UNIVERSIDAD DE TARAPACÁ

FACULTAD DE INGENIERÍA

DEPARTAMENTO DE INGENIERÍA CIVIL EN COMPUTACIÓN E INFORMÁTICA

Ingeniería@
Computación e Informática

Informe Fase 2
“Brazo Robótico de agarre con LEGO”

Alumnos: Carlos Cossio
Bryan Palacios
Franco Churata
Benjamín Aguilera
Joaquín Quezada

Asignatura: Proyecto I

Profesor: Baris Nikolai
Klobertanz Quiroz

Historial de Cambios

Fecha	Versión	Descripción	Autor(es)
26/09/2025	1.0	Concepción del documento.	Carlos Cossio
29/09/2025	1.1	Recopilación de datos del proyecto.	Benjamín Aguilera
1/10/2025	1.2	Redacción de ítems iniciales.	Franco Churata
6/10/2025	1.3	Inclusión de planificación de recursos y riesgos.	Carlos Cossio
10/10/2025	1.4	Última revisión y pulimento de detalles.	Bryan Palacios
17/10/2025	1.5	Finalización del informe 1.	Carlos Cossio
11/12/2025	1.6	Avance del informe 2.	Bryan Palacios
12/12/2025	1.7	Revisión de detalles del informe.	Carlos Cossio, Benjamín Aguilera
14/12/2025	1.8	Finalización del informe 2.	Carlos Cossio, Benjamín Aguilera, Bryan Palacios, Joaquin Quezada, Franco Churata

Tabla de Contenidos

1. Panel General

1.1. Introducción

El vehículo robótico requiere un sistema independiente que permita cargar los bloques de manera óptima y así completar correctamente la cadena de transporte. Durante el desarrollo del semestre, el grupo llevará a cabo la construcción, integración y programación de un brazo robótico de agarre. El objetivo principal es diseñar un sistema automatizado capaz de sujetar, levantar y depositar bloques en un vehículo robótico de transporte, el cual será ensamblado y programado por otro grupo dentro del mismo proyecto colaborativo.

Este proyecto forma parte de una colaboración entre tres equipos, donde cada uno cumple un rol definido: diseño del brazo de agarre, desarrollo del vehículo autónomo y coordinación del sistema conjunto mediante comunicación y control sincronizado.

El trabajo se centra en las etapas de experimentación mecánica, diseño estructural y cinemático, ensamblaje funcional, programación del sistema de control y documentación técnica del proceso. Asimismo, se aplican metodologías de trabajo colaborativo, gestión de tareas mediante herramientas digitales y principios de ingeniería de control y automatización, con el propósito de lograr una integración eficiente entre los distintos subsistemas del proyecto.

1.2. Objetivos

1.2.1. Objetivo General

Desarrollar y programar un brazo robótico de agarre que permita transferir bloques hacia un vehículo de carga automatizado, mediante una interfaz manual. Esta herramienta beneficiará al sistema completo del proyecto, ya que asegura una transferencia eficiente y confiable de materiales, mejorando la coordinación y el desempeño global dentro del entorno experimental.

1.2.2. Objetivos Específicos

- Analizar distintos diseños de sistemas de agarre y mecanismos de sujeción, evaluando su eficiencia estructural y capacidad de manipulación mediante piezas LEGO, con el fin de identificar el diseño más eficiente y funcional según los criterios de estabilidad y facilidad de uso.
- Diseñar y ensamblar un modelo funcional del brazo robótico, asegurando que presente estabilidad mecánica y precisión en el movimiento de agarre. Se evaluará la estabilidad mediante pruebas de carga, y el brazo robótico deberá ser capaz de realizar tareas de sujeción de objetos de manera eficiente y sin fallos durante su funcionamiento.
- Programar la secuencia de movimientos del sistema de agarre utilizando la plataforma de programación visual y posteriormente migrar el control a código Python para una mayor flexibilidad.
- Coordinar y colaborar con los demás grupos involucrados en el proyecto para sincronizar las funciones de carga y descarga entre el brazo robótico y el vehículo automatizado.
- Documentar el proceso de desarrollo de software mediante informes técnicos y una bitácora semanal, en la cual un integrante del equipo se encargará de registrar los avances, ajustes y resultados de cada etapa del proyecto. El encargado de hacer la bitácora cambiará semanalmente, rotando entre los miembros del equipo para fomentar el aprendizaje integral de todos los integrantes en la gestión de tareas del proyecto.

1.3. Restricciones

- Se debe programar en Python.
- Se debe utilizar el set LEGO Spike Prime y su kit de expansión.
- El robot debe ser capaz de tomar y soltar bloques sin errores.
- Se debe mantener conexión entre el computador y el robot mediante una conexión inalámbrica estable.
- Se debe mostrar una interfaz gráfica en el computador.
- Cantidad de integrantes limitada a cinco.

1.4. Entregables

- Bitácoras semanales: informe del avance del grupo, rotando el responsable cada semana.
- Carta Gantt: cronograma del proyecto con las actividades principales.
- Informe de Formulación: documento actual que describe la organización, planificación y objetivos.
- Presentación Final: exposición del funcionamiento del brazo robótico y los resultados obtenidos.
- Código base del cliente y servidor, publicados en un repositorio en Github.
- Una página del proyecto en la plataforma Redmine.

2. Organización del Personal

Para asegurar el desarrollo y eficiente del proyecto , definimos una estructura interna de trabajo basada en las habilidades de cada integrante del equipo, esta organización nos permite distribuir las responsabilidades de manera equilibrada, garantizando que todas las tareas sea abordadas de forma adecuada.

2.1. Descripción de los Roles

Para organizar de manera eficiente el trabajo en equipo y asegurar que el proyecto avance correctamente, se realizó un análisis de habilidades e intereses de cada integrante. A partir de esto, se definieron los roles que lograron cubrir todas las áreas necesarias para el desarrollo del robot, aparte se optó por un sistema de rotación semanal para que todos los integrantes pudieran experimentar cada función del proyecto.

Jefe de Proyecto / Documentador: coordina el grupo, redacta informes y bitácoras.

Ensamblador: encargado del armado y de probar el funcionamiento del brazo robótico.

Programador: desarrollan los movimientos y controles del brazo en la app Spike Prime y Python.

Diseñador: se encarga de proponer mejoras de diseño, asegurar el funcionamiento y que se vea bien estéticamente el brazo robótico.

2.2. Personal que Cumplirá los Roles

Rol	Responsable
Jefe de Proyecto / Documentador	Carlos Cossio
Ensamblador	Franco Churata, Bryan Palacios
Programador	Joaquín Quezada
Diseñador	Benjamin Aguilera

2.3. Métodos de Comunicación

- WhatsApp: para coordinación rápida y avisos inmediatos.
- Discord: para reuniones de trabajo online
- Reuniones presenciales: durante las clases de taller.
- Redmine: para entrega de documentos y bitácoras semanales.

3. Planificación del Proyecto

3.1. Actividades

A continuación, se detallan las actividades planificadas para el desarrollo del brazo robótico. Esta planificación está estructurada para seguir un flujo lógico y progresivo que se realizó durante el avance del proyecto.

Nombre	Descripción	Responsable	Producto
Investigación	Revisión de ideas, análisis de modelos y revisión de compatibilidad de piezas.	Todo el grupo	Diseño Conceptual

Ensamble	Construcción del modelo base y conexión de piezas	Franco Bryan	Garra parcialmente funcional
Documentación	Registro de bitácoras semanales y elaboración del informe	Todo el grupo	Robot en Movimiento
Codificación	Programación de movimientos básicos con bloques con el software Lego MINDSTORMS	Joaquín Quezada	Código en Python funcional
Presentación Final	Preparación de exposición y demostración	Todo el grupo	Presentación del proyecto

3.2. Carta Gantt

La Carta Gantt representa la visualización cronológica de todas las actividades detalladas, junto a su estado de avance. Esta herramienta es fundamental para la gestión de este proyecto, ya que permite al grupo ver los plazos, la asignación de cada tarea y el siguiente de esta, asegurando el cumplimiento de las actividades de manera organizada y eficiente.

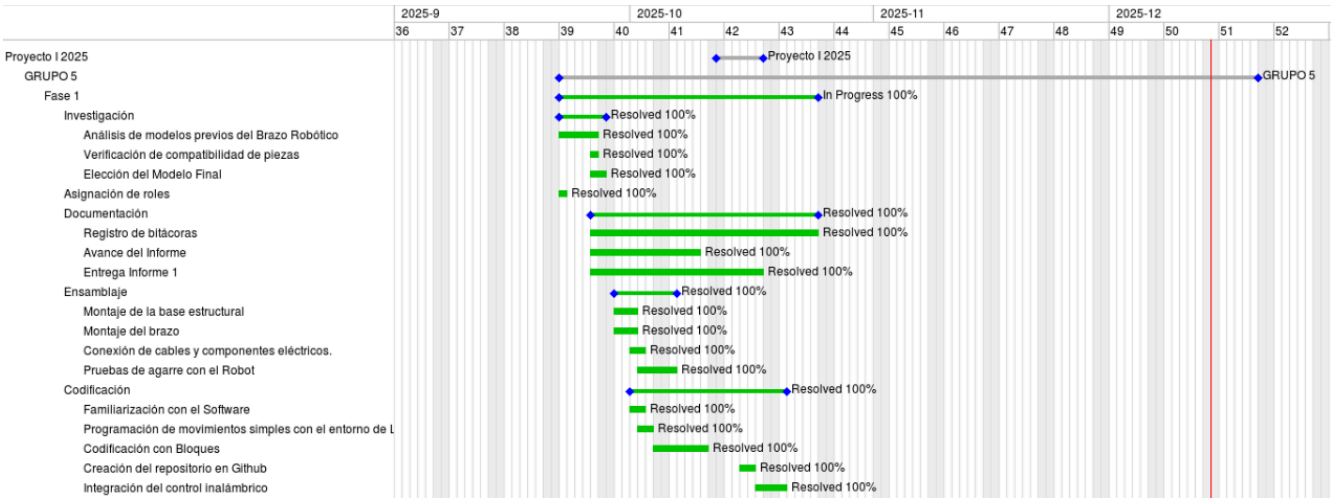


Figura 3.1: Carta Gantt del Proyecto “Brazo Robótico de agarre con LEGO”. Detalla la planificación temporal de la fase 1 del proyecto (Investigación, Documentación, Ensamble y Codificación), mostrando el avance que se realizó durante el semestre.

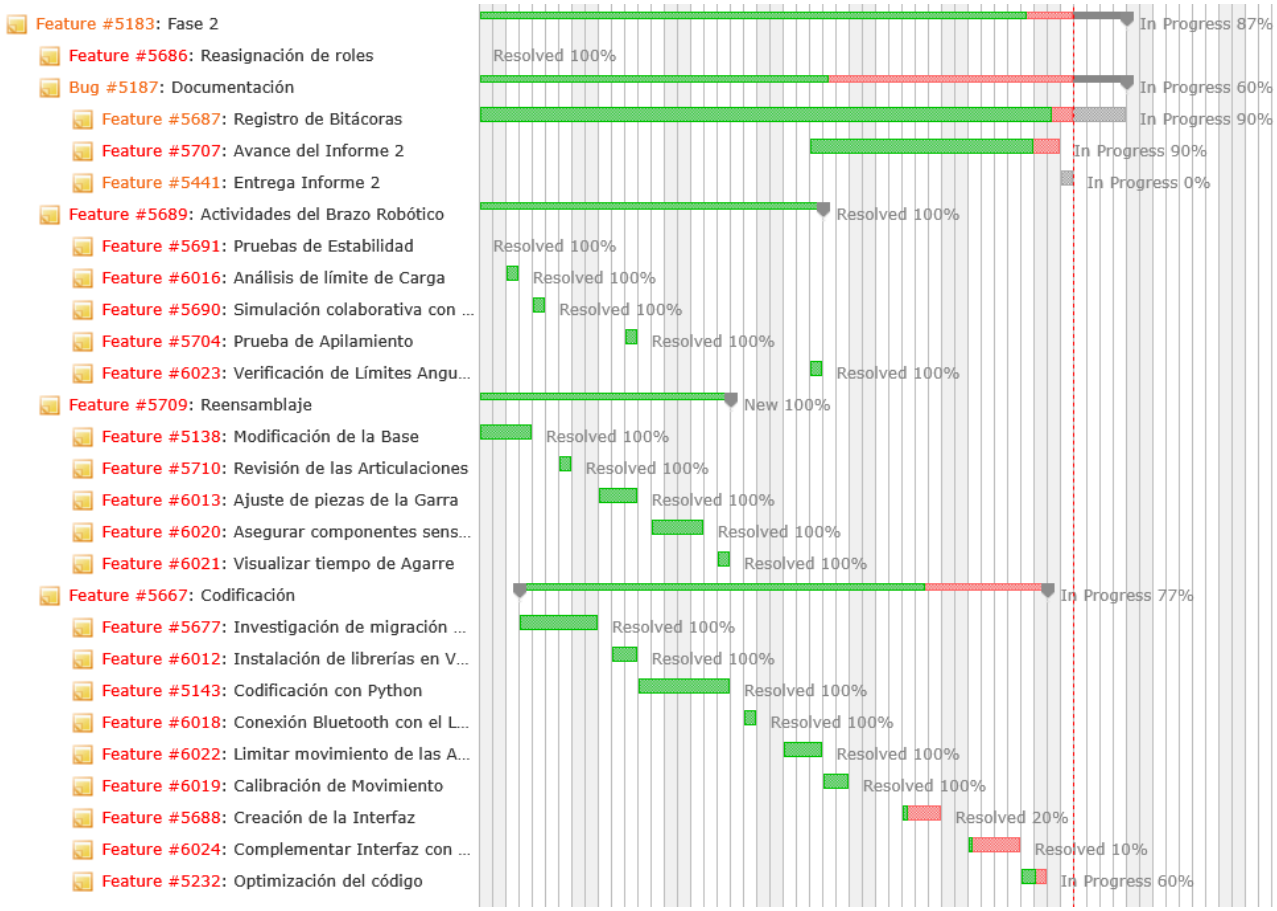


Figura 3.2: Carta Gantt del Proyecto “Brazo Robótico de agarre con LEGO”. Detalla la planificación temporal de la fase 2 del proyecto (Investigación, Documentación, Actividades del Brazo Robótico, Reensamble y Codificación), mostrando el avance que se realizó durante el semestre.

3.3. Gestión de Riesgos

En esta sección se realiza para identificar, analizar y planificar respuestas a eventos que podrían afectar negativamente los objetivos del proyecto. A continuación, se detallan los riesgos identificados, el nivel de su impacto y las acciones para solucionar esos tipos de eventos.

El Nivel de Impacto se definió utilizando una escala del 1 al 5, donde un valor más alto representa una amenaza más grave para el cumplimiento de los objetivos del proyecto:

- 1 (Bajo): riesgo que tiene una solución sencilla o con un impacto mínimo en el proyecto.
- 3 (Medio): riesgo que requiere tiempo y recursos adicionales, afectando el tiempo de trabajo.
- 5 (Alto): riesgo que amenaza la viabilidad o el alcance del proyecto.

Riesgos	Nivel de Impacto	Acción Remedial
Incompatibilidad con mando externo	1	Buscar alternativas dentro del software Spike Prime o mediante Python.
Pérdida del robot o desarme entre clases	2	Guardar el robot en un lugar seguro y registrar su estado al finalizar cada sesión.
Falta de tiempo y carga académica	3	Coordinar mejor los horarios y distribuir tareas entre los miembros.
Falta de piezas	3	Solicitar piezas de reemplazo o usar piezas del kit de expansión.
Errores de codificación	4	Revisar código en equipo y comparación ejemplos de Spike Prime.

4. Planificación de los Recursos

En esta sección presentamos la planificación de los recursos necesarios para llevar a adelante nuestro proyecto. Para organizar adecuadamente el trabajo , identificamos tres tipos de recursos fundamentales. Primero, los recursos de hardware, esto incluye todos los elementos físicos usados para construir y probar el Brazo robótico , después tenemos los recursos de software los cuales son gratuitos los cuales abarcan las herramientas digitales que usamos para programar , documentar y gestionar el proyecto , Por último consideramos los recursos financieros donde tenemos en cuenta los costos asociados tanto a los materiales , el tiempo invertido por el equipo.

Para calcular los costos, usamos un método basado en las horas dedicadas y un valor referencial por hora , todo esto en CLP. Esto nos permitió determinar un costo estimado por integrante, esto nos llevó a ver mejor el costo total del proyecto.

4.1. Hardware

Lego Spike Prime
Kit de Expansión
Portátiles

Software

Spike Prime App
Visual Studio Code
Python(MicroPython)
Redmine

4.2. Estimación de costos

La estimación de los costos totales del proyecto se basó en la suma de los recursos materiales (hardware) y el costo de la dedicación del equipo (mano de obra).

Cálculo de la Mano de Obra

El costo de la mano de obra se calculó considerando una tarifa de $3.000 CLP por hora y una dedicación individual de 20 horas.

Concepto	Valor por Hora	Horas por Integrante	Costo por Integrante
Mano de Obra	$3.000 CLP	20	$60.000 CLP

Recursos de Hardware

A continuación, se detalla el listado del equipo utilizado:

Recurso	Cantidad	Precio Unitario (CLP)
Lego Spike Prime	1	$600.000
Kit de Expansión	1	$150.000
Portátil (Suministrado por la Universidad)	5	$350.000

Resumen de la Estimación Total de Costos

El costo total estimado para el desarrollo del proyecto asciende a $2.800.000 CLP, distribuido de la siguiente manera:

Concepto	Costo Total (CLP)
Hardware	$2.500.000
Mano de Obra	$300.000
Total Estimado	**$2.800.000**

5. Análisis y Diseño

El análisis y diseño del sistema constituyen una etapa crítica dentro del desarrollo del brazo robótico de agarre, ya que permiten establecer con claridad los requisitos funcionales del sistema, las características de calidad que debe poseer para operar de manera confiable y la arquitectura de software necesaria para articular la interacción entre el usuario y el robot. Esta sección presenta el proceso de especificación de requerimientos, el diseño de la arquitectura cliente-servidor que soporta el control del prototipo y la propuesta preliminar de la interfaz gráfica de usuario, la cual facilitará la interacción entre el operador y el brazo robótico.

El propósito principal del diseño es asegurar que el sistema sea operable, mantenible y extensible, de manera que pueda integrarse adecuadamente dentro del proyecto colaborativo más amplio al cual pertenece, manteniendo coherencia tanto en su funcionamiento mecánico como en su estructura de software.

5.1. Especificación de requerimientos

Problema a resolver

En el contexto de una operación minera que ha experimentado accidentes asociados a labores de transporte, carga y descarga de mineral, surge la necesidad de automatizar estos procesos para reducir la exposición de los trabajadores a zonas de riesgo. La empresa requiere un sistema capaz de manipular materiales de manera segura, precisa y sin intervención humana directa.

Como representación en el ámbito educativo de este escenario industrial, el presente proyecto desarrolla un sistema automatizado compuesto por un organizador de material, un vehículo autónomo y un brazo robótico encargado de la carga y descarga. En particular, el brazo debe ser capaz de recoger y depositar bloques que en el contexto académico representan unidades de mineral de forma confiable. Para cumplir con esta necesidad, el equipo construirá y programará un brazo robótico utilizando LEGO Spike Prime, controlado mediante una interfaz gráfica o un mando externo.

Usuario del sistema

En el contexto minero, el usuario corresponde al operador encargado de supervisar el sistema automatizado desde una zona segura, monitoreando la manipulación del material sin intervenir físicamente en la faena. Este usuario requiere una interfaz clara, simple y confiable para minimizar riesgos y garantizar continuidad operativa.

En el contexto del proyecto universitario, el usuario es el integrante del equipo designado para operar el brazo robótico durante las pruebas y demostraciones. Esta persona interactúa directamente con la interfaz gráfica o con el mando para ejecutar las acciones del sistema,

verificar el funcionamiento del agarre y asegurar que el brazo cumpla correctamente las tareas de carga y descarga simuladas.

Cliente del sistema

En el entorno minero, el cliente corresponde a la empresa operadora que busca automatizar sus procesos de transporte y manipulación de mineral con el fin de mejorar la seguridad, reducir incidentes y optimizar la eficiencia de la faena.

En el marco académico, el cliente es el profesor a cargo del proyecto, quien define los objetivos, supervisa el desarrollo del prototipo y evalúa los resultados finales. El profesor actúa como la entidad que “solicita” la solución técnica que el equipo desarrolla.

5.1.1. Requerimientos Funcionales

Los siguientes requerimientos establecen las funciones esenciales que el sistema deberá cumplir. Estos se definieron considerando las capacidades técnicas del kit LEGO Spike Prime, las restricciones impuestas por el entorno de trabajo y los objetivos específicos del proyecto.

Requerimientos Funcionales (RF):

- **RF1:**El sistema logra abrir la garra mediante un comando enviado desde la interfaz gráfica.
- **RF2:**El sistema permite que la Garra pueda cerrarse cuando el usuario lo solicita.
- **RF3:** El brazo robótico debe de poder elevarse en una posición segura para el transporte del bloque.
- **RF4:** El brazo robótico debe poder descender su brazo hasta la posición de agarre.
- **RF5:** El mecanismo de apertura del brazo robótico debe permitir soltar el bloque cuando el comando correspondiente sea ejecutado.
- **RF6:** La interfaz gráfica debe enviar las instrucciones hacia el brazo robótico utilizando comunicación inalámbrica.
- **RF7:** El servidor instalado en el hub Spike Prime debe recibir y procesar los comandos enviados por el cliente, ejecutándose sobre los motores correspondientes.
- **RF8:** La interfaz gráfica debe ofrecer botones que permitan controlar manualmente

cada acción del brazo robótico.

- **RF9:** El brazo robótico debe ser capaz de rotar 360 grados sobre su eje principal para permitir la correcta orientación del bloque durante las etapas de agarre y depósito.

- **RF10:** El brazo debe iniciar su funcionamiento en una posición inicial definida, garantizando consistencia en las operaciones.

- **RF11:** El sistema debe mostrar información sobre el estado de la conexión entre cliente y interfaz gráfica del brazo robótico.

5.1.2. Requerimientos No Funcionales

- **Requerimientos No Funcionales (RNF): Criterios de Calidad del Prototipo**

  Los requerimientos no funcionales definen las características de calidad que el sistema debe poseer para garantizar que los requerimientos funcionales se cumplan de manera adecuada. Estas condiciones son indispensables para asegurar la estabilidad, seguridad y eficiencia del prototipo en operación.

ID	Requisito No Funcional	Descripción y Justificación	RF Relacionados
RNF1	**Estabilidad de la Comunicación**	La conexión inalámbrica entre el cliente y el servidor debe mantenerse ininterrumpida durante toda la operación del brazo, evitando cortes que afecten la ejecución de movimientos.	RF7, RF8, RF12
RNF2	**Tiempo de Respuesta (Latencia)**	El sistema debe procesar y ejecutar cada comando de la interfaz en un intervalo de 200–500 ms como máximo, asegurando un control fluido y en tiempo real del brazo robótico.	RF1, RF2, RF3, RF4, RF5, RF6, RF9

RNF3	**Precisión Posicional**	Los motores deben ejecutar los movimientos de elevación, descenso, agarre, apertura, cierre y rotación sin desviaciones que comprometan la estabilidad o la orientación del bloque manipulado.	RF3, RF4, RF5, RF10
RNF4	**Fiabilidad del Agarre**	El mecanismo de sujeción debe mantener el bloque firme durante todo el ciclo de movimiento, incluyendo rotaciones de 360°, eliminando el riesgo de caída.	RF5, RF10
RNF5	**Seguridad Operacional**	El sistema debe prevenir movimientos súbitos o velocidades excesivas que puedan dañar la estructura LEGO o resultar en una pérdida de control del bloque.	RF3, RF4, RF5, RF10
RNF6	**Usabilidad de la Interfaz (GUI)**	La interfaz gráfica debe ser intuitiva, con controles y botones claramente identificados, permitiendo que cualquier miembro del equipo pueda operar el sistema sin necesidad de entrenamiento previo.	RF1, RF2, RF3, RF4, RF6, RF9
RNF7	**Robustez y Tolerancia a Fallos**	El servidor debe ser capaz de tolerar la recepción de comandos repetidos, desordenados o con retraso sin experimentar fallos críticos, manteniendo la estabilidad operativa del sistema.	RF7, RF8, RF12
RNF8	**Mantenibilidad del Código**	El código fuente debe estar modularizado, bien estructurado y documentado con comentarios, facilitando futuros ajustes en la secuencia de movimientos, la GUI o la lógica cliente-servidor.	Todos los RF del sistema

RNF9	**Consistencia de Ejecución**	El brazo robótico debe exhibir el mismo comportamiento predecible ante un comando específico, garantizando uniformidad en las acciones de agarre, movimiento y rotación.	RF1, RF2, RF3, RF4, RF5, RF10
RNF10	**Disponibilidad Continua**	El sistema debe ser capaz de operar sin interrupciones durante las sesiones de prueba, evitando desconexiones, reinicios inesperados o fallos que paralicen su uso.	RF7, RF8, RF12

5.2. Arquitectura de software

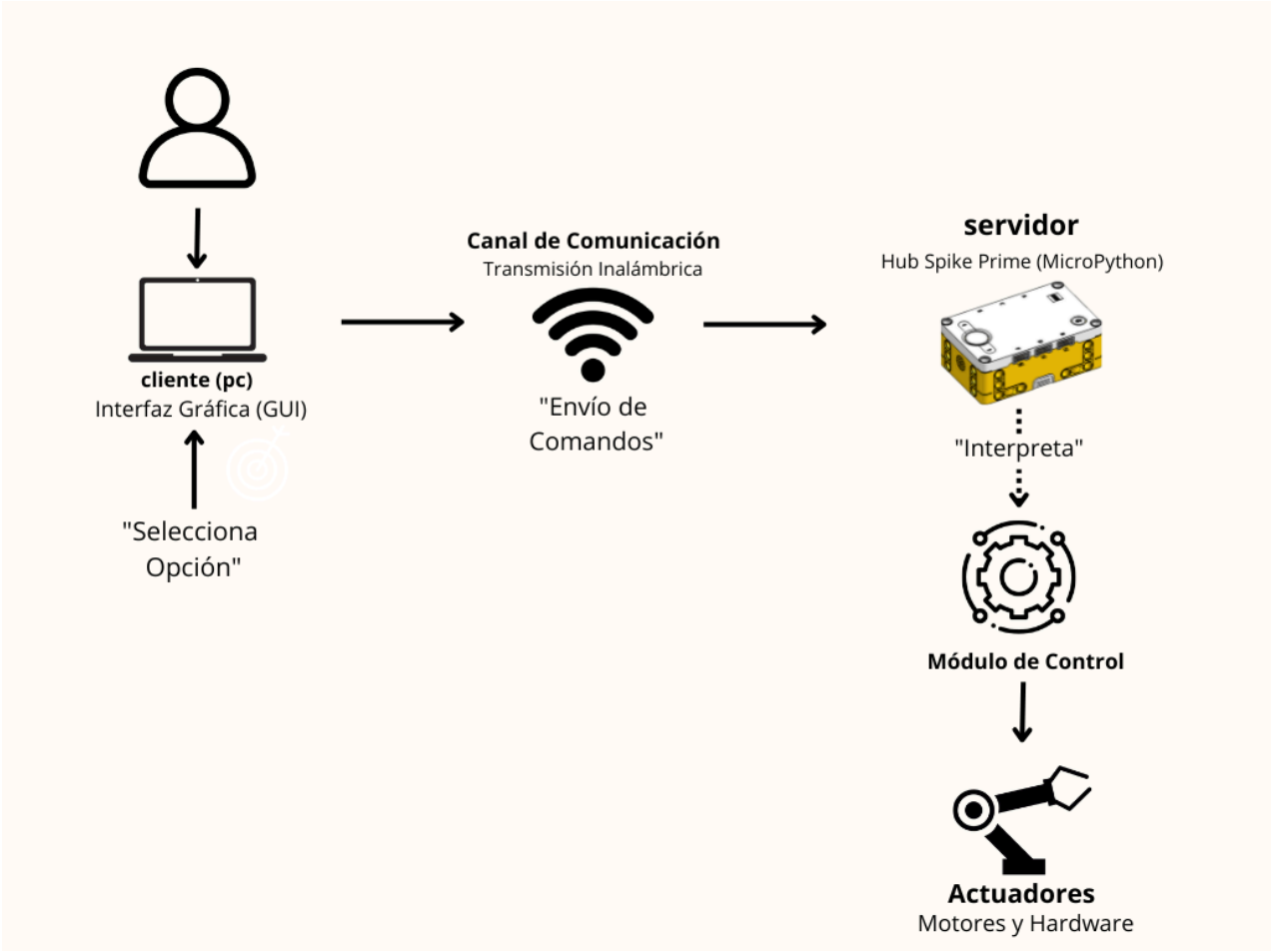
La arquitectura de software es clave para que este proyecto funcione. Básicamente, se encarga de organizar bien todos los componentes del sistema, cómo se relacionan entre sí y bajo qué reglas se diseñan y evalúan. En nuestro desarrollo, elegir e implementar la arquitectura correcta es súper importante porque es lo que asegura la flexibilidad que necesitamos para el control manual y la capacidad de procesamiento que tiene que dar el *hardware*.

Así es como funciona el sistema:

- **Cliente:** Va en una computadora personal y cuenta con una interfaz gráfica (GUI) cuyo diseño se asemeja al de un joystick virtual. A través de esta interfaz, el usuario puede controlar de forma intuitiva los movimientos del sistema, como mover de derecha a izquierda el brazo, subir o bajar y abrir o cerrar la garra de este mismo La interfaz gráfica (GUI) interpreta las acciones del usuario y las traduce en comandos que son enviados por la red para su ejecución.
- **Canal de comunicación:** Aquí usamos una conexión inalámbrica (normalmente Bluetooth o Wi-Fi, según lo que soporte el *hub* Spike Prime. Su trabajo principal es gestionar el envío y la recepción de mensajes entre el cliente y el servidor.
- **Servidor:** El *hub* Spike Prime corre un programa hecho en MicroPython o Pybricks. Su misión es recibir los comandos que manda el cliente, interpretarlos y, a partir de ahí, **ejecutar los movimientos en los motores** del brazo robótico.
- **Módulo de control del robot:** Este módulo es fundamental. Se encarga de traducir los comandos abstractos ("OPEN", "UP", "SEQUENCE") en movimientos concretos y precisos de los motores, controlando la velocidad, el torque y la posición.

Este diseño arquitectónico estratégico nos permite separar muy bien la lógica de control de la parte de interacción con el usuario. Esto hace que sea mucho más fácil corregir errores,

añadir mejoras o, si toca, cambiar algún componente sin que se caiga todo el sistema.

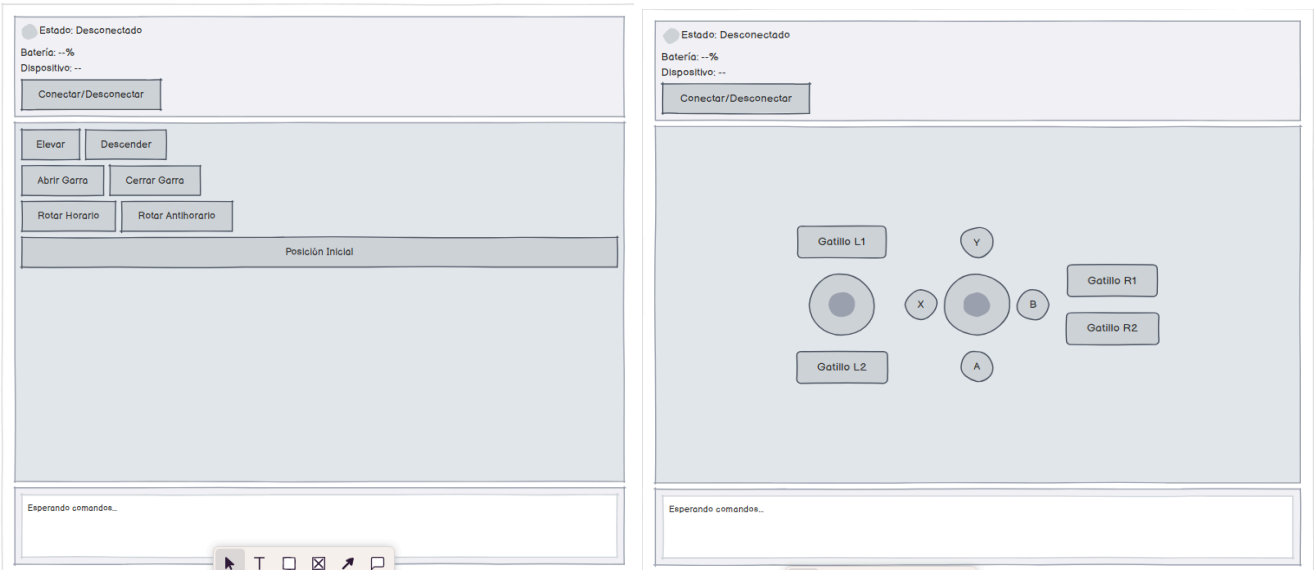


5.3. Diseño inicial de la interfaz gráfica de usuario (GUI)

La Interfaz Gráfica de Usuario (GUI) constituye el punto de contacto crítico entre el operador y el sistema robótico, siendo fundamental para la arquitectura Cliente-Servidor (sección 5.2). Para asegurar la operación eficiente del brazo robótico, se requiere que la interfaz cumpla con altos estándares de Usabilidad (RNF 4.1).

Inicialmente, el diseño se aborda mediante un bosquejo (wireframe) de baja fidelidad, un diagrama esquemático que representa la estructura y jerarquía de los elementos de control (botones, indicadores, áreas de texto) sin incluir aspectos estéticos. El objetivo de este paso es planificar la disposición lógica de los controles manuales (RF9, RF10), la visibilidad del feedback (RF12) y la coherencia del flujo de operación antes de proceder a la codificación final.

Este diseño será refinado durante la etapa de implementación, incorporando mejoras basadas en pruebas internas y retroalimentación del equipo.



6. Implementación

1. **Justificación Física y Validación del *Hardware***

   El diseño mecánico del brazo robótico se fundamenta en principios de movimiento, específicamente el Trabajo Mecánico. Se implementó un sistema de engranajes en la estructura para optimizar la relación velocidad-torque, esencial para la rotación y el posicionamiento preciso de las articulaciones dentro del área de trabajo.

   Los motores están configurados para generar el trabajo mecánico necesario para superar las fuerzas internas (peso del brazo y resistencia de los engranajes) y manipular la carga externa (el bloque). La integración de los engranajes es crucial, ya que facilita la transmisión del movimiento y aumenta el torque disponible en las articulaciones que requieren mayor potencia.

   La validación del *hardware* se confirmó mediante pruebas semanales exhaustivas, demostrando que **el brazo fue capaz de realizar** tareas repetitivas y estables de movimiento, levantamiento y agarre. Esto comprueba que el diseño y el trabajo mecánico ejecutado por el Brazo Robótico **fueron** plenamente adecuados para cumplir con los objetivos del proyecto.

6.1. Fundamentos de los movimientos

Esta sección tiene como objetivo justificar la configuración mecánica seleccionada para el robot, aplicando conceptos fundamentales de física estudiados en la asignatura Introducción a la Física. Este análisis se realiza considerando las capacidades reales de torque y velocidad de los motores **LEGO Spike Prime** y las limitaciones estructurales impuestas por el material.

6.1.1. Aplicación del Principio de Velocidad Angular (Justificación del Rendimiento)

Para el motor encargado de la **rotación de 360 grados de la garra** (RF10), se utiliza el concepto de **Velocidad Angular (ω)** para validar el cumplimiento del Requerimiento No Funcional de Rendimiento (RNF 1.2).

La Velocidad Angular se define como la razón de cambio del ángulo (Δθ) con respecto al tiempo (Δt):

$\omega = \Delta\theta/\Delta t$

1. Parámetros de Desempeño Ajustados:

- **Desplazamiento Angular (Δθ):** 360° (2π radianes).
- **Tiempo Objetivo (Δt):** Se establece un tiempo límite realista de Δt = 3 segundos para la rotación (ajustado del RNF 1.2).

2. Cálculo de la Velocidad Angular Requerida:

La velocidad angular mínima que debe alcanzar el motor es:

$$\omega\, requerida = 360°/3\ (s) = 120°/s$$

Convertido a revoluciones por minuto (RPM): ω requerida ≈ 20 RPM.

3. Conclusión de Configuración: El motor LEGO Spike Prime seleccionado tiene una capacidad superior a los 20 RPM. Al operar el motor a una velocidad que alcance este umbral, se garantiza que el requerimiento de 3 segundos para la rotación se cumpla eficientemente, permitiendo además margen para la aceleración y la inercia del brazo.

6.1.2. Aplicación del Principio de Trabajo Mecánico (Justificación de la Capacidad de Carga)

Se utiliza el concepto de **Trabajo Mecánico (W)** para justificar que el motor de elevación puede superar el peso del objeto a manipular (RF3, RF5). El trabajo es el producto de la fuerza aplicada (F) por la distancia recorrida (d):

$$W = F \cdot d$$

1. Parámetros de Carga Ajustados (Centrado en un Bloque):

- **Masa a Levantar (m):** Considerando únicamente un bloque ligero de LEGO (ej. un *brick* de 2×4), la masa estimada es m bloque = 0.003 kg (3 gramos).
- **Altura de Elevación (d):** La altura de elevación segura es d = 0.10 m (10 centímetros).
- **Aceleración de Gravedad (g):** g≈9.8 m/s2.

2. Cálculo de la Fuerza Requerida (Peso del Bloque):

La fuerza de elevación (Peso, P) es:

$$F = m\,bloque \cdot g$$
$$F = [0.003\ (kg)] \cdot [9.8\ (m/s^2)]$$
$$F \approx 0.0294\ (N)$$

3. Cálculo del Trabajo Mecánico Total:

El trabajo (W) que el motor debe realizar para elevar el bloque es:

$$W = F \cdot d$$
$$W = [0.0294\ (N)] \cdot [0.10\ (m)]$$
$$W \approx 0.00294\ Joules\ (J)$$

4. Conclusión de Configuración: El valor del trabajo requerido es extremadamente bajo (0.00294 J). Este resultado confirma que el motor de elevación no tendrá problemas para superar la carga de un solo bloque de LEGO. La principal consideración de diseño no es la fuerza, sino el **control de la posición y la precisión** para garantizar que el brazo se detenga exactamente en la posición de agarre (RF4) y la posición de depósito (RF3), utilizando los codificadores de los motores Spike Prime para asegurar una precisión angular requerida por el sistema (RNF 3.3).

6.2. Descripción del sistema

El sistema desarrollado constó de dos componentes de *software* principales: el Cliente y el Servidor, en línea con la arquitectura Cliente-Servidor definida en la Sección 5.2. Para facilitar la trazabilidad y la colaboración, el código fuente de ambos programas estuvo centralizado en un repositorio de GitHub.

- https://github.com/joaquinquezadaflores-cell/Proyecto-1/tree/main

6.2.1. Cliente y Servidor

Componente	Plataforma y Lenguaje	Función Principal	Componentes Clave del Código
Cliente	Python (Portátil)	**Control Manual e Interfaz:** Se encarga de recibir las instrucciones del operador y convertirlas en comandos digitales, los cuales son transmitidos de forma inalámbrica.	Los módulos clave son: Interfaz Gráfica de Usuario (GUI), Comunicación y Empaquetamiento de Comandos.
Servidor	MicroPython (Hub Spike Prime)	**Control de Actuadores:** Se encarga de recibir, decodificar y procesar las órdenes para ejecutar movimientos exactos mediante los motores.	Los módulos principales son el de Recepción Inalámbrica, el de Control de Motores (que gestiona las rutinas de rotación y elevación) y el de Reporte de Estado.

```
8   motor_D = Motor(Port.D, Direction.CLOCKWISE)     # motor de la base
9   motor_F = Motor(Port.F, Direction.CLOCKWISE)    # motor de la muñeca
10  motor_B = Motor(Port.B, Direction.CLOCKWISE)     # motor del brazo
11  motor_C = Motor(Port.C, Direction.CLOCKWISE)     # motor de la garra
12
```

De la línea 8 a la 11, se definen cuatro variables en las cuales se definen los motores y los puertos en los cuales están conectados.

```
15      while True:
```

En la línea 15 se crea un ciclo while True, el cual hace que lo que esté dentro se ejecute indefinidamente hasta que se cierre el programa.

```
17          buttons = controller.buttons.pressed()
18          lx, ly = controller.joystick_left()
19          rx, ry = controller.joystick_right()
20          lt, rt = controller.triggers()
```

En las líneas 17 a la 20 se definen las variables para los botones, los gatillos y las palancas.

```
22          #Movimiento de la base
23
24          if Button.RB in buttons:
25              motor_D.dc(30)
26          elif Button.LB in buttons:
27              motor_D.dc(-30)
28          else:
29              motor_D.stop()
```

De la línea 22 a la 29, se tiene la lógica de control de motores, si el botón RB está presionado, el motor D (correspondiente a la base), se va a mover en sentido horario, si el botón LB es presionado se va a mover en sentido antihorario, en caso de que ninguno de los botones esté presionado el motor se va a detener hasta la línea 56 es básicamente lo mismo para el resto de motores con sus respectivos valores y botones.

```
58          wait(200)
```

En la línea 58 al final del ciclo while, se hace un wait, el cual según su valor 200 hace que el programa se pause por 0,2 segundos para prevenir problemas al realizar cambios bruscos de dirección en los motores. la función wait de la librería Pybricks hace que el programa se pause por la cantidad de milisegundos entregados como parámetro, en el caso de este programa 200 milisegundos, lo que equivale a 0,2 segundos.

6.2.2. Interfaz gráfica de usuario (GUI)

Actualmente, el proyecto se encuentra en la fase de **desarrollo de la interfaz gráfica de usuario (GUI)**. Dado que esta etapa aún no ha concluido, **no se dispone de resultados visuales ni funcionales para presentar** en este apartado. La implementación y validación de la interfaz serán abordadas en las siguientes fases del proyecto y documentadas posteriormente.

7. Resultados

7.1. Estado actual del proyecto

El estado actual del proyecto corresponde a un sistema funcional pero básico. El brazo robótico es capaz de realizar correctamente las tareas principales para las cuales fue diseñado, demostrando un comportamiento estable durante las pruebas realizadas.

En cuanto a la funcionalidad alcanzada, el sistema permite abrir y cerrar la garra, elevar y descender el brazo, rotar 360 grados sobre su eje y manipular bloques de forma segura, logrando sostenerlos, moverlos y soltarlos sin inconvenientes. Además, se implementó una interfaz gráfica de control que permite al usuario operar manualmente el brazo y enviar comandos al robot mediante comunicación inalámbrica, la cual se ha comportado de manera estable.

Respecto a los requerimientos funcionales definidos inicialmente, en su mayoría han sido cumplidos en esta fase del proyecto, ya que el sistema responde a las acciones solicitadas por el usuario y ejecuta los movimientos esperados. No obstante, el cumplimiento se da dentro de un alcance limitado, propio de un prototipo funcional, por lo que aún existen aspectos susceptibles de mejora.

Entre las principales limitaciones del sistema se identifican pequeños retardos en la respuesta del robot frente a los comandos enviados desde la interfaz gráfica, así como dificultades ocasionales del motor encargado de elevar el brazo al trabajar con carga. También, el sistema no cuenta con procesos automatizados, operando completamente bajo control manual.

En resumen, el proyecto se encuentra en un estado operativo y cumple con la funcionalidad propuesta para esta primera fase, sentando una base sólida para futuras mejoras orientadas a la optimización del rendimiento y la incorporación de mayores niveles de automatización.

7.2 Problemas encontrados y solucionados

Durante el desarrollo del proyecto se presentaron diversos inconvenientes técnicos y de organización propios del proceso de construcción y programación del brazo robótico. Estas dificultades permitieron al equipo identificar ajustes necesarios para el funcionamiento general del sistema.

Uno de los principales problemas tiene relación con la disponibilidad limitada de las piezas LEGO, lo que obligó a cambiar el diseño original de los mecanismos. Esta situación se logró solucionar gracias al kit de extensión logrando una estructura estable y operativa.

También se encontraron problemas con la comunicación inalámbrica entre la interfaz gráfica y el hud del Spike Prime, lo que provocaba que algunos comandos tardarán más en ejecutarse. Para solucionar este problema, modificamos el envío y la ejecución de comandos, mejorando así la eficiencia de la comunicación.

Finalmente, debido al requerimiento de contar con un sistema de control mediante una interfaz gráfica, fue necesario diseñar e implementar una interfaz gráfica con funcionamiento tipo joystick virtual. Este proceso implicó un período de adaptación y pruebas, pero permitió lograr un control más intuitivo y acorde a los objetivos del proyecto.

8. Conclusión

En conclusión, el proyecto desarrollado cumple con el objetivo general propuesto, logrando implementar un brazo robótico funcional capaz de manipular bloques de manera segura y controlada. El sistema demostró un funcionamiento estable durante las pruebas realizadas, permitiendo ejecutar correctamente las acciones de agarre, transporte y liberación de bloques.

Entre los principales avances alcanzados se destaca la implementación de una interfaz gráfica de control, la cual permitió mejorar la forma en que el usuario interactúa con el sistema y

consolidar el modelo de control planteado para el proyecto. Esta incorporación facilitó el envío de comandos y el manejo manual del brazo robótico de manera más clara y organizada.

Durante el desarrollo del proyecto se enfrentaron diversos problemas, principalmente relacionados con la comunicación y la asignación de tareas dentro del grupo de trabajo. Como solución parcial a esta dificultad, se optó por realizar las sesiones de avance con todos los integrantes presentes de forma online, lo que permitió mejorar la coordinación y avanzar de manera más ordenada en las distintas etapas del proyecto.

Si bien el sistema cumple con lo solicitado, aún existe un margen de mejora. Entre las tareas más relevantes a desarrollar en etapas posteriores se encuentran la optimización de la interfaz gráfica y la automatización de algunos procesos del robot, lo que permitiría aumentar el nivel de eficiencia y funcionalidad del sistema.

9. Referencias

[1] Lego Brick & Tech. "Building a 360-Degree Rotating LEGO Robotic Arm | SpikePrime 【Tutorial】." YouTube. [Online]. Disponible en: https://www.youtube.com/watch?v=fAM7FBOOy74 (3 de enero de 2025).

[2] LEGO Education. "LEGO Education SPIKE Prime." [Online]. Disponible en: https://education.lego.com/es-es/product/spike-prime.

[3] Pybricks. "Pybricks Code." [Online]. Disponible en: https://code.pybricks.com.

[4] Universidad de Tarapacá (UTA). "Plataforma Redmine UTA." [Online]. Disponible en: https://redmine.uta.cl.

[5] IREA 국제로봇교육협의회. "How to connect Remote controller with Spike prime." YouTube. [Online]. Disponible en: https://www.youtube.com/watch?v=jdy6h0xA6lg (15 de junio de 2023).