Documentación oficial de CEFootV.4

Rodrigo Arce[1,†] **and David García**[2,†]

[1]*Ingeniería en Computadores, Tecnológico de Costa Rica*
[2]*Ingeniería en Computadores, Tecnológico de Costa Rica*
[†]*Todos los autores contribuyeron de manera igual al trabajo*

Compilado el 7 de mayo, 2024

Abstract

Internet of Things es un concepto que se ha popularizado en los últimos años. Este proyecto tiene como objetivo ponerlo en práctica mediante una Raspberry Pi Pico W. La idea del proyecto es crear una maqueta de un estadio de fútbol. Será un partido de penales, el primer equipo que meta 3 goles o que tenga más goles que el rival, gana. La detección de los goles funciona mediante la pulsación de uno de los 6 botones que están colocados detrás de la portería. El Raspberry Pi Pico W se va a conectar a una PC y va a enviar y recibir señales mediante un servidor HTTP. Se concluyó que, aunque el servidor HTTP funcionara la mayoría de veces, este suele presentar problemas sin solución aparente.

Keywords: Electrónica, Computador, Circuitos electrónicos, Servidor, HTTP, Raspberry pi pico w

Corresponding author: Estudiantes de Ingeniería en Computadores del Tecnológico de Costa Rica. *E-mail address:* r.arce.1@estudiantec.cr[1], d.garcia.1@estudiantec.cr[2].
URL: https://github.com/lVoidi/CEFootV.4

INTRODUCCIÓN

Internet of Things es un concepto que se ha popularizado en los últimos años. La posibilidad de poder desarrollar sistemas que se conectan entre sí, sin la necesidad de un cable de por medio ha abierto muchas posibilidades en el mundo de la tecnología. El presente proyecto busca ejemplificar el concepto de Internet of Things mediante una Raspberry Pi Pico W. Este dispositivo tiene la capacidad de conectarse mediante Wi-Fi, lo que permite desarrollar este proyecto.

REGLAS DEL JUEGO

Este mismo trata de un juego que busca simular una serie de penales mediante un circuito simple. Dos equipos van a competir en una tanda de cinco penales, el primer equipo que meta tres goles o meta más goles que el enemigo, es el ganador. El estadio va a ser una maqueta, y la clave del juego está en la cancha. La cancha va a contar con seis paletas. La idea es que cada paleta esté asociada a un boton que va a ser accionado por una bola.

FRONT-END Y BACK-END DEL PROYECTO

Para este juego, se desarrolló una interfaz gráfica hecha con Tkinter. La idea del juego es conectar la Raspberry Pi Pico W con la computadora mediante un servidor HTTP desarrollado con socket. La interfaz cuenta con la posibilidad de elegir visualmente entre tres equipos. Cada equipo cuenta con 6 jugadores, los cuales pueden ser elegidos tanto como artilleros como porteros a voluntad. La selección funciona mediante peticiones POST, las cuales son enviadas a la Raspberry Pi Pico W y se transforman en una de las tres señales: SIGTEAM, SIGPOT y SIGGOAL. Cuando se acciona la señal SIGTEAM, va a ejecutar un código que va a escuchar al botón de cambio de equipo y va a cambiar el color del led asociado.

```
selected = False
while not selected:
    print(selected)
    if change_player_signal.value() and
    current_team_playing == "blue":
        current_team_playing = "red"
        selected = True
        blue_team.value(0)
        red_team.value(1)
    elif change_player_signal.value() and
    current_team_playing == "red":
        current_team_playing = "blue"
        selected = True
        blue_team.value(1)
        red_team.value(0)
```

Code 1. Muestra del código de la señal SIGTEAM.

Cabe destacar que, al conectar la Raspberry Pi a la batería, esta va a encender por defecto la luz azul. Esta representa al equipo local, mientras que la luz roja va a ser el equipo visitante. Esto no significa que el equipo local sea siempre fijo. A nivel de código, el equipo local se elige mediante una función de números pseudoaleatorios en el código de cada una de las interfaces. La siguiente señal a analizar es la señal SIGPOT. Esta es la señal más simple de todas, pues lo que va a hacer es enviar un valor del potenciómetro que va a estar comprendido entre 0 y 6. Y por último, está la señal SIGGOAL. Esta señal va a esperar a que una de las 6 paletas sea activada, y además va a crear el algoritmo de anotación.

```
anotation_algorithm = random.randint(1, 3)

#  ndices  de las paletas en las que est  el
    portero
index_list = []
if anotation_algorithm == 1:
    goalkeeper_index = random.choice(goal)
    index_list = [goalkeeper_index,
    goalkeeper_index + 1] if goalkeeper_index +
    1 < len(goal) else [goalkeeper_index - 1,
    goalkeeper_index]
elif anotation_algorithm == 2:
    goalkeeper_index = random.choice(goal)
    index_list = []
    if goalkeeper_index == len(goal) - 1:
        index_list = [goalkeeper_index-2,
    goalkeeper_index-1, goalkeeper_index]
    elif goalkeeper_index == 0:
        index_list = [goalkeeper_index,
    goalkeeper_index + 1, goalkeeper_index + 2]
    else:
        index_list = [goalkeeper_index-1,
    goalkeeper_index, goalkeeper_index + 1]
else:
    group = random.randint(0, 1)
    index_list = []
    if group == 1:
```

```
21          index_list = list(filter(lambda x: (x %
        2 == 0), goal))
22      else:
23          index_list = list(filter(lambda x: (x %
        2 == 1), goal))
```

Code 2. Muestra del código de la señal SIGGOAL.

Hay tres maneras distintas en las que el programa escoge el portero. La primera manera que se planteó es que agarre un par de las paletas, la segunda manera es que agarre un trío de paletas juntas y la última manera es que agarre uno de los grupos de las paletas intercaladas. La selección de este algoritmo es pseudo-aleatoria y no es visible para el jugador, pues sucede del lado del servidor. Todos estos datos son enviados como respuesta a la solicitud enviada por el cliente a la Raspberry Pi.

```
addr = socket.getaddrinfo('0.0.0.0', 8080)
    [0][-1]
s = socket.socket()
s.bind(addr)
s.listen(1)
s.settimeout(100)
print(f"[DEBUG] Escuchando al puerto 8080, en {
    addr}")
```

Code 3. Creación del servidor.

Después de crear con éxito el socket, podemos recibir señales por parte de un cliente. Al recibir las señales y ejecutar las instrucciones designadas, se envía la señal mediante la función send_code.

```
def send_code(code, client, address, timeout=0):
    if timeout > 5:
        cl_file = client.makefile('r')
        while True:
            line = cl_file.readline()
            if not line or line == b'\r\n':
                break
    print(f'enviando {code}')
    client.send('HTTP/1.0 200 OK\r\n\r\n')
    print('enviando dato')
    client.sendall(code.encode())
    print('cerrando')
    client.close()
```

Code 4. Código para enviar código al cliente.

DIFICULTADES ENCONTRADAS

Aunque la Raspberry Pi Pico W sea un dispositivo muy útil, puede presentar problemas a la hora de armar un servidor en el mismo. La Raspberry Pi presentó problemas a la hora de recibir las señales. Específicamente, la señal SIGGOAL causó problemas, ya que el output solía ser el mismo: ConnectionResetError. Al parecer, este error no tiene solución bajo la lógica del circuito y del código. Este error sucede por ratos totalmente aleatorios y con las modificaciones actuales del código, este error tiene pocas posibilidades de ocurrir. Aún así, la solución temporal que se encontró a día 7 de mayo de 2024 es hacer un trycatch omitiendo el error, obligando al jugador a tirar la bola de nuevo.

```
try:
    # (...) Codigo de la interfaz
except Exception as e:
    print("Excepci n: ", e, ": Omitiendo")
```

Code 5. Try catch que evita los errores irremediables de lado servidor.

CONCLUSIONES

Se logró desarrollar un ambiente simulado mediante circuitos electrónicos y bajo el concepto de Internet of Things. La Raspberry Pi Pico W ha demostrado ser un dispositivo útil que tiene capacidad de montar un servidor simple para la facilitación de algunas tareas, como la conexión PC-Maqueta. Se ha desarrollado, con éxito, un servidor HTTP capaz de recibir peticiones POST y de responder a ellas en la mayoría de veces. A su vez, se ha logrado integrar este backend con una interfaz gráfica en Tkinter, la cual lleva el marcador del juego y tiene la capacidad de seleccionar al portero y al artillero para cada tiro.