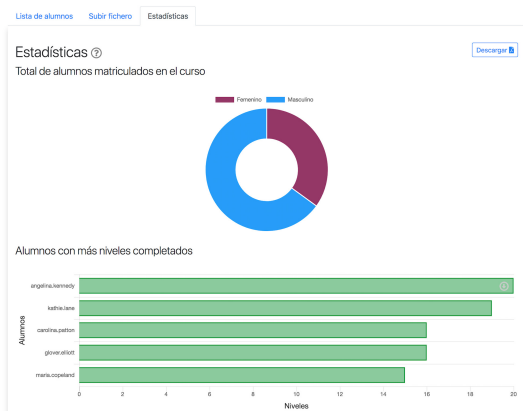
# Diseño de la aplicación

Los detalles del curso, junto con las secciones impartidas del mismo, se pueden observar en los detalles del curso.

The screenshot displays the 'CodeCharts' application interface. At the top, there is a header with the 'CodeCharts' logo, the text 'Mis cursos', and a user profile dropdown labeled 'Samuel'. The main content area is divided into two columns. The left column, titled 'Detalles del curso', contains the following information: 'Título: Laberinto de prueba', 'Descripción: Este curso consiste en el fomento del aprendizaje del pensamiento computacional.', 'Referencia: 7.25', and 'Creado por: Samuel'. At the bottom of this column are two buttons: 'Editar' (with a pencil icon) and 'Añadir sección+'. The right column, titled 'Secciones', lists three sections: 'Arona', 'La Laguna', and 'Granadilla'. Each section has two buttons: 'Administrar alumnos' (green) and 'Eliminar' (red). Below the 'Secciones' list is a section titled 'Resultados generales' with the text 'Total de alumnos matriculados en el curso'.

# Diseño de la aplicación

La representación gráfica de los resultados de los cursos/talleres estarán disponibles tanto para observarlos en la plataforma, como para descargar en formato PDF informe con los mismos.





# Pruebas y verificación

Para llevar a cabo el "testeo" de la plataforma, se usó RSpec, una herramienta utilizada con más frecuencia en el entorno de producción. Permite la verificación de diversas pruebas en modelos, controladores, etc.

```
RSpec.describe Course, :type => :model do
  subject { described_class.new }

  it "No se valida sin valor de referencia" do
    subject.value_reference = "Prueba de valor de referencia"
    expect(subject).not_to be_valid
  end

  it "Tiene uno o más usuarios en las secciones de ese curso" do
    should have_many(:users).through(:sections)
  end

  it "Se pueden añadir cursos" do
    @new_course = Course.new(title: "Titulo", body: "Descripción", value_reference: "18")
    @new_course.save
    expect(Course.find_by(title: "Titulo")).to eq(@new_course)
  end
end
```

```
RSpec.describe Section, :type => :model do
  subject { described_class.new }

  it "Se pueden añadir nuevos usuarios desde fichero" do
    @new_user = User.new(first_name: "Samuel", last_name: "Valcárcel", gender: "Chico", age: "23",
                        username: "cosaca", total_lines: "48", completed_levels: "28", section_id: "1",
                        user_value_reference: "9.5")
    @new_user.save
    expect(User.find_by(first_name: "Samuel")).to eq(@new_user)
  end

  it "Se pueden eliminar usuarios" do
    @new_user = User.new(first_name: "Samuel", last_name: "Valcárcel", gender: "Chico", age: "23",
                        username: "cosaca", total_lines: "48", completed_levels: "28", section_id: "1",
                        user_value_reference: "9.5")
    @new_user.destroy
  end

  it "Se pueden añadir secciones" do
    @new_section = Section.new(name: "Nombre")
    @new_section.save
    expect(Section.find_by(name: "Nombre")).to eq(@new_section)
  end
end
```

## Tests

Se realizaron las pruebas sobre algunos de los **modelos** de la aplicación (Curso, Secciones y Usuarios), que se comprueba que se pasan con éxito, como se observa en la figura.

```
[samuel@MacBook-Pro-de-Samuel:~/Desktop/Memoria/TFG/code$ rspec -fd

Course
  Se comprueba con atributos válidos
  No se valida sin título
  No se valida sin descripción
  No se valida sin valor de referencia
  Tiene uno o más usuarios en las secciones de ese curso
  Se pueden añadir cursos

Section
  Se comprueba con atributos válidos
  No se valida sin título
  Pertenece a un curso
  Debe tener uno o más usuarios
  Se pueden añadir nuevos usuarios desde fichero
  Se pueden eliminar usuarios
  No se validan los usuarios sin todos los atributos
  La edad debe ser en números
  Las líneas de código deben ser en números
  Los niveles completados deben ser en números
  Se pueden añadir secciones

User
  Se pueden añadir nuevos usuarios manualmente

Finished in 0.0781 seconds (files took 1.18 seconds to load)
18 examples, 0 failures
```

## Conclusions







- 1 *The developed platform, called CodeCharts, aims to make it easier for teachers to visualize the progress of their students, so that once the students have made the challenges, they can see if the courses or workshops are helping in the future promoting computational thinking when facing activities.*

## Conclusions

- 1 *The developed platform, called CodeCharts, aims to make it easier for teachers to visualize the progress of their students, so that once the students have made the challenges, they can see if the courses or workshops are helping in the future promoting computational thinking when facing activities.*
- 2 *One of the features of this application is that it is designed so that it is only used by the teaching staff, for that reason it has been developed for its use in the most simple and comfortable way.*

## Future improvements

- ① Creating an account for both teachers and students. As it is currently designed, it is designed so that only teachers are the only one able to gather all the information of the courses.
- ② Data stored in Code.org. As detailed in previous points, the main idea was to implement statistics on the Code.org platform. By failing to do this, the results of the workshops carried out on the Code.org platform could be obtained, so that the teacher can insert them directly into CodeCharts.

-  Code.org. <https://code.org/>
-  Chartkick. <https://www.chartkick.com/>
-  RSpec. <http://rspec.info/>
-  Wilson, Cameron. *Hour of Code: Bringing Research to Scale*  
<http://doi.acm.org.accedys2.bbtck.u11.es/10.1145/2746406>
-  Tumlin, Nath. *Teacher Configurable Coding Challenges for Block Languages* <http://doi.acm.org.accedys2.bbtck.u11.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.u11.es/10.1145/2839509.2844661>