

UNIVERSIDAD DE TARAPACÁ



**FACULTAD DE INGENIERÍA
DEPARTAMENTO DE INGENIERÍA CIVIL EN COMPUTACIÓN E
INFORMÁTICA**



**Plan de Proyecto
“Maquinaria de
clasificación de
materiales”**

Alumnos:	Bastian Hernandez Saoud Ahmed Alex Campillay Marcos Caldas Enzo Llancabure
Asignatura:	Proyecto I
Profesor:	Baris Klobertanz Quiroz

22 de Septiembre – 2025

Historial De Cambios

Fecha	Versión	Descripción	Autor(es)
22/09	1.0	Reconocimiento del Problema y Formalización del Proyecto	Todos
26/09	1.1	Finalización de la introducción	Marcos Caldas Enzo Llancabure Bastian Hernandez
08/10	1.2	Finalización de la organización de personal	Marcos Caldas Bastian Hernandez Enzo Llancabure
13/10	1.3	Finalización de la planificación del proyecto	Enzo Llancabure Marcos Caldas Bastian Hernandez
17/10	1.4	Finalización de la planificación de recursos	Enzo Llancabure Marcos Caldas
17/10	1.5	Finalización de la conclusión	Enzo LLancabure

17/10	1.6	Finalización de las referencias	Bastian Hernandez
-------	-----	---------------------------------	-------------------

Tabla N°1 “Historial de Cambios”.

➤ Índice

1. Panel General	5
1.1. Introducción	5
1.2. Objetivos	6
1.2.1. Objetivo General	6
1.2.2. Objetivos Específicos	6
1.3. Restricciones	8
1.4. Entregables	9
2. Organización del Personal	10
2.1. Descripción de los Roles	10
2.2. Personal que Cumplirá los Roles	10
2.3. Métodos de Comunicación	11
3. Planificación del Proyecto	12
3.1. Actividades	12
3.2. Carta Gantt	14
3.3. Gestión de Riesgos	15
4. Planificación de los Recursos	18
4.1. Hardware	18
4.2. Software	18
4.3. Estimación de Costos	19
5. Análisis y Diseño	22
5.1 Especificación de requerimientos	22
5.1.1 Requerimientos funcionales	22
5.1.2 Requerimientos no funcionales	22
5.2 Arquitectura de software	23
5.3 Diseño inicial de la interfaz gráfica de usuario (GUI)	25
6. Implementación	26
6.1 Fundamentos de los movimientos	26
6.2 Descripción del sistema	30
6.2.1 Cliente	35
6.2.2 Servidor	36
6.2.3 Interfaz gráfica de usuario (GUI)	37
7. Resultados	38
7.1 Estado actual del proyecto	38
7.2 Problemas encontrados y solucionados	39
8. Conclusión	40
9. Referencias	42
Anexos	44
9.Anexo 1	44

➤ Índice de tablas

Tabla N°1 “Historial de Cambios”.	1
Tabla N°2 “Roles”.	9
Tabla N°3 “Actividades”.	12
Tabla N°4 “Gestión de riesgos”	16
Tabla N°5 “Costo de Hardware”	18
Tabla N°6 “Costo Trabajador”	19
Tabla N°7 “Costo Software”	19
Tabla N°8 “Costo Total”.	20

➤ Índice de imagenes

Ilustración N°1 “Carta Gantt”	13
Ilustración N°2 “Wireframe GUI”	24
Ilustración N°3 “DCL”	26
Ilustración N°5 “Codigo Version1.0.0.py”.	28
Ilustración N°6 “Codigo Version1.0.1.py”.	30
Ilustración N°7 “Codigo app2_control”.	31
Ilustración N°8 “Codigo Version1.0.0.py”.	32
Ilustración N°9 “Codigo app2_control”.	33
Ilustración N°10 “Codigo Version1.0.0.py”.	33
Ilustración N°11 “Interfaz Grafica de Usuario”.	34
Ilustración N°12 “LEGO SPIKE PRIME”.	41
Ilustración N°13 “EXTENSION LEGO SPIKE PRIME”	41
Ilustración N°14 “Notebook loq gen 9”.	42
Ilustración N°15 “GALAXY TAB S8 PLUS”.	42
Ilustración N°16 “IDEAPAD 3 15”.	43
Ilustración N°17 “GAMER ASPIRE 5”.	43
Ilustración N°18 “HP PAVILION”.	44
Ilustración N°19 “Bloques usados”.	44

1. Panel General

1.1. Introducción

En la industria minera se evidencia múltiples veces que la clasificación de materiales es una parte importante de la operación; sin embargo, la ejecución ineficiente de esto puede llevar a cuellos de botella operativos. Aún más grave estos procesos pueden llevar a la exposición de los trabajadores a entornos peligrosos si no están debidamente automatizados, evidenciando el problema claramente: La necesidad de implementar un sistema que garantice eficiencia sin comprometer la integridad de los trabajadores.

Bajo esta premisa, en este informe se demostrará el trabajo colaborativo realizado por el equipo para cumplir el objetivo de mejorar la productividad, optimizar los procesos y elevar los estándares de seguridad de un equipo de minería. Para lograrlo se aplicaron conocimientos de ingeniería mediante una simulación de un clasificador de materiales fabricado con el set de Lego Spike Prime.

A partir de lo anterior, la estrategia seguida para la organización de actividades y asignación de responsabilidades se detalla en los siguientes apartados.

1.2. Objetivos

1.2.1. Objetivo General

Desarrollar y programar un robot con el kit LEGO Spike Prime que sea capaz de clasificar bloques en una casilla designada según su color, simulando un proceso de clasificación de materiales de una industria minera.

1.2.2. Objetivos Específicos

- Experimentar para ser capaz de usar correctamente el set Lego Spike Prime hasta el punto de poder aplicar correctamente todas las funciones necesarias para crear el robot clasificador en un tiempo estimado de 2 semanas.
- Asignar roles a cada integrante del grupo para mantener un responsable en cada área de trabajo y poder realizar el proyecto de forma eficiente, en un plazo de 1 semana.
- Planificar y ensamblar un prototipo inicial con el set LEGO Spike Prime que pueda detectar con el sensor los colores de los bloques y mover los motores respectivamente del color, en un plazo de 3 semanas.
- Programar con el lenguaje de programación Python en la aplicación LEGO Spike un código capaz de conectar el sensor de color junto con los motores al mismo tiempo para buscar la rapidez del ordenamiento de bloques, en el tiempo que el ensamblador lo tenga que necesitar.
- Construir el robot que cumpla con una buena coordinación y velocidad con el set Lego Spike Prime, librería Pybricks y Visual Studio Code, además que sea capaz de clasificar correctamente cada bloque en una casilla, antes de la presentación final.
- Documentar y subir los archivos en formato PDF a la plataforma de Redmine para registrar cada avance que se va a realizar hasta la fecha final del proyecto.

- Implementar una interfaz para el robot clasificador que sea capaz de realizar movimientos de este, dependiendo de los botones de la interfaz, utilizando Tkinter y Visual Studio Code, una vez terminado el robot, en un plazo de 3 semanas.
- Definir un repositorio en GitHub para tener un historial de cambios, además de un sitio donde cada integrante será capaz de ver el código, dentro de la web de GitHub, hasta que se finalice el trabajo de codificación.
- Crear un manual de usuario con las instrucciones de cómo usar el robot, en Microsoft Word, al finalizar la construcción y codificación del robot, en un plazo de 1 semana.
- Realizar una presentación que sea capaz de demostrar que el robot clasificador puede realizar una buena clasificación de bloques, utilizando la web Canva, en un tiempo estimado de 1 semana.

1.3. Restricciones

Las restricciones son requerimientos mínimos que deben ser obligatoriamente cumplidos.

Las restricciones que se tiene en este proyecto son:

- o Si es necesario, utilizar la extensión de Lego Spike Prime.
- o Solo se debe utilizar la plataforma Redmine para los documentos y avance del proyecto.
- o Se debe utilizar el Set de Lego Spike Prime.
- o Tiempo limitado para la finalización del proyecto, debido al término del semestre.
- o Cantidad de integrantes limitada a 5.
- o La disponibilidad del robot para su uso sea codificación y construcción, está limitada al horario del departamento de ICCI (Ingeniería civil en computacion e Informatica).
- o Robot debe ser capaz de reconocer el bloque y poder clasificarlo dependiendo de su color.
- o La conexión entre el pc y el robot debe ser inalámbrica.

1.4. Entregables

Bitácoras: Son informes semanales que describen el avance del equipo en el proyecto, abarcando actividades realizadas, dificultades encontradas, recomendaciones para mejorar y acciones tomadas. Preparadas por los integrantes del grupo, ofrecen un panorama exhaustivo para apoyar decisiones estratégicas, asignan responsabilidades y resaltan asuntos a tratar en grupo.

Carta Gantt: Representación visual de la programación del proyecto, mostrando en una línea de tiempo las tareas, su duración y secuencia, facilitando la gestión del tiempo y los recursos al visualizar la evolución de las actividades a lo largo del proyecto.

Informes de avance: Este documento detalla la organización y estrategia para alcanzar los objetivos de la asignatura. Se abordará la asignación de roles, las metas del equipo y las medidas que implementarán para lograr el propósito académico. Además, se comparten las primeras impresiones durante el proceso de desarrollo y se presenta la documentación relevante recopilada a lo largo del semestre.

Presentaciones: Se harán presentaciones con el formato de PDF, para presentar la información recopilada en los informes entregados, cada presentación tratará los temas de cada informe ya sea avance o posteriores a él.

Manual de Usuario: Se desarrolla un manual de usuario para el funcionamiento correcto del robot enfocado en el aprendizaje para el usuario.

2. Organización del Personal

La organización en un grupo es importante para lograr la finalización de este proyecto, como también un buen ambiente en el grupo.

2.1. Descripción de los Roles

Jefe de proyecto: Representante del equipo, supervisa y organiza el progreso del proyecto.

Ensamblador: Encargado del montaje y el armado de las piezas, monitorea el cumplimiento de las funcionalidades del robot, en conjunto con el programador.

Programador: Encargado del área de la codificación y la lógica de funcionamiento del robot, trabajando en colaboración con el ensamblador para integrar el software con el hardware.

Documentador: Encargado de registrar el avance del proyecto y la redacción de los informes. Para cumplir su función, debe mantener una comunicación constante con todos los miembros del equipo, con el fin de recopilar y unir la información de cada área.

2.2. Personal que Cumplirá los Roles

Rol	Responsable
Jefe de proyecto	Bastian Hernandez
Ensamblador	Alex Campillay
Programador	Saoud Ahmed
Documentador	Enzo Llancabure Marcos Caldas Bastian Hernandez

Tabla N°2 "Roles".

2.3. Métodos de Comunicación

Los principales medios de comunicación que usarán a lo largo del desarrollo del proyecto serán los siguientes:

-WhatsApp: Se utilizará para mensajería haciendo uso de los grupos que ofrece la Aplicación.

-Github: Como repositorio de codificación con avance e historial de versiones de este mismo, es una comunicación enfocada a tareas técnicas.

-Discord: Reuniones a distancia aprovechando los canales de texto y voz que ofrece esta Aplicación.

-Horario del Taller de Clase: Reuniones presenciales y avance de forma presencial para comunicar el avance del proyecto mismo.

3. Planificación del Proyecto

3.1. Actividades

O.E	Nombre	Encargado
O.E.1	Dominar el LEGO Spike Prime para una óptima creación de Robot.	Todos los integrantes.
O.E.2	Asignación de los Roles para cada integrante del grupo.	Todos los integrantes.
O.E.3	Diseño y Ensamblaje del primer prototipo con el set LEGO Spike Prime	Alex Campillay, Bastian Hernandez.
O.E.4	Programación del Robot Lego encargado de clasificar.	Saoud Ahmed Marcos Caldas
O.E.5	Construcción final del robot LEGO clasificador.	Alex Campillay, Bastian Hernandez, Enzo Llancabure.
O.E.6	Documentación y Registro en la plataforma Redmine.	Saoud Ahmed, Bastian Hernandez, Marcos Caldas.
O.E.7	Implementación de la Interfaz de control para el Robot LEGO clasificador.	Marcos Caldas, Saoud Ahmed, Alex Campillay.
O.E.8	Gestión del Repositorio Github con el código de control del Robot LEGO clasificador.	Todos los integrantes.
O.E.9	Elaboración del Manual de Usuario para tener las instrucciones de uso del Robot LEGO clasificador.	Todos los integrantes.

O.E	Nombre	Encargado
O.E.1 0	Presentación del Robot Clasificador para una demostración sólida del Robot LEGO clasificador.	Todos los integrantes.

Tabla N°3 “Actividades”.

3.2. Carta Gantt

Esta herramienta permite visualizar la secuencia de las actividades, organizadas en fases como la codificación, el ensamblaje y la documentación. A través de este diagrama, se muestra el porcentaje de avance de cada tarea, ayudando a la organización del grupo a la hora de completar las tareas.

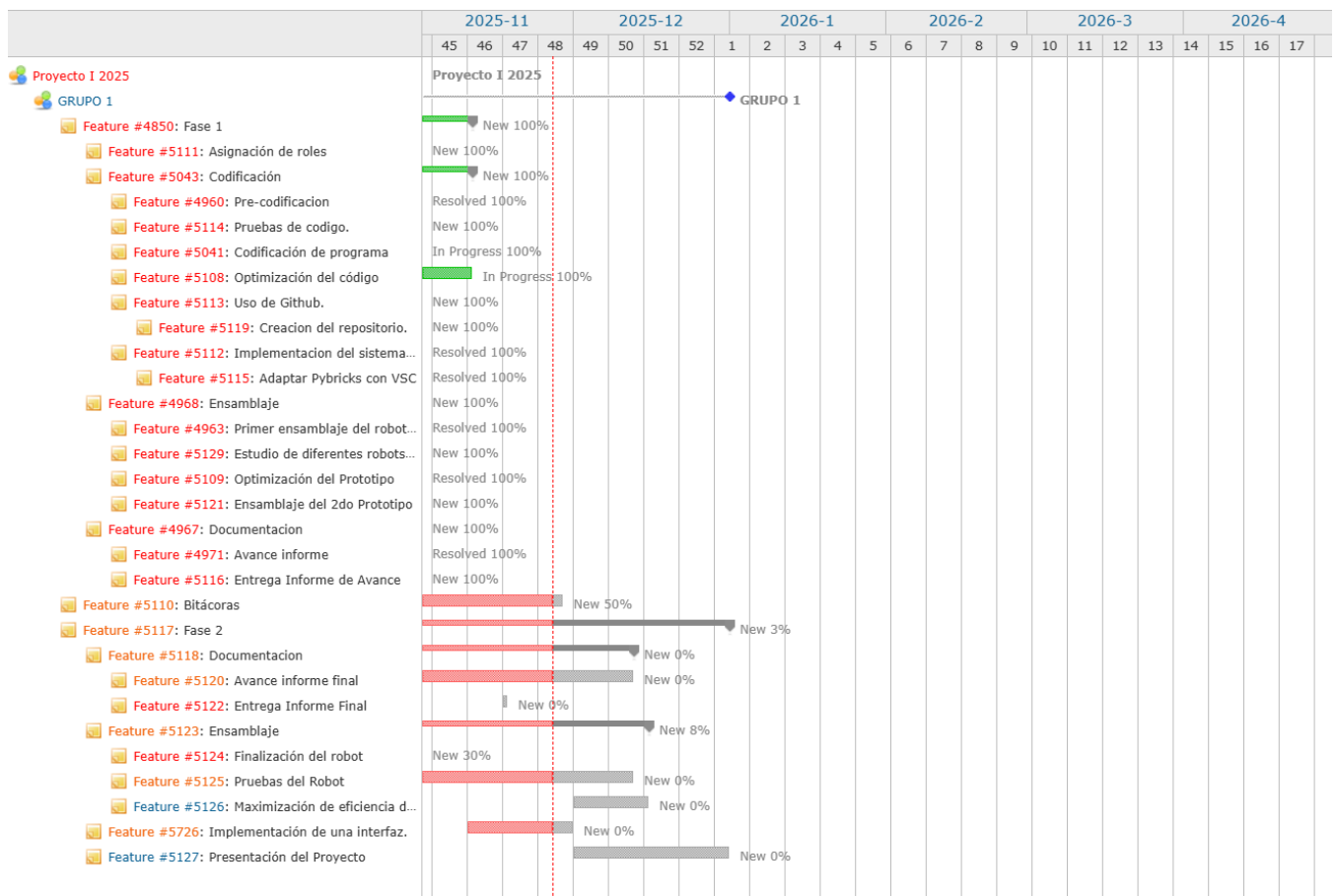


Ilustración N°1 "Carta Gantt"

3.3. Gestión de Riesgos

Para garantizar el cumplimiento de los objetivos del proyecto en los plazos establecidos, se ha elaborado una tabla de gestión de riesgos. Esta herramienta permite identificar, analizar y clasificar los posibles contratiempos que podrían afectar el desarrollo del prototipo robótico y la planificación general.

A continuación, se definen los niveles de severidad establecidos para clasificar cada riesgo según su impacto en el cronograma y en la operatividad del equipo:

1. **Impacto Crítico:** Problema al cual se le debe otorgar la máxima prioridad de resolución por parte del equipo, del caso contrario puede impactar en la entrega puntual del proyecto por retrasos o incluso un reinicio total de este.
2. **Impacto Alto:** Evento de máxima gravedad que compromete la viabilidad del proyecto. Requiere medidas inmediatas, pudiendo implicar el reinicio de etapas completas o la redefinición del alcance.
3. **Impacto Medio:** Riesgo que genera retrasos significativos en una o varias etapas clave. Exige una respuesta prioritaria para evitar que el desfase afecte la fecha de entrega final.
4. **Impacto Bajo:** Riesgo menor o imprevisto cotidiano que no altera la ruta del proyecto y puede ser resuelto con acciones simples sin afectar los entregables principales.

Riesgo	Nivel de Impacto	Acción Remedial
Daño grave de hardware	1	Verificar conexión del puerto, en caso de daños irreparables conseguir repuestos inmediatamente.
Horario insuficiente para el cumplimiento de tareas en conjunto	2	Coordinar los horarios disponibles del grupo.
Desempeño del robot poco eficiente	2	Ensamblar un robot más adecuado siguiendo guías en línea o un nuevo diseño adaptándolo a lo requerido.
Problema a la hora de experimentar con el robot.	2	Buscar la forma de modificar la posición de las piezas para que el robot cumpla su función.
Falla en el registro de redmine	2	Comunicar al profesor para buscar una solución.
Ausencia de piezas	2	Solicitar la extensión de Lego Spike o pedir piezas al ayudante.
Error en la codificación	2	Corregir errores sintácticos y lógicos en lo posible, de no serlo investigar una solución o explorar otro tipo de solución.
Falta de internet en la sala	3	Compartir internet del celular o usar cables ethernet.

Atraso en el cumplimiento de tareas	3	Hablar con el grupo para avanzar el proyecto fuera de clases.
Integrante falta a una clase	4	El Integrante tiene la responsabilidad de ponerse al día con lo que se avanzó en esa clase.

Tabla N°4 “Gestión de riesgos”

4. Planificación de los Recursos

Este punto muestra los recursos requeridos para la ejecución del proyecto, el cual se divide en herramientas físicas y digitales. Para la tabla de costo trabajador, se le asigna un sueldo basado en una tarifa por hora para cada rol, se toma en cuenta la complejidad de cada rol para la asignación de tarifa. El cálculo final es la suma de horas totales del equipo.

4.1. Hardware

- Set Lego Spike Prime.
- Set Lego Spike Prime Extension.
- Computador con el sistema operativo necesario para poder programar las instrucciones para el robot.
- Tablets para poder hacer la documentación necesaria.

4.2. Software

- Sistema operativo Windows para programar las funciones del robot.
- Redmine, página para la organización del proyecto.
- Canva.
- Plataforma Lego Education Spike (Code).
- Flutter para creación de aplicaciones.
- Pybricks en conjunto con VS code para la codificación pero más centralizada para los robots LEGO.

4.3. Estimación de Costos

Costo de Hardware:

Producto	Cantidad	Precio (CLP)
Set Lego Spike	1	\$ 622.879
Extension Lego Spike	1	\$ 167.202
Notebook LOQ Gen 9	1	\$ 769.993
Samsung Galaxy Tab S8 Ultra	1	\$ 1.150.000
Lenovo IdeaPad 3 15	1	\$ 699.990
Notebook Aspire G A515-58GM-56ZZ-1	1	\$ 1.049.990
HP Pavilion Laptop 14-dv2xxx	1	\$ 1.500.000
Total:	7	\$ 6.005.054

Tabla N°5 “Costo de Hardware”

Costo de Trabajador:

Rol	Horas / Mes	Horas Extra	Precio / Hora (CLP)
Jefe de proyecto	18 horas	5 horas	\$ 10.555
Programador	18 horas	5 horas	\$ 7.500
Ensamblador	18 horas	5 horas	\$ 7.000
Documentador	18 horas	5 horas	\$ 3.500
Total :	-	-	\$ 2.736.000

Tabla N°6 "Costo Trabajador"

Costo de Software:

Software	Precio (CLP)
Microsoft Office	\$ 8.990
Visual Studio Code	\$0
Redmine	\$0
Pybricks	\$0
Flutter	\$0
Total	\$8.990

Tabla N°7 "Costo Software"

Destacado:

- *La contabilización de las horas trabajadas comienza a partir de la formación del grupo de trabajo, la fecha de inicio fue el día 22 de septiembre del 2025.*
- *Para la contabilización de las horas de trabajo, se tuvo en cuenta el tiempo de trabajo en clases.*
- *Para la contabilización de las horas extras, se tuvo en cuenta el tiempo en las que se trabajó fuera del horario de clase.*
- *El cálculo por hora se basa en el sueldo mensual promedio de cada uno de los cargos del proyecto dividido en una jornada laboral de 180 horas mensuales.*
- *Para el pago de horas extras se aplicó un pago doble sobre el valor de la hora normal.*

Total de Costo:

Costo Hardware	\$ 6.005.054
Costo Empleados	\$ 2.736.000
Costo Software	\$ 8.990
Total :	\$ 8.750.044

Tabla N°8 “Costo Total”.

5. Análisis y Diseño

5.1 Especificación de requerimientos

Antes de especificar los requerimientos funcionales y no funcionales, se debe establecer que el cliente es una empresa minera, representada generalmente por el área de ingeniería y automatización, los cuales financian la implementación del sistema robótico encargado para clasificar los materiales, siendo su rol principal asegurar que la solución esperada por la empresa cumpla con los objetivos de mejorar la seguridad de los trabajadores, aumentar la eficiencia y optimizar el clasificación de materiales.

Y tomando en cuenta que el usuario no es necesariamente la misma empresa y que la

probabilidad de que interactúe de forma directa con el robot diariamente es escasa, se

Puede representar al usuario como un operador de maquinaria dentro de la empresa.

De esta manera teniendo en cuenta esos dos puntos se pueden definir los requerimiento

funcionales y no funcionales:

5.1.1 Requerimientos funcionales

Algunos de los requerimientos funcionales básicos que debe cumplir el robot a la hora de ser implementado y accionado son:

- El robot debe ser capaz de identificar los colores de bloques de lego.
- El robot debe ser capaz de funcionar de manera automática y manual.
- El robot debe ser capaz de administrar varios bloques de lego sucesivamente.
- El robot debe ser capaz de depositar cada bloque en celdas de compartimiento distintas según el color de los bloques.
- El robot debe ser capaz de almacenar bloques para luego organizarlos de acuerdo a su color.

5.1.2 Requerimientos no funcionales

Por otro lado algunos de los atributos de calidad del software que a los stakeholders les podrían interesar para cualificar el proyecto serían:

- Disponibilidad : El robot deberá mantener una disponibilidad mínima del 98% durante la jornada operativa de la empresa minera. Ante una falla, el sistema deberá restablecer su operaciones en un tiempo máximo de 10 minutos, garantizando la eficacia del proceso.

Métrica de Evaluación : Porcentaje de tiempo operativo

- Robustez : El sistema debe ser capaz de manejar errores de sensores, entradas incorrectas o interrupciones momentáneas de comunicación sin detenerse completamente.

Métrica de Evaluación : Número de fallos no controlados.

- Rendimiento : El robot automatizado debe responder a las órdenes del operador o del sistema de control en un tiempo máximo de 1 segundo, asegurando el flujo de trabajo eficiente, además deberá mantener su desempeño sin interrupciones durante al menos 8 horas de operación continua.

Métrica de Evaluación : Tiempo promedio de respuesta a comandos y estabilidad del rendimiento durante turno de trabajo.

- Usabilidad : La interfaz del control del sistema debe ser intuitiva, clara y autoexplicativa, permitiendo que un operador capacitado pueda utilizar el robot de forma autónoma tras una breve capacitación.

Métrica de Evaluación : Tiempo de capacitación requerido y cantidad de errores de operación cometidos.

5.2 Arquitectura de software

La arquitectura de software describe los componentes principales del sistema, la forma en que se organizan y cómo se comunican entre sí para cumplir con los objetivos del proyecto.

en este proyecto se utiliza una arquitectura cliente-servidor, donde la aplicación móvil actúa como cliente, mientras que el robot funciona como servidor que recibe, interpreta y ejecuta las instrucciones enviadas desde la interfaz gráfica

- Modelo de arquitectura cliente-servidor

en la arquitectura cliente-servidor, el sistema se divide en dos partes principales:

- Cliente: encargado de la interacción con el usuario y del envío de solicitudes.
- servidor: encargado de procesar las solicitudes recibidas y ejecutar las acciones correspondientes.

Este modelo permite separar la lógica de control del robot de la interfaz gráfica, facilitando el mantenimiento, las pruebas y futuras mejoras del sistema.

- componentes del sistema

el sistema está compuesto por los siguientes elementos:

cliente (aplicación flutter)

desarrollado en flutter.

proporciona la interfaz para el usuario.

permite al usuario controlar el robot sorter.

envía comandos de texto al robot utilizando bluetooth.

controla solo se encarga del envío de instrucciones por lo que no directamente el hardware.

interfaz gráfica de usuario (GUI)

forma parte del cliente.

permite la interacción directa del usuario.

traduce acciones del usuario como presionar botones o mover el joystick a comandos digitales.

muestra el estado de conexión y el modo de simulación.

Medio de comunicación (Bluetooth)

Es la línea de comunicación entre el cliente y el servidor.

Permite el envío de comandos en tiempo real.

no procesa información, solo se encarga de transmitirla.

servidor (Robot LEGO Spike Prime con Bricks)

Ejecuta el código en el entorno PyBricks usando el lenguaje python.

recibe los comandos enviados por la aplicación móvil.

interpreta los mensajes recibidos.

controla motores sensores del robot según el comando recibido.

Devuelve el estado de ejecución si es necesario.

- Flujo de comunicación del sistema

El usuario interactúa con la interfaz gráfica de la aplicación móvil, utilizando los botones o el joystick integrados en ella. Cada acción realizada es interpretada por la aplicación, que genera un comando de texto correspondiente. Este comando es enviado al hub del robot Spike Prime, el cual lo recibe, interpreta la instrucción y ejecuta la acción solicitada desde su propio código.

El control del movimiento se realiza mediante el joystick, mientras que el motor de empuje se acciona mediante dos botones: **L1**, que activa el movimiento hacia un lado, y **R1**, que lo dirige hacia el otro.

- Ventajas de la arquitectura utilizada

la utilización de esta interfaz otorga bastantes ventajas en temas de actualizaciones futuras u optimización. Existe una separación clara entre lo que es la interfaz y control físico, esto otorga mayor facilidad para depuración y pruebas.

al igual que una escalabilidad para agregar nuevas funciones incluso reutilización del código del robot con distintas interfaces

La arquitectura cliente-servidor permite que la aplicación móvil se enfoque únicamente en la experiencia del usuario, mientras que el robot se encarga del control del hardware, esta separación mejora la organización del sistema y permite modificar la lógica del robot sin necesidad de cambiar la interfaz gráfica.

5.3 Diseño inicial de la interfaz gráfica de usuario (GUI)

En esta sección se presenta el diseño inicial de la interfaz gráfica de usuario (GUI) del sistema de control del robot sorter. Este diseño corresponde a un wireframe de baja fidelidad, cuyo objetivo es definir la estructura general de la aplicación, la disposición de sus elementos y la interacción básica del usuario, sin considerar detalles visuales, tipografías o estilos gráficos finales.

El wireframe permite planificar la interfaz antes de su implementación, facilitando la comprensión del funcionamiento del sistema y sirviendo como base para el desarrollo posterior en flutter.

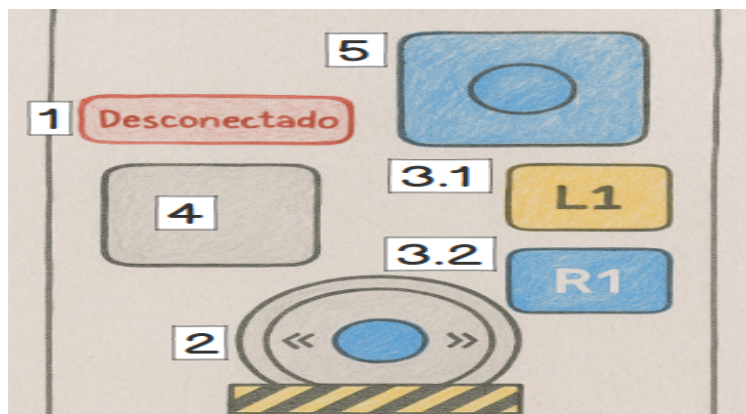


Ilustración N°2 “Wireframe GUI”

Este wireframe se utilizó como referencia directa para la implementación de la interfaz en flutter, donde cada forma del boceto se tradujo en widgets básicos como Container, Row, Column, Switch y controles táctiles. El diseño inicial permitió definir claramente la distribución de la pantalla y la lógica de interacción antes de aplicar estilos visuales definitivos.

6. Implementación

En esta sección del informe se presentan los resultados obtenidos hasta el momento en el desarrollo del proyecto. Se incluye la justificación de la configuración del robot a partir de principios físicos de movimiento básico estudiados en la asignatura FI 035—Introducción a la Física, una descripción de los componentes más relevantes del sistema implementado y una captura de la interfaz gráfica de usuario ya desarrollada, junto con la explicación de sus elementos y funciones.

6.1 Fundamentos de los movimientos

Se presentan los distintos tipos de movimientos utilizados para el correcto funcionamiento del robot desde la primera aplicación física-matemática hasta la del último uso del robot. Se divide en tres secciones fundamentales que justifican a detalle el funcionamiento matemático del robot, las cuales son:

- Caída del Bloque de Lego:

Se toma en consideración un diagrama de cuerpo libre para utilizar la segunda ley de newton para determinar la aceleración ideal requerida para el correcto funcionamiento; tomando en cuenta diferentes datos, tales como: Peso del Bloque, Ángulo de inclinación de la caída y El roce efectuado del mismo plástico (DINÁMICA).

Datos:

Masa (m): 0.00232 kg.

Coefficiente de roce cinético (uk): 0,654 (estimado para plástico ABS).

Gravedad (g): 9.8 m/s².

Ángulo de inclinación(theta) : 60°.

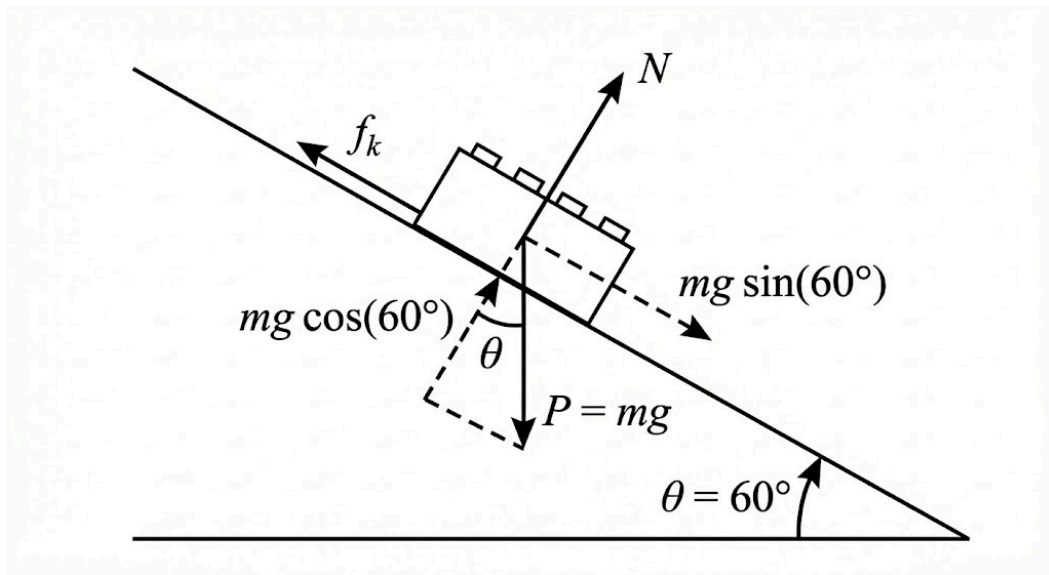


Ilustración N°3 “DCL”

Desarrollo: Analizar las fuerzas al ubicar el bloque sobre la superficie inclinada, que actúan dos fuerzas opuestas en el eje del movimiento: componente horizontal P_x que favorece la caída y la fuerza de roce f_k que se opone a ella.

Fuerza Normal (N): Es la fuerza de reacción perpendicular a la superficie.

$$N = m * g * \cos(60) \text{ N} = 0.00232 * 9.8 * 0.5 \text{ N} = \mathbf{0.01137 \text{ Newtons.}}$$

Fuerza de Roce Cinético (f_k): Es la resistencia opuesta al movimiento, calculada con el coeficiente estimado. $f_k = \mu_k * N$ $f_k = 0.654 * 0.01137$ $f_k = 0.00744 \text{ Newtons}$

Componente del Peso en el eje X (P_x): Es la fuerza generada por la gravedad que empuja el bloque hacia abajo por la rampa. $P_x = m * g * \sin(60)$ $P_x = 0.00232 * 9.8 * 0.866$

$$\mathbf{P_x = 0.01969 \text{ Newtons}}$$

Teniendo los datos ya resueltos se aplica la segunda ley de Newton (Suma de fuerzas = $m * a$) la cual dirá la aceleración resultante:

$$a = (P_x - f_k) / m \quad a = (0.01969 - 0.00744) / 0.00232 \quad a = 0.01225 / 0.00232 \quad \mathbf{a = 5.28 \text{ m/s}^2}$$

El dato de la aceleración de 5.28 m/s^2 es la ideal, menor a ella obtendrán

resultados poco eficientes en relación al tiempo de ejecución y mayor a ella se obtendrán múltiples errores desde caída con choques y el bloque expulsado hasta el no reconocimiento del sensor de color.

- **Cálculo del Tiempo Real de posicionamiento**

El giro de clasificación opera en un ángulo simétrico de -35° a $+35^\circ$ donde el ángulo 0° representa el reposo del clasificador. A diferencia del motor de empuje, que realiza un movimiento fijo ("golpe"), el motor de posicionamiento tiene trayectorias que cambian debido al sensor de color para el correcto ordenamiento del bloque.

Determinación de la velocidad real (w) para calcular la eficiencia del giro se calcula el recorrido total del robot de un extremo a otro, de -35° a $+35^\circ$.

Distancia Máxima = $35 + 35 = 70^\circ$

Fórmula = $t(\max) = \text{Diferencia Angular} / \text{velocidad real}$

La velocidad real está limitada por el mismo Hub del lego la cual es $300^\circ/\text{s}$.

$$t(\max) = 70^\circ / 300^\circ/\text{s} = 0.23 \text{ s}$$

el tiempo de recorrido es 0,23 segundos, junto con la velocidad real que esta limitada por el Hub del lego spike prime es lo mas eficiente debido a que el recorrido por detección de color es minimo para girar y volver del reposo, los datos y resultados son a base del peor caso posible, ir de izquierda a derecha, de extremo a extremo.

- **Eficiencia Motor de empuje (radio de empuje = 4 cm)**

el motor de empuje se encarga de colocar los bloques en la sección indicada de su color

se selecciona un recorrido de 180° para empujar el bloque, es eficiente ya que al seleccionar no debe dar un giro completo, como si fuera 360° tardaría más en ejecutarse, implementando así efectividad en el ángulo de empuje ya que da un ida y retorno simple y rápido, junto con el tope máximo de velocidad admitida por el Hub ($300^\circ/\text{s}$) para validar que esta configuración ofrece una fuerza óptima en un tiempo mínimo, se aplican las siguientes fórmulas:

Datos:

1. **Velocidad Angular Máxima (omega):** $300^\circ/\text{s}$ (Límite operativo establecido en

el Hub).

2. **Radio de Palanca (r):** 4 cm (Longitud del brazo de empuje).
3. **Desplazamiento (theta):** 180°.

-Cálculo del tiempo de ciclo: determina cuanto tiempo tarda el brazo en completar el empuje al color indicado:

$$t = \text{desplazamiento} / \text{velocidad angular} = 180^\circ / (300^\circ/\text{s}) = 0.6 \text{ segundos}$$

-Cálculo de la Fuerza de empuje (Newtons): determina la fuerza con la que se empuja el bloque (utilizamos el torque del motor = 18 N*CM).

$$\text{FUERZA} = \text{TORQUE} / \text{RADIO} = 18 \text{ N*CM} / 4\text{CM} = 4.5 \text{ NEWTONS}$$

Los siguientes datos extraídos son necesarios ya que se necesita calcular la fuerza de empuje para asegurar que el motor tiene potencia suficiente para mover el bloque y añadimos velocidad para generar impacto teniendo en cuenta que buscamos la eficiencia respecto al tiempo.

6.2 Descripción del sistema

Sobre el sistema usado para el manejo del robot, estamos usando Pybricks, mediante Pybricks tenemos distintas formas de manejar el robot.

Forma 1, archivo Version1.0.0.py: Este es nuestro código inicial el cual se opera de forma automática, es decir no es necesario una persona que controle el robot.

```
Version1.0.0.py > ...
1  from pybricks.hubs import PrimeHub
2  from pybricks.pupdevices import Motor, ColorSensor
3  from pybricks.parameters import Port, Color
4  from pybricks.tools import wait
5
6
7  hub = PrimeHub()
8
9  sensor_color = ColorSensor(Port.B)
10 motor_posicion = Motor(Port.A)
11 motor_empuje = Motor(Port.D)
12
13 while True:
14     color_detectado = sensor_color.color()
15
16     if color_detectado == Color.RED or color_detectado == Color.YELLOW:
17
18
19         motor_posicion.run_target(1000, -35)
20         if color_detectado == Color.RED:
21
22             motor_empuje.run_angle(1000, -180)
23
24         elif color_detectado == Color.YELLOW:
25             motor_empuje.run_angle(1000, 180)
26
27
28     elif color_detectado == Color.GREEN or color_detectado == Color.BLUE:
29
30         motor_posicion.run_target(1000, 35)
31         if color_detectado == Color.GREEN:
32             motor_empuje.run_angle(1000, 180)
33
34         elif color_detectado == Color.BLUE:
35             motor_empuje.run_angle(1000, -180)
36
37     wait(10)
```

Ilustración N°5 “Codigo Version1.0.0.py”.

Puntos claves de este código:

línea 9, el sensor que esta implementando en el robot se guarda como referencia en: sensor_color

línea 13, se tiene un bucle que permite que el robot siga funcionando hasta que se apague el programa de forma manual.

línea 14, se guarda el color detectado en: color_detectado.

línea 16-36, Este es el algoritmo que desarrollamos para guardar en 4 casillas, en la cual usamos condicionales if y elif dependiendo del color que se detecta.

Forma 2, archivo Version1.0.1.py: En esta versión se ocupa de base el código anterior pero se implementa una forma de manejar el robot, en este caso sería mediante teclas.

```
Version1.0.1.py > ...
1  # este codigo lee la entrada del teclado para controlar el robot
2  from pybricks.hubs import PrimeHub
3  from pybricks.pupdevices import Motor, ColorSensor
4  from pybricks.parameters import Port, Color
5  from pybricks.tools import wait
6  from sys import stdin
7  from select import poll
8
9
10 hub = PrimeHub()
11
12 sensor_color = ColorSensor(Port.B)
13 motor_posicion = Motor(Port.A)
14 motor_empuje = Motor(Port.D)
15
16 teclado = poll()
17 teclado.register(stdin)
18
19 while True:
20     if teclado.poll(0):
21         key = stdin.read(1)
22         if key=='w':
23             motor_posicion.run_target(1000, -35)
24             motor_empuje.run_angle(1000, -180)
25
26         elif key == 'a':
27             motor_posicion.run_target(1000, -35)
28             motor_empuje.run_angle(1000, 180)
29         elif key=='s':
30             motor_posicion.run_target(1000, 35)
31             motor_empuje.run_angle(1000, 180)
32
33         elif key=='d':
34             motor_posicion.run_target(1000, 35)
35             motor_empuje.run_angle(1000, -180)
36
37     wait(10)
38
```

Ilustración N°6 “Codigo Version1.0.1.py”.

Puntos claves de este código:

línea 6-7, se importan librerías que sí permite pybricks, para aplicar métodos que permitan mover el robot dependiendo a qué lugar desea el usuario.

Forma 3, carpeta ProyFlutter: En esta carpeta se tiene 2 versiones, app_control y app2_control, estas son 2 aplicaciones creadas para el manejo del robot mediante una conexión bluetooth, además de una interfaz gráfica para el usuario.

```
46 // INICIAR TECLADO BLUETOOTH
47 // -----
48 Future<void> startKeyboard() async {
49   try {
50     await keyboard.start();
51     setState(() {
52       connected = true;
53       status = "Conectado ✓ (Teclado Bluetooth)";
54     });
55   } catch (e) {
56     setState(() {
57       status = "Error: $e";
58     });
59   }
60 }
61
62 // ENVIAR TECLA AL PC
63 Future<void> sendKey(String key) async {
64   if (!connected) return;
65
66   try {
67     await keyboard.sendText(key);
68   } catch (e) {
69     debugPrint("Error enviando: $e");
70   }
71 }
72
73 @override
74 Widget build(BuildContext context) {
75   return Scaffold(
76     backgroundColor: Colors.black,
77     body: Padding(
78       padding: const EdgeInsets.all(12),
79       child: Column(
80         children: [
81           // ----- TOP BAR -----
82           Row(
83             children: [
84               Expanded(
85                 child: Container(
86                   padding: const EdgeInsets.all(10),
87                   decoration: BoxDecoration(
88                     color: connected ? Colors.green : Colors.red,
89                     borderRadius: BorderRadius.circular(10),
90                   ),
91                 child: Text(
92                   status,
93                   textAlign: TextAlign.center,
94                   overflow: TextOverflow.ellipsis,
95                   style: const TextStyle(
96                     fontSize: 18, fontWeight: FontWeight.bold),
97                 ),
98             ),
99           ],
100
```

```

121 // ----- LEFT BUTTONS -----
122 Column(
123   children: [
124     ElevatedButton(
125       onPressed: () => sendKey("w"),
126       style: ElevatedButton.styleFrom(
127         minimumSize: const Size(100, 80),
128       ),
129       child: const Text("L1 → w",
130         style: TextStyle(fontSize: 24)),
131     ),
132     const SizedBox(height: 20),
133     ElevatedButton(
134       onPressed: () => sendKey("s"),
135       style: ElevatedButton.styleFrom(
136         minimumSize: const Size(100, 80),
137       ),
138       child: const Text("R1 → s",
139         style: TextStyle(fontSize: 24)),
140     ),
141   ],
142 ),
143
144 // ----- COLOR BOX (decorativo) -----
145 Container(
146   width: 150,
147   height: 150,
148   decoration: BoxDecoration(
149     color: sensorColor,
150     borderRadius: BorderRadius.circular(20),
151     border: Border.all(color: Colors.white, width: 3),
152   ),
153 ),
154
155 // ----- JOYSTICK -----
156 SizedBox(
157   width: 220,
158   height: 220,
159   child: Joystick(
160     mode: JoystickMode.all,
161     listener: (details) {
162       // Usa WASD para movimientos
163       if (details.y < -0.5) sendKey("w");
164       else if (details.y > 0.5) sendKey("s");
165
166       if (details.x < -0.5) sendKey("a");
167       else if (details.x > 0.5) sendKey("d");
168     },
169   ),
170 ),

```

Ilustración N°7 “Codigo app2_control”.

línea 74-151, es como está decorado la aplicación, además de que envía texto hacia el computador.

línea 155-168, un joystick que envía texto como “WASD”.

URL:

<https://github.com/saoudahmedlanchipa-code/LegoSpikePrimeProyecto1Sorting>

6.2.1 Cliente

Como se representa el cliente en el sistema creado:

Primero que todo el cliente viene a ser las instrucciones que se van dando mediante el código realizado y el cual recibe el hub del lego Spike Prime. En este caso los mensajes se envían mediante bluetooth.

```
15
16     if color_detectado == Color.RED or color_detectado == Color.YELLOW:
17
18
19         motor_posicion.run_target(1000, -35)
20         if color_detectado == Color.RED:
21
22             motor_empuje.run_angle(1000, -180)
23
24         elif color_detectado == Color.YELLOW:
25             motor_empuje.run_angle(1000, 180)
26
27
28     elif color_detectado == Color.GREEN or color_detectado == Color.BLUE:
29
30         motor_posicion.run_target(1000, 35)
31         if color_detectado == Color.GREEN:
32             motor_empuje.run_angle(1000, 180)
33
34         elif color_detectado == Color.BLUE:
35             motor_empuje.run_angle(1000, -180)
36
```

Ilustración N°8 “Codigo Version1.0.0.py”.

Las instrucciones que se le envían al hub vienen ser:

Las líneas que tienen métodos de la librería Pybricks, en este caso `.run_target` y `.run_angle`.

El `.run_target` se usa a la hora de querer moverlo a una posición específica, este método se necesita para mover el motor de las casillas designadas.

El .run_angle se usa para mover por angulo el motor, se encarga de disparar el bloque para el lado izquierdo o lado derecho, por eso se representa los ángulos como 180 y -180.

```
155 // ----- JOYSTICK -----
156 SizedBox(
157   width: 220,
158   height: 220,
159   child: Joystick(
160     mode: JoystickMode.all,
161     listener: (details) {
162       // Usa WASD para movimientos
163       if (details.y < -0.5) sendKey("w");
164       else if (details.y > 0.5) sendKey("s");
165
166       if (details.x < -0.5) sendKey("a");
167       else if (details.x > 0.5) sendKey("d");
168     },
169   ),
170 ),
171 ],
172 ),
```

Ilustración N°9 “Codigo app2_control”.

De la propia app de flutter también se envían mensajes pero no directamente al hub, sino al computador y mediante la Version1.0.1.py se pueden controlar y usar los métodos anteriormente mencionados.

URL:

<https://github.com/saoudahmedlanchipa-code/LegoSpikePrimeProyecto1Sorting>

6.2.2 Servidor

El servidor en este caso viene ser el Lego Technic Large Hub, va ser el encargado de administrar los mensajes que se envían mediante bluetooth y transmitirnos otros mensajes.

```

7   hub = PrimeHub()
8
9   sensor_color = ColorSensor(Port.B)
10  motor_posicion = Motor(Port.A)
11  motor_empuje = Motor(Port.D)
12
13  while True:
14      color_detectado = sensor_color.color()
15

```

Ilustración N°10 “Codigo Version1.0.0.py”.

El hub nos permite ejecutar los motores y enviarnos mensajes, en este caso el color que detecta.

URL:

<https://github.com/saoudahmedlanchipa-code/LegoSpikePrimeProyecto1Sorting>

6.2.3 Interfaz gráfica de usuario (GUI)

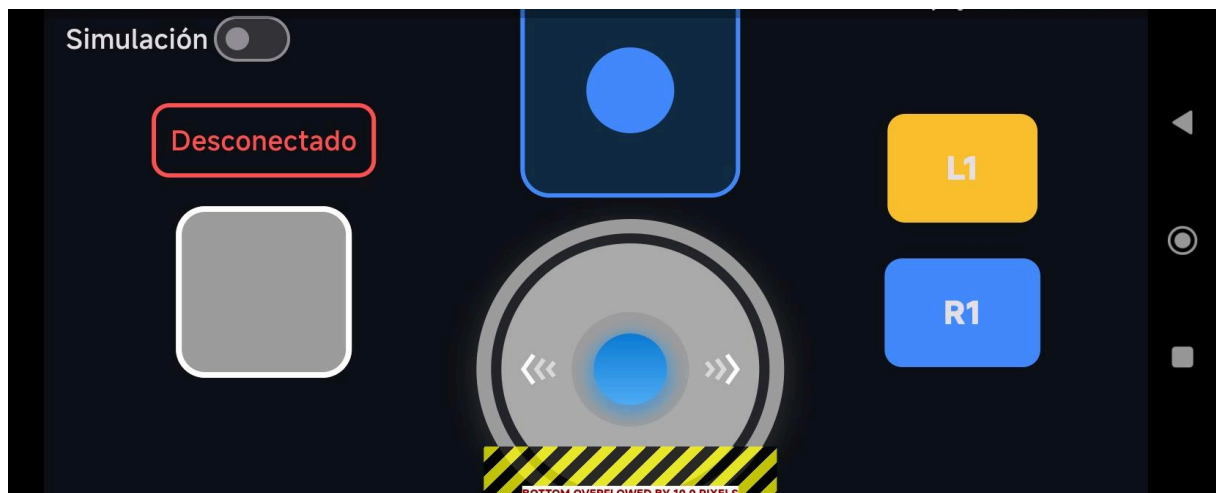


Ilustración N°11 “Interfaz Grafica de Usuario”.

1. Indicador de estado de conexión

Representado mediante un rectángulo que actualiza su estado de “desconectado” a la hora de hacer click en el. realiza una búsqueda por bluetooth de los dispositivos cambiando su estado de “desconectado” a “buscando” y en caso de encontrar la señal

específica del robot sorter el estado queda en “conectado”

2. Joystick

Representado como un círculo azul dentro de un círculo mayor como límites el cual a la hora de ser manipulado por el usuario. Este envía señales al hub del robot sorter para que este mueva el motor de movimiento y poder rotarlo de lado a lado. este solo permite movimiento horizontal

3. Botones (L1) y (R1)

Representados como dos botones de distintos colores e identificados como “L1” y “R1”, ambos envían instrucciones al hub del robot sorter para que este mueva el motor de empuje, siendo “L1” hacia el lado izquierdo y “R1” el lado derecho.

4. Muestreo de color

Representado como un cuadro gris cuando no hay conexión o el robot no tiene colores detectados por el sensor, en caso de si tener colores sobre el sensor este se mostrara en el recuadro de muestreo de color para que el usuario efectue el ordenamiento con efectividad.

5. Panel informativo de dirección

Representado como un cuadrado azul con un círculo dentro. este recuadro es una muestra de movimiento del joystick para otorgar comodidad visual al usuario

7. Resultados

7.1 Estado actual del proyecto

El estado actual de la solución planteada (Maquinaria de clasificación de materiales) se encuentra en una fase de funcionalidad completa respecto a su mecánica y

cumplimiento de los requerimientos funcionales definidos.

El robot ha logrado exitosamente la automatización total de clasificar bloques respecto a su color tanto de manera manual guiada por el usuario a través de un control remoto. La solución es capaz de operar de manera automática y manual replicando la tarea industrial sin exponer a funcionarios de empresas mineras.

El robot ha logrado alcanzar los siguientes requerimientos funcionales:

-Identificación de Materiales: Mediante el sensor de color el robot es capaz de reconocer el color del bloque y separarlo en base a ello.

-Modo de operación Híbrida: El robot ha sido capaz de operar de manera automática, o sea, un bucle de ordenamiento ejecutando la clasificación de estos mismo bloques hasta no detectar ninguno. Como a su vez un modo manual que implementa un control remoto para que el usuario pueda clasificar los bloques el mismo

-Flujo Continuo: El robot es capaz de reconocer y ordenar varios bloques de LEGO sucesivamente, manteniendo un constante control de la clasificación.

-Clasificación de Celdas por su color: El robot es capaz de moverse y empujar el bloque respecto al destino de su color, capaz de sincronizar el movimiento de posición con el empuje del bloque para que el bloque sea colocado en base a su color, enfatizando en su ordenamiento.

Respecto a los requerimientos no funcionales del robot el reporte actual son los siguientes:

-Disponibilidad: La disponibilidad del robot depende netamente de su conexión con el programa, ante una pérdida de conexión con el hub y Pybricks el sistema se detiene por completo. Ante estos casos el robot debe ser reiniciado o desconectado con el hub y reintegrar la misma conexión, siendo su restauración casi de manera inmediata.

-Robustez: El robot no es tan robusto respecto al manejo mecánico debido a que si llega a presentar obstrucción por bloques, necesita intervención inmediata para el correcto flujo, carece de un mecanismo de auto barrido ante estancamiento.

-Rendimiento: El robot en condiciones de flujo normales tanto automáticas como manual cumple con los tiempos de respuesta esperados.

-Usabilidad: Para mejorar la usabilidad y comodidad se implementó un esquema en base a un control DualShock 4 (PS4) ya que es más conocido a nivel del usuario con botones como R1 Y L1, se debe mantener una capacitación antes de usarlo para que el usuario conozca con detenimiento el accionar de cada botón con el fin de cumplir el requerimiento de una interfaz clara e intuitiva. La capacitación para su correcto uso es mínima debido a la familiaridad de los botones implementados en relación al DualShock 4, en un tiempo de intervalo de 5 a 10 minutos como máximo. Los botones al ser grandes y claros minimizan el posible error de pulsación fantasma o más bien error de selección de botón, esto garantiza un accionar claro y conciso.

7.2 Problemas encontrados y solucionados

Durante el desarrollo del proyecto se presentaron diversos problemas relevantes, principalmente asociados a la implementación de la interfaz gráfica y la comunicación entre la aplicación y el hub. Uno de los principales inconvenientes fue que, en una primera instancia, la aplicación no lograba detectar ni buscar el hub del robot mediante Bluetooth. Se descubrió que esto se debía a las restricciones del sistema operativo Android, el cual provoca que las aplicaciones solicitan explícitamente permisos de acceso a la ubicación y al Bluetooth para permitir la búsqueda y conexión con dispositivos externos. Este problema se solucionó incorporando una sección de código que solicita dichos permisos al usuario al momento de ejecutar la aplicación, permitiendo que la interfaz pudiera detectar el hub.

Otro problema fue con el framework Flutter, ya que se encontraba instalado solamente en el computador personal del encargado de la programación. Esto provocaba dificultades al momento de realizar pruebas, dado que cualquier error detectado en las clases debía ser corregido posteriormente en el hogar, retrasando el avance del proyecto. Para solucionar este problema, se coordinó la instalación de Flutter en todos los equipos del grupo.

8. Conclusión

En esta etapa del proyecto se lograron avances como la materialización del sistema, abordando el análisis, diseño e implementación del robot. Se definieron los requerimientos funcionales y no funcionales, se diseñó una arquitectura de software basada en el modelo cliente-servidor y se desarrollaron distintas formas de operación del robot, incluyendo un modo automático y control manual.

Durante el desarrollo se enfrentaron dificultades principalmente asociadas a ajustes en el ensamblaje del robot, errores de codificación y limitación de tiempo, las cuales fueron resueltas mediante pruebas, correcciones en el código y reuniones extras. Como trabajo pendiente para las siguientes etapas, se considera la optimización del rendimiento del robot, la ejecución de pruebas finales de funcionamiento y la presentación final.

6. Referencias

Ubuy. (s. f.). *LEGO Education SPIKE Prime Set (45678)*. Recuperado el 15 de octubre de 2025, de <https://www.ubuy.cl/sp/product/GMEH1T4-lego-education-spike-prime-set>.

Tradeinn. (s. f.). *LEGO Education SPIKE Prime Expansion Set (45681)*. Recuperado el 15 de octubre de 2025, de <https://www.tradeinn.com/kidinn/es/lego-juego-de-construccion-education-spike-prime-expansion-set-45681/141562226/p>.

Lenovo. (s. f.). *Lenovo LOQ 15/Ax9*. Recuperado el 15 de octubre de 2025, de <https://www.lenovo.com/cl/es/p/notebooks/loq-laptops/lenovo-loq-15iax9/len101q0006>.

RIPLEY. (s. f.). *TABLET SAMSUNG GALAXY TAB S8 PLUS + KEYBOARD COVER AMD RYZEN 9 8 GB RAM 12.4"*. Recuperado el 15 de octubre de 2025, de [TABLET SAMSUNG GALAXY TAB S8 PLUS + KEYBOARD COVER AMD RYZEN 9 8 GB RAM 12.4"](#)

Paris.cl. (s. f.). *Notebook IdeaPad Slim 3 AMD Ryzen7 5825U 16GB 512GB SSD Windows 11 Home 15.6" FHD Azul Abyss*. Recuperado el 15 de octubre de 2025, de <https://www.paris.cl/notebook-ideapad-slim-3-amd-ryzen7-5825u-16gb-512gb-ssd-windows-11-home-156-fhd-azul-abyss-106450999.html>.

Paris.cl. (s. f.). *Notebook Gamer Acer Aspire G A515-58GM-56XX-1: Intel Core i5, NVIDIA RTX 2050, 16GB RAM, 512GB SSD, 15.6" FHD*. Recuperado el 15 de octubre de 2025, de <https://www.paris.cl/notebook-gamer-aspire-g-a515-58gm-56xx-1-intel-core-i5-8-nucleos-nvidia-rtx-2050-16gb-ram-512gb-ssd-156-700915999.html>.

HP. (2025). *Notebook HP Pavilion Plus 14-ew1002la*. [Paris.cl](#). Recuperado el 27 de noviembre de 2025, de [Notebook HP Pavilion Plus 14-EW1002LA Intel Ultra 7 32GB](#)

[RAM 1TB SSD 14" 3K OLED Windows 11 Home HP | Paris.cl.](#)

Glassdoor. (2025). *Sueldos para Jefe de Proyecto en Chile (Basado en datos de Claro Chile)*. Recuperado el 26 de noviembre de 2025, de [KEY NOT FOUND: ei-salaries.seo-metadata.el-details.title.singular | Glassdoor.](#)

Glassdoor. (2025). *Sueldos para el puesto de Programador en Chile*. Recuperado el 26 de noviembre de 2025, de [Sueldo: Programador \(Noviembre, 2025\) | Glassdoor.](#)

Glassdoor. (2025). *Salario Mensual para Computer Hardware and Software en Latam (Chile)*. Recuperado el 26 de noviembre de 2025, de [KEY NOT FOUND: ei-salaries.seo-metadata.el-details.title.singular | Glassdoor.](#)

Glassdoor. (2025). *Salario mensual para Analista y Documentador Técnico en TrackTec (Chile)*. Recuperado el 26 de noviembre de 2025, de [KEY NOT FOUND: ei-salaries.seo-metadata.el-details.title.singular | Glassdoor.](#)

Bodega Digital. (2025). *Licencia Microsoft Office 2024 Professional Plus (Código Digital)*. Recuperado el 27 de noviembre de 2025, de [Office 2024 Professional Plus • Bodega Digital.](#)

BrickLink. (s.f.). *Part 3001: Brick 2 x 4*. Recuperado el 14 de diciembre de 2025, de <https://www.bricklink.com/v2/catalog/catalogitem.page?P=3001>

Wikipedia. (2024). *Acrilonitrilo butadieno estireno*. En Wikipedia, la enciclopedia libre. Recuperado el 15 de diciembre de 2025, de https://es.wikipedia.org/wiki/Acrilonitrilo_butadieno_estireno.

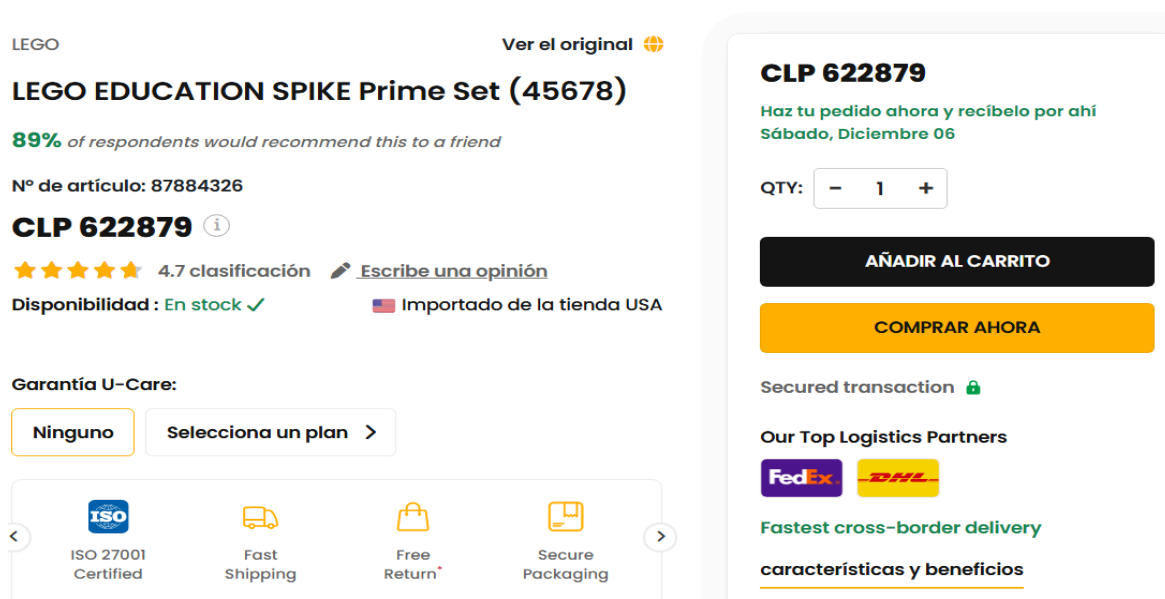
Hurbain, P. (2024). *LEGO Motor Characteristics: Comparative Test Data*. PhiloHome. Recuperado de <https://www.philohome.com/motors/motorcomp.htm>

Anexos

9.Anexo 1

Se adjunta fotografía de los precios reales de los productos (Hardware) mencionados en la tabla de la sección de costo de hardware.

1. SET LEGO SPIKE PRIME.



LEGO Ver el original

LEGO EDUCATION SPIKE Prime Set (45678)

89% of respondents would recommend this to a friend

Nº de artículo: 87884326

CLP 622879

★★★★★ 4.7 clasificación [Escribe una opinión](#)

Disponibilidad: **En stock** Importado de la tienda USA

Garantía U-Care:

Ninguno **Selecciona un plan** >

ISO 27001 Certified Fast Shipping Free Return Secure Packaging

CLP 622879

Haz tu pedido ahora y recíbelo por ahí
Sábado, Diciembre 06

QTY: - 1 +

AÑADIR AL CARRITO

COMPRAR AHORA

Secured transaction

Our Top Logistics Partners

FedEx **DHL**

Fastest cross-border delivery

características y beneficios

Ilustración N°12 “LEGO SPIKE PRIME”.

2. EXTENSION LEGO SPIKE PRIME.



Lego Juego de construcción Education Spike Prime Expansion Set 45681

167202 CLP\$

PVR: 183738 CLP\$
Ahorras: 16536.00 CLP\$ (-9%)

★★★★★ 1 Opiniones | 0 Preguntas

En stock. **Envío inmediato.** Recíbelo entre **Juev. 4 Dic.** y **Vier. 5 Dic.**

Añadir a la cesta

Compra este producto y gana 155 CoINNs / **1671 CLP\$**

Ilustración N°13 “EXTENSION LEGO SPIKE PRIME”

3. NOTEBOOK LOQ GEN 9.



Únete a la partida

Lenovo LOQ Gen 9 (15\" Intel)

★★★★★ 4.6 (70)

☐ Aplicar cupón \$15,000 de descuento extra!

Obtén un \$15,000 de descuento adicional en Lenovo LOQ Gen 9 (15\" Intel), aplicado automáticamente a tu carrito. ¡Por tiempo limitado!

A partir de

\$769.993

Ilustración N°14 “Notebook loq gen 9”.

4. SAMSUNG GALAXY TAB S8 ULTRA.

SAMSUNG

TABLET SAMSUNG GALAXY TAB S8 PLUS + KEYBOARD COVER AMD RYZEN 9 8 GB RAM 12.4"

SKU: 2000389779441P

Normal

~~\$1.149.990~~

Internet

\$749.990

-35%

COLOR:



Ilustración N°15 “GALAXY TAB S8 PLUS”.

5. LENOVO IDEAPAD 3 15.


Lenovo

**Notebook IdeaPad Slim 3 AMD
Ryzen 7 16GB RAM 512GB SSD...**

Vendido por [París](#)

SKU 106450999

24% **\$529.990**
~~\$699.990~~

 **10 cuotas sin interés** **Cupón APP: COMPUTACION10**

Envío Rápido ⚡

Comprar ahora **Agregar al carro**

Ilustración N°16 “IDEAPAD 3 15”.

6. NOTEBOOK ASPIRE G.

Acer


**Notebook Gamer Aspire G A515-
58GM-56XX-1 Intel Core i5 8...**

Vendido por [París](#)

SKU 700915999

★ ★ ★ ★ ★ 4.6 (15)

42% **\$599.990**
~~\$1.049.990~~

 **10 cuotas sin interés** **Cupón APP: COMPUTACION10**

Cupón APP: INTEL10 **Envío Rápido** ⚡

Ilustración N°17 “GAMER ASPIRE 5”.

7. HP PAVILION LAPTOP.

Producto no disponible

HP

Notebook HP Pavilion Plus 14-EW1002LA Intel Ultra 7 32GB RA...

Vendido por **Paris**

SKU 924154999

\$1.599.990

10 cuotas sin interés
 Cupón APP: COMPUTACION10

Cupón APP: INTEL10

Se agotaron las últimas unidades

¡Lo sentimos! Alguien más compró la última unidad de este producto y ya no está disponible.

Buscar similares

Ilustración N°18 “HP PAVILION”.

8. TAMAÑO DEL LEGO REAL PARA LA OBTENCIÓN DE DATOS APLICANDO LA FÍSICA

Catálogo: Piezas: Ladrillo: 3001

Ladrillo 2 x 4

Artículo n°: 3001 Artículo alternativo: 3001f1, 3556, 15589, 54534, 72841

[Consulta la guía de precios](#)
[Compra](#)

Seleccionar color ▼

Información del artículo

Años de lanzamiento: N/A

1978 - 2025

Peso: 2,32g

Vigas Dim.: 2 x 4 x 1 pulgada Pack

. Dim.: 1,6 x 3,2 x 1,15 cm

El ítem consta de

El objeto aparece en

[3929 Sets](#)
[19 Minifiguras](#)
[14 Partes](#)
[33 Libros](#)
[17 Equipo](#)

Inventario de mi tienda
[Añadir a mi inventario](#)
 145534 lotes a la venta

Mi lista de buscados
[Añadir a mi lista](#)
 de buscados en 1814872 listas de buscados

Mi colección
[Añadir a mi colección](#)
 En 16532 Colecciones

Notas adicionales:

Esta parte tiene una vertiginosa variedad de variantes, de las cuales solo unas pocas tienen entradas de catálogo separadas.

[Colapso ▲](#)

Elementos relacionados:

Este artículo es una variante de molde del(los) siguiente(s) artículo(s):

- Parte [3001 especial](#) de ladrillo 2 x 4 especiales (ladrillos especiales, ladrillos de prueba y/o prototipos)
- Parte [bhol04](#) Ladrillo 2 x 4 sin tubos inferiores
- Parte [3001old](#) Brick 2 x 4 sin soportes transversales
- Parte [3001oldb](#) Ladrillo 2 x 4 sin soportes cruzados, con agujero en la parte superior

[Colapso ▲](#)

Ilustración N°19 “Bloques usados”.