

UNIVERSIDAD DE TARAPACÁ



FACULTAD DE INGENIERÍA  
DEPARTAMENTO DE INGENIERÍA CIVIL EN COMPUTACIÓN E  
INFORMÁTICA



Plan de Proyecto

“EV3 Ball-E”

**Integrantes:** André Guerra,  
Alonso Kalise,  
Christopher Romo,  
Diego Pizarro,  
Fernando Díaz

**Asignatura:** Proyecto I

**Profesor:** Humberto Urrutia

Noviembre - 2024

## 1.- Historial de Cambios

Fecha	Versión	Descripción	Autor(es)
12/09/2024	1.0	Creación y formulación del proyecto	-André Guerra -Alonso Kalise -Christopher Romo -Diego Pizarro -Fernando Díaz
15/09/2024	1.5	Cambio de diseño de robot	-André Guerra -Alonso Kalise -Christopher Romo -Diego Pizarro -Fernando Díaz
17/09/2024	2.0	Reorganización de roles	-André Guerra -Alonso Kalise -Christopher Romo -Diego Pizarro -Fernando Díaz
24/09/2024	2.5	Primeras funciones del robot	-André Guerra -Alonso Kalise -Christopher Romo -Diego Pizarro -Fernando Díaz
20/10/2024	3.0	Desarrollo de la interfaz de la aplicación del robot	-André Guerra -Alonso Kalise -Christopher Romo -Diego Pizarro -Fernando Díaz
27/10/2024	3.5	Finalización del diseño de la interfaz, Desarrollo del funcionamiento de la interfaz	-André Guerra -Alonso Kalise -Christopher Romo -Diego Pizarro -Fernando Díaz
03/11/2024	4.0	Realización de la funcionalidad del robot	-André Guerra -Alonso Kalise -Christopher Romo -Diego Pizarro -Fernando Díaz
12/11/2024	5.0	Conexiones Exitosas	-André Guerra -Alonso Kalise -Christopher Romo -Diego Pizarro -Fernando Díaz

## Tabla de Contenidos

1. Panel General .....	4
1.1. Introducción .....	4
1.2. Objetivos .....	4
1.2.1. Objetivo General .....	4
1.2.2. Objetivos Específicos .....	4
1.3. Restricciones .....	5
1.4. Entregables .....	5
2. Organización del Personal .....	6
2.1. Descripción de los Roles .....	6
2.2. Personal que Cumplirá los Roles .....	6
2.3. Métodos de Comunicación .....	6
3. Planificación del Proyecto .....	7
3.1. Actividades .....	7
3.2. Carta Gantt .....	9
3.3. Gestión de Riesgos .....	10
4. Planificación de los Recursos .....	12
4.1. Hardware.....	12
4.2. Software .....	12
4.3. Estimación de Costos .....	12
5. Análisis – Diseño .....	14
5.1. Especificación de requerimientos .....	14
5.2. Arquitectura .....	14
5.3. Interfaz .....	15
6. Implementación .....	15
6.1. Fundamentos de proyectiles.....	15
6.2. Descripción de los programas .....	17
6.3. Diagramas .....	24
7. Resultados.....	24
7.1. Estado actual del proyecto .....	24
7.2. Problemas encontrados y soluciones propuestas .....	25
8. Conclusión .....	27
9. Referencias .....	28

# **1. Panorama General**

## **1.1 Introducción:**

El proyecto "Ball-E", desarrollado durante la asignatura Proyecto I, es un robot creado a partir de piezas de la empresa LEGO® MINDSTORMS® EV3.

Para realizar este proyecto, se requerirán todos los conocimientos adquiridos en la formación de los integrantes del grupo, como la aplicación de los conceptos enseñados en los cursos de Taller de Programación I y II.

## **1.2 Objetivos:**

### **1.2.1. Objetivo General:**

Planear, desarrollar y construir un robot EV3 de carga que permita al usuario mover una pelota de un lado a otro, mediante un programa hecho en Python.

### **1.2.2. Objetivo Específico:**

Aplicar los conocimientos adquiridos en taller de programación I y II, para abordar los desafíos específicos de este proyecto.

Identificar y seleccionar los componentes del kit LEGO® MINDSTORMS® EV3.

Diseñar un programa que permita al usuario controlar al robot.

## **1.3. Restricciones:**

- La fecha de entrega del primer informe es el 5 de septiembre.
- El equipo debe tener un máximo de 5 personas.
- La finalidad es que el robot pueda agarrar la pelota y moverse con ella.
- Se debe usar solo un sistema operativo basado en linux.
- Se debe usar el lenguaje de programación python.
- Se debe documentar todo el proyecto en la plataforma Redmine.
- Se debe tener una conexión inalámbrica entre el robot y el notebook.

## **1.4. Entregables:**

- Bitácoras semanales del avance realizado.
- Carta Gantt del detalle del proceso completo del proyecto.
- Informe y presentaciones.
- Robot "EV3 Ball-E".
- Wiki.
- Manual de usuario.

## 2. Organización del Personal

Se le designó una responsabilidad a cada miembro donde estarán a cargo de cumplir con el trabajo asignado.

### 2.1. Descripción de los roles.

ROL	DESCRIPCIÓN	ASIGNACIONES
Programador(es)	Encargado de crear, diseñar y perfeccionar el código con el que funcionará el robot.	-André Guerra -Diego Pizarro -Fernando Díaz
Ensamblador(es)	Encargado de armar el robot de forma que funcione óptimamente y cumpla con los objetivos señalados.	- Alonso Kalise - Christopher Romo
Escritor(es)	Encargado de realizar las bitácoras e informes de avances del equipo	- André Guerra - Diego Pizarro - Fernando Díaz
Fotógrafo(s)	Encargado de fotografiar y grabar los avances del equipo.	- Alonso Kalise - Christopher Romo
Documentador(es)	Encargado de ingresar los avances realizados por el equipo a la plataforma "Redmine".	- André Guerra - Diego Pizarro - Fernando Díaz

### 2.2. Mecanismos de comunicación:

Los dos principales mecanismos de comunicación serán Whatsapp y Discord. Whatsapp se usará para agendar reuniones y resolver dudas con respecto al proyecto. Mientras que discord sirve para organizar al grupo a la hora de realizar la bitácora, presentaciones, trabajos y otras tareas.

### 3. Planificación del Proyecto

Actividad	Descripción	Responsable	Involucrados
Encargado de bitácoras	Registro de todas las actividades que se desarrollan semanalmente.	André Guerra	- André Guerra - Diego Pizarro
Videos y fotos	Registro visual de los avances del proyecto.	Christopher Romo	- Alonso Kalise - Christopher Romo
Redacción de Carta Gantt	Planificación de las actividades a lo largo del semestre.	Diego Pizarro	- Diego Pizarro
Wiki	Se capturan y comparten ideas e información del proyecto.	Diego Pizarro	- Diego Pizarro - Fernando Díaz
Organización	Designación de la actividad que estará encargada cada integrante.	André Guerra	-André Guerra - Alonso Kalise -Christopher Romo -Diego Pizarro -Fernando Díaz
Contabilizar Piezas	Entrega del kit EV3 donde se contabilizaron las piezas.	Diego Pizarro	- André Guerra - Alonso Kalise - Diego Pizarro
Búsqueda de Ideas	Indagación de ideas que se podrían llevar a cabo	Fernando Díaz	- Alonso Kalise - Christopher Romo - Fernando Díaz
Construcción del Robot	Armado de la base del robot.	Alonso Kalise	- Alonso Kalise - Christopher Romo

### 3.1. Actividades

Instalar SO	Instalación de sistema operativo	André Guerra	- André Guerra
Administrar Redmine	Subir y organizar documentos en Redmine	Diego Pizarro	- André Guerra - Diego Pizarro - Fernando Díaz
Informe I	Creación de informe I	Diego Pizarro	- Diego Pizarro - Fernando Díaz
Estimación de Costos	Calcular costo total del proyecto	Diego Pizarro	- Diego Pizarro - Fernando Díaz
Presentación I	Creación de Presentación I	André Guerra	- André Guerra
Pruebas de Código	Programación del robot	Fernando Díaz	- André Guerra - Diego Pizarro - Fernando Díaz





### 3.3. Gestión de riesgos

#### Niveles de impacto:

1. Bajo
2. Medio
3. Alto
4. Critico

Riesgos	Probabilidad de concurrencia	Nivel de impacto	Acción remedial
Desarme por caída del robot	35%	Alto	Volver a construir el robot, de cómo estaba antes del desarme o caída.
Batería descargada	10%	Baja	Cargar la batería cada semana.
Rotura de pieza por caída del robot	10%	Alto	Pedir o comprar una nueva pieza, para reemplazar la rota.
Pérdida de pieza	40%	Medio	Intentar encontrar la pieza perdida, o reemplazarla con una nueva.
Incapacidad o inasistencia de un integrante	50%	Alto	Justificar la inasistencia del integrante.
Daño o pérdida de tarjeta SD	5%	Alto	Comprar una nueva tarjeta SD.
Escasez de piezas	2%	Medio	Comprar las piezas necesarias.
Mala estimación del tiempo	60%	Alto	Reorganizarse como grupo con el tiempo perdido y restante.
Reconstrucción del robot	40%	Medio	Implementar nuevas ideas.

## **4. Planificación de los Recursos**

### **4.1. Hardware:**

- Set Lego Mindstorm EV3.
- Micro SD, del set de Lego Mindstorm, en el cual se podrán ejecutar las instrucciones del robot. (micro Python)
- Computadores con las herramientas necesarias para el desarrollo del proyecto.

### **4.2. Software:**

- Sistema operativo Linux para programar las funciones del robot.
- Redmine: plataforma para la organización del proyecto.
- Visual Studio Code: editor de código.
- Canva: herramienta de diseño.

### 4.3. Costo Hardware y Software

La siguiente tabla refleja los productos y costos a utilizar en el proyecto ev3 "Ball-E"

Productos	Cantidad	Precio	Categoría
Arriendo Notebooks (4 meses)	3	\$60.000 c/u	Hardware
Kit Lego Mindstorm EV3	1	\$700.000	Hardware
Micro SD (8 GB)	1	\$6.696	Hardware
Dongle USB WIFI	1	\$8.000	Hardware
Router	1	\$45.000	Hardware
Python	3	GRATIS	Software
Ev3dev	1	GRATIS	Software
Whatsapp	5	GRATIS	Software
Discord	5	GRATIS	Software
Visual StudioCode	3	GRATIS	Software
Linux (Ubuntu)	1	GRATIS	SO
<b>Total</b>		<b>\$997.686</b>	

● **Costo Recursos Humanos**

Encargado	Personal	Horas trabajadas	Valor hora por trabajador	Sueldo mensual total	Sueldo total (4 meses)
Programador	2	30	\$20.000	\$2.400.000	\$9.600.000
Ensamblador	2	10	\$30.000	\$300.000	\$1.200.000
Jefe de Proyecto	1	32	\$25.000	\$800.000	\$3.200.000
Documentador	3	25	\$15.000	\$1.500.000	\$6.000.000
Diseñador	2	8	\$35.000	\$280.000	\$560.000
<b>Costo Total</b>					<b>\$20.560.000</b>

NOMBRE	COSTO TOTAL
Costos de Hardware	\$997.686
Costos de Software	\$0
Costos de Gestión	\$22.560.000
Costo total proyecto	\$23.557.686

## **5. Análisis – Diseño**

### 5.1. Especificación de requerimientos

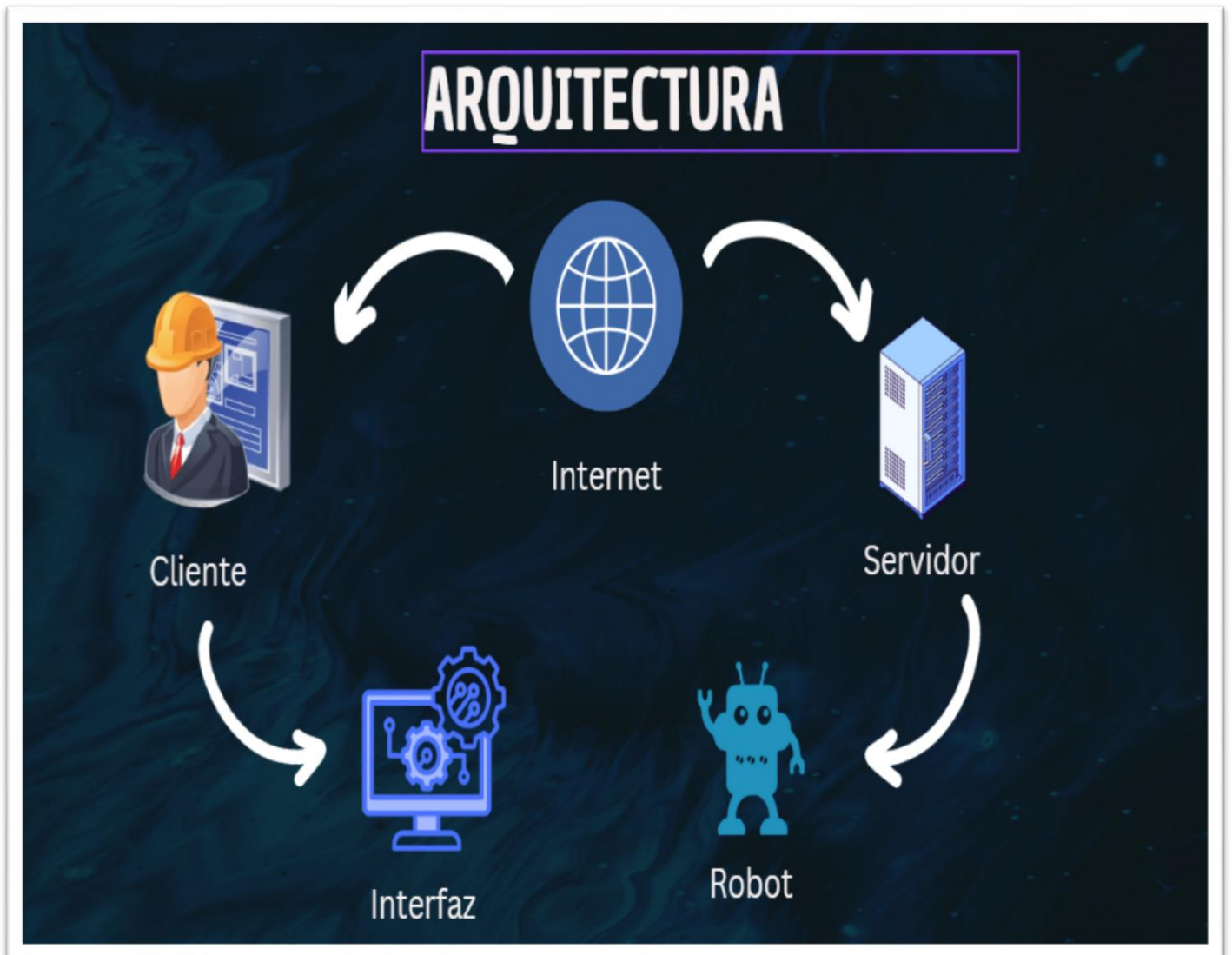
#### **5.1.1 Requerimientos funcionales:**

- Desarrollar un robot que se comuniqué vía wifi y permita al usuario controlarlo mediante una interfaz gráfica en Python.
- Capacidad para moverse en direcciones hacia adelante, atrás, izquierda y derecha.
- Capacidad para abrir, cerrar, subir y bajar la garra.

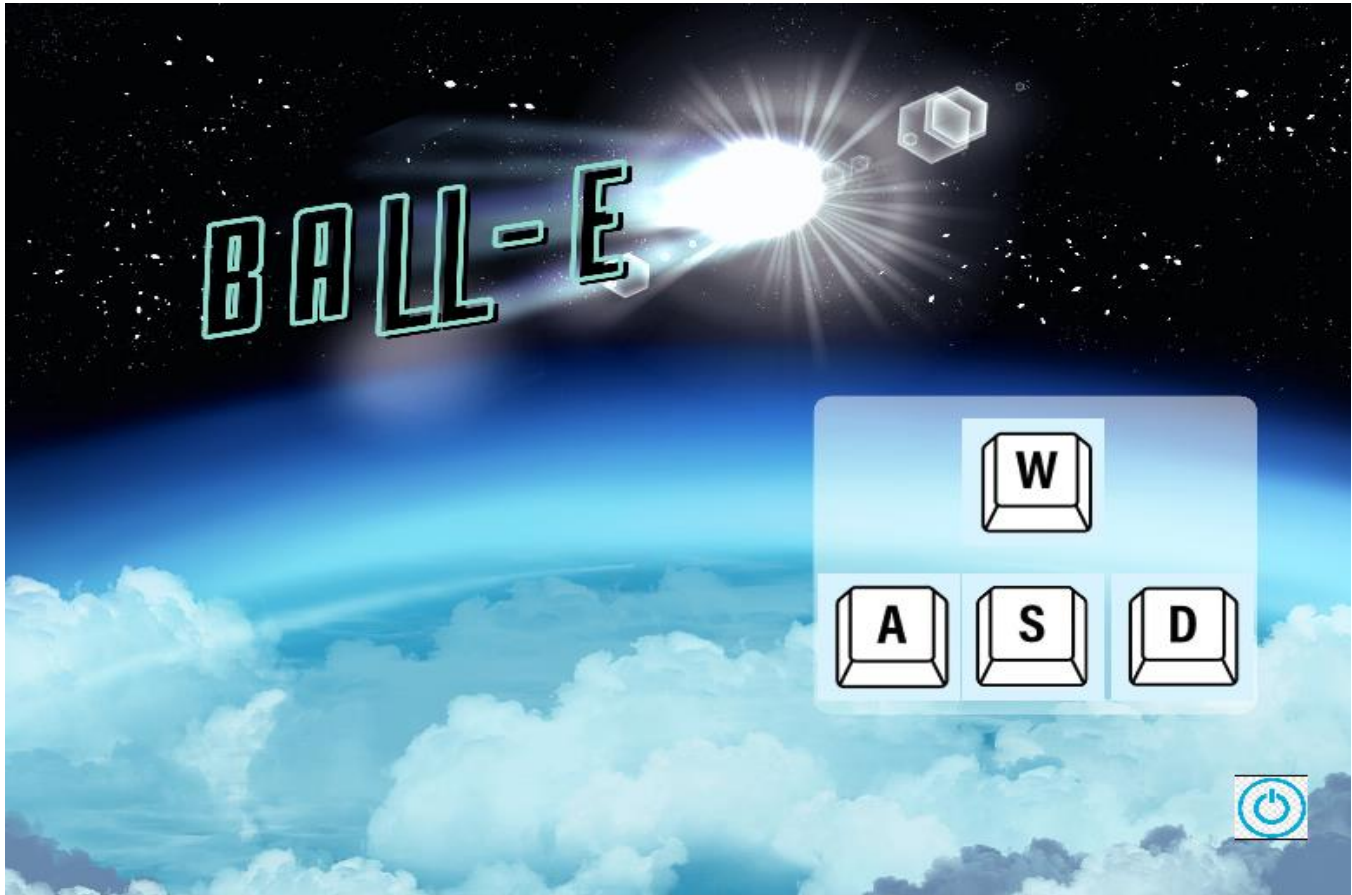
#### **5.1.2 Requerimientos no funcionales:**

- El proyecto debe incluir un manual detallado con instrucciones completas sobre el funcionamiento integral del robot.
- La interfaz gráfica debe contar con botones específicos para controlar el desplazamiento del robot y la garra.

## 5.2. Arquitectura:



### 5.3. Interfaz



Ingrese la dirección IP:

Conectar

## 6. Implementación

### 6.1 Fundamentos Físicos

Para la realización del proceso del robot, los fundamentos físicos básicos requeridos para la función correcta del robot, son de que se tiene que calcular el tiempo en el cual el robot se mueve del Punto A al Punto B en el cual toma la pelota y la lleva al punto C, todo esto también para calcular la fuerza necesaria que la garra necesita para tomar la pelota lo cual es lo más simple.

Fuerza de la garra al tomar la pelota:

Para esto solo necesitamos la estimación de la masa de la garra y la aceleración requerida será la aceleración de gravedad (9,8 m/s)

$F = m \cdot a$  por lo tanto si suponemos que la garra tiene una masa de 200 gramos (0.2 Kg)

Así que calculando sería

$$F = 0,2 \cdot 9,8 = 1.96 \text{ N}$$

Ahora calculando el tiempo que demoraría el robot en avanzar a los puntos (A, B, C):

Asumiendo por la estructura del robot teniendo una distancia aproximada del punto A al B con una aceleración constante de 0,3 m/s<sup>2</sup>

Tenemos los siguientes datos:

Datos del trayecto A hacia B:

Distancia D<sub>ab</sub>: 1,8m Distancia D<sub>bc</sub>: 2,5m Aceleración= 0,3 m/s<sup>2</sup>

La ecuación para distancia es:

$$d = \frac{1}{2} a \cdot t^2$$



La cual al despejarse para calcular el tiempo quedaría:

$$t = \sqrt{\frac{2 \cdot d}{a}}$$

Por lo tanto con los datos que tenemos inicialmente el cálculo para el punto A al B sería:

$$t_{AB} = \sqrt{\frac{2 \cdot d_{AB}}{a}} = \sqrt{\frac{2 \cdot 1,8}{0,3}} = \sqrt{\frac{3,6}{0,3}} = \sqrt{12} = 3,46 \text{ s}$$

Con esto determinamos que el tiempo de ir del punto A al B son 3,46 segundos

Ahora desde el punto B al C:

$$t_{BC} = \sqrt{\frac{2 \cdot 2,5}{0,3}} = \sqrt{\frac{5}{0,3}} = \sqrt{16,67} = 4,08 \text{ s}$$

Por lo tanto el robot se demoraría 4,08 segundos aproximadamente ir del punto B al C

Y con estos datos nos permite saber el tiempo total el cual sería

$$\text{Tiempo total} = 3,46 + 4,08 = 7,54 \text{ s}$$

Entonces el tiempo aproximado que se demora el robot en recorrer todo son 7,54 segundos

## 6.2. Descripción de los programas

### Servidor

```
1  import socket
2  from main import *
3
4  server = socket.socket()
5
6  PORT = 8080
7
8  server.bind(('', PORT))
9  server.listen(1)
10
11 connection, adress = server.accept()
12
13 actions = {
14     "w" : move_front,
15     "a" : move_left,
16     "d" : move_right,
17     "s" : move_back,
18     "e" : move_grab,
19 }
20
21 while True:
22     data = connection.recv(1)
23     keyword = data.decode("utf-8")
24
25     if keyword in actions:
26         actions[keyword]()
27
28     elif keyword == "q":
29         break
```

## Funciones

```
1  #!/usr/bin/env pybricks-micropython
2  from pybricks.hubs import EV3Brick
3  from pybricks.ev3devices import Motor
4  from pybricks.parameters import Port, Stop, Direction, Button, Color
5  from pybricks.tools import wait, Stopwatch, DataLog
6  from pybricks.robotics import DriveBase
7  from pybricks.media.ev3dev import SoundFile, ImageFile
8
9  #Initialize the EV3 brick.
10 ev3 = EV3Brick()
11
12 motor_right = Motor(Port.A, Direction.COUNTERCLOCKWISE)
13 motor_left = Motor(Port.D, Direction.COUNTERCLOCKWISE)
14 motor_garrra= Motor(Port.B)
15
16 robot = DriveBase(motor_left,motor_right, wheel_diameter=55.5, axle_track=84)
17 robot.settings(600, 500, 1000, 500)
18
19 def move_front():
20     robot.straight(150)
21
22 def move_back():
23     robot.straight(-150)
24
25 def move_right():
26     robot.turn(300)
27
28 def move_left():
29     robot.turn(-300)
30
31 def move_grab():
32     motor_garrra.run_until_stalled(-300, Stop.HOLD, 40)
33     motor_garrra.run_until_stalled(300, Stop.COAST, 40)
34
35 robot.stop()
36
37 ev3.speaker.beep()
```

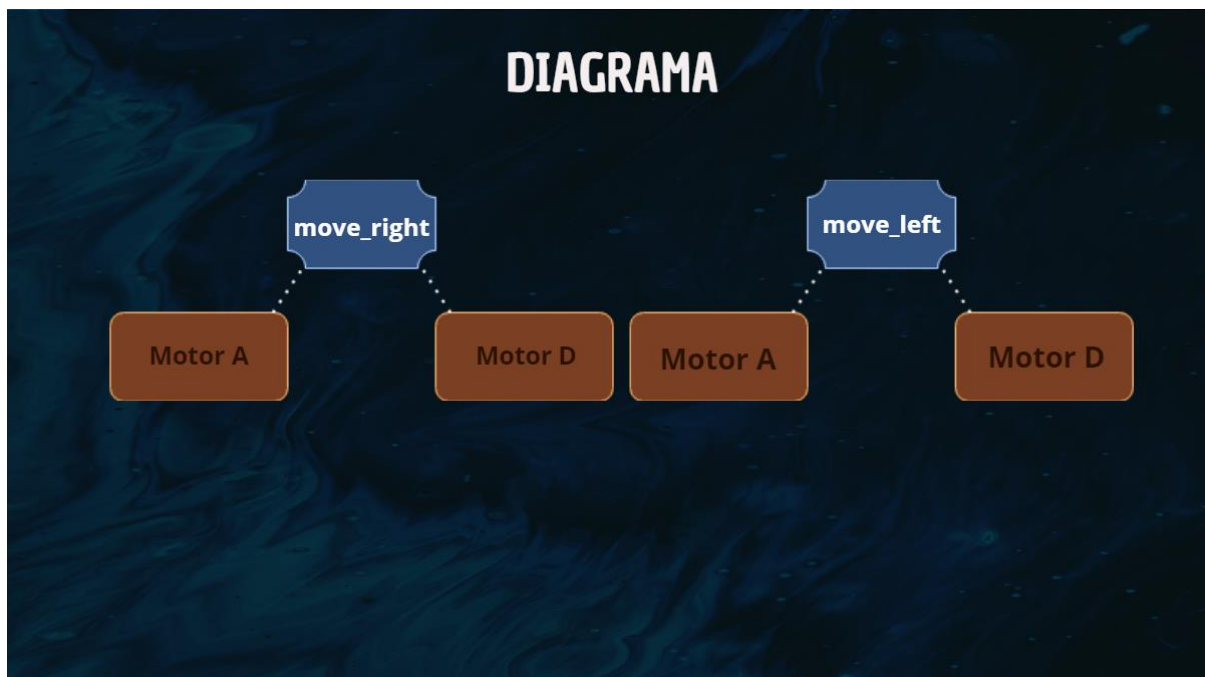
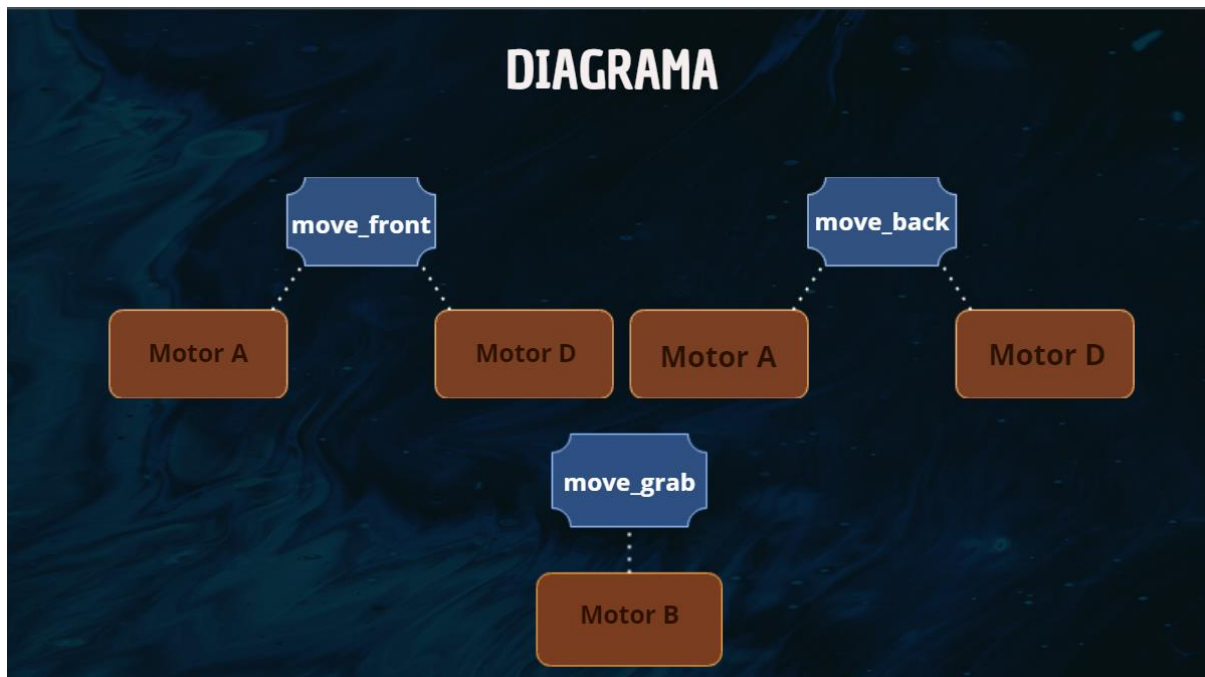
# Interfaz

```
1 import tkinter as tk
2 from tkinter import ttk
3 from tkinter import messagebox
4 import socket
5
6 class App(tk.Tk):
7     STATUS_CONNECTION = False
8     PORT = 8080
9
10     def __init__(self):
11         super().__init__()
12         self.title("Ball-e")
13         self.config(padx=10, pady=10, bg="#FFD8DB")
14         self.geometry("800x600")
15         self.canvas = tk.Canvas(self, width=800, height=600)
16         self.canvas.pack()
17         self.resizable(width=False, height=False)
18         self.imagen_fondo = tk.PhotoImage(file="fondo.png")
19         self.imagen_fondo = self.imagen_fondo.subsample(5,5)
20         self.canvas_create_image(0, 0, anchor=tk.NW, image=self.imagen_fondo)
21         self.init_images()
22         self.init_elements()
23         self.grab_open = True # Inicialmente, la garra está abierta
24
25     def init_images(self):
26         imagenes = ["encendido.png"]
27         self.imagenes = [tk.PhotoImage(file=imagen).subsample(12, 12) for imagen in imagenes]
28
29         self.garra_abierta_tng = tk.PhotoImage(file="garra_abierta.png").subsample(10, 10)
30         self.garra_cerrada_tng = tk.PhotoImage(file="garra_cerrada.png").subsample(10, 10)
31
32         imagenes1 = ["flechaFront.png", "flechaBack.png", "flechaLeft.png", "flechaRight.png"]
33         self.imagenes1 = [tk.PhotoImage(file=imagen).subsample(25, 25) for imagen in imagenes1]
34         self.flechaFront, self.flechaBack, self.flechaLeft, self.flechaRight = self.imagenes1
35
36     def init_elements(self):
37         btn_color = "#fff" if self.STATUS_CONNECTION else "#000"
38         self.btn_Connect = tk.Button(self, bg=0, bgcolor=btn_color, width=80, height=40, command=self.ip_connect, place=(710, y=500))
39         self.btn_Front = tk.Button(self, image=self.flechaFront, bg=0, bg="white", highlightbackground="white", width=80, height=80, command=lambda: self.mover_front_car(None)).place(x=545, y=100)
40         self.btn_Back = tk.Button(self, image=self.flechaBack, bg=0, width=80, height=80, command=lambda: self.mover_back_car(None)).place(x=545, y=300)
41         self.btn_Left = tk.Button(self, image=self.flechaLeft, bg=0, width=80, height=80, command=lambda: self.mover_left_car(None)).place(x=450, y=200)
42         self.btn_Right = tk.Button(self, image=self.flechaRight, bg=0, width=80, height=80, command=lambda: self.mover_right_car(None)).place(x=640, y=200)
43         self.btn_Grab = tk.Button(self, image=self.garra_abierta_tng, bg=0, command=lambda: self.mover_grab_car(None)).place(x=545, y=200)
```

```
1 def ip_connect(self):
2     if not self.STATUS_CONNECTION:
3         ventana_conexion = tk.Toplevel(self)
4         ventana_conexion.title("Ventana de Conexión")
5         ventana_conexion.geometry("200x150")
6         ventana_conexion.resizable(width=False, height=False)
7         etiqueta = tk.Label(ventana_conexion, text="Ingrese la dirección IP:")
8         etiqueta.pack(pady=10)
9         self.IP = tk.StringVar()
10        entry_ip = tk.Entry(ventana_conexion, textvariable=self.IP)
11        entry_ip.pack(pady=10)
12        btn_conectar = tk.Button(ventana_conexion, text="Conectar", command=lambda: self.robot_connection(self.IP.get()))
13        btn_conectar.pack(pady=10)
14    else:
15        self.robot_connection(self.IP.get())
16
17 def robot_connection(self, IP):
18     if self.STATUS_CONNECTION:
19         self.STATUS_CONNECTION = not self.STATUS_CONNECTION
20         self.init_elements()
21    else:
22        try:
23            self.client = socket.socket()
24            self.client.connect((IP, self.PORT))
25            self.STATUS_CONNECTION = True
26            self.init_elements()
27        except socket.error:
28            messagebox.showerror("Error", "No se pudo conectar al robot.")
29            self.STATUS_CONNECTION = False
```

```
1     def move_front_car(self, event):
2         self.client.send(bytes([ord('w')]))
3
4     def move_left_car(self, event):
5         self.client.send(bytes([ord('a')]))
6
7     def move_right_car(self, event):
8         self.client.send(bytes([ord('d')]))
9
10    def move_back_car(self, event):
11        self.client.send(bytes([ord('s')]))
12
13    def move_grab_car(self, event):
14        self.client.send(bytes([ord('e')]))
15        if self.grab_open:
16            self.grab_open = False # Ahora la garra está cerrada
17            self.btn_Grab.config(image=self.garra_cerrada_img)
18        else:
19            self.grab_open = True # Ahora la garra está abierta
20            self.btn_Grab.config(image=self.garra_abierta_img)
21
22    if __name__ == "__main__":
23        app = App()
24        app.mainloop()
```

### 6.3. Diagramas



## **7. Resultados**

### **7.1. Estado actual del proyecto**

- Funciones de movimiento de implementación.
- implementación de servidor.
- Interfaz gráfica desarrollada con tkinter.
- Conexión establecida.
- Documentación del proyecto.
- Registro de actividades, informe y presentación.

## **7.2. Problemas encontrados y soluciones propuestas**

### **7.2.1 Problemas encontrados:**

- Reconstrucción frecuente del robot.
- Modificación frecuente de la interfaz gráfica.
- Modificación de funciones.

### **7.2.2 Soluciones:**

- Terminar las modificaciones del robot para dar avance al desarrollo de las funciones y interfaz.
- Llegar a una idea en conjunto que cumpla con los objetivos establecidos.
- Reestructuración de la base y la garra del robot.

## **8. Conclusión:**

A lo largo de este proyecto, hemos logrado cumplir con los objetivos propuestos, desde la planificación hasta la implementación de "EV3 Ball-E". Cada integrante del equipo desempeñó su rol de manera efectiva, lo que permitió una distribución clara de responsabilidades y una comunicación fluida, utilizando herramientas colaborativas como WhatsApp y Discord. A pesar de los desafíos presentados, como la gestión del tiempo y los riesgos inherentes a la construcción y programación del robot, el equipo fue capaz de sortear las dificultades mediante estrategias de reorganización y ajustes oportunos. El proyecto no sólo refleja nuestra habilidad técnica y de gestión, sino también nuestra capacidad de trabajo colaborativo y resolución de problemas. En resumen, el desarrollo de "EV3 Ball-E" ha sido un ejercicio valioso que nos ha permitido poner en práctica los conocimientos adquiridos durante nuestra formación, demostrando un crecimiento significativo en nuestras competencias profesionales.

## **9. Referencias:**

Costo de piezas:

<https://www.brickowl.com/catalog/lego-mindstorms-ev3-set-31313/inventory>